NASA Contractor Report 180781

VAPEPS USER'S REFERENCE MANUAL

VERSION 5.0

Prepared by:

D. M. Park

Lockheed Missiles and Space Company
1111 Lockheed Way
Sunnyvale, California 94086

July 1988

Under
Contract no. NAS5-29452

NASA
National Aeronautics and Space Administration
for
Goddard Space Flight Center
Greenbelt, Maryland 20771
and
United States Air Force Space Division
Los Angeles, California 90009

Preface

A database management and prediction system named VAPEPS (VibroAcoustic Payload Environment Prediction System) has been developed for the vibration and acoustic data obtained from Space Shuttle and Expendable Launch Vehicle payload components. VAPEPS accepts vibroacoustic and structural data and constructs a database which the aerospace community can utilize to establish environments for new payload components. Additionally, VAPEPS provides a full range of software routines that can be used to manipulate and perform computational operations on data.

This document is the VAPEPS User's Reference Manual. It is intended as the primary reference to commands in the VAPEPS software. Before attempting to read this manual it is suggested that the VAPEPS User's Guide be read. It provides an introduction to the VAPEPS software and some of the most used commands. The majority of the text in this manual is contained in the on-line help files. The on-line help will always be more current than the manual. Therefore, if a question arises, that is not answered satisfactorily in this manual, the on-line documentation may be consulted to see if more current documentation exists.

This volume is so large that it is nearly impossible to provide the quality of documentation desired. It is hoped that the documentation is clear and concise. This document will be constantly updated as new commands and documentation errors are reported. Eventually it is the goal of the author to provide examples for every command in this volume. However, at present only a few of the commands have good examples.

VAPEPS is described in four self-contained volumes The VAPEPS User's Reference Manual is the most current of the documentation (other than the on-line help) and should be referred to before consulting the other documentation.

Volume I: VAPEPS Technical Manual, NASA CR 166823, June 1984, which illustrates the overall design concepts, database architecture, and the prediction schemes.

VAPEPS User's Reference Manual version 5.0, NASA CR 180781, July 1988, which describes the operations necessary to build a VAPEPS database and how to use the data processing and prediction routines. This replaces **Volume II: VAPEPS USER'S MANUAL**, NASA CR 166823, December 1982 and **Volume V: VAPEPS PLOT PACKAGE**, NASA CR 166823, June 1984.

Volume III: VAPEPS Programmer's Manual, NASA CR 166823, June 1982, which describes the internal workings of the VAPEPS computer program and its interaction with the database.

Volume IV: VAPEPS Sample Problems, NASA CR 166823, December 1982, shows typical operations required in data input, data retrieval, and environmental predictions with VAPEPS.

An introduction to the VAPEPS software may be found in the **VAPEPS User's Guide**, NASA CR 180782, July 1988.

Acknowledgements

This work is supported by NASA Goddard Space Flight Center under Contract NAS5-28577 and NAS5-29452. The technical monitor is Mr. Frank On. The funding is from the U.S. Air Force Space Division. Lt. C. Brown is the program monitor.

The author wishes to acknowledge the efforts of W. Henricks who has made numerous suggestions and comments about the content of this manual. Also the input of personnel at JPL for there comments concerning errors in the documentation.

Table of Contents

Table of Contents

Table of Contents

Table of Contents

Table of Contents

Table of Contents

Table of Contents

Table of Contents

Table of Contents

Table of Contents

Table of Contents

Table of Contents

Table of Contents

Table of Contents

List of Figures

List of Figures

1. INTRODUCTION

What is VAPEPS?

VAPEPS (VibroAcoustic Payload Environmental Prediction System) is a computer program for performing vibroacoustic environmental predictions and database management tasks. The program manages a database containing test data from a variety of Space Shuttle and Expendable Launch Vehicle payloads.

VAPEPS software

VAPEPS commands can be grouped into three functional categories:

* Computational operations

Including:

mathematical - add, subtract, multiply, divide, ...
statistical - average, percentile, confidence interval, max/min, ...
vibroacoustic - normalization, manipulation of vibration specifications, shock spectra calculation,fourier transform,...
general - printing, plotting, file housekeeping, ...

* Database operations

Including:

adding to the database - data from your own test, data from another site, ...
searching the database - for a particular test, for a "class" of tests, ...
retrieving data from the database - to support a prediction, to conduct statistical studies, ...

* Prediction operations

Including:

Statistical Energy Analysis (SEA) based predictions - theoretical and extrapolated
other predictions - scaling, stress, ...

VAPEPS Files

VAPEPS commands read and store data (numbers, words, etc) on specially structured files called DAL (Direct Access Library) files.

DAL files are used to store data such as:

* input to calculations
* calculation results
* the global database
* your own local database
* text for interactive help
* text for runstreams

In addition the program can read and write to symbolic (text) files native to your computer's operating system.

1.1. PURPOSE OF MANUAL

This manual is intended to be a reference manual for users with some basic knowledge of the VAPEPS software. See the document "VAPEPS User's Guide" for an introduction to VAPEPS.

1.2. CONVENTIONS USED IN MANUAL

This manual is designed for reference and is entirely generated from the help files contained in DAL unit 29. This gives the capability to quickly update the manual when new commands are created or changes are made to the documentation. This results in a large variation in page numbering from manual to manual.

An attempt was made to place all the commands in alphabetic order in chapter 3. Some commands have such lengthy documentation that they are placed in the following chapters. Also some topics are fairly specialized and deemed necessary to be placed in separate chapters.

1.3. ORGANIZATION OF MANUAL

This manual is divided into several chapters as follows:

Chapter 1 - Introduction: Gives a general introduction to VAPEPS and the use of this manual.

Chapter 2 - General Topics: Gives a complete description of DAL files, VAPEPS free field input, Command structure and how to use the online help facilities.

Chapter 3 - General Commands: An alphabetic listing of the general computation commands.

Chapter 4 - Printing and Plotting: Describes how to display your results in the form of tables or plots.

Chapter 5 - Data Entry: Discusses how to enter data into the DAL file structure in a form acceptable to the VAPEPS Database.

Chapter 6 - Database Preparation: Gives a description of the procedures required to prepare data for inclusion in the VAPEPS database.

Chapter 7 - Searching the Database: Describes how to find and extract data from the VAPEPS database.

Chapter 8 - Database Administration: Gives commands and procedures to correctly administer the VAPEPS database.

Chapter 9 - Direct Dictionary Interface: Describes the processor which maintains the Data Dictionary for the VAPEPS database.

Chapter 10 - Statistical Energy Modeler: Describes the processor that is used to build SEA models.

Chapter 11 - Old SEA Commands: Contains an early version of vibroacoustic prediction commands described in chapter 10; the commands given in chapter 10 are now recommended.

Chapter 12 - Transparent Commands: This is a description of a set of commands which are available at all times within VAPEPS.

Chapter 13 - Variables and Symbols: Description and use of variables and symbols.

Chapter 14 - Loops and Runstreams: General concepts and guidelines for creation of high level "programs" using VAPEPS commands.

Appendix A: Database Worksheets: Worksheets designed to assist in the data entry and preparation tasks. See chapters 4 and 5.

Appendix B: Semod Worksheets: Worksheets designed to assist in SEA modeling.

Appendix C: Frequency Tables: Tables of commonly used frequency standards.

Appendix D: Conversion Tables: Commonly used conversions and codes used in the VAPEPS software.

Index: Gives list of appropriate commands given a general word describing the operation of the command.

Subject Command Summaries: Lists commands associated with different subjects or disciplines.

2. GENERAL TOPICS

This chapter contains information on some general topics as follows:

* General command structure.

* DAL (Direct Access Library) files.

* VAPEPS free field input.

* Description of the online help facility.

* List of menus available with the MENU command.

2.1. COMMAND FORMAT

Command Format

The typical VAPEPS command is of the form

```
COMMAND/OPTION  N1,EL1,VR1  N2,EL2,VR2  ARG1,IARG2,.....
-Input Data-
```

Where COMMAND is the command name, OPTION is a modifier to the basic operation of the command, and N1,EL1,VR1 and N2,EL2,VR2 refer to DAL elements that will be read or written during the command.

```
N1    - DAL unit number (1-30) (1 = DAL001, 2 = DAL002, etc.)
EL1   - DAL element name. Four characters maximum
VR1   - DAL version name. Four characters maximum
ARG1  - A floating point argument
IARG1 - An integer argument
-Input Data -
      - Although most commands only require the specification of the
        command line to produce a result, some require additional input
        data following the command line.  In these cases the command
        line functions to start a process and the input data is
        required to finish it.
```

Each command string consists of a command name followed by a group of arguments. The command name may consist of up to 12 characters. Only enough characters to make the command unique are required. For RUN= commands the first four characters are required.

Some commands have options which are words of up to 12 characters. Options are specified by placing them after the command preceeded by a /. These options, if they exist, will be described in the description of the particular command itself.

The argument list associated with each command may vary from zero to ten or more items, depending on the command.

The argument list must be separated from the command and options by at least 1 blank.

Arguments may be separated by commas, spaces or equal signs.

Multiple spaces are treated as one space, but multiple commas or equal signs are treated as blank words (hollerith).

Arguments may be of integer, real or hollerith type. Each command expects a certain number of arguments. Each argument is expected to be a certain type. For some commands, trailing arguments may be omitted, resulting in default values. It is important to note that arguments must be specified in the order expected. Arguments may not be skipped, unless such skipping is specifically spelled out in the description of the particular command.

Each command is described in its own section. These descriptions are contained in this manual and will also be available on a DAL file. This DAL file may be accessed via the HELP command during program execution.

2.2. DAL FILES

What is a DAL file?

A DAL file is a specially formatted data file. Each file may contain up to 1760 data sets (although 500 is a more realistic number in terms of efficiency). Each data set may be thought of as a matrix. These data sets will be described later. Right now we'll consider the DAL file as a whole.

Naming a DAL file

Within a given processor (eg VAPEPS), DAL files are referenced by unit number. Up to 30 different DAL files may be referenced in a given execution. By default, the unit number is linked to a local name, as follows:

```
Unit 1 = DAL001
     2 = DAL002
     .     .
     .     .
    29 = DAL029
    30 = DAL030
```

To equate a particular file with a DAL unit number, the user normally equates one of the above local file names to the desired file. This is done through commands available in your host computer's operating system. For example:

UNISYS (aka: SPERRY (aka: UNIVAC))

```
@USE  DAL001.,MY*MYDATA.
@USE  DAL003.,DAL*MODEL.     etc.
```

VAX

```
$DEFINE  DAL001     DRA0:[MY.DIRECTORY]MYDATA.DAL
$DEFINE  DAL003     USER$DISK2:[MY.DATA]MODEL
```

or you may prefer the ASSIGN statement

```
$ASSIGN  DRA0:[MY.DIRECTORY]]MYDATA.DAL  DAL001
$ASSIGN  USER$DISK2:[MY.DATA]MODEL     DAL003
```

Note that on the VAX, you need not explicitly name the file type (.DAL), as DAL is the default type.

CDC

```
   ATTACH,DAL001=MYDATA,ID=ME.
   ATTACH,DAL003=MODEL,ID=MINE.
```

CRAY

```
   ATTACH,DN=DAL001,PDN=MYDATA,ID=ME.
   FETCH,DN=DAL003,DF=TR,TEXT='$DISK MODEL.CDL'.
```

An alternate means of associating a file with a DAL unit number is via the VAPEPS 'FNAME' command. The above assignments on any system could be made as follows:

```
FNAME 1,'MYDATA'
FNAME 3,'MODEL'
```

Note that the 'qualifier' has not been specified in the FNAME command. On the VAX, you could have put

```
FNAME 1,'DRA0:[MY.DIRECTORY]MYDATA'
```

Unfortunately the same is not true on UNIVAC and CDC systems. On the UNIVAC, the FNAME approach can currently only be used for files that have your account qualifier. Similarly, CDC users must always ATTACH their files prior to entering VAPEPS. eg:

```
ATTACH,MYDATA,ID=MY.
VAPEPS.
.
.
FNAME  1,'MYDATA'
```

FNAME may also be used to redefine the default naming convention.

```
FNAME  EAL0
```

causes unit 1 to be EAL001, unit 2 to be EAL002, etc. , and

```
FNAME 'UD1:[WHATEVER]TEST.L'
```

causes unit 1 to be UD1:[WHATEVER]TEST.L01, unit 2 is ...TEST.L02, etc. The basic idea is that each name is defined as the specified name, plus 01 for unit 1, 02 for unit 2, etc.

Units 28 through 30 may not be redefined in this manner. For more information on FNAME, see the FNAME command.

DAL Data Sets

As mentioned earlier, each DAL file can contain many data sets. Each such data set is identified by a four word name:

```
ELN,VER,IC1,IC2
```

Where

ELN - Element name -- Up to 4 characters long, the first of which should be alphabetic.
VER - Version name -- Up to 4 characters long, just like ELN
IC1 - First cycle number -- An integer, normally non-negative, and less than 10000.
IC2 - Second cycle number -- Same as IC1.

This four word name is always used to locate or create a data set.

In addition to their name, data sets have other attributes. These attributes are maintained in the Table of Contents of each DAL file. The following lists a typical Table of Contents as provided through the VAPEPS TOC command:

```
 TABLE OF CONTENTS FOR DAL UNIT  1     USER$DISK0:[JES.TEMP]TUTORIAL.DAL;1

SEQ      RR   DATE    TIME  E   WORDS     NR     NC  T ELN     VER     C1 C2
  1      16 840221   93156  0      10     10      1  1 SEQ     DAT      0  0
  2      20 840221   93226  0      10     10      1  1 LOG     DAT      0  0
  3     -24 840221   93315  0      10     10      1  1 LOG     MULT     0  0
  4      28 840221   93328  0      10     10      1  1 LOG     MULT     0  0
  5      32 840221   93352  0      10     10      1  1 LOG     SUB      0  0
  6      36 840221   93358  0      10     10      1  1 DIV     DAT      0  0
  7      40 840221   93444  0     100     10     10  1 TEN     TEN      0  0
  8      52 840221   93537  0     100     10     10  1 TEN     SEQ      0  0
  9     -64 840221   93603  0      16      4      4  1 FOUR    FOUR     0  0
 10      68 840221   93618  0      16      4      4  1 FOUR    FOUR     0  0
 11     -72 840221   93733  0      33     33      1  5 LOOP    LOOP     0  0
 12      76 840221   93901  0     100    100      1  1 TEN     EXP      0  0
 13      80 840221   93942  0      27     27      1  5 LOOP    LOOP     0  0
```

SEQ

The element sequence number. This number is not actually stored in the table of contents. The number always starts with 1 and continues sequentially through the last element. Some commands allow you to identify a data set by its sequence number.

RR

The starting sector address of the data set. For most users this is a meaningless number. It is useful for two reasons. First, you will notice that a few of the data sets in the above TOC have negative values for RR. This means that the data set has been disabled. A data set will be disabled if a new data set is entered with the same four word name. It is not physically removed from the file, however it may not be accessed unless it is 'enabled'. The ENABLE and DISABLE commands allows users to enable/disable elements by sequence number.

The second use for RR stems from the fact that data sets are are always written sequentially on the DAL file. Thus the RR numbers must always increase as you go from one data set to the next. If two RR's are the same, then the two data sets have been written in the same place, and the first data set has been overwritten. Similarly a decrease in RR indicates that a block of data sets have been overwritten. This should not happen, however it can happen if an execution terminates abnormally. Once it occurs, there's nothing you can do except disable the overwritten data sets.

The PACK command may be used to remove disabled data sets from the file. It checks the RR entries, and will not pack a file if any enabled data sets have been overwritten. If some have, it indicates that some data sets have a zero or negative length, and provides you with their sequence numbers. These must be DISABLEd before the file can be packed successfully.

DATE
Six digit integer YYMMDD indicating the date the data set was created.

TIME
Six digit integer HHMMSS indicating the time the data set was created.

E
Error code. A non zero value indicates that the data set is in error. It may not be accessed, however the error code may be changed via the CHAERR command (see CHANGE)

WORDS
The total number of words in the data set. Word in this case indicates a logical word. Ie for double precision data sets, WORDS indicates the number of double precision words. The number of physical words would be twice this number.

NR
The number of rows in the data set.

NC
The number of columns in the data set. Note that the use of NR and NC

imply that each data set is a two dimensional matrix. This is not always the case, but is true more often than not.

T

An integer type code. Legal values for type include the following:

0 - Integer (The data set contains integer data)
1 - Real (eg floating point, single precision)
2 - Double precision floating point
3 - Complex (Each logical word is in reality two words, the real part followed by the imaginary part)
4 - Hollerith (normally each word will contain a maximum of four characters. On some systems a word may contain more than 4 characters, but standard PRINT will show only the leftmost four characters)
5 - Packed hollerith text. This type of data set is created by SYMIN and CSYM and read by ##READ. They may also be PRINTed.
6 - Fixed record length hollerith text. These are used just like type 5 elements but are less storage efficient (each record is padded with blanks to achieve equal length records) The default mode for CSYM and SYMIN is to create type 5 elements. This may be reset by setting the variables NCR$ and NRB$.

```
SET NCR$=80,NRB$=60
```

will cause CSYM or SYMIN to create type 6 data sets. The data sets will have 80 characters per record, and 60 records per block. Users may choose any combination of characters/record and records/block. NCR$ should not exceed 132.

ELN,VER,IC1,IC2

Is the data name, as described earlier.

General notes:

Data set type:

Each data set has a type code. Ideally all data within a given data set will consist of values of the indicated type. Some commands key on the type code and perform specialized operations depending on the type code. The PRINT command is a good example. An integer type data set will be printed using integer format. A real type via a real format, etc. If a data set contains mixed types, the default PRINT command may not be used.

Eg. if column one contains integers and column 2 contains reals and the type code is 0, then column 1 will be printed correctly, and column 2 will contain extremely large integers and probably exceed the format and give asterisks. The solution is to use the PRINF format, and specify the format that will correctly print the data set. The point is, VAPEPS allows you to mix types however if you do, it is your responsibility to take care of any complications that arise.

NR,NC vs. NI,NINJ

Users of other DAL type processors have probably seen several different formats for the NR and NC columns of the DAL TOC. This can be very confusing. In reality the NR/NC NI/NJ question has a very simple answer and is best explained by looking at the manner in which DAL data sets are stored on a DAL file.

(The following paragraph is not meant to scare you away. It may be scanned, skipped or whatever)

For Advanced Users Only

First of all, all data sets have a starting address where the data exists. This is given by the RR entry in the table of contents. The address is called a sector address. The term sector originates from the addressing characteristics of rotating mass storage devices. Eg. when I write to a disk, I specify an address. If the device is word addressable, address 1 points to word 1, and address 2 points to word 2. If it is sector addressable, address 1 points to sector 1, and address 2 points to sector 2, etc. In the later case, a sector corresponds to a certain number of words.

Some disks have a sector length of 128 words, others have 256 words, etc. What this means is that if I am writing to a disk with a sector length of 128 words, I cannot write to the 2nd word on the disk directly. I can only begin input/output operations on words 1,129,257, etc. If I want to change word 2, I must read all of the first sector, change word 2, and then rewrite all of the first sector. As you can see, the simple change of a word can involve a lot of hidden I/O. For this reason, the DAL system was designed such that most I/O requests would automatically begin on a sector boundary. Thus the following constraints:

1) All data sets must begin on a sector boundary.
2) Data sets are written as a sequence of blocks. Each block must also start on a sector boundary.

and the following implications:

1) There will in general be 'wasted' space between data sets. Eg if your data set has only 2 words and the sector length is 128, there will be 126 unused words.
2) There will also be wasted space between blocks within a given data set.

With this in mind, the best way to minimize 'wasted' space is to minimize the number of data sets and to minimize the number of physical blocks per data set. A 3x3 matrix stored as three separate physical blocks is going to waste space, (unless of course the sector length is 3 or less -- which is unlikely). For this reason, the concept of logical blocks is used. The idea is to pack several logical records into one physical block. Eg write our 3x3 as a 9x1 (physically) and just remember that it is really a 3x3.

(You can continue reading now)

Blocking of DAL data sets

Each data set is written as a sequence of blocks. Eg

```
          blocks 1        2        3        4
            ---------------------------------------------
rows  1     !        !        !        !        !
      2     !        !        !        !        !
      3     !        !        !        !        !
      4     !        !        !        !        !
      5     !        !        !        !        !
      6     !        !        !        !        !
      7     !        !        !        !        !
      8     !        !        !        !        !
      9     !        !        !        !        !
     10     !        !        !        !        !
     11     !        !        !        !        !
     12     !        !        !        !        !
            ---------------------------------------------
```

In the NR/NC approach, the above data set has NR=12, NC=4. In the pure DAL approach, there is a hidden number called NI. NI is termed the logical block length. Let NI in this case be 4, then

```
               blocks 1        2        3        4
  j     i        ---------------------------------------------
        1      !        !        !        !        !
  1     2      !        !        !        !        !
        3      !        !        !        !        !
        4      !        !        !        !        !
               - - - - - - - - - - - - - - - - - - - - - - -
        1      !        !        !        !        !
  2     2      !        !        !        !        !
        3      !        !        !        !        !
        4      !        !        !        !        !
               - - - - - - - - - - - - - - - - - - - - - - -
```

```
       1     !          !          !          !          !
  3    2     !          !          !          !          !
       3     !          !          !          !          !
       4     !          !          !          !          !
             ---------------------------------------------
```

For the above configuration, NJ=3 (eg number of sub records) and NI = 4 (length of each sub record). The block length can be computed as NI*NJ (= 12 in this case). The above matrix can be considered a three dimensional matrix (4,3,4) or as a two dimensional matrix (4,12). The NI/NJ form is the basic definition of all DAL data sets. A data set is completely described by three numbers:

```
   NWDS = total number of words
   NJ   = number of logical groups per physical block
   NINJ = NI*NJ = physical block length
```

from which, any remaining items can be calculated

```
   NI = NINJ/NJ
   NC = number of blocks = (NWDS-1)/NINJ + 1
```

If the above calculation of NC seems odd, it is due to the fact that there is no requirement that the last block be completely filled. Eg consider the following:

```
             blocks 1          2          3          4
  j    i      ----------------------------------------------
       1     !          !          !          !          !
  1    2     !          !          !          !          !
       3     !          !          !          !          !
       4     !          !          !          !          !
             - - - - - - - - - - - - - - - - - - - - - - -
       1     !          !          !          !
  2    2     !          !          !          !
       3     !          !          !          !
       4     !          !          !          !
             - - - - - - - - - - - - - - - - -
       1     !          !          !          !
  3    2     !          !          !          !
       3     !          !          !          !
       4     !          !          !          !
             -----------------------------------
```

Here, NWDS = 3*12 + 4 = 40, NI = 4 (as before) and NJ = 3 (as before). In this case it is clear that this represents a (4,10) matrix.

Hopefully the above samples have cleared up the NI/NJ issue. You may still be wondering, however, why VAPEPS uses NR/NC. First of all, it doesn't really. The true table of contents as physically stored in the file, contains NJ and NINJ values, as described above. You may list the true TOC by resetting the TOC type via the TTYP command --

```
   TTYP 2
   TOC 1
```

yields an alternate listing for the above DAL file.

```
TABLE OF CONTENTS FOR DAL UNIT  1     USER$DISK0:[JES.TEMP]TUTORIAL.DAL;1

SEQ      RR   DATE    TIME   E    WORDS    NJ  NI*NJ   T ELN     VER     C1 C2
  1      16 840221   93156   0       10     1     10   1 SEQ     DAT      0  0
  2      20 840221   93226   0       10     1     10   1 LOG     DAT      0  0
  3     -24 840221   93315   0       10     1     10   1 LOG     MULT     0  0
  4      28 840221   93328   0       10     1     10   1 LOG     MULT     0  0
  5      32 840221   93352   0       10     1     10   1 LOG     SUB      0  0
  6      36 840221   93358   0       10     1     10   1 DIV     DAT      0  0
  7      40 840221   93444   0      100     1     10   1 TEN     TEN      0  0
  8      52 840221   93537   0      100     1     10   1 TEN     SEQ      0  0
  9     -64 840221   93603   0       16     1      4   1 FOUR    FOUR     0  0
 10      68 840221   93618   0       16     1      4   1 FOUR    FOUR     0  0
 11     -72 840221   93733   0       33    10     33   5 LOOP    LOOP     0  0
 12      76 840221   93901   0      100    10    100   1 TEN     EXP      0  0
 13      80 840221   93942   0       27     8     27   5 LOOP    LOOP     0  0
```

The NJ and NINJ values show up, as promised. The problem is that you must now do a calculation to determine NI and NC. Most VAPEPS commands interpret data sets as unblocked, two-dimensional matrices. Thus, to VAPEPS, all of the above data sets represent a 12x3 matrix. As a user convenience, VAPEPS computes NC as part of the TOC process, and renames NINJ to NR and shuffles them around to yield NR and NC. The idea is to make life easier for the user. Unfortunately when VAPEPS operations are used on data sets from other processors, eg SPAR, the end result is one of confusion, not simplicity.

As a means of bridging the gap between SPAR type and VAPEPS type conventions, VAPEPS provides the EXPAND command. The SPAR type joint location matrix JLOC BTAB 2 5 has NJ = number of joints, NI = 3. To VAPEPS this is a vector, NINJx1. EXPAND may be used to convert this to a NJx3 matrix, as follows:

```
   CYCLE 2,5
   EXPAND  1,JLOC,BTAB  1,XYZ,MAT  &NJ -1
```

(Note, EXPAND has recently been improved such that you can now leave off the &NJ and -1. VAPEPS now examines the table of contents entry for NJ. If

NJ .NE. 1, it assumes it is a SPAR type blocked data set and does the above automatically. If NJ = 1, then it reverses the process automatically. You can override this of course, by providing the new row length and NJ values directly in the command)

2.3. FORMAT

Dalpro Free-field Format

A free-field format is one which allows users to input a string of values as a group of words separated by commas. It eliminates the need to place each value in a particular location on the card image, and generally makes life a lot easier for the user. DALPRO free-field format is an extension of this type of format. It provides for the above format and also allows the use of variables, implied do-loops, multiple values and indirect strings.

2.3.1. BASIC APPROACH

The basic image is a string of words separated by commas, spaces or equal signs. Thus

```
1,3,5,7,10,9,8,7,3,12
1.1, 2.2, 3.56E7, -2.61-3, 7.804
ABCD, EFGH  IJK  H   '1234'
A=57.295, I=15, NAME=TEST ITEM=3
```

are all legitimate free field forms. Basic rules are that

1) Integers are numeric characters without a decimal point. Octal values may be specified by preceding them with an exclamation point (!200 = 128)

2) Reals are numeric characters that have a decimal point. These may have an exponent as indicated above. Many exponent forms are allowable. The only restriction is that spaces are not permitted between a number and its exponent.

3) Hollerith words consist of up to four alphanumeric characters. The first character must be alphabetic. Any words input as longer than four characters will be truncated to the first four characters. Other characters may be embedded in a hollerith word by enclosing the word in single quotes. Thus

```
' ', '!', '.=*', '1234'
```

Are all legitimate hollerith words.

4) Types may be mixed on one line, however, the values must correspond to the type of data that is expected by the reading program.

2.3.2. VARIABLES

Variables are named items that refer to values contained in an internal table. Variables may be created through the SET command or operated algebraically in the CALC subprocessor. When used in free-field input, the variable name must be preceeded by an ampersand (&) to distinguish it from a regular hollerith name. Thus

```
SET I=5, X=5.7
SET J=&I, Y=&X
```

represents the use of variables. In this case, the card image is the SET command which creates the variable I and sets it equal to 5. It then creates J and indirectly sets it equal to 5 by referencing the variable I.

The type of variable is determined by its first character. If this character is between I and N, the variable is assumed to be either integer or hollerith. If not, the variable is assumed to be real. There are many ways to trick DALPRO into not following the convention. It is not worthwhile, however, and you are advised to break convention at your own risk.

A special hollerith mode may be used in conjunction with integer variables:

```
A&I,BBB&I,'!'&I
```

would result in the hollerith words:

```
A1     BBB1  !1      FOR I = 1
```

and

```
A999   BBB*  !999    FOR I = 999
```

Note that an attempt to exceed 4 characters will result in asterisk fill. Also note that this mode is available only with integer type variables that contain non-negative values. The expression A-&I would be A-1 for I=1, while A&I would be A*** for I=-1. If the variable is zero, the zero is not included in the output string. Thus A&I would be A for I=0.

2.3.3. MULTIPLE VALUES

An asterisk may be used to indicate repeated values:

```
5*1.7, 3*ABCD, 2*-7

= 1.7,1.7,1.7,1.7,1.7,ABCD,ABCD,ABCD,-7,-7
```

The number to the left of the asterisk must be a non-negative integer. The word on the right may be of any type.

Implied Do-Loops

Two types of implied do-loops are permitted.

Colon mode: IST:LAST:INC or FIRST:FLAST:FINC

With the colon mode, the user specifies the starting value, the last value and the increment. Positive and negative values are permitted as long as the string makes sense. All values must be integer or they must all be real. The increment may be omitted. If omitted, the trailing colon must also be dropped and the increment defaults to 1 (or 1.). Note that the last value should be interpretted as an upper limit.

```
1:3 = 1,2,3   1:10:4 = 1,5,9   -1:-5:-3 = -1,-4
1.:3. = 1.,2.,3.      10.7:13. = 10.7,11.7,12.7
```

The colon mode may also be used with hollerith strings:

```
A:D,ABCD:ABCF,V1:V3,XX20:XX30:3,A:F:3
```

would yield:

```
A,B,C,D,ABCD,ABCE,ABCF,V1,V2,V3,XX20,XX23,XX26,XX29,A,D
```

Note that the increment defaults to 1, and that the increment must always be expressed as an integer. Also note that the hollerith portion of the last value must equal the hollerith portion of the starting value, and that the numeric portion must be of integer type.

Slash mode: IST/NUM/INC or FIRST/NUM/FINC

With the slash mode, the user specifies the starting value, the number of values and optionally the increment. The number of values must always be a non-negative integer. The starting value and increment must both be real or both be integer.

Super slash mode

The slash mode may be extended by adding an outer loop and an outer increment. This form is

```
IST/NUM/INC/LOOP/LINC
```

Basically, it causes the original loop to be issued LOOP times. The value of IST is incremented by LINC each time through the loop. Both LOOP and LINC default to 1.

```
1/3/1/5/6 = 1,2,3,7,8,9,13,14,15,19,20,21,25,26,27

10./5/10./2 = 10.,20.,30.,40.,50.,11.,21.,31.,41.,51.
```

Note that hollerith strings may be formed using either the slash or super slash mode. The approach is the same as for the colon mode, except that only the starting value may contain a hollerith field. The number of values, number of loops and both increments must be of integer type.

```
AA1/3/2/3/5 = AA1,AA3,AA5,AA6,AA8,AA10,AA11,AA13,AA15
```

2.3.4. INDIRECT MODE

The whole purpose of decoding a card image is to obtain the binary values that the characters represent. In many cases, these values are already present on DAL files. The indirect mode allows the user the capability of using a column of a matrix for input rather than typing the data in. The indirect mode is keyed by the presence of an equal sign in column 1. Ie,

```
= N1,EL1,VR1 ICOL

= N1,EL1,VR1,IC1,IC2,ICOL
```

the value of ICOL defaults to 1 if not supplied. Cycles of MASK,MASK may be used for IC1,IC2 if desired. Cycle numbers default to the current 'reading' cycles. Note that the entire column is read. Thus the program must be expecting the amount of data that is being input. If ICOL is input as zero, the entire matrix will be read, starting with column 1.

2.3.5. ERRORS

Error messages are written when errors are encountered. The following errors are possible:

1) Too much data -> you input more than was needed.
2) Invalid free-field format. The offending line is reprinted. This can be caused by starting a name with a digit, having inconsistent types in an implied do-loop, etc. Inspection of the line should reveal the problem.
3) Lib read error - this will occur if the element referenced does not exist, or exists with an error code.

2.4. HELP

HELP Command

Purpose

Provide online documentation for commands and processors.

Command

```
HELP   (topic (subtopic ...))
or
HELPC
or
HELPM (menu (submenu ...))
```

Where

```
HELP      - Get online documentation.

topic     - Is any main topic.  In general this is the command name.
subtopic  - Subtopics to the main topic.

HELPC     - List command pointers.


HELPM     - Display menus.

menu      - Same as topic except that the text is treated as a menu  and
            control is returned immediately to the command level.
submenu   - Same notes as for menu.
```

Main help display

The following is displayed when HELP is typed with out any arguments:

```
 Welcome to the VAPEPS online help facility.

- For documentation about a command, type the command. (e.g. TOC)

- For lists of available commands or general information,
  type in the appropriate code from the list below. (e.g. VT)

       Commands by Function                    General Information
 -----------------------------------    ----------------------------------
   Arithmetics, statistics.......AS       Introduction to VAPEPS........IN
   Advanced usage................AU       User guide (all user levels)..UG
   Element creation/manipulation.EC        Using HELP...................UH
   Element, file, system control.EF      ---------------------------------
   Graphics......................GR
   Input/Output..................IO
   Logical, sorting..............LS             New user Information
   All commands, runstreams......SUMMARY ---------------------------------
   Variable control..............VC        * To exit HELP, type QUIT
   Vibroacoustic database........VD        * To terminate VAPEPS execution
   Vibroacoustic prediction......VP          use the END command.
   Vibroacoustic tools...........VT        * Type UH to learn about HELP
------------------------------------     ----------------------------------
Help commands{..=>Quit  .=>Top level  ?=>Current info  ??=>subtopics} (see UH)
```

2.4.1. USING HELP

The following commands are available while in help:

QUIT - Exit help. If there is a subtopic called QUIT it will go to that and will not exit.

.. - Sure exit. This will take you out of help no matter where you are.

. - Go to top level of help. Takes you to the level with the prompt "help > ".

? - Redisplay current help section from the top.

?? - Displays the subtopics of the current section. When at the "help >" prompt it displays the main help section. This can be used as an argument as well. (eg. HELP ENTER ??)

SKIP - This works only when the help is stopped at the bottom of the screen. It has the same effect as ??.

Carriage return
- This will move you up one section or out of help if you are at the "help >" prompt. If you are in the middle of help output, it will give you the next screen of information.

2.5. MENU UTILITY

MENU Utility

Purpose

The menu utility provides summaries of commands by subject. This is a good way for a new user to find commands for a specific operation. The following is a list of the menus available. To turn menus on type MENU to turn them off type MENU OFF.

2.5.1. ARITHMETICS COMMANDS MENU

```
Arithmetics commands.           Menus:  AR AU EC EF GR IO LS ST VC VD VP VT
Often used commands:
ASMD      Term by term (+,-,*,/)        INVERT    Matrix inversion.
BUILD(X)  Term by term (+,-,*,/,..etc)  MULTIPLY  Matrix multiplication.
CALC      Calculator. (variables)       SEQUENCE  Sequence of real numbers.
FUN       Term by term (Sin,Cos,..etc)  TRAN      Matrix transposition.
Other useful commands:
FFT       MGRATE    NORM      RTOI      TURP
MDIFF     MTRAN     RUN=ROUN  TMUL
Type HELP command for more information.  (eg. HELP ASMD)
```

2.5.2. ADVANCED USAGE COMMANDS MENU

```
Advanced usage commands.        Menus:  AR AU EC EF GR IO LS ST VC VD VP VT

RUN=ADDC  CNFG      CSYM      RUN=GRLT  RUN=MEZ   RUN       RUN=TOCO
RUN=BCL   RUN=COPS  DPTOR     HELP      RUN=PGLS  RUN=SEND  RUN=UPDS
RUN=BOOT  CORE      EZOUT     RUN=INDE  RUN=PUTC  RUN=STAC  WTAPE
RUN=BSC   CPU       RUN=GCLS  RUN=INTE  RECOVER   SYMIN     XQT
CBIT      RUN=CPYS  RUN=GDOC  JPI       RLJUST    RUN=TCPY
CONTROL*  CRYPT     GLOSS     LOOPS*    RTABS     TDUMP
* General information.
Type HELP command for more information.  (eg. HELP CORE)
```

2.5.3. ELEMENT CREATION/MANIPULATION MENU

```
Element creation/manipulation.    Menus:  AR AU EC EF GR IO LS ST VC VD VP VT
Often used commands:
BUILD(X)  Combine matrices.             FORMAT   Information on formats.
CHOP      Create sub-matrices.          GROUP    Combine matrices.
ECOL      Input data.                   RANDOM   Random number generator.
Other useful commands:
GCOL      GREAD     RUN=LFU   MODIFY    RUN=RRAN SEQUENCE
Type HELP command for more information.  (eg. HELP CHOP)
```

2.5.4. ELEMENT, FILE AND SYSTEM CONTROL MENU

```
Element, file and system control. Menus: AR AU EC EF GR IO LS ST VC VD VP VT
Often used commands:
CHANGE    Change a DAL element name.    FIND     Find elements on a file.
COPY      Copy element to new element.  FNAME    Assign file to DAL unit.
COMMAND*  Typical command formats.      LAST     List last element on a file.
DISABLE   Disable DAL elements.         PACK     Remove disabled elements.
ENABLE    Enable DAL elements.          TOC      Table Of Contents of DAL file.
Other useful commands:
CYCLE     DAL*      EXPAND    FILES     LOCATE
* General information.
Type HELP command for more information.  (eg. HELP COPY)
```

2.5.5. GRAPHICS MENU

```
Graphics.                         Menus:  AR AU EC EF GR IO LS ST VC VD VP VT
Often used commands:
RUN=GENP  General plotting runstream.   RUN=PRSP  SEMOD plotting.
PLOT      Basic plotting command.       PSET      Set plotting options.
Other useful commands:
RUN=PGDB  RUN=PGRM  RUN=PCOMPARE
Type HELP command for more information.  (eg. HELP PLOT)
```

2.5.6. INPUT OUTPUT MENU

```
Input/Output.                    Menus:  AR AU EC EF GR IO LS ST VC VD VP VT
Often used commands:
ECOL      Input data.  (Free-field)      PRCRN     List corners of a matrix.
GREAD     Input data.  (Psuedo-format)   PRINT     Output DAL elements.
RUN=NEWS  List latest news item.         TABLE     Report quality tables.
Other useful commands:
CSYM      FORMAT*   GCOL      RUN=MANU   SYMIN     TDUMP     WRITE     WTAPE
* General information.
Type HELP command for more information.  (eg. HELP ECOL)
```

2.5.7. LOGIC AND SORTING MENU

```
Logic and sorting.               Menus:  AR AU EC EF GR IO LS ST VC VD VP VT
Often used commands:
ORDER     Sort contents of an element.   SEARCH    Search a DAL element.
Other useful commands:
AND       NOT       OR
Type HELP command for more information.  (eg. HELP ORDER)
```

2.5.8. STATISTICS COMMANDS MENU

```
Statistics commands.             Menus:  AR AU EC EF GR IO LS ST VC VD VP VT
Often used commands:
RUN=AVG   Find row or column average.  RANDOM    Random number generator.
RUN=CIMV  Confidence interval.         RUN=STAS  Normal & log-normal stats.
MAXMIN    Find maximums and minimums.  STAT      Normal statistics.
RUN=MVC   Mean Variance & Covariance.  RUN=TPVL  Theoretical percentile value.
Other useful commands.
RUN=GVAL  RUN=RRAN  RUN=TSTA
Type HELP command for more information.  (eg. HELP STAT)
```

2.5.9. VARIABLE CONTROL MENU

```
Variable control.                Menus:  AR AU EC EF GR IO LS ST VC VD VP VT
Often used commands:
CALC      Calculator.                 SET       Set variables.
Control   Transparent commands.       VAR       Remove variables.
LIST      List variables.
Other useful commands:
DEFAULT   RECOVER   RUN
Type HELP command for more information.  (eg. HELP STAT)
```

2.5.10. VIBROACOUSTIC DATABASE MENU

```
Vibroacoustic database.          Menus:  AR AU EC EF GR IO LS ST VC VD VP VT
Often used commands:
RUN=DATA   Get averaged data.           RUN=GMOD   Get parameters and data.
DICTIONARY Direct Dictionary interface. RUN=PADMIN Prepare PREP for Dictionary.
ENTER      Enter data.                  RUN=PICK   Data from specific channels.
RUN=GLST   List parameters. (RUN=GMOD)  PREP       Prep data.
Other useful commands:
ADMIN     RUN=SAVE  RUN=SPIL
Type HELP command for more information.  (eg. HELP PREP)
```

2.5.11. VIBROACOUSTIC PREDICTION MENU

```
Vibroacoustic prediction.        Menus:  AR AU EC EF GR IO LS ST VC VD VP VT
 _____________________The following menus are available._______________
   EXTRAP I...................VPX1      Statistical Energy MODeler..VPSM
   EXTRAP II..................VPX2
 ______________________________________________________________________
 - To obtain a menu listed type the code. (eg. VPX1 for  EXTRAP I.)
```

2.5.11.1. EXTRAP I PREDICTION MENU

```
EXTRAP I prediction.            Menus:  VPX1 VPX2 VPSM
Often used commands:
CNVRT     Specify units.              PRDICT    EXTRAP I prediction.
DAMO      Input parameters.           PREDICT   EXTRAP I, II and Theoretical.
DENS      Modal density.              TRNF      Transfer functions.
RUN=EQPL  Equivalent plate properties.
Type HELP command for more information.  (eg. HELP PRDICT)
```

2.5.11.2. EXTRAP II PREDICTION MENU

```
EXTRAP II prediction.           Menus:  VPX1 VPX2 VPSM
Often used commands:
DAMO      Input parameters.           PREDICT   EXTRAP I, II and Theoretical.
RUN=EQPL  Equivalent plate properties.  SCALE     EXTRAP II prediction.
Type HELP command for more information.  (eg. HELP SCALE)
```

2.5.11.3. SEA MODELER MENU

```
Statistical Energy MODeler.     Menus:  VPX1 VPX2 VPSM
Often used commands:
ATACALC   Coupling loss factors.      MDENS     Modal density.
ATACO     Coefficient matrix.         PREDICT   EXTRAP I, II and Theoretical.
CFAC      Specify units.              SEMOD     Statistical Energy MODeler.
RUN=EQPL  Equivalent plate properties.  TPRD      Theoretical prediction.
Other useful commands:
RUN=CLFGET          RUN=GETRESP         RUN=SAVG
RUN=CLFPUT          RUN=PRST            RUN=SHIFT
Type HELP command for more information.  (eg. HELP SEMOD)
```

2.5.12. VIBROACOUSTIC TOOLS MENU

```
Vibroacoustic tools.               Menus:  AR AU EC EF GR IO LS ST VC VD VP VT
Often used commands:
CONVERT   Convert units.                 RUN=NOML  Normalize data.
FREQUENCY Generate frequencies.          RUN=OCT3  Narrow band to 1/3 octave.
RUN=GRMS  Generate specs.                RUN=OVER  Calculate overall levels.
Other useful commands:
RUN=CIMV      RUN=LRMS      RUN=STAS      RUN=TPVL
RUN=GPSD      RUN=LTRP      SHOCK         RUN=VFIX
Type HELP command for more information.  (eg. HELP CONVERT)
```

3. GENERAL COMMANDS

This chapter contains a listing of all general commands. See the discussion on command format in chapter 2 for more information on the command structure. The following should give you an idea where to look if you can't find a command in this chapter.

Chapter 2 contains the HELP and MENU commands.

Chapter 4 contains the printing and plotting commands.

Chapter 5 contains the ENTER command.

Chapter 6 contains the PREP command.

Chapter 7 contains the SERCH command.

Chapter 8 contains the ADMIN command.

Chapter 9 contains the DICTIONARY command.

Chapter 10 contains the SEMOD command used for performing vibroacoustic predictions.

Chapter 11 contains an early version of vibroacoustic prediction commands described in Chapter 10; the commands given in Chapter 10 are now recommended.

Chapter 12 contains the transparent ## commands.

Chapter 13 contains the variable and symbol manipulation commands.

Chapter 14 contains the runstream commands.

3.1. RUN ADDC

RUN=ADDC Programmer's Command

Purpose

This command is used to add new commands to the SETCOM subroutine.

Command

```
   RUN=ADDC  OLD NEW  NIN NOUT

   Where
      OLD  = Version name of existing ABV, CALL, and EPS matrices.
      NEW  = Version name for new elements created here.
      NIN  = Unit number for input (default = 1)
      NOUT = Unit number for output (default = 2)
```

Example

Add the command ZAP to a setcom routine on DAL unit 1.

```
      RUN=ADDC VAP VAP 1 2
      YES                     -- Lists existing commands.
      1                       -- Number of commands to add.
      ZAP                     -- Name of command to add.
      YES                     -- Use first three characters for command name.
      YES                     -- Decode command.
```

Related Commands

RUN=ADNC, RUN=BCL, RUN=BSC, RUN=GCLS, RUN=SUBC

3.2. RUN ADNC

RUN=ADNC Programmer's Command

Purpose

This command will add a new style command and options to the command options file specified.

Command

```
RUN=ADNC N1(,NEL1 (N2(,NEL2))
or
RUN=ADNC N1,NEL1 'LIST'
or
RUN=ADNC N1,NEL1 'SPILL' NU
```

Where

```
N1,NEL1         - Input data set as follows.  NEL1 defaults to 'COM$'.
N1,NEL1,'NAM$'  - Contains the 12 character command names.
N1,NEL1,'NUM$'  - Contains the 3 character command name pointers.  Same
                  as 1st 3 characters of old style commands. There is
                  one entry here for each entry in NAM$.
N1,NEL1,'OPT$'  - Contains the 12 character option names.
N1,NEL1,'OPN$'  - Contains two columns as follows:
                  Column 1:  Contains the 3 character command name pointer
                             of the corresponding option in OPT$.
                  Column 2:  Contains the 1 character option to be passed
                             to the command.

N2,NEL2         - Output DAL unit and element name for new data set.
                  Default is N1,NEL1.

'LIST'          - Displays the data sets in an organized manner.

'SPILL'         - Creates a symbolic fortran file which when ##read will
                 recreate the dal data sets.
NU              - Is the fortran unit number to direct spill output to.
```

Example

Make the new command ENABLE be equivalent to ABLD.

```
RUN=ADNC 30,COM$
ENABLE
ABL,D
```

```
-blank line-
-blank line-
```

Notes

The above data sets are both input and output. If they do not exist a new set is created. The commands and options are sorted alphabetically.

Related Commands

RUN=ADDC, RUN=BCL, RUN=BSC, RUN=GCLS, RUN=SUBC

3.3. RUN AJNV

RUN=AJNV Command

Purpose

This command takes an input matrix and an approximate inverse and produces a better inverse.

Command

```
RUN=AJNV  N1,EL1,VR1  N2,EL2,VR2  N3,EL3,VR3  IPNT

Where
   N1,EL1,VR1 - Original matrix.
   N2,EL2,VR2 - Approximate inverse.
   N3,EL3,VR3 - Better inverse.
   IPNT       - Quality check flag.
              = 0 - No quality check done.
              > 0 - RUN=ORTH is executed to show the quality of the
                      original inverse and the new inverse.
                Default=0
```

Example

Invert then adjust the inverse of a matrix of random numbers and check the quality of the inversion..

```
RANDOM    28,A,A  50,50
INVERT    28,A,A  28,B,B
RUN=AJNV  28,A,A  28,B,B  28,C,C  1
```

Related Commands

INVERT, RUN=ORTH

3.4. RUN ALL

RUN=ALL Command

Purpose

Prints out the header and channel information section for all sections of all events on the specified DAL file.

Command

```
RUN=ALL NU (IOUT)
```

Where

```
NU   - DAL file to get events from.
IOUT - Optional fortran output file number.  Default = 6.
```

Example

Print out the header and channel information for all sections of all events on DAL unit 1 and direct the output to Fortran unit 2.

```
RUN=ALL 1 2
```

Related Commands

ENTER, RUN=ESAV, RUN=HEAD, PREP

3.5. AND

AND Command

Purpose

Reduce two lists into one list that contains all items that are present in both lists.

Command

```
AND  N1,EL1,VR1  N2,EL2,VR2  N3,EL3,VR3
```

Where

```
N1,EL1,VR1 = Input vector
N2,EL2,VR2 = Input vector
N3,EL3,VR3 = Output vector of common items.

M$ = Output variable set to indicate number of items in output list.
```

Example

Find the union of the two sets of integers (1,3,4) and (3,5,12).

```
ECOL 28,A,A 3,1
1,3,4
ECOL 28,B,B 4,1
3,5,10,12
AND 28,A,A 28,B,B 28,C,C
LIST M$
PRINT 28,C,C
```

Notes

The input vectors may be integer, real or hollerith, but both input vectors must be of the same type. Input vectors may contain repeated values. The output vector will contain each common item only once.

Related Commands

NOT, OR, ORDER

3.6. ASMD

ASMD Command

Purpose

Perform term by term addition, subtraction, multiplication, division of two matrices to form a third.

Command

```
ASMD    ITYP  N1,EL1,VR1  N2,EL2,VR2  N3,EL3,VR3
or
ASMDC   ITYP  N1,EL1,VR1  N2,EL2,VR2  N3,EL3,VR3
or
ASMD    ITYP  N1,EL1,VR1  N2,EL2,VR2  FAC
```

Where

```
ITYP       - 1 for add, 2 for sub, 3 for mul, 4 for div
N1,EL1,VR1 - Input DAL element.
N2,EL2,VR2 - Output DAL element (same size as EL1)
N3,EL3,VR3 - Optional second input element.
FAC        - Constant factor to be used in place of the second
             input element.
```

Example

Add the two vectors (1,2,3,4) and (3,8,4,9) and multiply the result by 3.

```
ECOL 28,A,A 4,1
1,2,3,4
ECOL 28,B,B 4,1
3,8,4,9
ASMD 1 28,A,A 28,C,C 28,B,B
ASMD 3 28,C,C 28,D,D 3
```

Description

The output matrix (N2,EL2,VR2) is always the same size as the first matrix (N1,EL1,VR1). Ie: Every input term is operated on and then output in the same position. The following sections outline the procedure. Note that the operator (N3,EL3,VR3) may be a matrix, a vector or a scalar. Assume the first element is A(NR,NC), the second is B(NR,NC), and the third is C(MR,MC). Also let ITYP be one (indicating addition). Then

```
IF(MR.EQ.NR.AND.MC.EQ.NC) B(I,J) = A(I,J) + C(I,J)
IF(MC.EQ.1.AND.MR.EQ.NR) B(I,J) = A(I,J) + C(I,1)
IF(MC.EQ.1.AND.MR.EQ.NC) B(I,J) = A(I,J) + C(J,1)
If MC not equal to one of the above, error ....
```

If A is square (NR=NC), and C is a vector (MR=NR=NC,MC=1), then both of the last two conditions hold. In this case, the first rule will be applied (ie: C is swept across A to produce B). If the second form is desired (ie: Transpose C and sweep it down A to get B), use the command ASMDC.

ASMD may also be used to operate on a matrix with a scalar. In this case the third element is not present and FAC is used.

Where the operation

```
B(I,J) = A(I,J) + FAC
```

is performed.

Notes

In the above, the plus is replaced by the other operations for ITYP other than 1.

ASMD will operate on single precision, double precision, complex or integer matrices. Both input matrices must be of the same type. In the case of complex, the scalar form may consist of a real and an imaginary value, or just a real.

Related Commands

BUILD, CALC, FUN

3.7. RUN AVG

RUN=AVG Command

Purpose

Compute the average of a group of data by rows or columns.

Command

```
RUN=AVG  N1,EL1,VR1  N2,EL2,VR2  ITYP
```

Where

```
N1,EL1,VR1 = Input array (NR,NC)
N2,EL2,VR2 = Output vector of averaged numbers.
             (NR,1) for ITYP='ROW' (NC,1) for ITYP='COL'
ITYP       = Character string indicating whether to average the rows
             ('ROW') or the columns ('COL') ITYP defaults to 'COL'
```

Example

Find the average of the following vector (1,3,4,5,4,6,5)

```
ECOL 28,A,A 7,1
1,3,4,5,4,6,5
RUN=AVG 28,A,A 28,B,B COL
PRINT 28,B,B
```

Related Commands

RUN=CMIV, RUN=STAS, STAT, RUN=TPVL

3.8. RUN BCL

RUN=BCL Programmer's Command

Purpose

This runstream builds a COMM/LIST element for saving the command list of a DAL processor.

Command

```
RUN=BCL  N1,IL1,IV1  NU
```

Where

```
N1,IL1,IV1 = Input command matrix (NR,3) or (NR,4)
NU         = Output Fortran unit
```

Example

Build a COMM/LIST element from the input matrix 1,BCL,DAT and direct the output to Fortran unit 1.

```
RUN=BCL  1,BCL,DAT 1
```

Related Commands

RUN=ADDC, RUN=ADNC, RUN=BSC, RUN=GCLS, RUN=SUBC

3.9. RUN BEAM

RUN=BEAM Command

Purpose

Calculate moments of inertia, centroidal distance and radius of gyration for a given beam built up of rectangular sections.

Command

```
RUN=BEAM N1,NVR1 (N2,NVR2)
```

Where

N1,NVR1 - DAL unit and version name for output. If N1 is negative and old set of parameters is read in from -N1,NVR1. The parameter values for the old model can be changed by stepping through the input sections and making changes as you go.

N2,NVR2 - Optional DAL unit and version name for output if N1 is negative. Default is N2,NVR2 = -N1,NVR1.

Notes

There are two sections of input.

1) General parameters. The following parameters need to be input:

```
NBL   = ---- - Number of rectangular sections in the beam.
RLEN  = ---- - Length of beam.
ER    = ---- - Reference Young's modulus. This is also used as the
               default for the E in the following section of input.
RHOR  = ---- - Reference mass density. This is also used as the
               default for RHO in the following section of input.
```

2) For each layer or section.

```
H    = ____ - Thickness of layer.
CENX = ____ - Distance from X reference to centroid of this layer.
CENY = ____ - Distance from Y reference to centroid of this layer.
E    = ____ - Young's modulus for this layer. Default = ER.
RHO  = ____ - Mass density for this layer. Default = RHOR.
W    = ____ - Width of this layer.
```

Output

There following DAL elements are output:

N1,'GENR',NVR1 - Input general parameters RLEN, ER, RHOR. (3 x 1)

N1,'GENI',NVR1 - Input general parameter NBL. (1 x 1)

N1,'BEML',NVR1 - Input parameters H, CENX, CENY, E, RHO, W for each beam layer. (NBL x 6)

N1,'BEAM',NVR1 - Output calculated values IX, IY, CENX, CENY, RGFX, RGFY, J, A, RGFJ.

The output parameters are:

IX - Moment of inertia about X axis.

IY - Moment of inertia about Y axis.

CENX - Distance from X reference to centroid.

CENY - Distance from Y reference to centroid.

RGFX - Radius of gyration about the centroid in X direction.

RGFY - Radius of gyration about the centroid in Y direction.

J - Polar moment of inertia. (IX + IY)

A - Cross section area.

RGFJ - Polar radius of gyration.

3.10. RUN BOOT

RUN=BOOT Programmer's Command

Purpose

Create a VAPEPS boot tape for the named system.

Command

```
RUN=BOOT  SYSTEM (OPT1(,OPT2(,...(,OPTN))))
```

Where

```
   SYSTEM - Computer system to generate tape for.
          = VAX     - All of the DEC VAX family of machines.
          = CDC     - CDC machines.  Very site dependant.
          = UNIVAC  - 1100 series of UNISYS (aka: SPERRY (aka: UNIVAC))
                      machines.
          = CRAY    - CRAY 1-S and CRAY XMP machines running COS.
          = MC500   - All MASSCOMP UNIX based machines.
          = SILIG   - Silicon Graphics Unix based machines.
          = SP7000  - Unisys Unix system.  Was Sperry 7000.
          = UNICOS  - Cray Unix operating system.
          = UNIX    - General UNIX based machines.  May experience some
                      problems depending on machine and UNIX version.
OPT1,OPT2,...,OPTN
          - One or more of the following options:
          = TAPE     - Creates Boot TAPE.  This assumes that the BOOT
                       files are on DAL002.  When this option is not
                       specified with the EXTRACT option no tape is
                       generated.
          = EASY     - This option includes the EASY symbolics on the
                       transmittal tape.  When TAPE is specified the EASY
                       deck must reside on DAL002.
          = EXTRACT  - This extracts the symbolics for the specified
                       system and places them on DAL002.  If EASY is
                       specified then the EASY symbolics are placed in an
                       element on DAL002 as well.
          = DISSPLA   - Includes symbolics for linking with the DISSPLA
                       plotting software.  This is only done when the site
                       requesting the tape has DISSPLA at their site
                       already.  EXTRACT must be specified with this
                       option.
          = NOPRIM   - This will cause primitive plotting software. To not
                       be included.  This is the default when DISSPLA is
                       specified.  EXTRACT must be specified with this
```

```
                    option.
     = NOPLOT  - This will cause no plotting software to be
                 included.  EXTRACT must be specified with this
                 option.
     = CSYM    - This uses the CSYM command to create symbolic VAX
                 files of the elements on DAL002.  This can be used
                 to create files which are to be trans- mitted over
                 a direct connection between a VAX and the machine
                 to which to symbolics are to be transmitted.
```

Notes

A) This stream only works from a VAX or MASSCOMP machine.
B) The file BOOTEASY.DAL must be assigned to DAL003
   The file EASY.DAL must be assigned to DAL004
   The file VAPEASY.DAL must be assigned to DAL005
C) The file GENERL.DAL must be assigned to DAL010
D) When booting to CDC or UNIVAC, this streams writes to a tape, identified as logical name TAPE04.
E) When booting to another VAX, this stream produces several files with version .TRN . These may be $COPYed to a labeled tape.
F) In each case, this stream produces a file BOOT.INS that contains instructions on how to use the tape on the destination machine.

Example

The following is a VAX runstream that may be used to create a BOOT tape for the CDC. The procedure for UNIVAC, MC500, and UNIX machines is similar.

```
$ASSIGN ?:[?]BOOTEASY DAL003
$ASSIGN ?:[?]EASY     DAL004
$ASSIGN ?:[?]VAPEASY  DAL005
$ASSIGN ?:[?]GENERL   DAL010
$VAPEPS
RUN=BOOT  CDC EXTRACT
END
$MOUNT MTA0:/FOREIGN      ! non VAX machines.
$ASSIGN MTA0:  TAPE04     ! non VAX machines.
$VAPEPS
RUN=BOOT  CDC TAPE
END
$DISMOUNT MTA0:
```

For a VAX machine the following runstream can be used:

```
$ASSIGN ?:[?]BOOTEASY DAL003
$ASSIGN ?:[?]EASY     DAL004
$ASSIGN ?:[?]VAPEASY  DAL005
```

```
$ASSIGN ?:[?]GENERL   DAL010
$VAPEPS
RUN=BOOT  VAX EXTRACT CSYM
END
$ALL MTA0:                  ! VAX
$INIT MTA0:    VAPEPS       ! VAX
$MOUNT MTA0: VAPEPS         ! VAX
$COPY *.TRN  MTA0:*.COM     ! VAX
$DEALL    MTA0:             ! VAX
```

Note: If you don't have BOOTEASY.DAL, EASY.DAL, VAPEASY.DAL or GENERL.DAL, you cannot generate a boot tape to another system.

The default boot does not include a copy of the master symbolics (EASY). If for some reason you want to send this to another site, SET ISYM=1 prior to issuing RUN=BOOT.

Related Commands

CSYM, EZOUT, RUN=SPILL, WTAPE

3.11. RUN BSC

RUN=BSC Programmer's Command

Purpose

Creates a SETCOM routine for VAPEPS.

Command

```
RUN=BSC  N1,EL1,VR1  N2,EL2,VR2 NAM1,NAM2 NOUT
```

Where

```
N1,EL1,VR1 = Input list of command abbreviations (N,1) (A3) or (N,2)
             where column two contains the decode option for this
             command (0==>decode, else don't)
N2,EL2,VR2 = Input list of command subroutines (N,2) (A3,A3)
NAM1,NAM2  = Program name (default =  DAL PRO)
NOUT       = Output Fortran unit  (output may require editing)
```

Example

Create a SETCOM routine from the input lists 1,ABV,VAP and 1,EPS,VAP and set the program name to VAP5.0 and direct the output to Fortran unit 2.

```
RUN=BSC  1,ABV,VAP  1,EPS,VAP  VAP '5.0' 2
```

Related Commands

RUN=ADDC, RUN=ADNC, RUN=BCL, RUN=GCLS, RUN=SUBC

3.12. BUILD

BUILD Command

Purpose

Create a new element by combining portions of other elements and/or adding constants to various portions of the element.

Command

```
BUILD  NO,ELO,VRO  NR,NC  ITYP
```

Where

```
NO,ELO,VRO - DAL unit, element and version name for new element.
NR,NC      - Number of rows and columns in new element.
ITYP       - (Optional) type of matrix.
           = 0 - Integer
           = 1 - Real  (default)
           = 2 - Double precision
           = 3 - Complex
           = 4 - Hollerith
```

Description

BUILD is actually a sub processor. It is entered with the VAPEPS BUILD command. This command contains information concerning the name and size of the element to be created. When the command is read, the new element is created as a zero array.

Sub-processor cards are then read, indicating portions of other elements and/or constants which are to be added to the new element.

The sub processor returns to VAPEPS when the DONE command entered.

Space

After the BUILD command has been issued, you have essentially dimensioned a matrix B(NR,NC) and filled it with zeros. You may now enter sub-commands to modify it.

Sub commands

Each sub-command initiates a dcuble do-loop that modifies the contents of the BUILD buffer. The modification may either be replacing the current value or adding something to it. The new data may be input directly as a scalar constant or indirectly from an existing DAL element.

Scalar type

```
FILL  IST,IFN  JST,JFN  ILC,JLC  LOOP
CONSTANT

ADDC  IST,IFN  JST,JFN  ILC,JLC  LOOP
CONSTANT

MULC  IST,IFN  JST,JFN  ILC,JLC  LOOP
CONSTANT

DIVC  IST,IFN  JST,JFN  ILC,JLC  LOOP
CONSTANT
```

Where

```
IST,IFN - Starting and ending rows to put constant.
          Defaults to the row dimension of the matrix.
JST,JFN - Starting and ending columns to put constant.
          Defaults to the column dimension of the matrix.
ILC,JLC - Amount to step for next output row and column respectively.
          Defaults to 1.
```

Fortran Equivalent

```
         DO 10 J = JST,JFN,JLC
         DO 10 I = IST,IFN,ILC
         B(I,J) = CONSTANT [FILL]
         B(I,J) = B(I,J) + CONSTANT [ for ADDC ]
         B(I,J) = B(I,J) * CONSTANT [ for MULC ]
         B(I,J) = B(I,J) / CONSTANT [ for DIVC]
    10   CONTINUE
```

Note

LOOP has not been included above, since it would only serve to complicate matters. Novice users should leave it off the command, or use a value of 1. LOOP is discussed under advanced usage.

Existing DAL element types

```
MIX   N1,EL1,VR1 I1,I2 J1,J2 IST,JST IC,JC ILC,JLC LOOP

PLUS  N1,EL1,VR1 I1,I2 J1,J2 IST,JST IC,JC ILC,JLC LOOP

MINUS N1,EL1,VR1 I1,I2 J1,J2 IST,JST IC,JC ILC,JLC LOOP

MUL   N1,EL1,VR1 I1,I2 J1,J2 IST,JST IC,JC ILC,JLC LOOP

DIV   N1,EL1,VR1 I1,I2 J1,J2 IST,JST IC,JC ILC,JLC LOOP
```

```
ADDX  N1,EL1,VR1 I1,I2 J1,J2 IST,JST IC,JC ILC,JLC LOOP
CONSTANT
```

Where

N1,EL1,VR1 - Name of an existing element. It has dimensions, so lets call it E(MR,MC)

I1,I2,IC - Indicate the row range and increment to use from N1,EL1,VR1.

J1,J2,JC - Indicate the column range and increment to use from NL1,EL1,VR1

Eg: Use rows I1 through I2 in increments of IC, and columns J1 through J2 in increments of JC.

IST,JST - Starting location of the output. Ie: The first term modified is B(IST,JST) [modified with E(I1,J1)]. This is where the upper left hand corner of the input matrix is to be placed in the output matrix.

ILC,JLC - Output row and column increment

CONSTANT - Constant value input by user for ADDX only.

Fortran Equivalent

```
      JJ=JST
      DO 30 J = J1,J2,JC
      II=IST
      DO 20 I=I1,I2,IC
      B(II,JJ)=E(I,J)               [MIX]
      B(II,JJ)=B(II,JJ) + E(I,J)    [PLUS]
      B(II,JJ)=B(II,JJ) - E(I,J)    [MINUS]
      B(II,JJ)=B(II,JJ) * E(I,J)    [MUL]
      B(II,JJ)=B(II,JJ) / E(I,J)    [DIV]
      B(II,JJ)=B(II,JJ) + E(I,J)*CONSTANT   [ADDX]
      II=II+ILC
 20   CONTINUE
      JJ=JJ+JLC
 30   CONTINUE
```

Notes

The user can specify any number of sub-commands during one BUILD. If a mistake is made, you can zero the matrix by

```
FILL
0
```

since I1,I2,... default to the whole matrix.

When you are through, issue the command:

```
DONE
```

This returns you to VAPEPS.

The division commands DIV and DIVC check for zero denominators. If one is found, the result is set to zero.

Cycles

Note that the transparent cycle command ##CYCL may be used in between sub-commands. This enables the creation of an array composed of input elements with varying cycle numbers.

Advanced Notes

This section discusses LOOP and also the command IAFJ. The discussion is merely an approximation of the FORTRAN code. Note that increments of zero are legal.

```
IAFJ IIAFJ,IOAFJ,IRAFJ
```

Means 'I' as a function of 'J'. If you look in the FORTRAN example above, 'II' is always reset to 'IST' at the start of each column loop. The purpose of IAFJ is to allow the starting row locations to vary from column to column, and to allow the number of row items considered to change from column to column.

IIAFJ - Increment of the starting input row as a function of the column loop.
IOAFJ - Increment of the starting output row as a function of the column loop.
IRAFJ - Increment of the number of terms in the row loop as a function of the column loop. IRAFJ defaults to zero. A value of 1 is for creating upper triangular matrices, -1 for lower.

IIAFJ, IOAFJ, and IRAFJ default to zero. Once IAFJ is used, the values remain as set until they are changed by another IAFJ command.

Fortran Equivalent

```
IR=I2 - I1 + 1
IF(IC.NE.0) IR=IR/IC
JR=J2-J1+1
IF(JC.NE.0)JR=JR/JC
JIN=J1
JOUT=JST
```

```
      DO 30 J=1,JR
      IIN=I1+(J-1)*IIAFJ-1
      IOUT=IST+(J-1)*IOAFJ-1
      DO 20 I=1,IR
      DO 15 L=1,LOOP
      B(IOUT+L,JOUT)=E(IIN+L,JIN)*CONSTANT [+B(IOUT+L,JOUT)]
15    CONTINUE
      IIN=IIN+IC
      IOUT=IOUT+ILC
20    CONTINUE
      JIN=JIN+JC
      JOUT=JOUT+JLC
      IR=IR+IRAFJ
30    CONTINUE
```

The use of IOAFJ and LOOP have similar consequences for the scalar type subcommand. The value of IIAFJ must be zero for the scalar case.

In the above sub commands, not all specifications must be given. If a specification is not given, it will be replaced by its default value. It is very important to include specs in their proper order. Specs may not be skipped. A zero will not yield the spec's default value.

Quantity	Default
I1	1
I2	MR
J1	1
J2	MC
IST	1
JST	1
IC	1
JC	1
ILC	1
JLC	1
IFN	NR
JFN	NC
IIAFJ	0
IOAFJ	0
IRAFJ	0
LOOP	1

BUILDX

If the command BUILDX is used instead of BUILD, the matrix NO,ELO,VRO is assumed to already exist with size NR,NC. In this case the matrix is not zeroed out, and all sub commands act to modify the original values in NO,ELO,VRO.

Example

Build a matrix of 2 columns from the two vectors (1, 5, 9, 10) and (8, 7, 3, 5).

```
ECOL 28,A,A 4,1
1,5,9,10
ECOL 28,B,B 4,1
8,7,3,5
BUILD 28,COMB,INE 4,2 0
MIX 28,A,A
MIX 28,B,B 1,4 1,1 1,2
DONE
```

Related Commands

ASMD, CALC, FUN, GROUP

3.13. RUN BWTH

RUN=BWTH Command (Obsolete Use FREQUENCY/BAND_WIDTH)

Purpose

This runstream is used to extract the standard 1/3 octave band- widths for the standard center frequencies in the range FLO to FHI. FLO and FHI need not be standard 1/3 octave center frequencies. However, an error message will be generated if no standard center frequencies lie between FLO and FHI.

Command

```
RUN=BWTH N,NM,IVN FLO,FHI
```

Where

```
N,NM,IVN = Name of output vector to put 1/3 octave band widths.
FLO,FHI  = The 1/3 octave center frequency range for which band-
           widths are to be calculated.
```

3.14. CALCULATOR

CALC Processor

Purpose

CALC is a sub processor which solves elementary algebraic equations. It is very powerful in that it allows the user to define and operate on a set of variables which may be used in subsequent VAPEPS command strings.

Description

The processor is entered when the command CALC is encountered. Equations are decoded and solved until a blank card or the DONE subcommand is read. Control is then returned to VAPEPS.

Equations

All equations must contain a variable name followed by an equal sign and one or more operands separated by operators. Allowable operators are +,-,*,/ and **(add,sub,mult,div,exp). **EQUATIONS ARE ALWAYS EVALUATED FROM LEFT TO RIGHT WITH NO REGARD TO THE HIERARCHY RULES OF FORTRAN.** In general, equations will take the following form:

```
NPTS=TLAST-TFRST/ DELT + 1.999
```

Where the difference TLAST-TFRST is divided by DELT and the result is added to 1.999 before being truncated to integer and stored in NPTS.

Constants

Both real and integer constants may be used to the right of the equal signs. Real constants must contain a decimal point and may have an exponent. Embedded blanks are not permitted. Integers beginning with a zero are assumed to be octal. Hollerith constants (4 chars max) may be entered by enclosing them in single quotes.

Named variables

Variable names may contain up to four alphanumeric characters. The first character determines the variable type(A-H,O-Z = real, I-N = integer) and must be alphabetic. These variables may be referenced in any future VAPEPS command by prefacing the variable name with an ampersand(&).

Indirect strings

Indirect strings are used to retrieve individual values from existing DAL elements. These strings can appear on the left or on the right hand side of the equation. They must be enclosed in parentheses and must contain five or seven words, separated by commas or blanks. The form is

```
(N1,EL1,VR1,IROW,ICOL)
or
(N1,EL1,VR1,IC1,IC2,IROW,ICOL)
```

where N1,EL1,VR1 identify the element, and IROW, ICOL pinpoint the value inside the element. IC1,IC2 indicate the cycles of the element. If omitted, they default to IC1,IC2 from the last CYCLE command. Note that any element referenced in an indirect string must already exist. If named variables are to be used inside an indirect string, they must be preceeded by the ampersand. ANY ITEM TAKEN FROM A DATASET IS ASSUMED TO HAVE THE SAME TYPE AS INDICATED IN THE TOC FOR THAT DATASET. Eg if you pull a value from a real array, it will be treated as a real value. If a dataset contains mixed types, then ##SET may be used to pull off values without doing any conversion.

Another form of the indirect string may be used to pull information from the table of contents of a file. In this case there must be either 4 or 6 words inside the parentheses.

```
(N1,EL1,VR1,ITEM)
or
(N1,EL1,VR1,IC1,IC2,ITEM)
```

where N1,EL1,VR1 is the element whose TOC line is to be accessed and ITEM is the item number. IC1 and IC2 may be included as cycle numbers. If they are omitted(ie: First case), the values presently set via CYCLE command for reading elements will be used. ITEM may have values from 0 to 12, and refer to the following quantities:

ITEM	Quantity
0	element number
1	sector address
2	creation date
3	creation time
4	error code
5	number of words
6	NJ
-6	NR
7	NI*NJ (or NR)
-7	NC
8	type code
9	element name
10	version name
11	first cycle number
12	second cycle number

Note that the number of columns may be calculated as

```
NC = (N1,EL1,VR1 5)/(N1,EL1,VR1 7)
or
NC = (N1,EL1,VR1 -7)
```

Indirect strings of this latter type are assumed to be of integer/hollerith type.

Functions

The following functions are currently available in CALC:

```
<IABS>,<ABS>,<COS>,<SIN>,<TAN>,<EXP>,<LOG>,<L10>,<E10>,<ASIN>,<ACOS>
<ATAN>,<JDAT>
```

They operate on the current value of the expression evaluated from left to right. As an example, the sine of the absolute value of the angle theta (in degrees) may be calculated as:

```
SINT = THTA<ABS>*PI$/180.<SIN>
```

ABS will work on either integer or real numbers without altering the type, but the others will convert integer input to real before proceeding. Note that <LOG> computes the natural log, and <EXP> uses base e. Ie:

```
X = X<LOG><EXP>
```

will return the original X. Common LOGS may be performed using <L10>.

Examples

1. Start time = .1 seconds, delta = .00375 seconds. Between what two points is TIME = .286
2. Find accelerations of station ten at time = .286, If time histories are in column 10 of 'BPLI','CASE' on DAL unit one, and rows correspond to the time description in example 1.
3. Array starts at sector address 32, find address at start of fifth record if records are 1024 words long

```
CALC
* EXAMPLE 1.
TIME = .286
TST = .1
N1 = TIME-TST/.00375 + 1
N2 = N1 + 1
*THE DESIRED POINTS ARE N1 AND N2
* EXAMPLE 2.
A1 = (1,BPLI,CASE,&N1,10)
```

```
ADIF = (1,BPLI,CASE1,&N2,10) - A1
* ADIF IS THE DIFFERENCE IN ACCELERATIONS BETWEEN N1 AND N2
T1 = N1-1*.00375+TST
ACC = TIME - T1/.00375 * ADIF + A1
*ACC IS THE DESIRED ACCELERATION
*EXAMPLE 3.
LENG = 1024-1/28 + 1
*LENG IS THE LENGTH OF EACH RECORD IN SECTORS (UNIVAC SECTORS)
IPOS = 5-1*LENG+12
```

Related Commands

ASMD, BUILD, FUN

3.15. CBIT

CBIT Command

Purpose

Check each column of a 'bit' matrix to find all rows that have a given set of bits. Save pointers to all columns that have at least one match.

Command

```
CBIT  N1,EL1,VR1  N2,EL2,VR2  N3,EL3,VR3
```

Where

```
N1,EL1,VR1 - Input bit configuration to check. If EL1=0, then two
             elements are read.
             ELT = 'CPNT' - list of columns to search.
             ELT = 'BMAP' - normal input bit configuration. (above).
N2,EL2,VR2 - Input bit matrix to be checked.
N3,EL3,VR3 - Output list of columns that have at least one match.  If
             EL3=0, then two elements are output.
             ELT = 'CPNT' - list of matching columns.
             ELT = 'BMAP' - Sequence of bits for each matching column.
                            Set bits indicate which rows matched.
```

Related Commands

SSTRNG

3.16. CEAL

CEAL Command

Purpose

Copy EAL elements to DAL files and visa versa. This command is system dependent and should only be used when DAL/EAL files are based on different sector lengths. The command was written for the PRIME system, but may be used on other systems as needed.

Command

```
CEAL  N1,ELN,VER  N2  IOP,LRAT
```

Where

```
N1,ELN,VER - Input EAL or DAL element.
N2         - Output EAL or DAL unit.
IOP        - option flag
           = 1   --- input is EAL element
                     output is DAL element
           = -1  --- input is DAL element
                     output is EAL element
LRAT       = ratio of EAL sector length to DAL sector length.
             default = 16  (512/32)
```

Optional form

```
CEALS N1,NS1,NS2  N2  IOP,LRAT
```

Where

```
N1         - Input EAL or DAL unit.
NS1,NS2    - sequence number range for input elements. NS2 may be input
             as a large number to transfer all datasets from NS1 on.
N2         - Output EAL or DAL unit.
IOP        - option flag
           = 1   --- input is EAL elements
                      output is DAL elements
           = -1  --- input is DAL elements
                      output is EAL elements
LRAT       - ratio of EAL sector length to DAL sector length. default =
             16  (512/32)
```

Description

In general, DAL files and EAL files are compatible in that they both use the DAL data management system to create data sets on user files. On some systems they may be incompatible, however, due to the use of different parameters in setting up the file structure.

On the PRIME system, EAL files use a sector length of 512 words. This is compared to the DAL sector length of 32 words. Because of this difference, EAL files may not be referenced directly in VAPEPS and DAL files may not be referenced directly in EAL (SPAR). This command may be used to transfer elements one at a time from one file type to the other.

Notes

CEAL can only modify an EAL file, not create one. Thus you must use EAL (or EALPRO if it exists) to create an EAL file before you try to CEAL anything into it.

CEAL may only be used when the EAL sector length is an integer multiple of the DAL sector length. If the sector lengths are equal, this command is not necessary. If you are on a system other than PRIME, and notice an incompatibility between DAL and SPAR or EAL, first check the following table for probable sector lengths on each system. The easiest way to tell is to examine the RR value in the TOC for the first element on file.

RR	Sector length
2	512
3	256
4	128
5	112
7	64
14	32
17	28

LRAT may then be computed as the ratio of EAL to DAL. If EAL is smaller than DAL, or the lengths are not integer multiples, please contact Jim Schafer at the address given in the VAPEPS section.

Digression

The reason for the different sector lengths is that systems operate more efficiently if you write in multiples of their preferred sector size. Some of these sizes are quite large (512 for prime). DAL data sets force each column to start on new sector boundary. Eg 3x3 matrix actually occupies 3 full sectors. Use of 512 would waste a lot of disk space, so VAPEPS 'software' sectors the files at a fraction of the preferred size, and actually performs writes at the larger size. One disadvantage of this approach is that several disk accesses may be required to read in a given data set (eg a 3x3 matrix may lie across a large sector boundary). Thus as many as three reads/writes may be required to satisfy a single read/write request. In order to minimize boundary crossing, VAPEPS has recently been changed to start all datasets at a preferred sector boundary. Thus the numbers in the table may be off slightly. (On VAX, preferred. size = 128, reduced size = 32. Normal dataset starting point = 14 = 3.5 full sectors -- this is rounded up to 4 full sectors, or 16 small sectors)

PRIME usage

EAL to DAL

1) execute VAPEPS

2) Use FNAME to set EAL unit (FNAME 1,L001)

3) CEAL 1,ELN,VER 2,1,16

DAL to EAL

1) execute VAPEPS

2) Use FNAME to set EAL unit (FNAME 1,L001)

3) CEAL 2,ELN,VER 1,-1,16

Note that the 16 may be omitted on both commands, since LRAT defaults to 16.

Related Commands

FNAME, TRDAL

3.17. CHANGE

CHANGE Command

Purpose

Change the name of an element or change the type or NJ value without modifying or restructuring the element.

Command

```
CHANGE  N1,EL1,VR1  (N1),EL2,VR2
or
CHANGE/TYPE  N1,EL1,VR1  ITYP
or
CHANGE/NJ  N1,EL1,VR1  NJ
```

Where

```
   N1,EL1,VR1 - Name of existing element.
   EL1,VR2    - New name of element. (DAL unit can't be changed.)
   ITYP       - New type of element
   NJ         - New NJ value for element
```

Example

Change the element 1,TRAN,DAT to 1,ZAP,DAT.

```
CHANGE 1,TRAN,DAT ZAP,DAT
```

Notes

The full element name includes cycle numbers which are not included in the command. The full input name uses input cycles from the last CYCLE command. The full output name uses cycles from the last CYCLE command.

When you change the type of an element using CHANGE, you are only changing the type flag. Thus it is not legal to change double precision to single precision with this command. Please see DPTOR and RTOI commands. CHANGE will let you get away with almost anything, so be careful. A common use for CHANGE/TYPE is to convert complex to single precision. Here the type is changed, and the number of rows is changed. The data is not actually changed, but it doesn't have to be. Another example is for a mixed type matrix. In this case you might want one type for one operation, and another type for a different operation.

The use of NJ and the need to change it is fairly involved. If you need to, just remember that CHANGE will do it. NJ is explained in the EXPAND command but it really isn't used very often in VAPEPS. The finite element program SPAR uses NJ a great deal and NJ is explained in the SPAR manual.

An alternate method of changing the TOC values is described in the command LOC/LOCH. In most cases, it is preferable to use the CHANGE command.

Related Commands

COPY, FIND, LOCATE, TOC, TTYP

3.18. RUN CHKO

RUN=CHKOUT Programmer's command

Purpose

This command is used to check out a data set for modification. The error code for that data set is set to the users name.

Command

```
RUN=CHKOUT  NU,NEL,NVR  IC1,IC2  TYPE
```

Where

```
NU,NEL,NVR - DAL unit element and version of file being checked out.
IC1,IC2    - Cycle numbers of file being checked out.
TYPE       - Type name of file being checked out.
```

Example

Checkout the file 30,RUN,CHKO 0,0.

```
CHKOUT 30,RUN,CHKO 0,0 DRS
```

Related Commands

CSYM, RUN=ERROR, RUN=EZDB, RUN=TRMT

3.19. CHL

CHL Command

Purpose

CHOLESKI decomposition of a real, positive definite, symmetric matrix. In addition, CHL may be used to invert the decomposed matrix. This latter feature is useful in obtaining the inverse of symmetric, positive definite matrices.

Command

```
CHL  N1,EL1,VR1  N2,EL2,VR2  IOP
```

Where

```
N1,EL1,VR1 - Input symmetric matrix. Only the upper triangular
             terms are used.
N2,EL2,VR2 - Output upper or lower triangular matrix
IOP        - Option flag
           = 0 -  Standard decomposition (upper triangular matrix)
           = 1 -  The above matrix inverted and transposed (lower)
           = 2 -  Matrix 0 inverted, ie. matrix 1 transposed (upper)
```

Description

Given a real, positive definite, symmetric matrix, S, CHOLESKI decomposition finds an upper triangular matrix, U, such that the transpose of U, times U, will equal S. For IOP=0, the matrix U is output.

```
CHL  1 SYM MAT  1 CHL UPP  0
TMUL 1 CHL UPP  1 CHL UPP  1 NEW SYM
```

would produce a NEW/SYM that equals SYM/MAT.

For IOP>0, the decomposed matrix U, is inverted. Since U is upper triangular, considerable savings are realized if this inverse is done in two parts. The first step is the computation of the transpose of U, which is a lower triangular matrix. This lower triangular matrix is output for IOP=1. This lower triangular matrix is useful in computing the inverse of the original matrix S.

```
CHL  1 SYM MAT  1 LOW INV  1
TMUL  1 LOW INV  1 LOW INV  1 SYM INV
```

will produce the inverse as SYM/INV.

For eigenvalue problems, the mass matrix must be decomposed and applied to both sides of the stiffness matrix, in order to retain the symmetry of the eigenvalue problem. In this case, computation is simplified if the output decomposed inverse is in upper triangular form, ie the transpose of the output for IOP = 1. IOP=2 will produce this matrix.

```
CHL  1,MASS,MAT  1,MASS,CHL  2
TMUL 1,STIFF,MAT 1,MASS,CHL 28,TMP,MAT
TMUL  28,TMP,MAT  1,MASS,CHL  28 GEN K
```

Would produce a symmetric matrix GEN/K that is suitable for a symmetric eigensolver such as the EIG command in DALPRO. Once the modes are computed, they must be backtransformed as

```
TMUL 1 MASS CHL  1 EIG MODES  1 REAL MODES
```

Since CHOLESKI decomposition is most often used as a part of an eigensolution, the default value of IOP is 2. ie, IOP may be omitted from the command, in which case an upper triangular inverse matrix will be output.

CHOLESKI may be used with either single precision or double precision matrices. If your matrix is symmetric and positive definite, it is much cheaper to use CHL to invert than the INVERT command. Also, CHL will work for any size matrix, whereas the INVERT command is only good for matrices that will fit in core.

Related Commands

TMUL, INVERT, MULT

3.20. CHOP

CHOP Command

Purpose

Create a new element by selecting rows and cols from an existing element.

Command

```
CHOP  N1,EL1,VR1  N2,EL2,VR2  NEWR,NEWC
-row numbers-
-column numbers-
```

Where

```
N1,EL1,VR1 - Input matrix, size = NR,NC
N2,EL2,VR2 - Output matrix, new size = NEWR,NEWC
NEWR       - Number of rows in the new element.
NEWC       - Number of columns in the new element.
-row numbers-
           - Row numbers from the input element to be placed in the
             output element.
-column numbers-
           - Column numbers from the input element to be placed in the
             output element.
```

Example

Given the vector (1,2,3,4,...,100) chop out rows 50-75.

```
ECOL 28,A,A 100,1
1:100
CHOP 28,A,A 28,B,B 25,1
51:75
PRINT 28,B,B
```

Description

CHOP is used to create a new matrix by selecting out various rows and columns from an existing matrix. The user specifies the number of rows and columns on the command line. The command then prompts the user to enter the list of row numbers to b selected, followed by the list of columns to be selected. In many cases the user wants all of the rows, or all of the columns. CHOP has been designed such that if NEWR=NR, all rows will be selected automatically, without specifying the input data list that normally follows the command line. Similarly, if NEWC=NC, all columns will be taken, without reading a list.

Other forms

In some cases, the automatic selection feature of CHOP can get in the way. This is especially true if you want to rearrange rows or columns. Take the case where NEWR=NR, but you want to specify a list of numbers anyway. The command form CHOPX turns off the automatic selection and requires user input for both row and column lists irregardless of the size of NEWR and NEWC. (The only exception is when NEWR and/or NEWC are input as negative--below)

Notes

The input lists of row/column numbers may consist of integer or floating point values. These values must be greater than or equal to zero, and less than or equal to the number of rows/columns of the input matrix. Zero values result in zero filled rows or columns. Note that these lists are read in DALPRO free-field format. Thus

```
1,2,3,4,5,5*0,20,18,16,14,12
1:5,0,0,0,0,0,20/5/-2
= 1 ROW LIST
```

are all legitimate forms of input.

When using CHOP, NEWR must be less than or equal to NR, and NEWC must be less than or equal to NC. In CHOPX, this restriction is removed. Thus CHOPX must be used in the case of enlarging a matrix. An enlarged matrix can only consist of a combination of the rows and columns contained by the input matrix.

Both command forms allow NEWR and NEWC to be negative numbers. This implies that the first -NEWR rows (or first -NEWC columns) are to be selected automatically, without user list input.

Related Commands

BUILD, GROUP

3.20.1. OPTIONAL FORMS OF CHOP

CHOPM Command

Purpose

Modifies an existing matrix. The output matrix is modified by replacing rows and columns with the specified rows and columns in the input matrix. If a zero is input as a row or column number then the corresponding row or column in the output matrix is not changed.

CHOPJ Command

Purpose

Uses the input matrix as an (NI, NJ, NC) array. The number of rows refer to the number of NJ items to be kept, and all NI values are taken. This can be thought of as choping a three dimensional array. (e.g. A CHOPJ on a SPAR VIBR/MODE dataset sould take all 6 deflections for the input joint numbers)

CHOPK Command

The CHOPK command is similar to the CHOPJ command but you don't need to keep all NI items. Instead, you add an arguement to the command line and specify the particular items right after the row and column inputs have been specified. For the VIBR/MODE example above, you could select only the translational terms by including NI as 3 and specify the line 1,2,3 following the row/column inputs, if any. When the K option is used, the list of item numbers must ALWAYS be present, even if NI is input as negative, and even if the X option is used as well.

CHOPY Command

Purpose

Allows the selection of rows to vary from column to column.

Command

```
CHOPY  N1,NEL1,NVR1  N2,NEL2,NVR2  N3,NEL3,NVR3  N4,NEL4,NVR4
or
CHOPY  N1,NEL1,NVR1  N2,NEL2,NVR2 N3,NEL3,NVR3 -NEWC
or
CHOPY  N1,NEL1,NVR1  N2,NEL2,NVR2 -NEWR N4,NEL4,NVR4
```

Where

N1,NEL1,NVR1 - Input dataset, same as for CHOP.

N2,NEL2,NVR2 - Output dataset, same as for CHOP.

N3,NEL3,NVR3 - A matrix containing lists of row numbers from N1,NEL1,NVR1 to be placed in N2,NEL2,NVR2. Each row in N3,NEL3,NVR3 points to a row in N1,NEL1,NVR1. Each column corresponds to columns in the output matrix.

N4,NEL4,NVR4 - A vector containing a list of column numbers from N1,NEL1,NVR1 which will be placed in N2,NEL2,NVR2. Each row corresponds to each column in N2,NEL2,NVR2 and N3,NEL3,NVR3.

NEWR, NEWC - Number of rows or columns. Will take the first NEWR rows or the first NEWC columns.

3.21. RUN CIMV

RUN=CIMV Command

PURPOSE:

This runstream calculates the confidence interval for a true mean of x given a particular confidence level. 'x' is a set of measured vibro-acoustic data (E.G. Mean square acceleration (g**2,g**2/Hz) or mean square sound pressure (psi**2, psi**2/Hz))

Command

```
RUN=CIMV N1,IEL1,IVR1  NU1,IVS1  CONF ITYP (IOUT)
```

Where

```
N1,IEL1,IVR1       - Input vector of mean square data.
NU1,IVS1           - DAL unit and version of output.
CONF               - Confidence level. (e.g. .95)
ITYP               - Type of distribution
                   = 0 - Normal distribution
                   = 1 - Log-normal distribution
                     (Default = 1)
IOUT               - Fortran unit for printed output - use 0 for no output.
                     (Default = 6 - terminal screen)
NU1,'CIMV',IVS1 - Contains three columns as follows:
                    Column 1 - Lower confidence limit.
                    Column 2 - Upper confidence limit.
                    Column 3 - Confidence interval.
```

Example

Find the confidence interval for the following vector (1.,2.,3.,...,20.).

```
ECOL 28,A,A 20,1
1.:20.
RUN=CIMV 28,A,A 28,TEST .95 0
PRINT 28,CIMV,TEST
```

3.22. RUN CLFG

RUN=CLFGET Command

Purpose

This command is uses to get a set of ata's for a particular path from an ata matrix and put them in an output matrix.

Command

```
RUN=CLFGET N1,NVR1 N2,NEL2,NVR2
```

Where

```
N1,NVR1      - The DAL unit number and version name of the model  from
               which to extract the ata's.
N2,NEL2,NVR2 - The DAL unit element and version of the matrix in
               which to put the extracted ata's.
```

Example

Get the damping loss factor associated with the element SKIN in the model TEST. The ata matrix for model TEST is assumed to exist.

```
RUN=CLFGET  1,TEST 28,DLF,SKIN
SKIN,SKIN
```

Note

There are two ata's associated with each path so both ata's will be put into N2,NEL2,NVR2. If you specified a Damping Loss Factor (DLF) to be extracted then N2,NEL2,NVR2 will have only 1 column. The order that you specify for a path is the order that the ata's will be extracted.

Related Commands

SEMOD, RUN=CLFPUT

3.23. RUN CLFP

RUN=CLFPUT Command

Purpose

This command is used to put a set of ata's for a particular path into an ata matrix.

Command

```
RUN=CLFPUT N1,IVR1 N2,IEL2,IVR2
```

Where

```
N1,IVR1        - The DAL unit number and version name for the
                 model.
N2,IEL2,IVR2 - The dal unit, element and version of the ata's
                 to be placed in the ata matrix.
```

Example

Put the coupling loss factors in the matrix 28,CLF,NEW into the path EXTA,SKIN in model 1,TEST. Model 1,TEST and 28,CLF,NEW are assumed to exist.

```
RUN=CLFPUT 1,TEST 28,CLF,NEW
EXTA,SKIN
```

Note

There are two ata's associated with each path so both ata's must be in N2,IEL2,IVR2. If you are replacing a DLF then N2,IEL2,IVR2 has only one column. The element order that you specify for a path is for the ata's in the first column. The second column contains the ata's for the reverse order.

Related Commands

SEMOD, RUN=CLFGET

3.24. CMOD

CMOD Command

Purpose

Find the modulus or modulus and phase of a complex array. CMOD may also be used to convert modulus and phase to a complex array. Note that this process is identical to getting the length and angle of a vector in the x-y plane. A variation of the command will go from polar to rectangular coordinates or visa versa.

Command

```
CMOD   N1,EL1,VR1  N2,EL2,VR2
or
CMODP  N1,EL1,VR1  N2,EL2,VR2
```

Where

```
N1,EL1,VR1 - Input complex or real array
N2,EL2,VR2 - Output real or complex array
```

Example

Convert the complex matrix 28,COMP,LEX to a real/imaginary matrix.

```
CMODR  28,COMP,LEX 28,REAL,MAT
```

Description

If the first matrix is complex (type 3), the output matrix will be real (type 1), and will contain the modulus of the complex matrix. If the command CMODP was used, the output matrix will also contain the phase in radians. If the input dimensions are (NR,NC) the output dimensions will be (NR,NC) or (NR,2*NC). Note that a complex array of size (NR,NC) actually has NR*2 rows, which alternate real,imaginary,real,imaginary,...

If the input matrix is real, it must contain both the modulus and phase. These must be stored as alternate columns (modulus,phase,modulus,phase,...). Thus the real matrix will have an even number of columns (eg (NR,NC*2)). The output element will be an (NR,NC) complex matrix. Note that the conversion of modulus/phase to complex is allowed only with the CMODP form of the command.

Additional forms

CMOD has several special cases, as follows:

```
CMODR    (COMPLEX)     (REAL,IMAGINARY)
or
CMODR    (REAL,IMAGINARY)  (COMPLEX)

CMODPR   (POLAR)       (RECTANGULAR)
or
CMODRP   (RECTANGULAR) (POLAR)
```

Where

```
(COMPLEX)      - Refers to a complex matrix.
(REAL,IMAGINARY)  - Refers to a real matrix with an even number of columns.
                 (e.g. REAL,IMAGINARY,REAL,IMAGINARY,...)
(POLAR)        - Refers to a real matrix of polar coordinates This may
                 be (R,THETA,Z) OR (R,THETA) (i.e. 3 columns or two
                 columns)
(RECTANGULAR) - Refers to a real matrix of rectangular coordinates. This
                 may be (X,Y) or (X,Y,Z).
```

Related Commands

DPTOR, RTOI

3.25. RUN CMPF

RUN=CMPF Command

Purpose

Compares the TOC lines of two DAL files. The TOC's are merged, then sorted alphabetically according to name. The result is printed to FORTRAN unit NOUT.

Command

```
RUN=CMPF  N1,N2  NOUT
```

Where

```
N1,N2 - DAL unit numbers of TOC's to be merged.
NOUT  - Fortran unit to direct output. Default = 6.
```

Example

Compare the TOC's of DAL units 1 and 2.

```
RUN=CMPF 1,2
```

Notes

If N1 < 0 then file names,dates,times are compared and only datasets with a different TOC line are kept in the list

If N2 < 0 then file names,sizes are compared and only datasets with a different TOC line are kept in the list

Related Commands

FIND, LAST, RUN=OTOC, TOC, RUN=TOCO

3.26. CNFG

CNFG Command

Purpose

CNFG creates and/or edits a configuration tree for the VAPEPS system. It is included as a stand alone command to aid users in creating general configuration trees which apply to many events. This general tree may then be modified within the CONF portion of PREP as necessary for a particular event.

Since the configuration tree is described in detail in the users manual, only a brief summary of the command will be included here.

Command

```
CNFG  N1,VR1  N2,VR2
```

Where

```
N1,VR1 - Input configuration (unit and name). If N1,VR1 are omitted,
         the tree will be built from scratch.
N2,VR2 - Output configuration. This will be written unless the OMIT
         command is used to terminate CNFG.
```

Once inside CNFG (or PREP CONF) the user is prompted for subcommands. These serve to create, edit or list the configuration tree.

Related Commands

PREP

3.26.1. INPUT

INPUT Command

Purpose

Causes CNFG to enter input mode. User input is of the form:

```
LTGSME> LAST  THIS  GENERIC  SPECIFIC  MOUNT  ENCLOSURE
```

Where

```
LAST       - Local name of parent node. A dollar sign is used for the
             trunk (main node) since it has no parent.
THIS       - Local name of this node. All nodes in a tree must have a
             unique local name. The term local name is used to imply
             that these names may not be used in later searching.
GENERIC    - Generic name for this node. (12 characters max)
             Examples:  SHUTTLE  PALLET  CARGO-BAY
SPECIFIC   - Specific name for this node. (12 chars) This name serves
             to subdivide the generic class into a number of
             subclasses. Examples: COLUMBIA 6-METER  CONF-43A
MOUNT      - Local name of a node which serves as a support structure
             for the current node.
ENCLOSURE  - Local name of a node which serves as an enclosing
             structure to the current node.
```

Note

Each node is implicitly linked to all nodes between itself and the trunk. The MOUNT and ENCLOSURE fields allow the user to add additional linkages to this node.

The input mode is terminated when less than three items are input on a given line.

3.26.2. READ

READ Command

Purpose

Reads the specified configuration tree and places the new nodes at the end of the tree. Any previous nodes having the same local node name will be disabled.

Command

```
READ   NU,VER
```

Where

NU,VER - DAL unit and version name of existing configuration tree.

3.26.3. DISABLE

DISABLE Command

Purpose

Disables nodes in the configuration tree.

Command

```
DISABLE  I,J
```

Where

I,J - Node numbers in the tree to disable.

Example

Disable nodes 1,3.

```
DISABLE 1,3
```

3.26.4. ENABLE

ENABLE Command

Purpose

Enables previously disabled nodes in the configuration tree.

Command

```
ENABLE   I,J
```

Where

I,J - Node numbers in the tree to enable.

Example

Enable nodes 1,3.

```
ENABLE 1,3
```

Note

Note that enabling nodes may cause other nodes to be disabled.

3.26.5. PACK

PACK Command

Purpose

Removes any disabled nodes. Nodes are renumbered accordingly.

Command

```
PACK
```

3.26.6. CHECK

CHECK Command

Purpose

Performs a consistency check and writes any applicable diagnostics.

Command

```
CHECK
```

3.26.7. LIST

LIST Command

Purpose

List the current configuration tree.

Command

```
LIST  (I,J)
```

Where

I,J - Node number to list. Default is all node numbers.

Example

List node numbers 1 through 3.

```
LIST 1,3
```

3.26.8. DONE

DONE Command

Purpose

Performs PACK, CHECK and then saves the tree. A blank line is equivalent to DONE.

Command

```
DONE
```

3.26.9. OMIT

OMIT Command

Purpose

Terminate CNFG without writing the tree.

Command

```
OMIT
```

3.27. RUN COG

RUN=COG Command

Purpose

Computes the coordinates of the center of gravity and the inertia tensor about the cg.

Command

```
RUN=COG  N1,EL1,VR1  N2,EL2,VR2  N3,EL3,VR3  N4,VR4
```

Where

```
N1,EL1,VR1 - Input DOF table (joints,6)
N2,EL2,VR2 - Input station matrix (joints,3)
N3,EL3,VR3 - Input mass matrix or vector (DOF,DOF) or (DOF,1)

N4,'COOR',VR4 - Output coordinates (3,2)
N4,'RBMA',VR4 - Output inertia tensor (6,6)
```

Note

Two columns are produced for the output coordinates. If your nodes have the same translational mass in X,Y, and Z then both columns will be the same.

If the two columns come out different, then at least one node has different masses associated with the three translational directions.

Related Commands

RUN=JTOD

3.28. CONVERT

CONVERT command

Purpose

This command is used to convert between various sets of units.

Command

```
CONVERT/option N1,NEL1,NVR1 N2,NEL2,NVR2 INTYP,OUTYP (F1,F2)
or
CONVERT/USER N1,NEL1,NVR1 N2,NEL2,NVR2 INTYP,OUTYP PL,PM,(PT (ISHFL,F1,F2))
or
CONVERT/COLUMNS N1,NEL1,NVR1 N2,NEL2,NVR2 N3,NEL3,NVR3 (F1,F2)
```

Where

```
option is:
SOUND        - Sound Pressure Level data.
ACCELERATION - Acceleration data.
USER         - User specified conversion.
COLUMNS      - Convert each column independently depending on the flags
               in N3,NEL3,NVR3.

N1,NEL1,NVR1 - Input matrix to be converted.
N2,NEL2,NVR2 - Output matrix to put converted data.
INTYP,OUTYP  - Input and output unit types from the following lists:
```

SOUND option

Flag	Units
1	dB
2	(N/M**2)**2
3	(DYNES/CM**2)**2
4	(LBF/FT**2)**2
5	(LBF/IN**2)**2
6	(N/M**2)**2/HZ
7	(DYNES/CM)**2)**2/HZ
8	(LBF/FT**2)**2/HZ
9	(LBF/IN**2)**2/HZ
10	N/M**2
11	DYNES/CM**2
12	LBF/FT**2
13	LBF/IN**2

ACCELERATION option

Flag	Units
1	dB
2	G**2
3	(M/S**2)**2
4	(CM/S**2)**2
5	(FT/S**2)**2
6	(IN/S**2)**2
7	G**2/HZ
8	(M/S**2)**2/HZ
9	(CM/S**2)**2/HZ
10	(FT/S**2)**2/HZ
11	(IN/S**2)**2/HZ
12	G
13	M/S**2
14	CM/S**2
15	FT/S**2
16	IN/S**2

USER option

Flag	Units
1	M, KG, S
2	CM, G, S
3	FT, SLUG(LBF-SEC**2/FT), S
4	IN, SNAIL (LBF-SEC**2/IN), S

Special units

5	dB
6	Gees**2 or Gees**2/Hz (see ISHFL below)

PM - Power to take M to.

PL - Power to take L to.

PT - Power to take T to.

ISHFL - Flag for special handling.

Flag	Function
0	Nothing extra done.
1	Multiply input by bandwidth before converting.
2	Divide output by bandwidth after converting.
3	Square input before converting.
4	Take square root of output after converting.
5	Do both 1 and 4.
6	Do both 2 and 3.

F1 - Lower frequency range (if required).

F2 - Upper frequency range (if required).

N3,NEL3,NVR3 - Input matrix of flags from SOUND or ACCELERATION options. Positive numbers indicate SOUND option. Negative numbers indicate ACCELERATION option.

Column one - Input unit flag one for each column in N1,NEL1,NVR1.

Column two - Output unit flag one for each column in N2,NEL2,NVR2.

Comments

Units of vibration are referenced to 8.4144E-18 G**2. Units of pressure are referenced to 8.4144E-18 Psi**2 (2.E-5 N/M**2).

Related Commands

FREQUENCY, PREDICT, SEMOD

3.28.1. EXAMPLES

Example 1 - conversion from dB to Psi**2.

```
* Try the following sequence of commands:
ECOL 28,A,A 31,1
31*100.
CONVERT/SOUND 28,A,A 28,B,B 1,5
PRINT 28,B,B
```

Example 2 - Conversion from G**2 to G**2/Hz

```
ECOL 28,A,A 31,1
31*1.
CONVERT/ACCELERATION 28,A,A 28,B,B 2,7 10.,10000.
PRINT 28,B,B
```

Example 3 - Conversion from ft**3 to in**3.

```
ECOL 28,A,A 1,1
1.
CONVERT/USER 28,A,A 28,B,B 3,4 3.,0.,0.
PRINT 28,B,B
```

Example 4 - Conversion from SLUG/IN**3 to SNAIL/IN**3

```
ECOL 28,A,A 1,1
1.
CONVERT/USER 28,A,A 28,B,B 3,4 -3.,1.,0.
PRINT 28,B,B
```

Example 5 - Conversion from Psi to N/M**2 (SNAIL/(IN*S**2) to KG/(M*S**2))

```
ECOL 28,A,A 1,1
1.
CONVERT/USER 28,A,A 28,B,B 4,1 -1.,1.,-2.
or
CONVERT/SOUND 28,A,A 28,B,B 13,10
PRINT 28,B,B
```

3.29. RUN CONV

RUN=CONVERT Command (Obsolete - use CONVERT command)

Purpose

Does conversions from one set of units to another.

Command

```
RUN=CONVERT  N1,NEL1,NVR1  N2,NEL2,NVR2 NIN,NOUT (F1,F2)
```

Where

N1 NEL1 NVR1 - Input matrix to be converted.
N2 NEL2 NVR2 - Output matrix of converted data.
NIN NOUT - Flag referring to the input and output unit types.
F1 F2 - Frequency range to use when required.

The following conversions are supported:

FLAG	units
1	DB
2	PSI**2
3	G**2
4	(N/M**2)**2
5	PSI**2/HZ
6	G**2/HZ
7	(N/M**2)**2/HZ

Related Commands

CONVERT

3.30. RUN COPS

RUN=COPS Command

Purpose

Writes all of the symbolic (types 5 and 6) elements from DAL unit 'NDAL' to Fortran unit 'NSYM'.

Command

```
RUN=COPS  NDAL,NSYM,IOP
```

Where

```
NDAL - Is the DAL unit from which to get symbolic elements.
NSYM - Is the Fortran unit to direct output.
IOP  - Option flag.
     = 0 - All symbolic element will be written.
     > 0 - Only symbolic elements in 28,TOC,LIST 0,0 will be used.
           28,TOC,LIST can be formed by the following:
           CYCLE 0 0
           FINES &NU
           28,TOC,LIST
```

Example

Copy all symbolic elements from DAL unit 1 to Fortran unit 1.

```
RUN=COPS 1,1
```

Output format

The general output format is as follows:

```
SYMEND  &NU  'NAME'  'NAMV'  'IC1'  'IC2'  ZZZZ
-THE DATA-
SEOF
```

Related Commands

COPY, RUN=COPY, RUN=CPYS, CSYM, PRINT, RUN=SEND, RUN=SNDP, RUN=SPIN, RUN=UPDS

3.31. COPY

COPY Command

Purpose

Copy an element from one file to another. COPY may also be used in a 'mask' mode, where a group of elements sharing a common name or cycle may be copied. COPYF may be used to copy an entire DAL file into a DAL element on another file.

Command

```
COPY  N1,EL1,VR1  N2,EL2,VR2
```

Where

```
N1,EL1,VR1 - Existing element.
N2,EL2,VR2 - New element.
```

Example

Copy 1,ZAP,DAT to DAL unit 2 using the same name.

```
COPY 1,ZAP,DAT 2
```

Notes

Times and dates of creation are updated for the new element. Disabled elements may not be copied. Cycle names are derived from current input and output cycles as defined by the CYCLE command.

MASK mode

The command COPYM may be used instead of COPY to transfer a group of elements.

```
CYCLE  1,MASK  2,MASK
COPYM  1,MASK,DATA  3,MASK,NEWV
```

would transfer all elements from unit 1 that have a version name of DATA and a first cycle of 1 to unit 3, with different version names of NEWV and new cycle 1 = 2.

Related Commands

RUN=COPY, RUN=COPS, RUN=CPYS, PACK, TDUMP, TRDAL, WTAPE

3.31.1. COPYF

COPYF (COPY File)

Purpose

Copy file to/from DAL dataset. Copy file to/from tape.

Command

```
COPYFO  NU  N1,EL1,VR1
or
COPYFO  NU  NTAP,IFIL  LENG
or
COPYFI  NU  N1,EL1,VR1
or
COPYFI  NU  NTAP,IFIL  LENG,LTOT
```

Where

```
NU          - Is the DAL unit to be copied.
N1,EL1,VR1 - DAL unit NU is stored in this element as a type 9
              element.

NTAP        - Tape unit number.
IFIL        - File number on tape.
LENG        - For COPYFI:
              Maximum number of words per tape block ( if omitted, LENG
              defaults to 1000) If in doubt, specify a large number for
              LENG, eg 5000
            - For COPYFO:
              LENG is still the length of a tape block, but this length
              will be ROUNDED to an integer number of host sectors. If
              the file originally came from another machine (ie Cray)
              then you should set LENG such that it is a multiple of the
              original sector size. (512 cray words = 1024 Vax words, so
              LENG=1024 or a multiple thereof.)
LTOT        - Number of sectors in the file.
              If NU is a true DAL file, both LENG and LTOT should be
              omitted.
```

DAL file to/from DAL element

COPYFI will create N1,EL1,VR1 and COPYFO will recreate DAL unit NU. DAL unit NU must be an empty file prior to the COPYFI command.

DAL file to/from Tape.

As an example, consider writing a VAX resident Cray DAL file to tape. In this case, the file is a DAL file, but not a VAX DAL file. So, assume the file is OUTPUT.CDL and that if you do a $DIR/SIZE OUTPUT.CDL it shows up with 45123 blocks. Then 45123 blocks is 45123*128 words or 45123*4 sectors of 32 words each. (45123*4 = 180492)

```
$ DEFINE TAPE04 MTA0:
$ MOUNT  TAPE04/FOREIGN
$ DALPRO
  FNAME  1,'OUTPUT.CDL'
  COPYFO 1  4,1  1024,180492
  END
```

would copy the file to tape. You could then use TDUMP and TRDAL on the Univac to read the tape.

3.32. RUN COPY

RUN=COPY Command

Purpose

This runstream may be used to copy groups of elements from one DAL unit to another, with different cycle numbers, if desired.

Command

```
RUN=COPY  N1,EL1,VR1  C1,C2  N2  C3,C4
```

Where

```
N1,EL1,VR1 - DAL unit, element name and version of the element(s) to
             be copied.
C1,C2      - Cycle numbers of the element(s) to be copied.
N2         - DAL unit to which to copy the element(s).
C3,C4      - Desired output cycle numbers.
```

Example

Copy all element from DAL unit 1 to DAL unit 2.

```
RUN=COPY 1,MASK,MASK MASK,MASK 2 0 0
```

Note

The word 'MASK' may be used for any of the arguments except N1 and N2 as a match all. It is recommended that actual numbers be used for C3 and C4.

Related Commands

COPY, PACK

3.33. CORE

CORE Command

Purpose

Allows user to reset maximum data space. CORE also may be used to release unused core after core has been expanded.

Command

```
CORE MAXDSP
```

Where

MAXDSP - Maximum data space in words.

Example

Set maximum core size to 65000 words.

```
CORE 65000
```

Notes

VAPEPS uses one large array for the storage of data in core. In addition, on some systems, heap storage is used for arrays too large to fit in the allocated storage area. In general, the size of this array may be increased or decreased dynamically. When VAPEPS is executed, two parameters are initialized for this area. The first is the actual size of the area, the second is the maximum size for the area. The maximum defaults to the largest size a user should be using, considering the environment at the time (demand runs have a lower core limit than batch,etc).

If the user has permission to exceed demand guidelines, the CORE command may be used to adjust the maximum size to normal batch limits. Note that a maximum size varies from system to system.

The use of CORE will never increase your core allocation. It merely allows future commands to allocate more core if the need arises. It will, however, physically decrease your core if the maximum specified is less than the current physical size.

Related Commands

FNAME

3.34. CPU

CPU Command

Purpose

Print CPU (or SUP on UNISYS) time that has accumulated during various portions of the run.

Command

```
CPU
```

Example

Find out how long it takes, in CPU seconds, to invert a 100 x 100 matrix.

```
RANDOM 28,TEST,MAT 100,100
CPU
INVERT 28,TEST,MAT 28,INV,MAT
CPU
```

Notes

When the CPU command is encountered, three numbers will be written to unit 6. The first indicates the number of CPU seconds that have been used since the last call to CPU. The second indicates the number of CPUs during this execution. The third indicates the number of CPUs during the entire run.

Related Commands

END

3.35. RUN CPYS

RUN=CPYS Command

Purpose

Copy all matching symbolic element from a DAL unit and create a file of card images, with embedded 'SYMEND' cards. This file may then be written to tape and later re-entered on a DAL file by adding (##READ) it to VAPEPS.

Command

```
RUN=CPYS  NFOR,NDAL,ELN,VER,IC1,IC2  IDAT
```

Where

```
NFOR  - Fortran unit to direct output.
NDAL,ELN,VER,IC1,IC2
      - DAL unit, element, version and cycle numbers of elements  to
        match.  DAL unit must be specified.  Element, version and cycle
        numbers can be MASKed.
IDAT  - Date before which no items will be copied.   Default = 0
```

Example

Copy all symbolic elements from DAL unit 27 to Fortran unit 3.

```
RUN=CPYS 3,27
```

Note

If NDAL is entered as a negative number, the element 28,ELN,VER is assumed to contain the result of a transposed FINES operation. IC1, IC2 and IDAT are ignored. Elements are read from DAL unit -NDAL.

Related Commands

COPY, RUN=COPY, RUN=COPS, CSYM, PRINT, RUN=SEND, RUN=SNDP, RUN=SPIN, RUN=UPDS

3.36. CRYPT

CRYPT Command

Purpose

Perform conversion from one character set to another (ASCII to EBCDIC, etc). This operation is usually done only on type 5 or type 6 DAL elements.

CRYPT may also be used to pack or unpack bit fields, ie convert a packed string of 12 bit integers into a string of full word integers or visa versa.

Command

```
CRYPT  N1,EL1,VR1  N2,EL2,VR2  IBPC,JBPC  IST,LST
-DATA-
```

Where

```
N1,EL1,VR1 - Input DAL element.
N2,EL2,VR2 - Output DAL element.
IBPC       - Number of bits per input character.
JBPC       - Number of bits per output character.
IST        - Lowest input character value.
LST        - Highest input character value.
-DATA-     - A list of new character values (integers).
```

Example

Convert the symbolic file 1,MY,FILE from ASCII to FIELDATA.

```
CRYPT 1,MY,FILE 1,MY2,FILE 8,6 0,127
=30,ASCI,FLDT
```

Notes

CRYPT is used to convert ASCII to EBCDIC or FIELDATA, etc. The input array consists of symbolic images as will the output array. The row length output will be different if (JBPC.NE.IBPC).

The basic approach is to read the input data set and collect the first IBPC bits. This forms an integer. This integer is used as a pointer into the input table. If I is the integer, then a new integer J is found as J = DATA(I-IST+1).

Notes on character conversion

The hardest thing about converting from ASCII to EBCDIC is preparing the appropriate cross-reference table to use as data to CRYPT. Several of these tables have been prepared already, and exist as data elements on DAL unit 30 (the master RUN file). The following names are used:

```
ASCI  -- ASCII code (7 bit code, 128 values)
CYBR  -- CYBER (CDC) code (6 bit code, 64 values)
EBCD  -- EBCDIC code (8 bit code, 256 values)
FLDT  -- FIELDATA (UNIVAC) code (6 bit code, 64 values)
```

The element name of the element determines the current code, and the version name determines the desired code. (eg 30,ASCI,EBCD contains the table for ASCII to EBCDIC) The length of the element corresponds the the number of values in the current code (ASCII = 128). You should always set IST=0, and LST=length-1. Ie:

```
CRYPT  1,ASCI,DATA  1,EBCD,DATA  8,8  0,127
= 30,ASCI,EBCD
```

would convert from ASCII TO EBCDIC. Note that ASCII is a 7 bit code, and yet 8 was used for IBPC. This is due to the fact that the data was written as 8 bits per character, but the left most bit is always 0. For the UNIVAC, ASCII characters are normally stored as 9 bit characters. This is also true for EBCDIC when //////Q is used to read an offsite tape.

Notes on bit manipulation

If IST and LST are omitted from the CRYPT command, CRYPT will not read a data table, and will not do any trans- lations. In this case, it will simply read IBPC bits, right justify them into JBPC bits, and write them.

VAX considerations

VAX is a byte addressable machine and stores character data in successive byte addresses. If you look at a long word that contains 'ABCD' it appears to contain CDAB. CRYPT expects this, and pulls the input bits a byte at a time, and writes the output bits a byte at a time. This allows 8-bit to 8-bit character conversions with no problem. Sometimes (especially with bit manipulation) it is desirable to operate on a full long word at a time, rather than byte by byte. This may be accomplished by setting the variables NIN$ and NOT$ prior to using CRYPT.

```
NIN$ - Number of bytes to grab at once (default=1 max = 4)
NOT$ - Number of bytes to write at once (default=1 max = 4)
```

If you are interested, try this on a VAX

```
* CREATE A TYPE 5 ELEMENT THAT HAS ONE LINE
SYMIN 1,TEST,SYM
THIS IS MY ONE LINE
SEOF
* PRINT THE OCTAL CONTENTS
PRICF  1,TEST,SYM  1,6
(6O12)
* SET NIN$ AND NOT$
SET NIN$=1,NOT$=4
CRYPT  1,TEST,SYM  1,NEW,TEST
PRICF  1,NEW,TEST
(6O12)
PRICF  1,NEW,TEST
(1X,6A4)
```

Related Commands

COPY, RUN=SPILL, TDUMP

3.37. RUN CSD2

RUN=CSD2 Command

Purpose

Coupled Spectral Density calculation part 2. Gives the response of an FEM model to some acoustic loading based on a correlation function. See SCSD for a SIN(kx)/kx correlation function.

Command

```
RUN=CSD2  NUMD,NVMD  NUSD,NVSD  NOUT,NVOU  IDVA
```

Where

```
NUMD,NVMD - Unit and version name of input model datasets
            FN   - Input natural frequencies   (#mode,1)
            RN   - Input generalized mass      (#mode,1)
            QN   - Input generalized gain      (#mode,1)
            PX   - Input response eigenvectors (#resp,#mode)
            CSDF - Input frequencies at which responses are to be
                   calculated(#freq,1).

NUSD,NVSD - Unit and version name of input spectral datasets
            MSDF - Input MSD frequency vector      (#msdfreq,1)
            MSD  - Input Modal Spectral Density matrix (#mode,#mode)
                   There is one MSD matrix for every frequency in
                   NUSD,'MSDF',NVSD.  The second cycle number of the
                   MSD matrix corresponds with a row number in the
                   frequency matrix.

NOUT,NVOU - Unit and version name for saving PSD (RESP) results.
            RESP  - Output Power Spectral Density (#resp,#freq)
            RMS   - Output RMS of PSD (RESP) (#resp,1)

IDVA      - Option indicator.
            = 1 - Uses a displacement transfer function.
            = 2 - Uses a velocity transfer function.
            = 3 - Uses an acceleration transfer function.
```

Summary of control variables

```
IC2$ - For restarted runs is set to the last frequency completed.
IC3$ - Maximum frequency to be calculated.
ICV$ - Conversion flag.
     = 0 - Conversion is made from (in/sec**2)**2/hz to g**2/hz
     > 0 - No conversion is done.
FDIF - Factor for determining if a frequency is equal to the input
       msd matrix frequency.  See notes below.
```

Example

```
* 1/3 Octave Center frequencies.
FREQUENCY/CENTER 1,EXFQ,PRES 125.,160.
* Input Sound Pressure Level in dB.
ECOL 1,DBEX,EXMP 2,1
2*100
* Convert to PSI**/Hz
CONVERT/SOUND 1,DBEX,EXMP 1,EXC,PRES 1,9 125.,160.
* Prepare input for RUN=SCSD
RUN=MSDSETUP 1,PRES 125.,160.
1,JLOC,BTAB MASK,MASK
1,VIBR,MODE MASK,MASK
1,VIBR,EVAL MASK,MASK
1,EQNF,FORC MASK,MASK
* Calculate Modal Spectral Density
RUN=SCSD 1,PRES 2,PRES
* Joints at which responses are desired.
ECOL 1,RESP,LIST 18,1
1,51,101,151,201,251,301,351 401,451
501,551,601,651,701,751,801,851
* Convert joint numbers to DOF's
RUN=JTOD 1,RESP,LIST 1,DOF,LIST 1,2,3
* Set up input to RUN=CSD2
RUN=CSDSETUP 1,PRES 1,DOF,LIST 2,PRES 50.,.5
1,VIBR,MODE MASK,MASK
1,VIBR,EVAL MASK,MASK
* Calculate response.  Results will be in G**2/HZ.
RUN=CSD2 1,PRES 2,PRES 3,PRES 3
TOC 1
TOC 2
TOC 3
```

Execution time

CSD2 can be very expensive, but it can be restarted, just like SCSD. In this case, if a run maxtimes, check for the last non-zero column of the PSD (RESP) matrix and set the variable IC2$ to that column number prior to executing the runstream. For checkout a maximum frequency number can be set with IC3$ equal to the last frequency number to be calculated.

Notes

CSD2 takes an input MSD matrix and interpolates to the MSD matrix needed for each frequency calculated. If a frequency is closer to one of the input frequencies than FDIF*DELTAF (where DELTAF is the difference between two appropriate input frequencies) the MSD matrix for that frequency will be used. FDIF is defaulted to 1.0E-6. Care should be taken in SCSD to calculate sufficient frequencies for the accuracy desired.

Related Commands

RUN=CSDF, RUN=CSDS, RUN=MSDF, RUN=MSDS, RUN=SCSD, RUN=JTOD

3.38. RUN CSDF

RUN=CSDFREQ Command

Purpose

This command calculates the frequencies required for the RUN=CSD2 command. This command is executed by RUN=CSDSETUP.

Command

```
RUN=CSDFREQ  N1,NAME  N2,NVSD  (Q  (SPC  (DZFAC)))
```

Where

```
N1,'CSDF',NAME - DAL unit and version name for output.  The output
                 element name is N1,'CSDF',NAME.
N2,'MSDF',NVSD - DAL unit and version name for input MSD frequencies.
Q              - Amplification factor.
SPC            - Frequency spacing.  (eg.  For 1/2 power spacing SPC=.5)
                 Default = .5
DZFAC          - Frequencies closer together than this factor are
                 considered to be identical.  Default=1.E-6.
```

Example

See RUN=CSD2 for a complete example of this analysis.

Related Commands

RUN=CSD2, RUN=CSDS, RUN=JTOD, RUN=MSDF, RUN=MSDS, RUN=SCSD

3.39. RUN CSDS

RUN=CSDSETUP Command

Purpose

This command does the setup for the RUN=CSD2 command.

Command

```
RUN=CSDSETUP N1,NAME  N2,NEL2,NVR2  N3,NVSD  (Q  (SPC  (DZFAC)))
```

Where

```
N1,NAME        - DAL unit and version name for RUN=CSD2 input.
N2,NEL2,NVR2 - Input vector of response degrees of freedom desired.
               Use RUN=JTOD to calculate degrees of freedom from joint
               numbers.
N3,NVSD        - DAL unit and version name for MSDFrequency vector.
Q              - Amplification factor.  Default = 50.
SPC            - Frequency spacing.  (eg.  For 1/2 power spacing SPC=.5)
               Default = .5
DZFAC          - Frequencies closer together than this factor are
               considered to be identical.  Default=1.E-6.
```

Input

This runstream prompts for the following input matrices:

```
'VIBR','MODE' - Mode shape matrix as output by EAL.
'VIBR','EVAL' - Natural frequencies as output by EAL.
```

Output

The following matrices are output:

```
N1,'RN  ',NAME - Generalized mass.  (#modes,1)  This is output as a
                 vector of 1.'s.
N1,'QN  ',NAME - Generalized gain.  This is output as a vector of Q.
                 (#modes,1)
N1,'PX  ',NAME - Response eigenvectors. (#resp,#modes)
N1,'FREQ',NAME - Frequencies at which responses are desired.
                 (#freqs,1)
```

Example

Refer to the RUN=CSD2 command for a complete example of this analysis.

Related Commands

RUN=CSD2, RUN=CSDF, RUN=JTOD, RUN=MSDF, RUN=MSDS, RUN=SCSD

3.40. CSYM

CSYM Command

Purpose

Read/write DAL symbolic (types 5/6) elements from/to system files or elements. The use of this command varies between computers

Command

```
CSYM  NDAL,EL1,VR1  NAMF
or
CSYM  NAMF,EL1,VR1  NDAL
```

Where

```
NDAL     - Integer DAL unit (1-30)
EL1,VR1  - Element name
NAMF     - System file name or dummy.
           The first form copies a DAL element to a system file. The
           second form copies a system file to a DAL element.
```

Example

Write the symbolic element 30,RUN,GRMS to a symbolic file.

```
CSYM 30,RUN,GRMS DRS
```

On VAX and UNIX machines this will create the file RUNBRMS.DRS. On UNISYS machines this will create the file DRS.RUNGRMS.

Notes

In all cases, the system file or element name is one word (max 8 characters) that is derived by combining the DAL element/version names.

On UNIVAC, NAMF is the name of a program file for reading or writing the system element. NAMF is limited to a max of 4 characters.

On VAX, NAMF is the version (or type) name of the element. Ie. DUMDUM.DAT is a typical VAX file name. DAT is used for NAMF. CSYM DAT DUM DUM 1 reads the file and creates a DAL element. CSYM 1 DUM DUM DAT writes the DAL element to the VAX file. Unix systems work the same as the VAX.

On PRIME, the 8 character name is the actual PRIME file name. NAMF is used only as a key on which direction the transfer is to go. NAMF should be any hollerith name, or it may be set to zero.

Currently there is no CDC version.

DAL element type

The type of DAL element produced by CSYM is controlled by the values of the variables NRB$ and NCR$. If these variables are not set, CSYM will produce a type 5 element (variable length records). To get a type 6 element (fixed record length), set NCR$= number of characters per record, and NRB$= number of records per block. e.g.

```
SET  NCR$=80,NRB$=60
CSYM .......
```

The use of type 5 elements is recommended, as they make the best use of disk storage. Type 6 elements are convenient for writing symbolic files to tape, ie. for offsite transmittals. As an example, if you have a group of card images on DAT.DATADECK (UNIVAC) or DATADECK.DAT (VAX), the following stream would write the deck to tape as 80 chars per record, 60 records per block.

```
SET NCR$=80,NRB$=60
CSYM  DAT  DATA,DECK  1
WTAPE  1,DATA,DECK  4,1,1
```

where images are written to tape 4, starting at file 1, record 1.

Related Commands

PRINT, SYMIN

3.41. CYCLE

CYCLE Command

Purpose

Change preset cycle numbers during an execution. Recover current cycle numbers.

Command

```
CYCLE  IC1,IC2  JC1,JC2
or
CYCSET  NAM1,NAM2,NAM3,NAM4
```

Where

```
IC1,IC2 - Cycle numbers to be used when finding elements.
JC1,JC2 - Cycle numbers to be used when creating new elements.
NAM1    - Name of variable to receive current value of IC1
NAM2    - Name of variable to receive current value of IC2
NAM3    - Name of variable to receive current value of JC1
NAM4    - Name of variable to receive current value of JC2
```

Example

Set the input and output cycles to 1,0.

```
CYCLE 1,0
```

Notes

If either command form is used without any arguments, a message is printed indicating the current cycle values.

If JC1 and JC2 are omitted, they are set equal to IC1 and IC2. The word 'MASK' may be used for any or all of the cycles which indicates that that cycle number is to be ignored when searching for the element.

Caution is advised when input and output cycles are different (unless input cycles are 'MASK'ed) as newly created elements may not be referenced without modifying cycle names.

The use of cycles and thus the CYCLE command is very important in many VAPEPS executions. The basic theory is as follows:

All DAL elements are identified by four names

```
ELN,VER,IC1,IC2
```

The first two names are hollerith words, the last two names are normally integers. All four names must be specified when reading an existing element and when creating a new element. Contrast this statement with the following general VAPEPS command:

```
Command  N1,EL1,VR1  N2,EL2,VR2
```

Note that the user only specifies the unit, element name and version name of each DAL data set. The cycle names are not specified and yet they must come from somewhere.

The CYCLE command sets default values for cycle numbers. Two pairs of cycle numbers are defined. The first pair is used as cycles when a VAPEPS command is reading an existing DAL element. The second pair is used when a VAPEPS command is creating a new element.

Related Commands

##CYCLE

3.42. RUN DATA

RUN=DATA Command

Purpose

Calls RUN=PICK to pick out selected channels from an event. Converts the data returned by RUN=PICK from dB to PSI**2 and averages it then outputs it in dB.

Command

```
RUN=DATA NU,IVNT N1,EL1,VR1 N2,EL2,VR2 FLOW,FHI IFLG
```

Where

```
NU,VENT     - Input event unit and name.
N1,IL1,IV1  - Input vector of channel names to be picked.
N2,IL2,IV2  - Name of output element.
FLOW,FHI    - Desired frequency range limits ( default to 10 and 10000
              hz, respectively )
IFLG        - Flag used to specify output units. Default is zero (output
              in dB) If specified as a value other than zero, output is
              in mean square values.
```

Example

Get the average of microphones M1, M2, M3 and M4 from the event STD1.

```
ECOL 28,MIC,NAMS 4,1
M1:M4
RUN=DATA 1,SDTD 28,MIC,NAMS 28,AVG,MICS 31.5,2000.
```

Related Commands

CONVERT, RUN=PICK

3.43. RUN DCON

RUN=DCON command

Purpose

This runstream takes an input DAMO parameter element and converts the specified element to a symbolic deck of var=val. This is useful as an aid in converting DAMO data sets to SEMOD data sets.

Command

```
RUN=DCON  N1,NAME  N2,NEL2,NVR2  ELT
```

Where

```
N1,NAME       - DAL unit and version name of DAMO parameter matrix.
N2,NEL2,NVR2 - DAL unit element and version for symbolic output deck.
ELT           - Name of SEA element to extract.  Valid values are:
                EXTA,SKIN,INTA,MONT,INST,FRAM.
```

Example

Convert the element EXTA in DAMO model STD1 to element EXTERNAL in SEMOD model TEST.

```
RUN=DCON  1,STD1 1,SYM,EXTA EXTA
SEMOD 28,TEST
 ELNAME
  EXTERNAL,1
   ##READ 1,SYM,EXTA
   LIST
   EXIT
  DONE
 DONE
```

Notes

The output element may require editing. Use CSYM or PRINT and SYMIN to get to a system file so it can be edited.

Related Commands

DAMO, RUN=GMOD, SEMOD

3.44. RUN DIAG

RUN=DIAG Command

Purpose

Create a vector which is the diagonal of the input matrix or create a square matrix given its diagonal as a vector.

Command

```
RUN=DIAG  N1,EL1,VR1  N2,EL2,VR2  MBLK
```

Where

```
N1,EL1,VR1 - Input matrix.  (Square or diagonal).
N2,EL2,VR2 - Output matrix.  (Square or diagonal).
MBLK - Number of terms to be taken along the diagonal,
       Default = 1.
```

Example

Create a square matrix whose diagonal is (1.,2.,8.,4.,3.).

```
ECOL 28,DIAG,VECT 5,1
1.,2.,8.,4.,3.
RUN=DIAG 28,DIAG,VECT 28,DIAG,MAT
```

Notes

For the vector diagonal mode, MBLK=1. MBLK may be set to 6, for example, if you want all 6x6 matrices along the diagonal of a square matrix.

RUN=DIAG works for single, double, and complex matrices.

Related Commands

RUN=LFU

3.45. DISABLE

DISABLE Command

Purpose

Disable a group of elements in a DAL file.

Command

```
DISABLE  NU,I1(,I2)

where
   NU = DAL unit
   I1 = Sequence number of first element in group.
   I2 = Sequence number of last element in group.
        If omitted, I2 = I1.
```

Example

Disable the first 10 elements on DAL unit 28.

```
DISABLE 28,1,10
```

Notes

Whenever a new element is created, the TOC is examined for an element of the same name. If found, the previous element is disabled (ie: marked for deletion). The DIABLE command allows the user to mark a group of elements for deletion. Note: Elements are not physically removed until the file is PACKed.

Related Commands

ENABLE, PACK, TOC, FIND

3.46. DOC

DOC Command

Purpose

This command will take the input help files and format them into a printable document complete with page numbers, table of contents and other features.

Command

```
DOC MAXITEM NLPP ISTP
-DATA-
```

Where

```
MAXITEM - Maximum number of sections in the manual.  This turns  out to
          be the combined total number of table of contents lines,
          figures, and tables.
NLPP    - Maximum number of lines per page.
ISTP    - Shifts output to the right.  This is the word offset  where
          output is to start.  For example ISTP = 1 would cause output
          to be shifted right 4 characters.  This includes the fortran
          carriage control character so it is effectivly 3 characters.
-DATA-  - Input lists of DAL files to include in the document.
```

DATA input

Input is in the following form:

```
LEVEL,NU,ELN,VER,IC1,IC2
```

Where

```
LEVEL - Is the section level to start this DAL element at.  This  may
        be input as a negative integer in which case the line that
        follows is read as a the section title.
NU,ELN,VER,IC1,IC2
      - DAL unit, element, version and cycle numbers of the DAL
        element to be used.
```

Control variables

The following DAL variables can be set prior to execution of the DOC command.

```
ITM$ - This sets the device type.  The following devices are supported:
     = XEROX - Output is formated for the SEND/SFONT command.
LVL$ - This controls how page ejects are performed on sections.  All
       section levels up to and including the LVL$ are forced to start on
       a new page. This number can be appended to the DOC command.  (eg.
       DOC2 has the same effect as if LVL$ were set to 2.)  The DOC2 form
       superseeds the LVL$ form.
```

3.46.1. FORMATING OF OUTPUT

The DAL elements input above can have statements embeded in them to cause special formatting of the output. The statements are summarized as follows:

3.46.1.1. DECK

DECK Command

Purpose

Starts a new deck.

Command

```
C=DECK NAME ...
```

Where

NAME - Text entered here is used as the title of the section unless something else is specified via the C=TITLE command. Other text may follow NAME. If NAME has embeded blanks it must be surrounded by ''.

Example

The file that generated this text starts with:

```
C=DECK DOC VAPEPS MANUAL
```

Notes

This is normally the first command in a DOC deck.

3.46.1.2. KEYS

KEYS Command

Purpose

This command is used by the GLOSS processor to produce an index.

Command

```
C=KEYS KEY1 KEY2 KEY3 ...
```

Where

```
KEY1 KEY2 KEY3 ...
      - Keys words to be put in the index.
```

Notes

This command must follow the DECK command and comes before the C=BLOCK DESCRIPTION command.

3.46.1.3. DESCRIPTION

DESCRIPTION Command

Purpose

This command is used to create a summary listing of 1 line descriptions for all commands.

Command

```
C=DESCRIPTION TEXT
```

Where

TEXT - Line of text to be put in the command summary.

Example

The C=DESCRIPTION command used in this deck is as follows:

```
C=DESCRIPTION DOC.........Document formater.
```

3.46.1.4. BLOCK

BLOCK Command

Purpose

This command breaks up the text into sections and sub-sections etc.

Command

```
C=BLOCK BLOCKNAME
```

Where

BLOCKNAME - The name of the block that follows the command.

Notes

The C=END command ends the block that it specifies. Blocks may be nested up to 10 levels.

Special case

When BLOCKNAME is DESCRIPTION this is taken as the outer most block in the file. Any text before this is ignored. Any text following a C=END DESCRIPTION is also ignored.

3.46.1.5. END

END Command

Purpose

This command ends a block that was started with the C=BLOCK command.

Command

```
C=END BLOCKNAME
```

Where

BLOCKNAME - Name of block as it appeared in the C=BLOCK command.

3.46.1.6. ON-OFF

ON-OFF Command

Purpose

This command turns on or off the DOC and HELP modes.

Command

```
C=ON DOC
C=OFF DOC
C=ON HELP
C=OFF HELP
```

Where

DOC-HELP - Refer to the DOC or HELP processors respectivly.

Notes

The command C=OFF DOC will cause all text following it to be ignored until a C=ON DOC is encountered while in the DOC processor. The same is true for the C=ON HELP and C=OFF HELP commands in the HELP processor. Commands that are found in these sections are ignored the same as text.

3.46.1.7. TITLE

TITLE Command

Purpose

Provide a title for the section.

Command

```
C=TITLE TEXT
```

Where

TEXT - Is the title to be used for this section.

Notes

This command should follow the C=BLOCK command.

3.46.1.8. FIGURE

FIGURE Command

Purpose

Provides blank space to put in a figure and creates a table of contents line for the figure.

Command

```
C=FIGURE LINES TEXT
```

Where

```
LINES - Number of blank lines to leave for the figure.
TEXT  - Text to be used as a title for the figure.
```

3.46.1.9. TABLE

TABLE Command

Purpose

Provides blank space to put in a table and creates a table of contents line for the table.

Command

```
C=TABLE LINES TEXT
```

Where

```
LINES - Number of blank lines to leave for the table.
TEXT  - Text to be used as a title for the table.
```

3.46.1.10. ENHANCE

ENHANCE Command

Purpose

Turns on or off in line enhancements such as underlining.

Command

```
C=ENHANCE ON
C=ENHANCE OFF
```

Where

ON - Turns on enhancment decoding.
OFF - Turns off enhancment decoding.

Enhancements

The following enhancement codes are available:

\ul - Text following this is underlined.
\bf - Text following is bold face.
\bl - Text following is blinking (VT100 terminals only.)
\rv - Text following is in reverse video (VT100 terminals only.)
\of - Turns off all of the above modes.

XEROX printer fonts.
\f1 - Normal typewriter font. (Default)
\f2 - Italic font.
\f3 - Bold font. \bf gives this font.
\f4 - Special characters.
\f5 - Double high characters.

The XEROX codes assume that the SEND/DFONT command or equivalent is available on your VAX.

3.46.1.11. EJECT-REMAIN

EJECT-REMAIN Commands

Purpose

Cause unconditional and conditional page ejects.

Commands

```
C=EJECT
C=REMAIN LINES
```

Where

LINES - Number of lines that must be left on the page for a page eject not to be done.

3.47. DPTOR

DPTOR Command

Purpose

DPTOR performs the conversion between single and double precision matrices.

Command

```
DPTOR  N1,EL1,VR1  N2,EL2,VR2  IFLAG
```

Where

```
N1,EL1,VR1 - Input matrix.
N2,EL2,VR2 - Output matrix.
IFLAG      - Direction flag (optional).
           =  1 - Implies input matrix is double precision.
           = -1 - Implies input matrix is single precision.
```

Example

Convert the single precision vector (1.,2.,5.,4.,3.) to double precision.

```
ECOL 28,A,A 5,1
1.,2.,5.,4.,3.
DPTOR 28,A,A 28,B,B
FIND 28,A,A
FIND 28,B,B
```

Notes

Double precision arrays are identified by a type code of 2 (or -2). Single precision arrays have a type of 1 (or -1). DPTOR checks the type of the input element and performs the necessary conversion to the complement type.

If IFLAG is used, the indicated direction must match the operation that DPTOR would have done if it were omitted. If there is a conflict, an error is returned and the conversion does not take place.

Related Commands

CMOD, RTOI

3.48. RUN DTAB

RUN=DTABLE Command

Purpose

This command creates a table from the data set(s) specified.

Command

```
RUN=DTABLE
```

Notes

This runstream prompts for the following input:

```
1) N1,NVR1 ITYPE - Input DAL unit and version of data and type.

   Where

       N1,'RDAT',NVR1 - This contains the data.  Number of frequencies
                        by number  of channels.
       N1,'RCHN',NVR1 - This contains the channel names.  One for each
                        column in  RDAT.
       N1,'RFRQ',NVR1 - Contains the frequencies of the data.  This
                        should be a  single vector.  If it is not a
                        vector then the first column is used.
       ITYPE          - This is the first character of the element
                        name in the above data sets.  The default is
                        'R'.  The following are used for specific
                        types of data:
                      = M - Microphone data.
                      = A - Accelerometer data.
                      = S - Strain gauge data.

2) NC, NOUT

   Where

       NC   - Number of data columns per output line.  Default =  6.
       NOUT - Fortran unit to direct output.  Default = 6.

3) FTYPE, WIDTH, DIGITS

   Where

       FTYPE  - Is the Fortran format type. (E,F,I,H,...) Default = E.
```

WIDTH - Number of total digits in the output format. Default = 11.
DIGITS - Number of significant digits in output. Default = 4.

The above defaults yield a format of (E11.4) for each output column.

Related Commands

RUN=PCOMPARE

3.49. ECOL

ECOL Command

Purpose

Enter data by columns. A rectangular matrix is created. The first column of the matrix will contain values input on the first data records, etc. Two basic forms are allowed: ECOL AND ECOSYM. ECOL is used to input real, integer, or hollerith data in free field format. ECOSYM is used to read hollerith data as card images.

Command

```
ECOL  N1,EL1,VR1  NR,NC
- DATA -

ECOSYM  N1,EL1,VR1  NCHAR,NC
- DATA -
```

Where

```
N1,EL1,VR1 - New element name.
NR         - Number of rows per column.
NC         - Number of columns.
NCHAR      - Number of characters per column.
```

Notes

ECOL reads data using the VAPEPS free-field interpreter, and keeps track of the type of data being entered. If all of the data is of the same type, the matrix type will be set accordingly (0=integer, 1=real, 4=hollerith). If a mixture of types is encountered, a type of 0 is set.

Several variations of the ECOL command exist. These forms explicitly set the type of the output element. No checking is performed internally:

```
ECOI - Produces an integer matrix (type 0)
ECOR - Produces a real matrix  (type 1)
ECOD - Produces a double precision matrix (type 2)
ECOC - Produces a complex matrix (type 3). Note that two numbers are
       required per complex word.
ECOH - Produces a type 4 element.
```

The idea is to enter NC columns of data. The data for each column must begin on a new card. You are allowed as many cards as necessary per column, however, exactly NR values must be specified before moving on to the next column. In the case of complex (ECOC), NR*2 numbers are required per column.

The command ECOSYM is different than ECOL (et al) in that free field input is not used. Cards are read using a hollerith format such that all words are filled. Ie., On UNIVAC = (13A6), VAX = (20A4), CDC = (8A10), etc. A maximum of 80 characters may be read per line (column). Thus one line is expected per column. The VAPEPS precoder ($$ notation) may be used to decode variables into hollerith as part of the read process.

All three commands read data until either NC columns have been filled or an EOF is sensed. In the latter case, the output matrix size will be adjusted to indicate the actual number of columns read.

Examples

```
SET NUM=100,INC=5,F=27.3
ECOL  1,INT,DATA  100,3      - Standard integer
1:100
&NUM*20
10/&NUM/&INC
ECOL 1,MIX,DATA 5,2          - Mixture of types. Note That the resulting
1.,2,3.,4,5.                   Element should be used carefully since its
A,B,&NUM,6.7,3                 type code does not represent its contents.
ECOR 1,REAL,DATA 100,3
1./&NUM/3.                   - Column 1
1.:100.:2.                   - Column 2 (1,3,...,99)
2./50/2.                     - Column 2 cont.  (2,4,...,100)
100*&F                       - Column 3

ECOSYM 1,PLOT,TTLS 80,3
THIS IS TITLE ONE
$$ FREQUENCY = &F(F5.2)      - Results in 'FREQUENCY = 27.30'
THIS IS THE LAST LINE
```

Related Commands

CSYM, GREAD, SREAD, SYMIN, ##READ, ##OPEN, ##CLOSE

3.50. ENABLE

ENABLE Command

Purpose

ENABLE a group of elements in a DAL file.

Command

```
ENABLE  NU,I1(,I2)

where
   NU = DAL unit
   I1 = Sequence number of first element in group.
   I2 = Sequence number of last element in group.
        If omitted, I2 = I1.
```

Example

ENABLE the first 10 elements on DAL unit 28.

```
ENABLE 28,1,10
```

Notes

Whenever a new element is created, the TOC is examined for an element of the same name. If found, the previous element is disabled (ie: marked for deletion). The ENABLE command allows the user to enable (reinstate) any element that has been disabled. Note: Elements are not physically removed until the file is PACKed.

Related Commands

DISABLE, PACK, TOC, FIND

3.51. RUN ENTR

RUN=ENTR Command

Purpose

Duplicate ENTER output for the case where data already is in a DAL file.

Command

```
RUN=ENTR NU,IVNT ISEC
-DATA-
```

Where

```
NU,IVNT - DAL unit and version name for output DAL elements.
ISEC    - ENTER section number.
-DATA-  - Input lines of data as follows:

          - Four lines of description.

          - Number of channels

            For each channel:
          - NAME,TYPE,UNITS
          - CHANNEL DESCRIPTION

          - 1/3 Octave Frequency Range.

          - Name of DAL file containing the data.
```

3.52. RUN EQPL

RUN=EQPL Command

Purpose

Use RUN=EQPL to model complex (honeycomb, beam stiffened - orthotropic) skin structures for SEA predictions. It calculates a set of equivalent homogeneous parameters (RHO, RHOS, H and E) given specific parameters of a structure. Formulas and assumptions for these calculations are show under the EQUATIONS heading.

To describe a structure for this routine, isolate a rectangular cutout of it, then break it down by slicing it up into co-planar segments (layers) of rectangular cross-section, and group them into the following categories:

-bare skin layers (omni-directional stiffness) e.g. sheet metal, honeycomb.

-lengthwise running beam layers (uni-directional stiffness) e.g. cylinder longerons, if present.

-widthwise running beam layers (uni-directional stiffness) e.g. cylinder stiffening rings, if present.

Command

```
RUN=EQPL N1,VER1 (N2,VER2)
-input-
```

Where

N1,VER1 - Dal unit and version name for output. If N1 is input as a negative number, an old set of parameters is read in from -N1,VER1 (modification mode). The parameter values for the old model can be changed by stepping through the input sections and making changes as you go.

N2,VER2 - (optional) If N1 is negative, Dal unit and version for the output modified parameters. Default: output is to -N1,VER1 (new replaces old).

Input

You are prompted to input parameters by section. The sections are:

1 general parameters,

2 bare skin parameters - each layer,
3 lengthwise beam parameters - each layer (if specified)
4 widthwise beam parameters ... (if specified)

The form of input is PARAMETER1=VALUE, PARAMETER2=VALUE, etc. - the order is not important. There are four command that can be issued typed in during parameter input, they are:

- * LIST lists out the parameters and their values.
- * CHECK lists out what still needs to be specified.
- * DONE takes you on to the next step.
- * QUIT cancels the run.

Prompted input

1 General parameters:

NS =_____ - Number of bare skin layers (e.g typical honeycomb NS=3 - top sheet, core, bottom sheet).
NBL =_____ - Number of lengthwise running beams layers (e.g a single "L" beam NBL=2 - vertical web, horizontal web). Default: NBL=0.
NBW =_____ - Number of widthwise running beams (uni-directional stiffness) layers. Default: NBW=0.
RLEN=_____ - Length of the structure.
WID =_____ - Width of the structure.
ER =_____ - Reference Young's modulus (E of the most common material of the structure - e.g. aluminum ER=10.6E+6 psi).
RHOR=_____ - Reference mass density (RHO of the most common material of the structure - e.g. aluminum RHOR=2.591E-4 lb*sec**2/in**4).
CL =_____ - Longitudinal wave speed of bare skin - e.g. aluminum CL=2.01E+5 in/sec.

2 For each bare skin layer:

H =_____ - Thickness.
CEN =_____ - Distance from the centroid of this layer to an arbitrary reference point.
E =_____ - (optional) Young's modulus of this layer. Defaults to ER.
RHO =_____ - (optional) mass density of this layer. Defaults to RHOR.

3 For each lengthwise running beam layer (if any):

H =_____ - Thickness.
CEN =_____ - Distance from the centroid of this layer to an arbitrary reference point.
E =_____ - (optional) Young's modulus of this layer. Defaults to

ER.
RHO =_____ - (optional) mass density of this layer. Defaults to RHOR.
W =_____ - Width of this beam layer.

4 For each widthwise running beam layer (if any):

- same as above -

Output

RUN=EQPL prints out what was input and the resulting equivalent parameters.

The following DAL elements are output:

N1,'GENR',NVR1 = Input parameters EREF,RHOR,LEN,WID,CL (5 x 1).
N1,'GENI',NVR1 = Input parameters NS, NBW, NBL (3 x 1).
N1,'BARS',NVR1 = Input parameters H, CEN, E, RHO for each skin layer (NS x 4).
N1,'BEML',NVR1 = Input parameters H, CEN, E, RHO, W for each lengthwise running beam layer, if present (NBL x 5).
N1,'BEMW',NVR1 = Input parameters H, CEN, E, RHO, W for each widthwise running beam layer, if present (NBW x 5).
N1,'EQPL',NVR1 = Equivalent parameters - H, E, RHO, RHOS, CL and TOTM (total mass), RMBS (bare skin mass), RMBL (lengthwise beam mass) and RMBW (widthwise beam mass) (9 x 1).

Equations

The equivalent parameters derived from these methods results in SEA predictions that correlate well with empirical data.

Equivalent parameter equations:

H - Match the critical frequency of structural panel

$$H = (0.551*2*pi/CL) * (D/ps)^{1/2}$$

RHO - Match the surface mass density

$$RHO = ps/H$$

E - Match the bending stiffness

$$E = 12*D / H^{3}$$

Where

pi = 3.1416...
CL = Longitudinal wave speed of the structural panel material (e.g. for a honeycomb panel use the cl of the face sheets)
ps = Surface mass density of the structural panel
D = Equivalent stiffness of the structural panel. The equation for an orthotropic structure is shown below.

Equivalent stiffness calculation for beam stiffened structure:

The equivalent stiffness calculation matches the stiffness of a simply supported orthotropic panel to that of a simply supported isotropic panel.

For cylinders, unroll it into a flat panel.

```
                  WID
  b ---->  ______  /\__/\______
           \       \ \ \ \      \
       L    \  ~bs  \ \ \ \      \
        E   /\       \ \ \ \      \
         N  \/\       \ \ \ \      \
              \_______\/__\/______\
       b ----->
              ↑                    ↑
               \                    \
                \                    \
                 a                    a
```

Section xx:
Lxx - width of cross-section
Exx - Young's modulus of cross-section
Ixx - Moment of inertia of cross-section (about it's neutral axis)

Sections:

```
   _______|__|_______   : aa - orthotropic section (isotropic
                               skin with beams)
          ________
          |_|        : bb -   "       "       "

   __________________   : bs - isotropic skin without beams
                               (bare skin)
```

Calculate D using this equation:

```
         Eaa*Iaa/Laa      Ebs*Ibs/Lbs      Ebb*Ibb/Lbb
         ----------- + 2*-----------   +  -----------
               4             2     2            4
           Lbb            Laa * Lbb          Laa
   D = --------------------------------------------------
              1            2            1
            -----  +  ---------  +  -----
   4        2     2        4
Lbb      Laa * Lbb      Laa
```

3.53. RUN ERRO

RUN=ERROR command

Purpose

This command is used to keep track of errors reported for DALPRO, VAPEPS etc.

Command

```
RUN=ERROR  NU option
```

Where

```
NU     - DAL unit of file containing error information.
option - Is one of the following:
       = LIST - lists out data.
```

This runstream will prompt for the appropriate information.

Related Commands

RUN=EZDB, RUN=CHKOUT, RUN=TRMT

3.54. RUN ESAV

RUN=ESAVE Command

Purpose

Writes the enter deck for event 'NAME', section 'ISEC' from DAL file 'NU' to Fortran unit 'MU'.

Command

```
RUN=ESAV  NU,NAME,ISEC  MU
```

Where

```
NU,NAME - DAL unit and event name of event enter deck to be written.
ISEC    - Section from event enter deck to be written.
MU      - Fortran unit to which to write output.
          If MU>0, the Fortran unit is closed after write.
          If MU<0, the fortran unit is left open after write.
```

3.55. EULER

EULER Command

Purpose

EULER creates 3 by 3 transformation matrices which represent successive coordinate rotations about various axes.

Command

```
EULER  N1,EL1,VR1  N2,EL2,VR2
-DATA-
```

Where

N1,EL1,VR1 - Input DAL element containing rotation angles Each row corresponds to a different matrix, each column is for a rotation about a different axis

N2,EL2,VR2 - Output DAL element. It will always have 3 columns, and will have three times as many rows as N1,EL1,VR1

-DATA- - One card of data containing as many integers as there are columns in N1,EL1,VR1. Each integer must be between 1 and 3, indicating rotation about X,Y and Z axis respectively.

Notes

Assume N1,EL1,VR1 is NR by NC, then you will be computing NR 3 by 3 matrices. Each matrix will correspond to NC successive rotations. The output element (N2,EL2,VR2) will be dimensioned 3*NR by 3.

In general you can assume that EULER builds a rotation matrix such that:

```
X' = ROT * X
```

Note though that if each joint is rotated through the same angles, then the output (eg ROT/MAT) is a 3x3, and if the original XYZ matrix is XYZ/ORIG(20,3) then

```
MTRAN  1,XYZ,ORIG  1,ROT,MAT  1,XYZ,NEW
```

will give the coordinates relative to the new system.

Important

Angles in N1,EL1,VR1 must be in radians. If you have the angles in degrees, use EULED for the command name.

Example

Joint one is to be rotated 30 degrees about Z axis, and 20 degrees about X axis. Joint two is rotated 50 degrees and -45 degrees. Compute the matrices

```
ECOL  1,DUM,ANG 2,2
30.,50.
20.,-45.
EULED 1 DUM ANG 1 ROT MATS
3,1
```

3.56. RUN EVEN

RUN=EVENT command

Purpose

This command is used to display information about various parts of an event on the data base.

Command

```
RUN=EVENT EVENT options
```

Where

```
EVENT   - Is the name of an event in the master file.
options - One or more from the following list:
          ALL    - Lists all information on the event.
          BOOK   - Lists all book information.
          CHAN   - Lists all channel information including descriptions.
          CNFG   - Lists configuration tree.
          MODS   - Lists summary of modules.
          MOD=N  - Lists information on module N.  If N=ALL then all
                   module information is displayed.
          MODC=N - Lists only channel information for module N.
          MODP=N - Lists only parameter information for module N.
          MODN=N - Lists only nondimensional parameters and descriptions.
          FOR=N  - Directs output to fortran unit N.
          UNIT=N - Looks on DAL unit N for EVENT instead of searching
                   the master file first.  This is useful if the
                   database file is already assigned to a DAL unit.
```

Notes

On some machines it may be necessary to assign the database file prior to using this runstream then use the UNIT=N option. This runstream should work fine on VAX and UNIX machines.

If ALL is used it must be the first option on the line.

3.57. EXPAND

EXPAND Command

Purpose

Modify the structure of an existing DAL element, without changing the contents.

Command

```
EXPAND  N1,EL1,VR1  N2,EL2,VR2  NEWR  (NJ)
```

Where

```
N1,EL1,VR1 - Existing element
N2,EL2,VR2 - New element
NEWR       - New number of rows (block length)
NJ         - Optional special purpose flag
```

Notes

In order to use EXPAND properly, the user must know the various data structures that exist, and how to recognize what structure a given element has or should have.

First of all, all elements are written one block at a time. In a SPAR Table of Contents, an element's structure is described by the parameters NWDS,NJ,NI*NJ. NI*NJ is the block length, NJ indicates either the number of joints, or the number of columns packed into one block. NWDS gives the total number of words in the array. VAPEPS commands assume that NJ is equal to one, ie that there is only one column per block, and that the block length is equal to the number of rows in the array. For this reason, the VAPEPS TOC lists NWDS,NR,NC where NWDS always equals NR*NC, NR is the block length, and NC is the number of blocks or columns. To find the number of rows and columns using the SPAR Table of Contents, one must do two divisions: NI = NI*NJ/NJ = NR and NC = NWDS/NI.

The rule of thumb to use is as follows. Give some thought to what matrix the element is supposed to represent, particularly the number of rows and columns you think the matrix should have (ie. MR and MC). If NI*NJ=MR and NWDS/(NI*NJ) = MC, then the element is already VAPEPS compatible. If NWDS = MR*MC, but NI*NJ is some multiple of MR (ie. NI*NJ = NJ*MR), then EXPAND may be used to create an element with block size MR. If NWDS is much less than MR*MC, the element is probably in sparse matrix format, and the VAPEPS command 'SPARSE' or 'DSPAR' must be used.

Description

EXPAND is used to increase or decrease the block size of an element. In all cases expand either combines an integer number of blocks into one block or breaks up each block into an integer number of blocks. NEWR is the new block length and must always be a multiple or divisor of the original block length. Assume in the following that JC is this multiple (or divisor), and NR is the original block size. Thus for NEWR > NR, JC = NEWR/NR and JC blocks are combined to form one output block. For NR > NEWR, JC = NR/NEWR and each input block becomes JC output blocks.

There are two logical ways that this blocking (or deblocking) may be done. The first (and most obvious) occurs when NJ is omitted or set to a non-negative number. Note that omitting NJ or setting it to zero will yield NJ=1

NJ > 0, NEWR > NR

The first block is formed by placing the first JC input blocks one after the other. The next JC blocks are then put in the second new block, etc.

NJ > 0, NR > NEWR

The first block is formed using the first NEWR values of the first original block. The next output block uses the next NEWR values from the same input block.

The second method rearranges the values in the blocks, as well as changing the block length. This occurs when NJ is input as less that zero.

NJ < 0, NEWR > NR

The first output block is formed, by using the first value in the first block, then the first value in the second block etc, through JC input blocks, and then getting the second value in the first block. The second output block is formed in the same manner, starting with input block JC + 1.

NJ < 0, NR > NEWR

The first output block will contain values 1,1+JC,1+2*JC,etc from the first input block. Output block two contains values 2,2+JC,2+2*JC,... etc from the first input block. Output block JC+1 begins again using input block number two.

Note

The absolute value of NJ will be used for the value of NJ in the new element

Examples

1. Convert SPAR blocked matrix 'VIBR','BACK' on DAL001., to a VAPEPS compatible matrix 'DAL','BACK' on DAL002. Assume NWDS = 9234, NJ = 20, and NI*NJ = 1140.

   First, NI = 1140/20 = 57, ie there are 57 rows in the matrix. Also, NC = 9234/57 = 162 blocks or columns.

   ```
   EXPAND 1 VIBR BACK 2 DAL BACK 57
   ```

   will create a 57 by 162 matrix on DAL002.

2. Convert 'DAL','BACK' above to a packed SPAR element on DAL005., only this time put 25 columns per block, or NEWR = 25*57 = 1425

   ```
   EXPAND 2 DAL BACK 5 NEW BACK 1425,25
   ```

3. Build a DOF table for a model with 10 stations, where all nodes have six degrees of freedom. First create a sequential vector from 1. to 60.

   ```
   SEQ 2 DOF VEC 60 1. 1.
   ```

   If DOFs 1-10 are all the X direction DOFS, 11-20 = Y'S, etc. then

   ```
   EXPAND 2 DOF VEC 1 DOF MAT 10
   ```

   builds the matrix:

   ```
    1.   11.   21.   31.   41.   51.
    2.   12.   22.   32.   42.   52.
    3.   13.   23.   33.   43.   53.
    4.   14.   24.   34.   44.   54.
    5.   15.   25.   35.   45.   55.
    6.   16.   26.   36.   46.   56.
    7.   17.   27.   37.   47.   57.
    8.   18.   28.   38.   48.   58.
    9.   19.   29.   39.   49.   59.
   10.   20.   30.   40.   50.   60.
   ```

   If DOFs 1-6 are for the first node, etc., then use

   ```
   EXPAND 2 DOF VEC 1 DOF MAT 10,-1
   ```

   to get

   ```
    1.    2.    3.    4.    5.    6.
    7.    8.    9.   10.   11.   12.
   13.   14.   15.   16.   17.   18.
   ```

```
19.   20.   21.   22.   23.   24.
25.   26.   27.   28.   29.   30.
31.   32.   33.   34.   35.   36.
37.   38.   39.   40.   41.   42.
43.   44.   45.   46.   47.   48.
49.   50.   51.   52.   53.   54.
55.   56.   57.   58.   59.   60.
```

3.58. RUN EZDB

RUN=EZDB command

Purpose

This command is used to keep track of modifications to the easy file for DALPRO, VAPEPS etc.

Command

```
RUN=EZDB  NU option unit
```

Where

```
NU     - DAL unit of file being maintained.
option - Is one of the following:
         LIST - lists out changes.
         SAVE - collects changes and transposes to unit 28
unit   - output fortran unit for listing changes (default=6)
```

This runstream will prompt for the appropriate information.

3.59. EZOUT

EZOUT Command

Purpose

Produce listings of EASY decks. The user provides a list of block names which designate the portions of the decks that are to be listed. In addition, the user may specify any number of headers that will be written prior to each deck listing. Following these specifications, the user inputs the EASY decks as data to the command. EZOUT is used to produce compilation runstreams as well as the VAPEPS manual.

Command

```
EZOUT NHEAD,NOUT
BLOCK1 BLOCK2 ... BLOCKN
BLOCKI ... BLOCKJ
('FORTRAN FORMAT THAT INCLUDES ',3A4)
...

BLOCKK
('LAST HEADER FORMAT',A4,2X2A4)
EASY DECKS
```

Where

NHEAD - Number of header pairs to be read.
NOUT - FORTRAN unit for output.
BLOCKI - A block name specification. These may include up to twelve characters.
FORMAT - Standard FORTRAN format statement, enclosed in parentheses. Each format must provide for the writing of 3 hollerith words, ie, 3A4.

Description - EASY

EASY decks contain card images. These images include control cards that act to describe the deck. Each control card begins with the letters 'C=', followed by a control command. In general, any number of commands may be present, but EZOUT only recognizes 3 types:

```
C=DECK DECKNAME1 DECKNAME2 DECKNAME3
C=BLOCK SPEC-1 SPEC-2 ... LAST-SPEC
C=END SPEC-1 SPEC-2 ... LAST-SPEC
```

The C=DECK command identifies the current deck of cards and must be the first card in each EASY deck. Note that three names must be present on the card. Each name may contain up to 12 characters.

The C=BLOCK and the C=END cards serve as dividers and provide each internal deck section with a group of local labels. Each data card image has a label status. Initially (after C=DECK) the label status is zero. When a C=BLOCK card is encountered, the included specs are added to the label status. The C=END card removes specs from the label status.

Consider the following:

```
C=DECK EXAMPLE1 CHECKOUT DECK
C=BLOCK ONE
     ONE LABEL IS CURRENT:  ONE
C=BLOCK TWO THREE
     THREE LABELS ARE PRESENT:  ONE TWO THREE
C=END ONE THREE
     ONE LABEL IS CURRENT:  TWO
C=END TWO
```

Description - EZOUT

The block specifications that are input to EZOUT control which sections will be listed. If a blank card is used, the entire deck will be listed, including control cards. If one or more blocks are specified, then

1) At least one of these block labels must be current for the line to be printed.
2) All block labels that are current must be among the original specifications in order for the line to be printed.
3) Control cards are not printed.
4) If a negative block label (NOT SPEC) is encountered, the lines will be printed as long as conditions 1 and 2 are met and SPEC was not among the original specifications.

The printing of a header involves writing the first deck name (DECKNAME1) using one of the header formats. Note that the first name is composed of 3 hollerith words and thus should be printed as 3A4.

Normally, headers are printed once per deck. The approach is to detect when the first card image is about to be printed. The label status at this point is compared to the specs that were included with each header definition. If all specs of a particular header are present in the current label status, the header is printed. If not, the header is not printed. If the label status returns to zero at any later point in the deck, the 'detect first card' flag is reset. Thus if another 'good' section is encountered, the appropriate header will be printed.

At times it is desirable to include a trailing line after a deck has been processed. The use of a header specification with only the label TRL$, will cause EZOUT to write the line indicated by the header format. Note that in this case, no deck names will be written as part of the write.

Examples

The following examples indicate how various features of EZOUT work. Note that the above EASY deck is used in the example.

```
EZOUT 0,6                                  - write the whole deck
- BLANK -
```

produces

```
C=DECK EXAMPLE1 CHECKOUT DECK
C=BLOCK ONE
     ONE LABLE IS CURRENT:  ONE
C=BLOCK TWO THREE
     THREE LABELS ARE CURRENT:  ONE TWO THREE
C=END ONE THREE
     ONE LABEL IS CURRENT:  TWO
C=END TWO
```

and

```
EZOUT 2,6
ONE TWO
ONE
('THIS DECK IS ',3A4)
TRL$
('TRAILER')
```

produces

```
THIS DECK IS EXAMPLE 1
ONE LABEL IS CURRENT:  ONE
ONE LABEL IS CURRENT:  TWO
TRAILER
```

3.60. FFT

FFT Command

Purpose

Fast Fourier Transform of a set of equal spaced vectors.

Command

```
FFT  N1,EL1,VR1  N2,EL2,VR2  NPTS  DT  SW
```

Where

```
N1,EL1,VR1 - Input matrix
N2,EL2,VR2 - Output matrix
NPTS       - Number of points used in transform, must be a power of
             two.
DT         - Time increment
SW         - -1. For transform, 1. For inverse transform
```

Notes

This FFT routine operates solely on real functions of time and their complex transforms. If SW=-1., then each col of N1,EL1,VR1 corresponds to a real function of time. If NPTS is less than the number of rows, only NPTS will be used if NPTS is larger, the remainder is filled with the last input value. The output element for SW = -1. is a complex array with NPTS/2+1 rows. Note that complex arrays have type=3 and really have twice as many rows as indicated in the table of contents. They may be changed to regular arrays using the CHANGT command. The resulting matrix will have odd rows = real, and even rows = imaginary portion. The frequency increment of the output (radians) is DW = 2*PI/(NPTS*DT).

For SW= 1., Element N1,EL1,VR1 is the complex transform to be inverted. It must be a complex array (type 3) with NPTS/2+1 rows. If fewer are input, zeros are added. If more are input, only the first NPTS/2 + 1 words are used. The output element will have NPTS rows and the same number of columns as input.

Dt and SW must always be real numbers. Be sure to include the decimal points. Typically DT will be the time step, but other factors may be used, as discussed below.

Compatability

This command uses the Cooley-Tookey algorithm for computing FFT's. This does not mean, however, that results from this command will be compatible with FFT's from other programs, even those which use Cooley-Tookey. The differences, if they exist, will be in the form of multiplication factors. For a forward transform, values are multiplied by 'DT' prior to output. For an inverse transform, values are divided by (NPTS*DT) prior to output. If you are comparing VAPEPS FFT's with other programs, or if you are back transforming FFT's computed with another program, it is very important that you are aware of the factors used in the other program. In most cases, VAPEPS can be made compatible with other programs by using a modified value for 'DT'. As an example, some programs give dimensionless FFT's that may be obtained by setting DT = 1./NPTS . Note that in this case, DT no longer refers to the time step.

3.61. FILES

FILES Command

Purpose

List names of all DAL units currently assigned to this execution. Note that this includes only those DAL files that have been referenced in the current execution of VAPEPS .

Command

```
FILES
```

Notes

The FILES command lists the following information about each active file:

1) Unit number
2) File name (QUAL*FILE) (device:[directory]filename.type;version)
3) Number of sectors in file
4) Number of writes during this execution.
5) Number of reads during this execution.

3.62. FIND

FIND Command

Purpose

Prints a TOC line for all elements that satisfy the input name.

Command

```
FIND    NU,ELN,VER IC1,IC2   (IDAT,JDAT)
or
FINE    NU,ELN,VER IC1,IC2   (IDAT,JDAT)
```

Where

```
NU   - DAL unit number
ELN  - Element name or  MASK
VER  - Version name or  MASK
IC1  - Cycle number 1 or MASK
IC2  - Cycle number 2 or MASK
IDAT - Optional integer date (YYMMDD). Only elements
       created on or after this date will be listed.
JDAT - Optional integer date (YYMMDD). Only elements created on or
       before this date will be listed.
```

Notes

IDAT defaults to zero, and JDAT defaults to the current date. If the dates are to be used, all element names must be specified. IDAT may be used without specifying JDAT.

The command FIND examines the TOC of the indicated unit and lists the TOC line for all elements that match the input names. Note that MASKed items are considered automatic matches. The FINE command is the same as FIND, except that disabled elements are ignored during the search.

If fewer than 5 items are specified, trailing items will be MASK filled. Thus the following commands are identical (both result in the entire TOC being listed).

```
FIND  1,MASK,MASK,MASK,MASK
FIND  1,MASK,MASK
FIND  1
```

Saving TOC lines

Two additional options are available for FIND, which allow the user to save a copy of the matching TOC lines:

```
FINDS  NU,ELN,VER,IC1,IC2   (IDAT,JDAT)
N1,EL1,VR1,JC1,JC2
```

and

```
FINES  NU,ELN,VER,IC1,IC2   (IDAT,JDAT)
N1,EL1,VR1,JC1,JC2
```

These operate in the same manner as FIND and FINE, except that the user must specify a DAL element name following the command. The matching TOC lines are written to this element instead of being printed.

```
N1,EL1,VR1(,JC1,JC2) - Name of output element JC1 and JC2  default to
                       current output cycles (CYCLE command)
```

The size of the element will be (13,N), where N = number of matching items.

Wild card usage

Element and version names specified in the FIND (FINES,etc) command may contain one or more asterisks. Ie.

```
FINE  1,'A**C','***Z'
```

would find all enabled elements on unit 1 that have an element name starting with A and ending with C, and a version name ending in Z. Note that the apostrophes surrounding the names are required, since asterisks normally imply multiple values (see FORMAT).

3.63. FIT

Purpose

Fits an mth order polynomial to an ordered set of data using the least squares method, and writes out the approximate ordinate.

Command

```
FIT  NX,ELX,VRX  NY,ELY,VRY  NO,VRO  MM,MODE
- DATA -
```

Where

```
NX,ELX,VRX - Abscissa array.
NY,ELY,VRY - Ordinate array.
NO,VRO     - Unit and version name of output elements.
MM         - M + 1 (ie. If MM = 4, a third order approx.)
           > 0 - Powers 0 through MM-1 are used.
           < 0 - Program prompts for user to specify powers.
MODE       - Output option flag.
           = 0 - Output one polynomial per input Y column.
           = 1 - Treat all columns as a population.  Find average
                 coefficients and output one ordinate only.  One RMS
                 value is output per column.
```

Example

Fit a straight line through the following set of data.
(1,1),(2,3),(3,2),(4,3),(5,8)

```
ECOL 28,X,VEC 5,1
1,2,3,4,5
ECOL 28,Y,VEC 5,1
1,3,2,3,8
FIT 28,X,VEC 28,Y,VEC 28,FIT 2,0
```

Notes

The input X and Y arrays essentially define a group of functions:

```
Y(I,1) = FUNC1(X(I,1))
Y(I,2) = FUNC2(X(I,2))
etc.
```

FIT does a least squares fit to express each function as a polynomial in X. For MM>0 the polynomial has order MM-1 (powers 0,1,...,MM-1). For MM<0, the user inputs the powers that are to be used.

The X and Y arrays must have the same number of rows. They must also have the same number of columns, unless the same X vector applies to all Y columns, in which case the X matrix may be input as a vector. In this last case, the user may set MODE=1. Then each Y column will be used to obtain one best-fit polynomial. The resulting polynomial coefficients are the average of the individual coefficients that would have resulted for MODE=0.

For MODE=0, a set of coefficients and an expanded poly- nomial will be output for each column. For MODE=1, only one set of coefficients and one polynomial will be output. A seperate RMS value is output per column in either mode.

Output

The output from FIT consists of 3 arrays:

```
NO,'APE',VRO  - expanded polynomials (NR,MC)
NO,'RMS',VRO  - root mean squares (NC,1)
NO,'COEF',VRO - coefficient array (MM,MC)
```

Where

```
NR = number of input rows
MC = number of output columns
   = number of input Y columns for MODE=0
   = 1 for MODE=1
```

Related Commands

POWER

3.64. FNAME

FNAME Command

Purpose

The default logical name associated with DAL unit 1 is DAL001. The default for DAL unit 26 is DAL030. These may be altered, either for all units or just a few, by use of the FNAME command.

Command

```
FNAME  DAL000
or
FNAME  NU,'AAAABB'
or
FNAME  NU,'filename'  (VAX, CRAY & UNIX machines)
or
FNAME  NU,CORE
```

In the first case, the basic logical name is set to DAL000 (which is default). Thus unit one becomes DAL001 etc.

In the second case, the specific unit takes on the name AAAABB where BB may be omitted if the name is four characters or less. Note that the second form only affects the named unit.

In the third case, the filename input is assigned to DAL unit NU. This doesn't work on the UNISYS.

In the fourth case, an "in core DAL file" is created with the number of bytes specified by CORE. The use of this option is useful if you have a runstream that uses a moderate amount of storage but does a lot of I/O. To determine the number of I/O requests use the FILES command. Use this option with caution, since when execution stops all information on this "in core DAL file" is lost.

The basic file name may be reset at any time during a run. This resetting will affect only any new units that are referenced. The logical name of a file is fixed once it is referenced. The second form immediately fixes the name of the file and may not be changed.

Note

DAL units 27-30 are automatically fixed at DAL027,DAL028, DAL029, and DAL030. They may not be changed with the FNAME command and they are not affected by the setting of a general name through FNAME.

3.65. RUN FPAS

RUN=FPASS command

Purpose

This runstream is used to combine acoustically induced and mechanically induced loads.

Command

```
RUN=FPASS  N1,NVR1  N2,NVR2  (SLOW,SHI  (NINC (IFLG (T)))
```

Where

```
N1,NVR1   - DAL unit and version name for the following input matrices:
            N1,'TRAN',NVR1 - Transient response time history.
                             (#Time points x #DOF's).
            N1,'TIME',NVR1 - Time points for input time history in seconds.
                             (#Time points x 1).
            N1,'RESP',NVR1 - PSD for each DOF.  (#Frequencies x #DOF's).
            N1,'FREQ',NVR1 - Frequencies for PSD.  (#Frequencies x 1).
N2,NVR1   - DAL unit and version name for output.  The following elements
            are output:
            N2,'SDOT',NVR2 - The standard deviation of the time rate of
                             change of the acoustically induced response.
            N2,'SIG ',NVR2 - The standard deviation of the acoustically
                             induced response.
            N2,'MXMN',NVR2 - Maximum and minimum values of the transient
                             response.
                             Column 1 contains maximum values.
                             Column 2 contains the time when the maximum
                             occured.
                             Column 3 contains minimum values.
                             Column 4 contains the time when the minimum
                             occured.
            N2,'PASS',NVR2 - Table of predicted values. Column 1 contains
                             the percentile value and each following
                             column contains the predicted values.
            N2,'NSIG',NVR2 - Table of N values of MAX+N*SIGMA
                             corresponding to the predictions.
SLOW,SHI  - Lower and upper N sigma values for calculation of barrier
            levels.  Barrier levels are calculated as follows:
              BLOW = MAX+SLOW*SIGMA
              BHI  = MAX+SHI*SIGMA
                Where
                  BLOW,BHI - Lower and upper barrier levels.
```

MAX - Maximum data value.
SIGMA - Standard deviation.
Default values are (SLOW=2.,SHI=7.).

NINC - Number of increments to calculate boundary levels between BLOW and BHI. Default=10.

IFLG - Integer to multiply transient by. A value of -1 will give the -95th percentile value of the lower bound. Default=1.

T - Duration of phenomenon. Defaluts to total transient response time.

Notes

The following elements must reside on DAL unit 30. These are used to evaluate an integral:

TOTL,ALPH
TOTL,TABL

The element 30,PER,CENT in generaly is not problem peculiar. It is a list of percentiles (eg. 1., 2., ..., 100.). This may be modified if there is a need. Since it resides on DAL unit 30 only the some one with permission to write to DAL unit 30 may make this modification.

3.66. RUN FREQ

RUN=FREQ Command (Obsolete - Use FREQUENCY/CENTER Command)

Purpose

This runstream is used to extract the standard 1/3 octave band center frequencies in the range FLO to FHI.

Command

```
RUN=FREQ N,NM,IVN FLO,FHI
```

Where

```
N,NM,IVN - Name of output vector to put 1/3 octave band frequencies.
FLO,FHI  - The 1/3 octave center frequency range desired.
```

3.67. FREQUENCY

FREQUENCY Command

Purpose

This command returns the values for the center frequencies, bandwidths and bandedges.

Command

```
FREQUENCY/CENTER  N1,NEL1,NVR1  F1,F2
or
FREQUENCY/WIDTH   N1,NEL1,NVR1  F1,F2
or
FREQUENCY/EDGE    N1,NEL1,NVR1  F1,F2
or
FREQUENCY/SET     (IFOP,(FO))
```

Where

N1,NEL1,NVR1 - Dal unit, element and version to output to.

F1,F2 - Frequency range.

IFOP - Frequency calculation option as follows:

IFOP	Description
1	Standard 1/3 octave frequencies. (Default)
2	Standard 1/6 octave frequencies.
3	Calculated frequencies based on 10**(N/10) where N is the band number. (FO needed.)
4	Calculated frequencies based on (2**(FO))**I where I is the number of frequencies from 1 Hz to the current frequency.
5	Calculated frequencies based on (2**(FO))**I where I is the number of frequencies from F1 to the current frequency.
6	Calculate constant bandwidth frequencies. FO is the bandwidth.

FO - Fractional octave. (1/3,1/6,1/10...) This is only needed for options 3, 4, and 5.

/CENTER Option

This option returns the center frequencies for the frequency range requested.

/WIDTH Option (BAND_WIDTH, BANDWIDTH)

This option returns the bandwidths for the center frequency range requested.

/EDGE Option (BAND_EDGE, BANDEDGE)

This option returns the bandedges for the center frequency range requested.

/SET Option

This sets the option requested.

If FO is omitted it will remain the same as it was.

If no arguments are input the current option will be displayed.

3.67.1. PROGRAM

PROGRAMMER'S information

There are four function subroutines that have been written to return for the calculation of frequencies. These functions are as follows:

SETNFR(F1,F2) - Calculates the number of frequency points from F1 to F2.
F1,F2 - The center frequencies for the range requested.

SETFRQ(SF,I) - Calculates the Ith center frequency.
SF - The starting center frequency.
I - A counter indicating what frequency band you want. If I=1 then SETFRQ=SF.

SETBWT(SF,I) - Calculates the band width for the Ith center frequency.
SF - The starting center frequency.
I - Counter. Same as above.

SETLBE(SF,I) - Calculates the lower band edge for the Ith center frequency.

SF - The starting center frequency.
I - Counter. Same as above.

Note

If it is desired to get the upper band edge at a given frequency, just use I+1 when the function is called.

It is hoped that all frequency calculations in VAPEPS be done using these functions as it will make it possible to add new frequency calculation methods in the future and allow for other than standard 1/3 octave frequency calculations to be made.

3.68. FUN

FUN Command

Purpose

Given an abscissa vector X, generate an ordinate vector Y as some function of X.

Ie. Y(I) = B*(FUNC(A*X(I))**C)

Where FUNC is ABS, COS, SIN, TAN, EXP, ALOG, COSH, SINH ACOS, ASIN, TANH, LOC, E10, L10, SKX or nothing.

Command

```
FUNX  NX,ELX,VRX  NY,ELY,VRY  A,B  IP,JP
```

Where

```
C           - FLOAT(IP/JP)
NX,ELX,VRX - Input vector or array.
NY,ELY,VRY - Output vector or array.
```

The X in FUNX may be replaced by ABS,COS,SIN,TAN,EXP,CSH, SNH,ACO,ASI,TNH,LOG,E10,L10 as indicated. The X will yield

```
Y(I) = B*((A*X(I))**C)
```

FUNCOS would yield

```
Y(I) = B*(COS(A*X(I))**C)
etc.
```

The following is a list of operations (assuming A=1.,B=1.,I=1,J=1)

FUN	Y=X	FUNE10	Y=10.**X
FUNX	Y=X	FUNL10	Y=ALOG10(X)
FUNCOS	Y=COS(X)	FUNCSH	Y=COSH(X)
FUNSIN	Y=SIN(X)	FUNSNH	Y=SINH(X)
FUNTAN	Y=TAN(X)	FUNACO	Y=ACOS(X)
FUNEXP	Y=EXP(X)	FUNASI	Y=ASIN(X)
FUNLOG	Y=ALOG(X)	FUNSKX	Y=SIN(X)/X

Trig functions expect input in radians, and inverse trig functions return values in radians. Please consult a FORTRAN manual for more information about the functions.

Notes

A and B must be floating point numbers. Thus a decimal point is required, or in the case of variables, real variables must be used. IP and JP are integers, and decimal points may not be used.

If A,B,IP,JP are to be 1, it is not necessary to include them. It is important to remember that B cannot be omitted if IP or JP is to be included,etc.

If X is input with more than one column, then the calculation is done for each column and the Y element will be the same size as X.

The output dataset has the same type as the input. Input array normally is single precision, but double precision is allowed on most machines (not Cray). If integer array is input, it is converted to real, then function is performed, then it is converted back to integer prior to output.

Example

If the element SYS/EVAL contains eigenvalues (omega**2), the following commands may be used to compute the corresponding natural frequencies (in hz.).

```
CALC
FAC = 1./PI$
DONE
FUNX  1,SYS,EVAL  1,NAT,FREQ  .25,&FAC,1,2
```

3.69. RUN GCLS

RUN=GCLS Programmer's Command

Purpose

Order command abbreviations and entry points of a DAL processor. this stream produces the element NOUT:CLST/NAME,ICYC,JCYC it reads 'ABV' and 'EPS' elements from unit NIN.

Command

```
RUN=GCLS  NAME  (,NIN,NOUT,ICYC,JCYC)
```

Where

```
NAME - Version name for ABV and EPS elements.  (Default = VAP)
NIN  - Input DAL unit. (Default = 1)
NOUT - Output DAL unit. (Default = 10)
ICYC,JCYC - Input cycle numbers.  (Default = 6,1)
```

Related Commands

RUN=BSC, RUN=ADDC

3.70. GCOL

GCOL Command

Purpose

GCOL reads a matrix by columns. Each column consists of a fixed number of groups. Each group is composed of a fixed number of words. The words may be integer, floating or hollerith in nature.

Command

```
GCOL  NU,ELN,VER  NGPC,MAXC,NWPG,NCPW
-DATA-
```

Where

```
NU,ELN,VER - Name of output element.
NGPC       - Number of groups per column (must be specified).
MAXC       - Maximum number of columns (defaults to 1000).
NWPG       - Number of words per group (defaults to 1).
NCPW       - Number of characters per word (defaults to 4).
           - 0 Implies integer or floating point words. else = Number
             of hollerith characters per word.
```

Notes

The use of GCOL essentially allows the user to input a three dimensional matrix. NGPC represents the length of each logical column. The resulting physical record length (NR) is the product NGPC*NWPG.

The default mode for GCOL is that each line of data causes a complete column to be written to the output element. If fewer than NGPC groups are read on a given line, remaining groups are zero (or blank) filled. NGPC may be input as a negative number to allow a column to extend across several records. In this case, make sure that the last record for each column contains exactly the correct number of groups to complete the column.

For hollerith input, a group is a string of characters that do not contain a space, comma, or equal sign. They may be any length, but only the first NWPG*NCPW characters are used. If you want to include a comma, space or equal sign in a particular group, enclose that group in single quotes.

Sometimes it is desirable to read hollerith information without regard for group separators (eg for titles,etc) In this case, use a negative number for NCPW (eg -4)

The output element will have NR = NGPC*NWPG
NJ = NGPC
NC = MAXC (unless EOF ends input prematurely)

```
The output element will have    NR = NGPC*NWPG
                                NJ = NGPC
                                NC = MAXC (unless EOF ends input
                                     prematurely)
```

A super special case exists where only one line of data is expected, but you don't know how many groups will be input on the line. Normally, the output element will contain as many groups as were specified on the command, with non- entered groups blank filled. If MAXC is set to -1, GCOL will truncate the element to the actual number of groups input. (See 30,RUN,INDE for an example)

3.71. RUN GDOC

RUN=GDOC Programmer's Command

Purpose

Prepares a DOC file, using unit NG as a general file and NH as the help unit. NTMP is a temporary Fortran unit and defaults to 2.

Command

```
RUN=GDOC  NG,NH,NTMP
```

Where

```
NG   - DAL unit containing General file. (Default = 10)
NH   - DAL unit containing Help file. (Default = 29)
NTMP - Temporary fortran unit.  (Default = 2)
```

3.72. RUN GENP

RUN=GENPLOT Command

Purpose

This runstream allows you to plot specified items of a matrix. (eg. certain channels out of a set of data)

Command

```
RUN=GENP
```

Input

This runstream prompts user for

1) Name of abscissa array.
2) Name of ordinate matrix.
3) Name of name vector -- must contain names of items in ordinate matrix (one name per ordinate column)
4) The items that you wan plotted.

3.73. RUN GETR

RUN=GETRESP Command

Purpose

This command gets the specified responses for a given SEMOD model.

Command

```
RUN=GETRESP  N1,IVR1  N2,IEL2,IVR2  (IOP)
```

Where

```
N1,IVR1       - The DAL unit and version name of your model.
N2,IEL2,IVR2  - DAL element to place output in.  One column for each
                response.
IOP           - The number of response elements that you wish to
                extract.  Excitation elements can be extracted as well
                as response elements.  If this parameter is omitted then
                up to 10 elements can be specified.
              = 'ALLR' all response elements are extracted. Excitation
                elements are not extracted.
              = 'ALLE' all excitation elements are extracted. Response
                elements are not extracted.
              = 'ACRO' all acoustic response elements.
              = 'ACEO' all acoustic excitation elements.
              = 'ACER' all acoustic excitation and response elements.
              = 'STRO' all structural response elements.
              = 'STEO' all structural excitation elements.
              = 'STER' all structural excitation and response elements.
```

3.74. GLOSSARY

GLOSSARY Command

Purpose

Compares input list (N1,EL1,VR1) with the glossary (N2,EL2,VR2). The variables 'M$' and 'N$' are returned indicating the number of matches and non-matches. The element N3,EL3,VR3 may be omitted, but if it is not, a vector will be written indicating the glossary numbers associated with each item on the input list.

Command

```
GLOSS   N1,EL1,VR1  N2,EL2,VR2  N3,EL3,VR3  NWPG
```

Where

```
N1,EL1,VR1 - Input list.
N2,EL2,VR2 - Input glossary.
N3,EL3,VR3 - List of glossary items.
NWPG       - Number of words per group (optional). Its default value
             depends on the value of NJ for input element 1.  If
             (NJ.EQ.1) NWPG=1. If (NJ.NE.1) NWPG=NI.
```

Notes

When the command GLOSS is used, the glossary is updated to include any new items encountered during processing. If the command GLOCHK is used instead, the glossary is not updated, and is accessed in a 'read-only' mode.

GLOSS is part of VAPEPS's word processing capability. It will normally be used in conjunction with GCOL and PREP. These will be documented in a special section. At present, this special section is not available.

3.75. RUN GLST

RUN=GLST Command

Purpose

Lists output data elements from RUN=GMOD.

Command

```
RUN=GLST NU,NAME NOUT I99F
```

Where

```
NU   - DAL unit number containing output elements from RUN=GMOD.
NAME - DAL version name associated with output elements from RUN=GMOD.
NOUT - Optional Fortran output unit.  If not specified, output is to the
       output device.  Otherwise output is to Fortran unit NOUT.
I99F - Non-zero value prints out parameters with a value of -99.
       Default is to not print out these parameters.
```

3.76. RUN GMOD

RUN=GMOD Command

Purpose

Retrieves data from a module in the master file.

Command

```
RUN=GMOD  MOD,NIN  NOUT,NAME  F1,F2 NAV
```

Where

MOD - Absolute module number as returned by SERCH.
NIN - DAL unit where actual event is located.
NOUT - DAL unit where information is to be written.
NAME - Version name to be associated with output data.
F1 - Low frequency value.
F2 - High frequency value.
NAV - Option code. If not specified, spectral data are averaged for output. If specified as a value other than zero, the data are not averaged.

Output Elements

NOUT,'PARA',NAME - Parameter matrix (30,6)
NOUT,'DATA',NAME - Averaged spectral data (NFRQ,6)
NOUT,'FREQ',NAME - Frequency matrix (1/3 octave) (NFRQ,3)

Note

Each column of the output data matrix contains the averaged measurements for all channels in that section. Section 1 corresponds to EXTA, 2 to SKIN, etc. Data values are in dB.

If output spectral data are not averaged, NOUT,'DATA',NAME is replaced by up to six elements, one for each sea element which has channels attached to it in the module. Element names are:

NOUT,SEANAM,NAME

Where SEANAM is replaced by the actual sea element name in each case (Eg., NOUT,'SKIN',NAME).

3.77. RUN GPSD

RUN=GPSD Command

Purpose

Get PSD from time history. This runstream calculates the power spectral density in the frequency domain of a function in the time domain.

Command

```
RUN=GPSD   NUI,NVI  NUO,NVO  NPRT
```

Where

```
NUI,NVI - DAL unit number and version name for input data.
NUO,NVO - DAL unit number and version name for output data.
NPRT    - +1 if rms values and analysis frequency bandwidth are to be
          printed, 0 if not.  (Default = 1)
```

Input element names (&NUI,XXXX,&NVI)

```
1) TIME - Vector of N equally spaced time points corresponding to function
          data points, seconds (N,1)
          Note - Number of time points N must equal an integer power of 2,
          I.E. N=2**M, M=1,2,3,...,etc.
2) DATA - Matrix of function data points corresponding to time vector time
          points, usually in g's (N equal functions)
```

Output elements (&NOU,XXXX,&NVO)

```
1) PSD  - Matrix of power spectral densities in frequency domain, usually
          g**2/Hz (N/2+1,=functions)
2) HZ   - Vector N/2+1 equally spaced frequency points corresponding to
          PSD row values, Hz (N/2+1,1)
3) RMS  - Vector of RMS values of each function, usually g's,
          (=functions,1)
```

Comments

The method of computation used is as follows:

1) The input time history functions in the data matrix are multiplied by a 10 per cent cosine taper window. (Ref. Bendat and Piersol 'Random Data: Analysis and Measurement Procedures', pp. 322,327
2) The VAPEPS FFT routine is used to take the Fourier Transform of the modified data set.

3) The square of the Fourier transform moduli are computed and multiplied by (2./.875*T) to give the power spectral densities.

3.78. GREAD

GREAD Command

Purpose

GREAD is used to read card image data into a DAL element. It is similar to ECOL except for the following:

1) Each column need not start on a separate line.

2) You specify a 'pseudo format'. This allows you to skip portions of each data card. It may also be used to separate two fields that run together. Eg if two I5 type numbers appear as 1234554321 (12345 and 54321), ECOL would see only one number whereas the pseudo format of 5,5 would limit each field to 5 characters.

Command

```
GREAD  N1,EL1,VR1  NR,NC  ITYP
- PSEUDO FORMAT -
- DATA CARDS -
```

Where

N1,EL1,VR1 - Name of data set to be created.

NR,NC - Desired number of rows and columns. NC may be input larger than you expect to read. NC may also be input as a negative number, which fills the matrix a row at a time, rather than the default column at a time.

ITYP - Desired type of output matrix. If omitted, the data type is set automatically. Allowable types are:

0 = Integer
1 = Single precision floating
2 = Double precision floating
3 = Complex
4 = Hollerith

Pseudo Format
- List of integers. Each integer represents a field length on the input card images. Consider the following examples:
= 80 (eg, just one number input) GREAD does a free-field read for the first 80 characters on each card. Commas and/or blanks are used to separate numbers.
= 16*5 (or 5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5) GREAD again uses 80 characters, however it internally places a blank

character (see notes) between every 5th character. Now the free field reader will separate each 5 character field into separate words.

= -12,68 Negative values indicate fields that are to be skipped. Thus the first 12 characters on each card will be ignored. The next 68 characters will be interpreted according to DALPRO free-field format rules.

Data Cards

This section contains the actual data to be read. Several possibilities exist.

1) Data immediately follows the command in the runstream.

2) You can access a FORTRAN file via transparent commands -

```
##READ 1,0     (Read FORTRAN unit 1, return EOF when done)
```

3) You can access a DAL symbolic element (type 5 or 6) via

```
##READ 1,MY,DATA 0  (The trailing 0 again requests EOF return when
                     done)
```

4) The ##READ feature of DALPRO can be used to full advantage by preceeding your reads with

```
##READ 1  DATA    or    ##READ 1,MY,DATA  DATA
```

The trailing DATA identifies the unit (or dataset) as the data unit. Once identified, you can read the unit in sections via

```
*
* READ NUMBER OF ROWS, COLUMNS  (for example)
*
GREAD  1,NUM,RC  2,1
2*5  (eg 2I5 )
##READ  DATA  AUTO
*
* NOW RECOVER NR AND NC
*
##SET  NR=1,NUM,RC,1,1    (get NR from row 1, column 1)
##SET  NC=1,NUM,RC,2,1    (get NC from row 2, column 1)
*
*  AND READ THE ACTUAL MATRIX
*
GREAD  1,MAT,RIX  &NR,&NC
##READ  DATA  AUTO
```

Here the AUTO's imply that there is more on the DATA unit than will be read by GREAD. DALPRO will automatically stop reading from the alternate source as soon as GREAD has read NR*NC words. It will the DATA unit open to the next card, so that subsequent GREAD's or whatever may take off from where this read stopped. (Note: The DATA and AUTO technique is not limited to use with GREAD)

Notes

In the normal mode, GREAD will separate your input line into fields, by inserting a blank character between the fields. In some cases you may have blank fields. Thus the free field reader will not pick up the blank field. You can force it to pick up the blank field by using the command GREADC in which case GREAD will insert commas instead of spaces. Note that a blank field yields a word of all blanks (NOT A ZERO).

3.79. RUN GRLT

RUN=GLRT Command

Purpose

Updates the READ ELTS element in the EASY file.

Command

```
RUN=GRLT  NU,IC1,IC2  JOUT,JOT2
```

Where

```
NU         - Dal unit assigned to the EASY file.
IC1,IC2    - Cycles to be read.
JOUT,JOT2 - Cycles of output READ ELTS.
```

3.80. RUN GRMS

RUN=GRMS Command

Purpose

This program computes the Overall GRMS value of a spec given the corner points of the spec or the slope and corner points. It also allows you to modify a spec up or down by a constant dB value.

Command

```
RUN=GRMS (N1,VR1 (N2(,NOUT))))
```

Where

```
N1,VR1 - Output dal unit and version for results.
         N1,'DATA',VR1 contains all intermediate data.
         Column 1 and 2 contain starting and ending frequency for each
         segment.
         Column 3 and 4 contain starting and ending G**2 values
         Column 5 contains the slope of the segment.
         Column 6 contains the area in G**2/HZ for each segment.
         N1,'GRMS',VR1 contains the overall level
         NOTE:  For N1 < 0 an already existing data set created by
         RUN=GRMS is expected for input.  This allows you to obtain a
         listing, plots or modify the spec up or down by a constant dB.
N2     - Maximum number of corner points in curve.  (default = 20)
NOUT   - Fortran unit for printed output.  Default is 6.
```

Notes

This runstream is fully interactive and prompts for all input. NOTE: To obtain overall levels for 1/3 octave data use RUN=OVER

Equations

This program is based on the following.

For a given line on a log-log plot.

```
     y = a*x**b
     a = the y intercept
     b = the slope     =  log(y2/y1)/log(x2/x1)
     a = y/x**b
  Area = integral of a*x**b = a*/(b+1)*(x2**(b+1)-x1**(b+1))
  GRMS = summation of the areas**.5
```

```
SLOPE(dB/OCTAVE)=10*LOG10(G2/G1)/(LOG(F2/F1)/LOG(2))
```

3.81. GROUP

GROUP Command

Purpose

Build one element from several other elements. This is a simplified alternative to BUILD where input elements are stacked one after the other rather than being overlayed. It was designed for use with plotting several curves on one plot but has applications in other areas as well.

Command

```
GROUP  N1,EL1,VR1  (N2,EL2,VR2)  (MAXG(,MAXC))
```

Where

```
N1,EL1,VR1 - Name of element to be created.
N2,EL2,VR2 - Name of optional output group directive vector (for plots)
MAXG       - Maximum number of groups per output column (default=10)
MAXC       - Maximum number of output columns (default=10)
-DATA-     - Input data cards which provide the names of the input
             elements to be included. These cards take the following
             form:

                M1,IL1,IV1  (JC1(,IC1(,ID1)))
                M2,IL2,IV2  (JC2(,IC2(,ID2)))
                    .
                    .
                MN,ILN,IVN  (JCN(,ICN(,IDN)))
                DONE

                Where

                   MI,ILI,IVI - Name of ith input element.
                   JCI        - Column in output matrix where leftmost
                                column of ith input matrix should be
                                placed. (default=1)  A dollar sign may
                                be used to indicate the next empty
                                column.
                   ICI        - Input column number to be included (
                                default = 0, which implies that all
                                columns are to be included)
                   IDI        - Group descriptor for this element
                                (default = 1)
```

Notes

As each input data card is read, GROUP keeps track of the number of output columns and the number and length of groups in each column. Every time an input column is placed in a given output column, the next available row position is used as the starting location.

The output group directive vector always corresponds to the groups in column 1 of the output matrix. Three numbers are output per group: I1,I2,ID. I1 is the starting row location for the group. I2 is the last row position for the group. ID is the group descriptor.

Uses

Plotting

Suppose you have 3 elements that contain xy pairs that are to be plotted on the same graph: (XY,ONE XY,TWO XY,THRE). Also assume that each has two columns but different lengths.

```
GROUP  1,XY,TOT  1,DIR,TOT
1,XY,ONE  1,0,5
1,XY,TWO  1,0,5
1,XY,THRE 1,0,5
DONE
```

would create 1,XY,TOT as a two column matrix with points from ONE followed by those from TWO, etc. They may be plotted using:

```
PLOM03  1,XY,TOT  1,XY,TOT  0,2
= 1,DIR,TOT
```

Note that ID=5 causes each plot group to be connected with every fifth point marked.

General

Suppose you have N vectors that are separate elements. They have the same name except for C1, which are numbered from 1 to N. It is desired to put these in one matrix, with each vector becoming a separate column.

First create a loop element for the GROUP data cards:

```
SYMIN  28,GROP,LOOP
##INC  IC1,1
##CYCLE  &IC1,0
1,INPT,VEC  &IC1
SEOF
```

Now initialize IC1 and start GROUP

```
SET  IC1=0
GROUP  1,FULL,MAT  &N
##READ 28,GROP,LOOP &N
DONE
CYCLE 0,0
```

3.82. RUN GVAL

RUN=GVAL Command

Purpose

This command looks up values in the distribution named distribution table.

Command

```
RUN=GVAL  NAME=NTAB N,CONF
```

Where

```
NAME - Name of variable to store result.
NTAB - Name of the distribution table.
N    - Sample size.
CONF - Confidence level.
```

Example

Get the 95th percentile CHI squared distribution for a sample size of 15.

```
GVAL  XXX=CHIS 15,.95
```

3.83. RUN HEAD

RUN=HEAD Command

Purpose

Print the header and channel information section for a single section of an ENTERed event.

Command

```
RUN=HEAD  NU,VENT,ISEC
```

Where

```
NU,VENT - DAL unit and event name.
ISEC    - Section of ENTERed event which is to be printed.
```

Related Commands

RUN=ESAV

3.84. RUN INDE

RUN=INDEX Command

Purpose

Provides a way of locating commands that server a given function.

Command

```
RUN=INDEX
```

Notes

For instance if you are looking for commands that perform multiplication, use RUN=INDEX, and enter the string MULTIPLICATION when it asks for items. It will give you a list of VAPEPS commands that are concerned with multiplication.

3.85. INDEX

INDEX Command

Purpose

Indexing scheme where data values from a matrix are compared with an index vector. If the value is found in the index, then the value in the matrix is replaced with the index identifier.

Command

```
INDEX N1,EL1,VR1  N2,EL2,VR2  N3,EL3,VR3  (N4,EL4,VR4)  (NOFIND)
or
INDX  N1,EL1,VR1  N2,EL2,VR2  N3,EL3,VR3  (N4,EL4,VR4)  NOFIND
```

Where

```
N1,EL1,VR1 - Input matrix of data values A(NR,NC).
N2,EL2,VR2 - Input INDEX element B(MR,1) or B(MR,2).
N3,EL3,VR3 - Output modified data matrix C(NR,NC).
N4,EL4,VR4 - Optional output 'lump' matrix (MR,NC). This keeps track of
             how many times each index item is matched, on a per column
             basis.
NOFIND     - Optional 'no-match' number--see below.
```

Notes

Basically, each item in the first matrix is compared to the values contained in column 1 of the second element. If an item matches a value, then the item is reset either to the index row number of the match or to the contents of the second column of the index element for that row. If a no-match condition occurs, the value is either left unchanged or set to a NOFIND value. The rules are discussed below. The variable N$ is initially set to zero. It is incremented by one for each no-match condition. This variable may be used after the command to determine the success of the INDEX operation.

```
Each item A(I,J) is compared with the index items in first column of B. If
   1) A(I,J) = B(K,1) then
      a) If B has two columns then C(I,J) is set to B(K,2)
      b) If B has one column, then C(I,J) is set to K
   2) A(I,J) does not match any B(K,1), then
      a) If NOFIND is present, C(I,J) is set to NOFIND.
      b) If NOFIND was not included on the command, C(I,J) = A(I,J).
      c) If NOFIND is present on the command and the command was issued as
         INDX instead of INDEX, and the input index element has two
```

columns, the C(I,J) = B(NOFIND,2).

Examples

If you have a matrix where each column refers to some item, eg. J1, J5, X15, B47, X10,... and you want to be able to select a group of columns by name, then

a) Construct the index vector

```
ECOL  1,IND,EX   10,1    (i.e. matrix has ten columns)
J1,J5,X15,B47,X10,...
```

b) Create a name vector for each group of columns you want:

```
ECOL  1,GRP,ONE  3,1
J1,X15,J5
```

c) INDEX the list

```
INDEX  1,GRP,ONE  1,IND,EX  1,CHOP,ONE  0
```

d) See if any names didn't match

```
##JEQZ  &N$ OKAY
##ON COMMENT
$$*  WARNING, &N$(I3) ITEMS WEREN'T FOUND
* THEY WILL BE SET TO ZERO.
>>OKAY
```

e) Use CHOPX to chop out desired values from matrix

```
CHOPX  1,ORIG,MAT  1,SEL,ONE  -&NROW,3
= 1,CHOP,ONE
```

The use of NOFIND=0 will cause any non-existent names to result in zeros in the output selected matrix.

If you want to change selected values within a matrix, ie. change all ones to fives, threes to twos, etc

```
ECOL   1,CHNG,SPEC  2,2
1,3
5,2
INDEX  1,ORIG,MAT  1,CHNG,SPEC  1,CORR,MAT
```

Special case -- NJ,MJ

In general, the first element will have size NR,NC and the second element will have size MR,MC, where MC = 1 or 2. All DAL elements may be further partitioned such that each column is considered a collection of logical blocks. The number of logical blocks per column is NJ. Normally NJ = 1. INDEX checks NJ for both input elements, and computes the quantities NI=NR/NJ and MI=MR/MJ. If NI=MI and MJ.NE.1, then INDEX assumes that the user really wants to consider NI words at a time in doing the indexing. The result is straight forward except for the following notes. For this case, MC must = 2. If NOFIND is specified, it must be used with the INDX form of the command, and it will always imply that non-matches should be set to the NOFINDth logical record in the second element.

3.86. INTERPOLATE

INTERPOLATE Command

Purpose

Given X,Y, and X', find Y' using LINEAR , LAGRANGIAN, CUBIC SPLINE, or SINE interpolation.

Command

```
INTERPOLATE  N1,EL1,VR1  N2,EL2,VR2  N3,EL3,VR3  N4,EL4,VR4  NTYPE
```

Where

```
N1,EL1,VR1 - Original abscissa vector or matrix (X)  (*)
N2,EL2,VR2 - Original ordinate vector or matrix  (Y)
N3,EL3,VR3 - Desired abscissa vector or matrix (XP)  (*)
N4,EL4,VR4 - Output ordinate matrix or vector    (YP)
NTYPE      - Type of interpolation.
           = 2 or omitted, LINEAR
           = 6, 6 point LAGRANGIAN
           = -2, SINE interpolation
           =  1, CUBIC SPLINE - Parabolic end condition
           = -1, CUBIC SPLINE - Linear end condition
```

Notes

Normally X and XP must be vectors. Either may be a matrix as long as all other input datasets are vectors. If Y is a matrix then each column is considered as a separate ordinate vector with X as the common abscissa. The output dataset YP will have a many columns as Y (or X or XP, whichever is input as a matrix) and as many rows as XP. X and Y must have the same number of rows. X must be monotonically increasing.

(The command INTERPOLATE/TRANSPOSE may be used instead of INTERPOLATE. In this case, the X and XP datasets must be vectors. Y may be a matrix, and must have as many columns as X has rows. In this case, YP will be output with as many rows as Y, and as many columns as XP has rows.)

The following rules govern points in XP not covered by points in X.

```
IF(XP(I).LE.X(1)) THEN YP(I,J) = Y(1,J)
IF(XP(I).GE.X(N)) THEN YP(I,J) = Y(N,J)
```

where N is the number of rows in X and Y

Special case

If you have an array which has times as the odd columns, and values as the even columns, leave off N1,EL1,VR1 and input only three element names. The only restriction on this is that if some time-value pairs do not have the full number of points as the rest, make sure the last valid abscissa value for that pair is followed by a zero. (This method is not legal for the Transpose mode)

3.87. RUN INTE

RUN=INTE (Programmer's Command)

Purpose

Provides a way to interactively obtain input that is not passed in the argument list of a runstream.

Command

See RUN=FREQ, RUN=BWTH, RUN=NORM, or RUN=STTS for examples of it's use.

3.88. INVERT

INVERT Command

Purpose

Matrix inversion (single precision or double precision or complex).

Command

```
INVERT  N1,EL1,VR1  N2,EL2,VR2
```

Where

```
N1,EL1,VR1 = Element to be inverted.
N2,EL2,VR2 = Inverse.
```

Notes

The input element must be square. Since the array is loaded into core for the inversion, difficulty may arise with very large matrices.

The type of the matrix is checked prior to inversion. The type of the output inverse will be the same as the input.

For single and double precision mats, the variable DEF$ is set to the value of the determinant. For complex, DTR$ and DTI$ give the real and imaginary parts to the determinant respectively. If the exponent of the determinant exceeds that which is allowed for VAPEPS variables, it is set to zero, and a warning is printed.

3.89. JPI

JPI Command

Purpose

Provide job process information for the user. This command is machine dependent. On the UNIVAC, it accesses the PCT table and returns one or more values. On the VAX, it uses the system service routine SYS$GET_JPI.

Command

```
JPI  ITEM
```

Where

On UNISYS (aka: SPERRY, UNIVAC)
ITEM - An offset from the start of the PCT table. Consult UNISYS manuals for a definition of offsets.

On VAX
ITEM - The value of $JPIDEF item. See the VAX system services manual for values and what they represent. Item 1034 is the page-fault count and can be monitored to see how badly the execution is thrashing.

The contents of the items will be printed in octal.

Alternate form for UNISYS usage

```
JPI  N1,EL1,VR1  LENG,IOFF
```

Will cause LENG values to be taken from the PCT table (words IOFF to IOFF+LENG-1) and written to the element N1,EL1,VR1. Note that the first item in the PCT has IOFF=0.

3.90. RUN JTOD

RUN=JTOD Command

Purpose

This command is converts joint numbers to the appropriate degree of freedom numbers.

Command

```
RUN=JTOD  N1,NEL1,NVR1  N2,NEL2,NVR2  ID1(,ID2(,ID3(,ID4(,ID5(,ID6)))))
```

Where

```
N1,NEL1,NVR1 - Input list of joint numbers.
N2,NEL2,NVR2 - Output list of degree of freedom numbers.
ID1,...,ID6  - Degrees of freedom desired at each joint.
```

The following variable can be preset if needed.

IDF$ - Actual number of degrees of freedom per joint. Default=6

Example

Compute the degree of freedom numbers for the rotational degrees of freedom of joints 1,5,10,70 for a model with 6 degrees of freedom per joint.

```
ECOL 28,JOIN,TS 4,1
1,5,10,70
RUN=JTOD 28,JOIN,TS 28,DOF,VEC 4,5,6
```

3.91. LAST

LAST Command

Purpose

Print the last table of contents line in a DAL file.

Command

```
LAST  NU
```

Where

NU = DAL unit

Notes

LAST is most often used to find the number of elements in a DAL file.

The type of TOC line written depends on the current TTYP (as set by the command TTYP).

3.92. RUN LFU

RUN=LFU Command

Purpose

Create a lower triangular, upper triangular or full matrix from a lower triangular, upper triangular or full matrix.

Command

```
RUN=LFU  N1,EL1,VR1  N2,EL2,VR2  ILFU,JLFU
```

Where

```
N1,EL1,VR1 - Input square matrix.
N2,EL2,VR2 - Output square matrix.
ILFU       - Input matrix type.
           = -1  Input matrix is lower triangular.
           =  0  Input matrix is full.
           =  1  Input matrix is upper triangular.
JLFU       - Output matrix type.
           = -1  Output matrix will be lower triangular.
           =  0  Output matrix will be full.
           =  1  Output matrix will be upper triangular.
```

3.93. LOCATE

LOCATE Command

Purpose

To print a TOC line, change it, or write the TOC line to an element as desired.

Command

```
LOCATE  N1,EL1,VR1  N2,EL2,VR2
or
LOCH  N1,EL1,VR1  N2,EL2,VR2
```

Where

N1,EL1,VR1 - Input DAL element.
N2,EL2,VR2 - Optional output DAL element for TOC line.

Notes

Cycles for the element to be located are taken from the current input cycles. If the second element is specified, it will be written as a vector of length 13. Since it is obtained directly from the TOC, items 7 and 8 are NJ, NINJ irregardless of the value of TOCTYP.

The command LOCH locates the element, and then prompts the user to enter changes to the TOC line. Changes start with the value of NWDS, and continue through CYC2. If a particular item is not to be changed, use a single '$' in its place.

Example

Change the value of NJ and ITYP for the element TEST/MAT[5,5] on unit 3

```
CYCLE 5,5
LOCH  3,TEST,MAT
$,&NEWJ,$,&NTYP
```

LOCH lets you change any or all four element names. This can be dangerous, since it does not check to see if the new names are unique.

The preferred method for changing element names, type codes, and NJ values is through use of the CHANGE command, as described earlier in this manual.

3.94. LOOK

LOOK processor

Purpose

This processor is intended to provide a quick way of looking at a data set via listing or plotting in a meaningful fashion on a terminal.

Command

```
RUN=LOOK  N1,NEL1,NVR1  N2,NVR2  N3,NEL3,NVR3  N4,NEL4,NVR4
```

Where

```
N1,NEL1,NVR1 - Dal unit version and element name of data to be  worked
               on.
N2,NVR2      - Dal unit and version name to keep working files.
N3,NEL3,NVR3 - Range of values for each row.  (eg. Frequency range.)
N4,NEL4,NVR4 - Range of values for each column.  (eg. Channel names.)
```

3.94.1. EXIT

EXIT command

Purpose

This command exits the LOOK processor.

Command

```
EXIT
```

3.94.2. HELP

HELP command

Purpose

This command provides help for the LOOK processor.

Command

```
HELP
```

3.94.3. LIST

LIST command

Purpose

This command displays 1 screen of the input data set.

Command

```
LIST
```

Notes

This command will display the portion of the matrix starting at the location specified by the WINDOW command. The WINDOW command specifies the upper right hand corner of the block of data to be displayed. LIST will then display up to 16 rows and 6 columns of the data. The default location for window is 1,1.

3.94.4. MENU

LOOK Processor menu

LOOK processor

Misc.	Tables	Plotting	Choping	IO
EXIT	LIST	PLOT	SELECT	READ
HELP	WINDOW	PLOT LABELS		WRITE
STATUS	TITLES	PLOT RANGE		
		PLOT TYPE		
		TITLES		

3.94.5. PLOT

PLOT command

Purpose

This command is used to plot the entire data set on one plot frame or plot a specific column of the data set.

Command

```
PLOT (NC)
```

Where

NC - The single column to be plotted. If omitted the entire data set is plotted.

Note

If there are a lot of columns in your data set plot may not be able to handle them all and could cause an error. As a rule of thumb don't use more than 20 columns per plot.

3.94.5.1. LABELS

PLOT LABELS command

Purpose

This command is used to input plot labels.

Command

```
PLOT LABELS
```

The program prompts for input of two plot labels (1 for each axis.)

3.94.5.2. RANGE

PLOT RANGE command

Purpose

This command allows the input of ranges for the data to be plotted.

Command

```
PLOT RANGE
```

The program prompts for input of a range for the X and Y axes. If a value of 0.,0. is input then automatic scaling will be done for that axis. Default is 0.,0. for each axis.

3.94.5.3. TYPE

PLOT TYPE command

Purpose

This command allows the specification of a plot type. The two types allowed are DB and ACCL.

Command

```
PLOT TYPE=IPTYP

Where

   IPTYP - The plot type.  The valid values are:
         = DB   - Sets the plot to Linear Y and Logarithmic X.
         = ACCL - Sets the plot to Logarithmic Y and Logarithmic X.
                  Default is ACCL.
```

3.94.6. READ

READ command

Purpose

This command is used to read a new data set in.

Command

```
READ NEL
or
READ NU,NVR
or
READ NU,NEL,NVR

Where

   CASE I -- 1 argument.

   NEL - The data set N2,NEL,NVR2 replaces your current working data set.
         See the command line input for N2 and NVR2
         The new data set must have the same dimensions as the old data
         set.

   CASE II -- 2 arguments.

   NU,NVR - The following data sets replace your current working set.
            This data set could have been created with the WRITE command.
               NU,'DATA',NVR - Data set.
               NU,'ROW ',NVR - Frequency range.
               NU,'COL ',NVR - Channel names.

   CASE III -- 3 arguments.

   NU,NEL,NVR - This data set replaces your current working data set. The
```

new data set must have the same dimensions as the old data set.

3.94.7. SELECT

SELECT command

Purpose

This command is used to select a subset of the current working file to replace the current working file. Similar to the CHOP command.

Command

```
SELECT
```

The command prompts for the number of rows and columns to be selected. Then prompts for individual row and column numbers. These numbers can be input using the free-field input rules. (eg. 1:10)

3.94.8. STATUS

STATUS command

Purpose

This command displays the status of the LOOK processor.

Command

```
STATUS
```

The following items are displayed:

Default working DAL unit and version name.
Title lines.
Labels.
Plot type.
Ranges for plotting.
Window status.

3.94.9. TITLES

TITLES command

Purpose

This command allows the input of three title lines for use in plotting and the LIST command.

Command

```
TITLES
```

The program then prompts for three title lines.

3.94.10. WINDOW

WINDOW command

Purpose

This command sets the upper left hand corner location for use in the LIST command.

Command

```
WINDOW IWR,IWC
```

Where

IWR,IWC - Is the upper left hand corner of the portion of the matrix to be displayed using the LIST command.

3.94.11. WRITE

WRITE command

Purpose

This command is used to save a data set in a DAL file.

Command

```
WRITE NEL
or
WRITE NU,NVR
or
WRITE NU,NEL,NVR
```

Where

CASE I -- 1 argument.

NEL - The data set N2,NEL,NVR2 created.

CASE II -- 2 arguments.

NU,NVR - The following data sets are created. This data can now be read by the READ command.

```
NU,'DATA',NVR - Data set.
NU,'ROW ',NVR - Frequency range.
NU,'COL ',NVR - Channel names.
```

CASE III -- 3 arguments.

NU,NEL,NVR - This data set is created.

3.95. RUN LOOK

RUN=LOOK Processor

Purpose

This command is used to look at a data set, give it a range of rows and a range of columns, and give it a default working dal unit and version name. A set of commands are then set up to work on this data set.

Command

This command is documented in the main help section. See LOOK for more information.

3.96. RUN LRMS

RUN=LRMS Command

Purpose

This runstream will calculate a root-mean-square (rms) value for a function whose log-log plot is piecewise linear (see example plot).

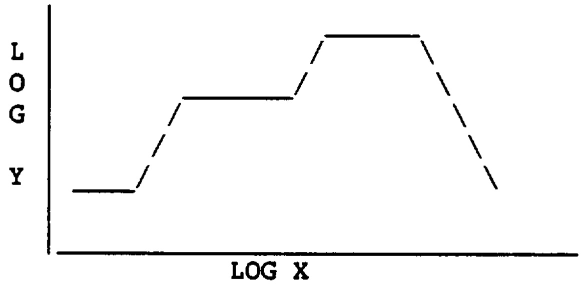
Command

```
RUN=LRMS N1,EL1,VR1  N2,EL2,VR2  N3,VR3
```

Where

```
N1,EL1,VR1      - Input abscissa vector.
N2,EL2,VR2      - Input ordinate vector.
N3,VR3          - Output unit and element name. The following elements
                  are output:
N3,VR3,'AREA'   - Area of each segment (segments X 1).
N3,VR3,'TOTA'   - Total area under curve (1 X 1).
```

Example plot



Notes

The first input vector contains the abscissa coordinates of the end points of the linear segments (i.e. frequency points). The second vector contains the corresponding ordinate values (i.e. mean square spectral density).

The output will consist of two vectors. the one named 'AREA' will contain the areas underneath the individual linear segments. The one named 'TOTA' will contain the RMS value for the entire function.

Related Commands

RUN=GRMS, RUN=OVER

3.97. RUN LTRP

RUN=LTRP Command

Purpose

This runstream interpolates an input spec level to create an element containing the spec levels at standard 1/3-octave center frequencies. It is assumed that the spec level was derived from 1/3 octave vibration data and so was not really intended to be a continuous spectrum, so no integration is done, only interpolation.

Command

```
RUN=LTRP  N1,L1,V1  N2,L2,V2  N3,L3,V3  F1,F2
```

Where

```
N1,L1,V1 - This element contains the spec to be interpolated.
N2,L2,V2 - This element contains the frequencies that go with the
           values in N1,L1,V1.
N3,L3,V3 - This is the output element.  A frequency element,
           N3,'FREQ',V3, is also produced.
F1,F2    - Values will be interpolated at standard 1/3 octave center
           frequencies between F1 and F2 inclusive.
```

3.98. RUN MANU

RUN=MANUAL Command

Purpose

This command interrogate the VAPEPS HELP and RUN files to produce a VAPEPS user's Manual.

Command

```
RUN=MANUAL NLPP (ITYP)
```

Where

```
    NLPP - Is the number of lines per page.
    ITYP - Is one of the following:
         = XEROX - Generates output suitable for a XEROX printer. If this
           option is omitted no special formating will be done.
```

Notes

Output is directed to Fortran unit 6. THE MANUAL IS ABOUT 600 PAGES SO USE THIS COMMAND WITH CARE.

3.99. MAXMIN

MAXMIN Command

Purpose

Get maximum and minimum values for a set of time functions. Times of occurrence may also be saved.

Command

```
MAXMIN  N1,EL1,VR1  N2,EL2,VR2  N3,EL3,VR3
```

Where

```
N1,EL1,VR1 - Input array. Max-min values are found for each column.
N2,EL2,VR2 - Output max-min array. Output number of rows = input number
             of columns. Number of cols is either 2 or 4, depending on
             whether times are saved.
N3,EL3,VR3 - Optional input time vector. If input, times are read and
             saved on output array.
```

Notes

If input times are not included (ie. 7 Words in command), the output array will have two columns, maximums and minimums. If input time vector is included in command (ie. 10 Words in command), then output array will have four columns: max,max times,min,min times. If input, the time vector must have as least as many rows as the input array.

MAXABS

If absolute value max-mins are desired, use MAXABS instead of MAXMIN. The format is the same except tests are made on absolute values. Output max-mins reflect input signs.

MAXT

Since the output from many response programs is stored as one column per time point (ie the transpose of what MAXMIN needs), MAXMIN has been modified to operate on this type of matrix. To initiate this mode, use the command

```
MAXT  N1,EL1,VR1  N2,EL2,VR2  N3,EL3,VR3
```

where maxmins are found on each ROW of N1,EL1,VR1, and the number of rows output for N2,EL2,VR2 will equal the number of rows of N1,EL1,VR1. Note that if times are to be included, the third element must still be a vector, but its length must be at least as big as the number of columns of N1,EL1,VR1.

This transpose feature can also be used with MAXABS, by using the command MAXAT, or MAXTA

3.100. MDIFF

MDIFF Command

Purpose

Differentiate an ordered function using either straight line techniques or by first fitting it with a cubic spline.

Command

```
MDIFF  N1,EL1,VR1  N2,EL2,VR2  N3,EL3,VR3  IOP
```

Where

```
N1,EL1,VR1 - Input abscissa vector or array ie (X) Each abscissa vector
             must be monotonically increasing.
N2,EL2,VR2 - Input ordinate vector or array ie (Y)
N3,EL3,VR3 - Output vector or array of derivatives ie (A)
IOP        - Differentiation type
           =-1   - (or omitted) use straight line method
           = 0   - (or omitted) use IOP=-1
           > 0   - Cubic spline, IOP determines end condition
           = 1   - Linear end condition
           = 2   - Parabolic end condition
           = 3   - Extrapolated end condition (dangerous)
```

Notes

MDIFF operates in normal and transposed modes. For the normal mode, the number of rows of X must equal the number of rows of Y. The number of columns in X must either = 1, or equal the number of columns in Y. Each column in Y is treated as a separate function.

For the transpose mode, X must be a vector. The number of rows in X must equal the number of columns in Y. Ie for a time function, each column in Y corresponds to a time point and each row of Y corresponds to a separate function. Note that the number of rows in X must not equal the number of rows in Y, or else the normal mode (above) will be used.

In either mode, the output matrix (A) will be the same size as the matrix Y. In the normal mode A(I,J) will be the derivative of the Jth function at the Ith point. In the transpose mode, A(I,J) will be the derivative of the Ith function at the Jth point.

Transposed differentiation will always be done using straight line techniques.

3.101. RUN MEZ

RUN=MEZ Programmer's Command

Purpose

This streams reads the files generated by the MEZ$ option of EZOUT. Its purpose is to construct a matrix which relates routine names to C=BLOCK references.

Command

```
RUN=MEZ NUNIT,NAME
```

Where

NUNIT - Is the Fortran unit that EZOUT output was directed to.
NAME - Is the version name for the output DAL file.

3.102. MGRATE

MGRATE Command

Purpose

Integrate an ordered function using either trapezoidal method or by first fitting it with a cubic spline.

Command

```
MGRATE  N1,EL1,VR1  N2,EL2,VR2  N3,EL3,VR3  IOP
```

Where

```
N1,EL1,VR1 - Input abscissa vector or array ie (X) Each abscissa vector
             must be monotonically increasing.
N2,EL2,VR2 - Input ordinate vector or array ie (Y)
N3,EL3,VR3 - Output integral vector or array ie (A)
IOP        - Integration type
           =-1   - (or omitted) use trapezoidal method
           = 0   - (or omitted) use IOP=-1
           > 0   - Cubic spline, IOP determines end condition
           = 0   - (or omitted) use IOP=2
           = 1   - Linear end condition
           = 2   - Parabolic end condition
           = 3   - Extrapolated end condition (dangerous)
```

Notes

MGRATE operates in normal and transposed modes. For the normal mode, the number of rows of X must equal the number of rows of Y. The number of columns in X must either = 1, or equal the number of columns in Y. Each column in Y is treated as a separate integral.

For the transpose mode, X must be a vector. The number of rows in X must equal the number of columns in Y. Ie for a time function, each column in Y corresponds to a time point and each row of Y corresponds to a separate function. Note that the number of rows in X must not equal the number of rows in Y, or else the normal mode (above) will be used.

In either mode, the output matrix (A) will be the same size as the matrix Y. In the normal mode A(I,J) will be the integral of the Jth function at the Ith point. In the transpose mode, A(I,J) will be the integral of the Ith function at the Jth point.

Transposed integration will always be done using trapezoidal method.

3.103. MODIFY

MODIFY Command

Purpose

MODIFY the contents of an element by specifying the row, column and new value for each change.

Command

```
MODIFY  N1,EL1,VR1  N2,EL2,VR2  MAX
-DATA-
```

Where

```
N1,EL1,VR1 - Input element (real, integer or hollerith)
N2,EL2,VR2 - Optional output element. If omitted, the input element
             will be corrected in place.
MAX        - Combination size parameter/option flag
           > 0   - Maximum changes per line - Mode 1
           < 0   - Total number of changes  - Mode 2
```

Notes

Mode 1

Modify the element by entering as many combinations of changes as desired. The general format is:

```
IROW,ICOL,VALUE,IROW,ICOL,VALUE,...
```

As many as MAX sets may be entered per line. Max defaults to 10 (ie 30 numbers). Cards will be read until a blank card is read or until a card with fewer than three items is read.

Mode 2

In this case, -MAX indicates the total number of changes that will be made. This time the data will be read as a list of row numbers followed by a list of column numbers, followed by a list of new values. Each list must begin on a new line although each list may occupy more than one line.

Example

The first form is more handy for a few changes:

```
MODIFY  1,SAM,PLE
1,1,10.
DONE

MODIFY  1,SAM,PLE  -1
1
1
10.
```

But for many changes mode 2 is better. For example the following use will zero out all diagonal terms

```
MODIFY  1,SAM,PLE  -&NR
1:&NR
1:&NR
&NR*0.
```

Special form

The command MODSUM may be used in place of MODIFY. In this case, all input items will be added to the current value rather than replacing it.

3.104. RUN MPNT

RUN=MPNT Command

Purpose

Print a summary of master file information for the specified event numbers.

Command

```
RUN=MPNT  (I1,(I2))
```

Where

I1,I2 - The first and last event number to be printed. Default is all events.

3.105. MSAP

MSAP Command

Purpose

Perform multiplication of a NEPSAP stiffness matrix by a standard VAPEPS matrix. The stiffness matrix may be single or double precision, while the other input and the output matrix will always be single precision.

Command

```
MSAP  NSEP,ITEM  N2,EL2,VR2  N3,EL3,VR3
```

Where

```
NSEP        - DAL unit where 'KEMX' and 'BIGK' reside.
            > 0, stiffness matrix is single precision
            < 0, stiffness matrix is double precision
ITEM        - first cycle number of 'BIGK' element
            = 0, use K matrix  -- BIGK,NEP1,0,1
            = 1, use K' matrix -- BIGK,NEP1,1,1
            =-1, use KG matrix -- BIKG,NEP1,1,1
N2,EL2,VR2 - input multiplicand. size = (NDOF,NVEC)
              where NDOF = size of stiffness matrix
                    NVEC = any number of columns
N3,EL3,VR3 - output matrix, size = (NDOF,NVEC)
```

Notes

The sign of NSEP indicates the precision of the input stiffness matrix. The other matrices are always single precision.

Stiffness extractions

If N3,EL3,VR3 is omitted, then N2,EL2,VR2 will be output as the full VAPEPS type single precision stiffness matrix. ie. size = (NDOF,NDOF). Warning, this process can be time consuming.

Cycles

The VAPEPS 'CYCLE' command has no effect on the NEPSAP matrices that are processed by this command. The cycles do control the VAPEPS matrices that are input and output.

3.106. RUN MSDF

RUN=MSDFREQ command

Purpose

This command calculates the frequencies for input into RUN=SCSD at which the MSD matrix will be calculated.

Command

```
RUN=MSDFREQ N1,NAME  FL,FH  (FBMX (BFAC))
```

Where

```
N1,NAME - DAL unit and version name for output.  The output element
          name is N1,'MSDF',NAME.
FL,FH   - 1/3 octave frequency range.
FBMX    - Maximum allowable frequency range between calculations.
          Default is to use only the band edges.
BFAC    - Factor to calculate band edge frequencies.  At each band edge
          two frequencies are calculated.  One is slightly lower than
          the band edge and the other is slightly higher.  The amount
          is:

             10**(LOG10(band edge) +- BFAC)

          BFAC defaults to .0001
```

Notes

In many cases it is sufficient to use only the band edges in the MSD calculation. RUN=CSD2 uses a linear interpolation to determine frequencies other than those calculated here.

3.107. RUN MSDS

RUN=MSDSETUP Command

Purpose

This command does the setup for RUN=SCSD.

Command

```
RUN=MSDSETUP  N1,NAME  FL,FH  (FBMX (BFAC) (ZFAC))
```

Where

```
N1,NAME - DAL unit and version name for SCSD input.
FL,FH   - 1/3 octave frequency range.
FBMX    - Maximum allowable frequency range between calculations.
          Default is to use only the band edges.
BFAC    - Factor to calculate band edge frequencies.  At each band edge
          two frequencies are calculated.  One is slightly lower than
          the band edge and the other is slightly higher.  The amount
          is:

             10**(LOG10(band edge) +- BFAC)

          BFAC defaults to .0001
ZFAC    - Equivalent nodal forces below this absolute value will be
          assumed to be zero.  Default = .01
```

This runstream prompts for the following input matrices:

```
'JLOC','BTAB' - Joint location matrix as output by EAL.
'VIBR','MODE' - Mode shape matrix as output by EAL.
'VIBR','EVAL' - Natural frequencies as output by EAL.
'EQNF','FORC' - Equivalent nodal forces for 1 psi pressure, as output
                by EAL.
```

The following matrices are assumed to already exist:

```
N1,'EXC ',NAME - Excitation acoustic field (#frequencies,#Force nodes)
                 or (#frequencies,1).  In this case this matrix is
                 expanded to (#frequencies,#Force nodes) Input is in
                 psi**2/hz
N1,'EXFQ',NAME - Frequency vector for acoustic field (#frequencies,1)
```

Notes

In many cases it is sufficient to use only the band edges in the MSD calculation. RUN=CSD2 uses a linear interpolation to determine frequencies other than those calculated here.

3.108. MSPAR

MSPAR Command

Purpose

Standard matrix multiplication (C = A*B), where A is a 'SPAR' sparse matrix. The sparse matrix may be single or double precision.

Command

```
MSPAR  N1,EL1,VR1  N2,EL2,VR2  N3,EL3,VR3  FAC
```

Where

N1,EL1,VR1 - Input sparse matrix. This is a special matrix, and is not subject to the regular 'VAPEPS' conventions. First of all, its version name must be 'SPAR' (ie VR1 = 'SPAR'). Secondly, the first cycle number must indicate the number of DOF per joint squared. Thus the current input cycles from the cycle command are not used. Instead, the element is read with 'MASKED' cycles. I repeat, the use of the CYCLE command has no effect on the reading of this element. The matrix may be single precision or double precision. In either case, it must follow SPAR's convention for matrix type. For single precision, the type must be 1, for double precision, the type must be 2, and the number of words and number of rows, must indicate the number of double precision words, not the number of single precision words.

N2,EL2,VR2 - Input second matrix. This is a standard VAPEPS matrix. Here the cycles from the last CYCLE command are used to read the element. It must be of the same precision as the sparse matrix above. For this element, double precision must be type 2 with NWDS = number of double precision words.

N3,EL3,VR3 - Output matrix. It will be of the same type as the above two matrices. Its cycle numbers will be as indicated in the last CYCLE command. In the case of double precision, the type will be 2.

Description

Matrix multiplication is fairly straight forward. The only thing to be added is that the dimensions of the arrays must be compatible. A sparse matrix is always a square array, but its dimension is not always obvious. If it is a genuine 'SPAR' sparse matrix, the size is NR = NC =

NJ*SQRT(C1). This will be the output number of rows for C, and must also be the number of rows for B. The output number of columns for C will be equal to the number of input columns for B.

This command is not meant to replace the 'PROD' command in SPAR's AUS. For one thing, it is more expensive. The reason for the expense, is that the intermediate products are always summed in double precision, even in the single precision case. The other reason is that 'PROD' does not allow the input of double precision mode shapes (ie. matrix B), and it does not yield a double precision answer. Thus AUS's PROD command should be used unless you want more precision.

3.109. MTJ

MTJ Command

Purpose

Multiply a matrix with 3 dimensions by a matrix with two dimensions. Eg.

```
     A(NI,NJ,NC) * B(NC,NI) = C(NC,NJ)
     or
     A(NI,NJ,NC) * B(NC,NJ) = C(NC,NI)
```

Command

```
MTJ  N1,EL1,VR1  N2,EL2,VR2  N3,EL2,VR2
or
MTJT  N1,EL1,VR1  N2,EL2,VR2  N3,EL3,VR3
```

Where

```
   N1,EL1,VR1 - Input 3d matrix:  A(NI,NJ,NC)
   N2,EL2,VR2 - Input 2d matrix:  B(MR,MC)
   N3,EL3,VR3 - Output 2d matrix: C(LR,LC)
```

Notes

The value of NJ for the first matrix is used to determine the length of its first dimension - NI = NR/NJ

```
   For MTJ    MR must = NC
              MC must = NJ
       and    LR will = NC
              LC will = NI

   For MTJT   MR must = NC
              MC must = NI
              LR will = NC
              LC will = NJ
```

3.110. MTRAN

MTRAN Command

Purpose

Perform the matrix multiplication

$$A B^t = C$$

without having to transpose B first.

Command

```
MTRAN  N1,EL1,VR1  N2,EL2,VR2  N3,EL3,VR3  (FAC)
```

Where

```
N1,EL1,VR1 - Input matrix.   (A)
N2,EL2,VR2 - Input matrix.   (B)
N3,EL3,VR3 - Output matrix.  (C)
FAC        - Optional multiplication factor which is carried along at
             no extra cost.
```

Notes

If the dimensions of A are (NR,NC) and the dimensions of B are (MR,MC), then NC must equal MC and Dimensions of C are (NR,MR)

MTRAN will handle virtually any size matrix, however batch mode is advised for large matrices (>200 by 200), since more core is available.

MTRAN works with single or double precision matrices as well as complex. Single matrices may be mixed with either double or complex, but double may not be multiplied with complex. If any matrices are double, output is double. If any matrices are complex, output is complex.

If the FAC option is used with complex multiplication, the factor is still assumed to be strictly real and thus multiplies both real and imaginary portions of the answer.

3.111. MULTIPLY

MULTIPLY Command

Purpose

Matrix multiplication.

Command

```
MULT  N1,EL1,VR1  N2,EL2,VR2  N3,EL3,VR3  (FAC)
```

Where

```
N1,EL1,VR1 - Input matrix. (ie. A(NR,NC))
N2,EL2,VR2 - Input matrix. (ie. B(NC,MC))
N3,EL3,VR3 - Output matrix. (ie. C(NR,MC))
FAC        - Optional multiplication factor which is carried along at no
             extra cost.
```

Notes

Where A, B and C are indicated above, the following operation is performed:

```
 DO 10 I=1,NR
 DO 10 J=1,MC
 C(I,J) = 0.
 DO 10 K=1,NC
 10 C(I,J) = C(I,J) + A(I,K)*B(K,J)
(10 C(I,J) = C(I,J) + A(I,K)*B(K,J)*FAC)
```

An error message is printed if dimensions of the input arrays are not compatible. Large matrices are broken into chunks. Chunk size depends on the maximum data space allowed (see CORE command).

If FAC is used with double precision or complex multipli- cation, FAC is still assumed to be a real number, and is used to multiply both the real and imaginary parts.

Single precision*complex and single*double products are allowed, but double*complex are not. If A or B is double, C will be double. If A or B is complex, C will be complex.

3.112. RUN MVC

RUN=MVC Command

Purpose

Calculate mean, variance and covariance of a data set.

Command

```
RUN=MVC  N1,EL1,VR1  N2,EL2,VR2  N3,EL3,VR3  IOP
```

Where

```
N1,EL1,VR1 - Input matrix (NR = # of variables, NC=#of cases)
N2,EL2,VR2 - Output average (NR=#of variables, NC=1)
N3,EL3,VR3 - Output co-variance matrix (#VAR,#VAR)
IOP        - Option flag as follows:
           = 0 - Division by number of variables. (Default = 0)
           = 1 - DIVISION BY #VAR-1
```

3.113. RUN NEWS

RUN=NEWS Command

Purpose

Display the latest news bulletin.

Command

```
RUN=NEWS
```

Notes

The element 30,'NEWS','NEWS' will be printed if it exists. Otherwise a message will be printed.

3.114. RUN NOML

RUN=NOML Command

Purpose

RUN=NOML linearly normalizes your baseline response/excitation data to a new excitation.

Command

```
RUN=NOML N1,EL1,VR1 N2,EL2,VR2 N3,EL3,VR3 N4,EL4,VR4 (UDAT)
```

Where

```
N1,EL1,VR1 - Input element containing the baseline excitation,
             size: (# of frequencies x 1).
N2,EL2,VR2 - Input element containing the baseline response,
             size: (# of frequencies x # of responses).
N3,EL3,VR3 - Input element containing the new excitation,
             size: (# of frequencies x 1).
N4,EL4,VR4 - Output element containing the normalized response,
             size: (# of frequencies x # of responses).
UDAT       - (optional) Data units:
           = 0 - (default) input and output units are linear,
           = 1 - input and output units are dB.
```

Notes

If N1,EL1,VR1 or N3,EL3,VR1 have more than one column, only the values in column 1 will be used in the calculations.

Equations

For linear data:

$$Rn= En*Rb/Eb$$

For dB data:

$$Rn= 10xLog[base10](10^{Rn/10} x10^{Rb/10} / 10^{Eb/10})$$

Where

Rn - Response normalized to the new excitation.
En - New excitation.
Rb - Baseline response.

Eb - Baseline excitation.

3.115. NORM

NORM Command

Purpose

Normalize an array such that the maximum value in each column is equal to 1.

Command

```
NORM  N1,EL1,VR1  N2,EL2,VR2
```

Where

```
N1,EL1,VR1 - Input integer, real, or double precision array.
N2,EL2,VR2 - Output normalized array.
```

Notes

NORM treats each column separately. Each term in the column is divided by the value of the term with the largest absolute value.

Alternate form

```
NORMABS  N1,EL1,VR1  N2,EL2,VR2
```

May be used instead of NORM. In this case, each column is divided by the absolute value of the term with the largest absolute value in the column. This may produce a normalized maximum of -1.

3.116. RUN NORM

RUN=NORM Command

Purpose

Normalize data from the VAPEPS data base to a different input spectrum.

Command

```
RUN=NORM NCAS NU1,VSN1 F1,F2
```

Where

```
NCAS      - Number of separate cases.
NU1,VSN1  - Output will go here.
F1,F2     - Frequency limits.
```

General

This runstream is fully interactive.

Each case pertains to particular excitation and normalization spectra. When either or both of these spectra change, a new case must be considered. For each case, the runstream will prompt for the event name and dal unit for the data to be normalized. Both excitation and response data must be from a single event for any given case. The runstream will then prompt for the names of excitation transducers. If there are more than one, the average will be used as excitation. Next, the runstream prompts for the names of data channels to be normalized. Lastly, the runstream prompts for the normalization spectrum. This process is repeated for the number of cases specified by ncas.

Input restrictions

Only one line of input is allowed for response to any of the prompts. If your response to any prompt might possibly require more than one line of input, store the response in an element ahead of time using the command ECOL. Then, in response to the prompt, type an equal sign followed by the unit, element name and version associated with the ele- ment containing the response.

Output

Output from this runstream consists of three elements. The first contains the normalized data. Each row represents a frequency band and each column represents a data channel. The data channels are entered into the element in the order in which they were specified. The units of the normalized data are db. The second output element is a list of channel names. the order is the same as input. The third element contains the overall levels. The names of these elements are:

```
1) NU1,'NORM',VSN1  Normalized data (dB)
2) NU1,'CHAN',VSN1  Channel names
3) NU1,'OVRL',VSN1  Overall levels (GRMS)
4) NU1,'NMGS',VSN1  Normalized data (g**2/Hz)
5) NU1,'FREQ',VSN1  1/3 octave frequencies (Hz)
```

These elements are used by the statistics runstream and the plot command STPLT.

Special Case

A special case exists when data is used that does not come from the database. In that case put in the actual unit,element,version for the data instead of the unit and event name of the database event.

Input is different in that channel names are not asked for.

Output is the same except that NU1,'CHAN',VSN1 will not be there.

3.117. NOT

NOT Command

Purpose

Create a vector which contains all numbers in a given range (or list) which are not in the input vector.

Command

```
NOT  N1,EL1,VR1  N2,EL2,VR2  IST,IFN,INC
or
NOT  N1,EL1,VR1  N2,EL2,VR2  FST,FLST,FINC
or
NOT  N1,EL1,VR1  N2,EL2,VR2  N3,EL3,VR3
```

Where

N1,EL1,VR1 - Input vector containing a list of values. For command type 1 above, the values must be integers, for 2, they must be real. For type 3, they may be integer, real or hollerith. See notes.

N2,EL2,VR2 - Output vector. The values will be of the same type as those input in N1,EL1,VR1.

IST,IFN,INC - First value, last value and increment of a sequence of integer numbers.

FST,FLST,FINC - First value, last value and increment of a sequence of real numbers.

N3,EL3,VR3 - Input vector (used instead of a sequence). Note that the values in this vector must be of the same type as N1,EL1,VR1.

Notes

The basic approach is to construct the item pool, either as a sequence of numbers for types 1 and 2, or by reading dataset N3,EL3,VR3 for type 3. Then, take each item in the pool and try to find it in the dataset N1,EL1,VR1. Any pool items not found in N1,EL1,VR1 are added to the output list

NOT sets the variable N$ to the number of terms that weren't found (eg the length of the output vector). If all input values are found in the sequence, N$=0, and no output element is produced. THUS YOU SHOULD USE N$ TO TEST FOR SUCCESS OR FAILURE AFTER 'NOT'. (If you check the size of the output element, and none was produced, you could accidentally pick up a previously generated element)

If INC (or FINC) is omitted from the command, it defaults to 1 (or 1.). If only one sequence definition term is present, it is assumed to be IFN (or FLST). In this case, IST (or FST) defaults to 1 (or 1.).

The type of input vector (real or integer only) determines the type of output vector. The choice of sequence terms (IST,IFN,INC or FST,FLST,FINC) must be consistent with this type.

For command type 3, either integer, real, or hollerith comparisons may be made. Note that elements 1 and 3 must be of the same type.

CAUTION --- Floating point comparisons

NOT may be used to compare floating point values, however it is highly unlikely (in general) for two floating point numbers to be exactly equal. Therefore, command type 2 uses the test

```
if ABS( (X1(I)-XS(J))/X1(I) ) .LE. 1.E-5  then
```

for all intents and purposes they are equal, and X1(I) will not be put in the output list. (it does protect against X1(I) being zero). (XS(J) = jth term in sequence)

Command type 3, on the other hand, insists that X1(I) must exactly equal some X3(J). Choose your method according to your needs.

3.118. NTOW

NTOW Command

Purpose

Convert narrow band data to wide band data.

Command

```
NTOW N1,NEL1,NVR1 N2,NEL2,NVR2 F1,F2 N3,NEL3,NVR3 N4,NEL4,NVR4
or
NTOW N1,NEL1,NVR1 N2,NEL2,NVR2 F1,F2 N3,NEL3,NVR3 CBAND
```

Where

```
N1,NEL1,NVR1 - Input narrow band center frequencies.
N2,NEL2,NVR2 - Input narrow band data.
F1,F2        - Input wide band center frequencies.
               These may be set using the FREQUENCY/SET command.
               Default is standard 1/3 octave.
N3,NEL3,NVR3 - Output data set to store wide band data.
N4,NEL4,NVR4 - Input narrow band band widths.
CBAND        - Constant narrow band band width.  Used in place
               of N4,NEL4,NVR4.
```

Example

Given the input data sets:

```
1,FREQ,PSD  - Narrow band center frequencies.
1,DATA,PSD  - Narrow band data.
```

And knowing that the data has a constant bandwidth of 1.5259 hz. Assuming that the data covers 1/3 octave freqeuncy bands from 20. to 2000. hz. Calculate the 1/3 octave psd for the data.

```
NTOW 1,FREQ,PSD 1,DATA,PSD 20.,2000. 1,THRD,OCTV 1.5259
PRINT 1,THRD,OCTV
```

Related commands

RUN=OCT3, INTEGRATE, INTERPOLATE

Discussion

This command will output zeros if there is no narrow band data in a band. Also if there is not a full band of data the level is averaged with zero for the rest of the band.

Moral: Make sure that the narrow band data covers the entired band width for each of the wide bands desired.

3.119. RUN OCT3

RUN=OCT3 Command

Purpose

This runstream takes a narrow band psd and converts it to a standard 1/3 octave band psd.

Command

```
RUN=OCT3   NU1,NEL1,NVR1  F1,F2  (NU2,NEL2,NVR2 (NU3,NEL3,NVR3))
```

Where

```
NU1,NEL1,NVR1   - Input narrow band psd (# FREQS X # RESP)
F1 ,F2          - Frequency range for 1/3 octave output
NU2,NEL2,NVR2   - Output unit and version. defaults to:  NU1,NEL1,NVR1
NU3,NEL3,NVR3   - Non-standard input frequency vector defaults to:
                  NU1,'FREQ',NVR1 or if the number of columns in NU1
                  NEL1 NVR1 is equal to 2 then the frequency vector is
                  taken from column one of NU1 NEL1 NVR1 and the psd
                  vector is taken from column 2.
NU2,NEL2,NVR2   - Output 1/3 octave PSD (# FREQS X # RESP)
NU2,'FOC3',NVR2 - 1/3 octave center frequencies.
```

Notes

If other than 1/3 octave frequencies are desired use the FREQUENCY/SET command to set the frequencies used. (eg. FREQUENCY/SET 2 will cause the routine to use standard 1/6 octave frequencies.) For more information see the documentation for the FREQUENCY command.

3.120. OR

OR Command

Purpose

Combine two lists into one list that contains all items from each list.

Command

```
OR  N1,EL1,VR1  N2,EL2,VR2  N3,EL3,VR3
```

Where

```
N1,EL1,VR1 - Input vector
N2,EL2,VR2 - Input vector
N3,EL3,VR3 - Output vector of all items.
M$         - Output variable set to indicate number of items in output
             list.
```

Notes

The input vectors may be integer, real or hollerith, but both input vectors must be of the same type. Input vectors may contain repeated values. The output vector will contain each item only once.

3.121. ORDER

ORDER Command

Purpose

Order the rows of a matrix by arranging the values of a particular column in ascending order. ORDER may also be used to 'pack' a data set by eliminating zeros, duplicate values or values within a certain range of each other.

Command

```
ORDER  N1,EL1,VR1  N2,EL2,VR2  ICOL  N3,EL3,VR3  N4,EL4,VR4
- column numbers (for negative ICOL)
- FACTOR(s)      (for F option)
```

Where

```
N1,EL1,VR1 - Input matrix to be sorted.
N2,EL2,VR2 - Output sorted matrix.
ICOL       - Sort flag.
           > 0 - ICOL = Column number to be sorted. (default = 1)
           = 0 - Sort all columns independently.
           < 0 - -ICOL = Number of columns to be used during sort
                   procedure.
N3,EL3,VR3 - Optional output integer dataset which contains resulting
             sort pointers. First output value is the number of the row
             that contains the smallest value. This will always be a
             vector unless the input matrix has more than one column
             and ICOL = 0.
N4,EL4,VR4 - Optional output integer dataset which contains the index
             for determining which rows were removed/kept as the result
             of a NoTie, Factor, or NoZero operation. This will always
             be a vector, with as many rows as the input matrix. If no
             ties or zeros occur, the vector will be 1 through NR. If
             20th item matched 14th, and T option was used, then row 20
             would contain a 14.
```

Notes

In its basic form, ORDER reads the column indicated by ICOL, and forms a list of pointers indicating which row is lowest, second lowest etc. It then reads the input matrix column by column, reorders it, and writes the reordered columns to the second matrix. For ICOL<0,the order process is to first read -ICOL numbers directly following the command. The first column indicated is ordered. If there are any ties, then the second column number input is used to break the ties. If there are still ties, the next column is used, and so on.

The last option is sort each column independently.

The third element is a list of pointers. This will be a vector except for the case where ICOL=0, in which case a column of pointers will be produced for each column in the input matrix.

The type of sorting is determined by the type of the element input. (Type 0=integer, 1=floating, 2,3=error >3 = logical sorting).

Additional forms

```
ORDNT    N1,EL1,VR1 .....
ORDNZ    N1,EL1,VR1 .....
ORDNTZ   N1,EL1,VR1 .....
ORDF     N1,EL1,VR1 .....
- FACTOR(s)
```

The NT option stands for 'No Ties'. Any duplicate values will result in only one of the values being output. In the case of a multiple column (ICOL < 0) ORDNT, all specified columns must match before a 'tie' is indicated. The NZ option stands for 'No Zeros'. Any rows that contain a zero will be dropped. Again in the multiple column case, all specified columns must be zero for a row to be dropped. ORDNTZ is a combination of the two forms. Note: These options may not be used with a column by column ORDER (ICOL = 0), since it could result in different row lengths for each column. The F option loosens the definition of a 'tie'. You must input a factor or factors which define how close two values must be in order to be considered the same. For multiple column ORDERS (ICOL < 0), you may enter a different FACTOR for each column (all FACTORS must be input on the same line) or you can enter 1 FACTOR that will then apply to all columns.

3.122. RUN ORTH

RUN=ORTH Command

Purpose

Check the orthogonality of a matrix.

Command

```
RUN=ORTH  N1,EL1,VR1  N2,EL2,VR2
```

Where

N1,EL1,VR1 - Input matrix to be checked.
N2,EL2,VR2 - Output normalized matrix.

Notes

This runstream normalized the input matrix, prints the diagonal of the resulting matrix and prints the value of the maximum off diagonal term as well as it's location.

3.123. RUN OTOC

RUN=OTOC Command

Purpose

Prints an ordered table of contents that contains all datasets that match the input spec. Eg OTOC 1 would produce an ordered TOC of the whole file.

Command

```
RUN=OTOC  NU,ELN,VER,IC1,IC2  IDATE NOUT
```

Where

```
NU,ELN,VER,IC1,IC2 - Are the input DAL unit, element, version and
                     cycle numbers of data sets to match.  DAL unit
                     must always be input. Element, version and cycle
                     can use the word MASK as in the FIND command.
IDATE              - Any elements created before this date will  be
                     ignored.  Default = 0
NOUT               - Fortran unit to direct output.  Default = 6
```

Notes

The output format will not have the sector address. Instead, it will include NWDS,NI,NJ,NR,NC instead of just NJ,NINJ or NR,NC

3.124. RUN OVER

RUN=OVER Command

Purpose

Compute the overall level of the input matrix.

Command

```
RUN=OVER  N1,IL1,IV1  N2,IL2,IV2 (F1,F2)
```

Where

```
N1,IL1,IV1 - Input matrix  (NFRQ,NCHN)
N2,IL2,IV2 - Output overall matrix  (NCHN,1)
F1,F2      - Frequency limits (if frequency limits are present then
             input is assumed to be in G**2/HZ and output is overall in
             GRMS)
```

Notes

This runstream assumes that the input element is in dB unless frequency limits are present then it assumes that input is in G**2/Hz.

3.125. PACK

PACK Command

Purpose

Physically compact a DAL file by removing all disabled elements, or copy a group of elements from one file to another.

Command

```
PACK  N1
PACK  N1,N2
PACK  N1,N2,I1,I2
```

Where

```
N1     - DAL unit to be packed or DAL unit that contains elements to be
         transferred.
N2     - Output DAL unit.
I1,I2  - Range of sequence numbers.  All non-disabled elements from
         sequence number I1 through I2 on DAL unit N1 will be copied to
         the end of DAL unit N2.
```

Example

Remove all disables elements from DAL unit 1.

```
PACK 1
```

Notes

DAL files have a tendency to collect disabled elements. PACK removes these elements. Note that the first form of PACK physically overwrites itself. This can be disastrous if the process is interrupted (max-time error, etc.). For this reason, the second method is preferred. In this case, the method is to 'PACK N1,N2' and then exit VAPEPS and copy the second file back to the first using available system processors.

The third method provides an easy method of copying a block of elements from one file to another. Note that only non-disabled elements will be transferred.

Related Commands

ENABLE, DISABLE, TOC, COPY, FIND

3.126. RUN PADM

RUN=PADMIN command

Purpose

This command extracts all words to be defined from the specified event and creates a list for input to the DICTIONARY/PREADMIN command..

Command

```
RUN=PADMIN  NU,VENT
```

Where

NU,VENT - DAL unit and event name.

Notes

This command creates a list of all the words (NU,'WORD',VENT) in an event which are to be entered into the Data Dictionary.

3.127. RUN PCOM

RUN=PCOMPARE Command

Purpose

This runstream plots the specified inputs on the same plot.

Command

```
RUN=PCOMPARE
```

Input

1 - Number of curves.
2 - Channel numbers of curves to compare. (#Channels X #Curves) This matrix must be integer or hollerith (Type 0 or 4 respectively)
3 - For each curve:

```
N1,NVR1  MARKER  FLAG  TYPE
```

Where

```
N1,NVR1   - DAL unit and version of input data sets.
MARKER    - Curve marker flag.
          <     0 - Mark every MARKER curves with no lines
                    connecting each marker.
          =     0 - Connect each point with a solid line no curve
                    markers.
          >     1 - Mark every MARKER curves and connect each
                    marker with a solid line.
          = 10000 - Connect each point with a dashed line no
                    curve markers.
FLAG      - Special purpose flag.
          =  0 - Input data sets are as follows:
                 N1,'RDAT',NVR1 - Data.
                 N1,'RFRQ',NVR1 - Frequencies.
                 N1,'RCHN',NVR1 - Channel names.
          =  1 - Input data set has the form:
                 N1,'N'I,NVR1
                 Data set contains two columns of data.
                 The first column contains frequencies.
                 The second column contains data.
                 Where I is an integer input in the Channel
                 numbers matrix. If I=1 then 'N'I = N1
TYPE      - For FLAG = 0:
            Character to replace the 'R' in RDAT, RCHN, and RFRQ
```

data sets. The represents the type of the DATA. The conventions used are:

- M - Microphone data.
- A - Acceleration data.
- S - Strain gauge data.
- R - Raw data. This data can be of any type desired.

The default is R.

- For FLAG = 1:
  Character to replace the 'N' in 'N'I element name. Default is N.

4 - Abscissa label.
5 - Ordinate label.
6 - Three title lines.
7 - Lower and upper limits for ordinate axis. The default is 0.,0. which gives automatic scaling. If YMIN is input as negative then an approximately square 3 x 3 cycle log log plot will be produced. If both YMIN and YMAX are negative then a normal -YMAX x 3 cycle log log plot will be produced.
8 - Type of plot.
SPL - Data is in dB units and will be plotted on a linear - log plot.
ACC - Data is in linear units and will be plotted on a log - log plot.
Default is ACC.

Labels

For normal curves the label is set to the channel number identifier as it is input in 2 above.

At this time labels for specs are set to &ITYP&I where &ITYP is the character input for the first character of the spec curve number.

Titles

At this time only one set of titles per plot set can be input. This will change in the future.

Example

```
* First set up some data sets.
ECOL 1,RDAT,TAPZ 10,3
10*.002
10*.005
10*.2
ECOL 1,RDAT,SQZ  10,3
10*.015
10*.025
10*.003
FREQUENCY/CENTER 1,RFRQ,TAPZ 50.,315.
FREQUENCY/CENTER 1,RFRQ,SQZ  50.,315.
```

```
ECOL 1,RCHN,TAPZ 3,1
A1,A2,A3
ECOL 1,RCHN,SQZ  3,1
A9,A12,A34
* Now create a spec.
ECOL 1,N1,SPEC 4,2
31.5 ,100. ,250. ,500.
  .01,    .2,    .2,    .001
* Now create the channel pointers.
ECOL 1,CHAN,PNT 3,3
1,1,1
A2,A1,A3
A34,A9,A12
* Now select plot device
PSET TEK
* Now do comparisons.
RUN=PCOMPARE
3                  -- Number of curves
1,CHAN,PNT         -- Data set containing appropriate channel pointers.
1,SPEC,     0,1,N  -- Information for curve 1.  Solid line, no markers.
1,TAPZ,     1,0,R  -- Information for curve 2.  Solid line with markers.
1,SQZ ,10000,0,R   -- Information for curve 3.  Dashed line, no markers.
Frequency, (Hz)    -- Abscissa label.
G**2/Hz            -- Ordinate label.
This               -- Three title lines.
is an
example
.001,1.            -- Ordinate limits.
ACC                -- Type of plot.
```

3.128. RUN PDBG

RUN=PDBG Command

Purpose

Does plots of data in g**2/Hz, dB or delta dB. Prompts you for titles and ordinate range

Command

```
RUN=PDBG  N1,IEL1,IVR1  N2,IEL2,IVR2  DTYP
```

Where

```
N1,IEL1,IVR1 - Frequency vector
N2,IEL2,IVR2 - Data matrix
DTYP         - Type of data
             = G2 for g**2/Hz
             = DB for dB
             = DDB for delta dB.
             = USER prompts for all plot parameters.
```

When this runstream is executed you will be asked to input three title lines.

3.129. RUN PGLS

RUN=PGLS Command

Purpose

Sort a glossary alphabetically then print it.

Command

```
RUN=PGLS  NU,ELN,VER
```

Where

NU,ELN,VER - DAL unit, element and version of glossary.

Related Commands

GLOSSARY

3.130. RUN PGRM

RUN=PGRMS Command

Purpose

Plot output matrix of RUN=GRMS command.

Command

```
RUN=PGRM N1,VR1 DEVI
```

Where

```
N1,VR1 - Dal unit and version of data created by RUN=GRMS
DEVI   - Device to send plot to.  Defaults to TEK.
```

Related Commands

RUN=GRMS

3.131. RUN PICK

RUN=PICK Command

Purpose

Extract selected channels from an event stored in the VAPEPS database.

Command

```
RUN=PICK  NU,VENT  N1,EL1,VR1  N2,EL2,VR2  FLOW,FHI
```

Where

```
NU,VENT     - Input event DAL unit and name.
N1,IL1,IV1  - Input vector of channel names to be extracted.
N2,IL2,IV2  - Name of output element.
FLOW,FHI    - Desired frequency range (default = 10.,10000.)
```

Related Commands

RUN=DATA, RUN=NORM, RUN=GMOD

3.132. RUN PMDS

RUN=PMDS command

Purpose

This command finds the modes of a simply supported rectangular plate.

Command

```
RUN=PMDS  N1,NVR1
```

Where

N1,NVR1 - DAL unit and version name to store output. If N1 is input as a negative value then N1,NVR1 is assumed to exist and all the variables are preset to their previous values.

Notes

The following variables are to be input at the "Input parameters >" prompt:

```
A    = Length of plate
B    = Width of plate
E    = Young's modulus
H    = Thickness.
RHO  = Mass density.
ANU  = Poisson's ratio.
M    = Mode number (default=1)
N    = Mode number (default=1)
```

The following commands are available at the "Input parameters >" prompt:

```
LIST  - List the values of the above variables.
DONE  - Terminate input and calculate frequency.  A blank line is
        equivalent to DONE.
QUIT  - Exit runstream and save input.
CHECK - Checks for missing variables.
```

3.133. POW

POW Command

Purpose

Raise a vector to successive powers. This may be used to construct polynomials.

Command

```
POW  N1,EL1,VR1  N2,EL2,VR2  MM
```

Where

```
N1,EL1,VR1 - Input DAL element, a vector (NR,1).
N2,EL2,VR2 - Output DAL element, an array (NR,MM).
MM         - Number of powers to use.
             If MM > 0, use powers 0 through MM-1
             If MM < 0, read -MM powers from card immediately following
             command.
```

Notes

The POW command works as follows:

Given an input vector X and a list of integer powers, compute the output matrix Y as

```
Y(I,J) = X(I)**IPOW(J)
```

3.134. RUN POWE

RUN=POWER Command

Purpose

This command is used to find the power flow between any two elements via the specified connection.

Command

```
RUN=POWER N1,NAME N2,NEL2,NVR2
-DATA-
```

Where

```
N1,NAME       - Name of SEMOD model.
N2,NEL2,NR2   - Name of output vector with power flow. Flow is in
                direction specified by order that the path is input.
                Output units are consistent with the units the user
                specified in CFAC.

                   1 - Meters, Kilograms, and seconds  (Watts)
                   2 - Centimeters, Grams, Seconds.    (Ergs/sec)
                   3 - Feet, Slugs, Seconds.           (Ft Lbf/sec)
                   4 - Inches, Snails, Seconds.        (In Lbf/sec)

                   Power = Mass * Length**2 / Time**3

-DATA-        - Path for which power flow is to be calculated.
```

3.135. RUN POWI

RUN=POWI Command

Purpose

Create or modify power inputs into an SEA theoretical or EXTRAP I model.

Command

```
RUN=POWI  N1,NAME  N2,NEL2,NVR2  (IOP)
-ELEMENT-
```

Where

```
N1,NAME       - DAL unit and version name of SEA model for power input.
N2,NEL2,NVR2  - DAL unit, element and version name of power input.
IOP           - Option to determine mode for existing power input.
              = REPLACE - Totally replaces any power inputs into the
                          specified element.  Default.
              = ADD     - Adds the value of this power input to any
                          existing power input to the specified element.
              = NEW     - Will cause any exiting power inputs to
                          any elements to be replaced.
-ELEMENT-     - Element name where power is to be input.
```

Example

Put a constant power input of 10. into the SEA element SHROUD in SEA model TEST. 1/3 octave frequency range is 10. to 10000. hz.

```
ECOL 1,POWE,R 31,1
  31*10.
RUN=POWIN  1,TEST  1,POWE,R
  SHROUD
```

Notes

If a multiple number of power inputs were ADDed together for a single power input and only one of them is to be replaced then the others will have to be reADDed to that input.

Related Commands

RUN=TBLP, SEMOD, PREDICT/THEORETICAL, PREDICT/EXTRAP/I

3.136. RUN PRSP

RUN=PRSP Command

Purpose

This runstream is designed to take input SEA element names from a SEMOD model and plot the names you specified in the units that you specify.

Command

```
RUN=PRSP  N1,NVR1  (IOP)
```

Where

N1,NVR1 - Dal unit and version name of SEMOD model.
IOP - The number of elements that you wish to plot. Excitation elements can be extracted as well as response elements. If this parameter is omitted then up to 10 elements can be specified.
= 'ACRO' all acoustic response elements.
= 'ACEO' all acoustic excitation elements.
= 'ACER' all acoustic excitation and response elements.
= 'STRO' all structural response elements.
= 'STEO' all structural excitation elements.
= 'STER' all structural excitation and response elements.

Input

The following elements must be present:

N1,'FREQ',NVR1 = 1/3 octave center frequency vector.
N1,'RESP',NVR1 = Output from PRDICT. These are the responses of each element in your model. (#freqs X #resps)
N1,'MNAM',NVR1 = This is a vector of element names.
N1,'IDES',NVR1 = This is the descriptor matrix which tells which elements are excitation and which are response.
N1,'DBFG',NVR1 = Unit flag matrix. Output by CFAC.

3.137. RANDOM

RANDOM Command

Purpose

Creates an array of uniformly distributed random numbers between 0. and 1.

Command

```
RANDOM  N1,EL1,VR1  NR,NC
or
RANC  N1,EL1,VR1  NR,NC
```

Where

```
N1,EL1,VR1 - Name of output vector.
NR,NC      - Number of rows and number of cols. (default NC = 1)
```

RANC creates an array of random complex numbers.

Notes

RANDOM creates a unique set of random numbers every time it is called. If a sequence is desired that can be repeated, enter NR as a negative number, and the same sequence will always be written.

3.138. RLJUST

RLJUST Command

Purpose

This command will right justify or left justify (and blank fill) a matrix containing four character hollerith words.

Command

```
RLJUST  N1,EL1,VR1  N2,EL2,VR2  IOP
```

Where

```
N1,EL1,VR1 - Input DAL element containing hollerith data.
N2,EL2,VR2 - Output DAL element containing right or left justified
             hollerith data.
IOP        - Operation code. 1 to right justify and 2 to left justify.
```

Related Commands

ORDER

3.139. RUN ROUN

RUN=ROUND Command

Purpose

Convert floating point elements to integers by rounding to the nearest integer.

Command

```
RUN=ROUND  N1,EL1,VR1  N2,EL2,VR2
```

Where

```
N1,EL1,VR1 - Input floating point array.
N1,EL2,VR2 - Output integer array.
```

Related Commands

RTOI

3.140. RUN RRAN

RUN=RRAN Command

Purpose

Generate a matrix of random numbers that lie between the specified values.

Command

```
RUN=RRAN  N1,EL1,VR1  NR,NC  XL,XU
```

Where

```
N1,EL1,VR1 - Name of array to store random numbers.
NR,NC      - Number of rows and columns in matrix.
XL,XU      - Values between which to generate random numbers.
```

Related Commands

RANDOM

3.141. RTOI

RTOI Command

Purpose

Real to integer conversion and visa versa.

Command

```
RTOI  N1,EL1,VR1  N2,EL2,VR2  IFLG
```

Where

```
N1,EL1,VR1 - Input element.
N2,EL2,VR2 - Output element.
IFLG       - Conversion indicator (optional).
           < 0 - Implies integer to real conversion.
           > 0 - Implies real to integer conversion.
```

Notes

RTOI allows the truncation of floating point, or lets the user convert reals to integers for I/O purposes.

Single precision real arrays are identified by a type code of 1 or -1. Integer arrays have a type code of 0. RTOI checks the type of the input element and performs the necessary conversion to the complement type.

If IFLAG is used, the indicated direction must match the operation that RTOI would have done if it were omitted. If there is a conflict, an error is returned and the conversion does not take place.

3.142. RUN SAVE

RUN=SAVE Command

Purpose

Save all PREPed events on the specified DAL file to the master file.

Command

```
RUN=SAVE  NPRP  MAST
```

Where

```
NPRP - DAL unit containing PREPed events. (Default = 1)
MAST - DAL unit containing master file.  (Default = 13)
```

Note

This stream looks for all elements CFDF/MASK on unit NPRP to determine which events are worthy of saving. This means that any events that do not contain a CONFiguration section will be ignored.

3.143. RUN SAVG

RUN=SAVG Command

Purpose

This command is used to average two input responses from a SEMOD model.

Command

```
RUN=SAVG  N1,NVR1  N2,NVR2  N3,NVR3
```

Where

```
N1,NVR1 - First model of which responses are to be averaged.
N2,NVR2 - Second model of which responses are to be averaged.
N3,NVR3 - Output model to put averaged responses.
```

3.144. RUN SCSD

RUN=SCSD Command

Purpose

Calculate Modal Spectral Density based on a SIN(KX/KX) correlation function.

Command

```
RUN=SCSD  NUFD,NVFD  NURD,NVRD    SPEED
```

Where

```
NUFD,NVFD - DAL unit and version name of input data sets
            PF   - Input force eigenvectors (#forces,#modes)
            JLOC - Input force locations (#forces,3)
            SIGN - Input signs (#forces,1)
            PSD  - Input force PSD (#forces,#frequencies)
            MSDF - Frequencies for PSD (#frequencies,1)
NURD,NVRD - DAL unit and version name of output data sets
            MSD  - Modal Spectral Density matrix (modes,modes)
            Second cycle number ranges from 1 to #frequencies.
SPEED     - Propagation speed for coupling (Speed of sound).  (Default
            = air = 13548. in/s)  Coupling equation between forces is:

               SQRT( PSD(I)*PSD(J) ) * SIGN(I) * SIGN(J) * SIN(KX)/KX

               Where

                  K - 2*PI*FREQ/SPEED
                  X - distance between force locations
```

Notes

This runstream presents an alternate approach to RUN=PSD. It is useful when you have a large number of coupled forces, as well as a large number of response points. This runstream is normally used in conjunction with RUN=CSD2 which essentially back transforms the Modal Spectral Density into the Response Spectral Density as produced by RUN=PSD.

Warning

This routine uses PF, but it requires that PF be dimensioned FORCES by MODES which is the transpose of RUN=PSD.

Execution time

This run can really eat up CPU time, especially if you have a large number of forces. The good news is that it can be restarted. The output MSD matrix is initialized to 0. If your run errors off for max time, check MSD for the last non-zero column. Set the parameter ISC$ to this column number and the loop will start there.

3.145. SEARCH

SEARCH Command

Purpose

Find locations in a matrix where values lie within a given range or match an input list of values. SEARCH may also be in a search and change mode.

Command

```
SEARCH  N1,EL1,VR1  N2,EL2,VR2  ICOL  LOW,HIGH  (IBUF,ISCAL)
-DATA- (If ICOL < 0)
or
SEARCH  N1,EL1,VR1  N2,EL2,VR2  ICOL  HIGH,LOW  (IBUF,ISCAL)
-DATA- (If ICOL < 0)
or
SEARCH  N1,EL1,VR1  N2,EL2,VR2  ICOL  NUM  (IBUF,ISCAL)
-DATA- (If ICOL < 0)
-DATA-  (Num values, or other, as discussed below)
```

Where

```
N1,EL1,VR1 - Input matrix to be searched.
N2,EL2,VR2 - Output list of matching items.
ICOL       - Column search indicator.
           > 0  - The particular column to search.
           = 0  - Search all columns
           < 0  - Search -ICOL columns. User is prompted to enter
                  column numbers as a single line of data following the
                  command.
LOW        - Values must be .GE. LOW to match
HIGH       - Values must be .LE. HIGH to match
NUM        - Number of specific values to look for.
           > 0  - User will be asked to input NUM values. This data is
                  read until NUM values have been read. For ICOL < 0,
                  this data follows the column number data, and applies
                  to all columns searched.
           = 0  - Valid for ICOL < 0 only. The user must enter a
                  separate LOW,HIGH or HIGH,LOW pair for for each
                  column searched. Each pair must be entered on a
                  separate line.
           < 0  - Valid for ICOL < 0 only. The user must enter -NUM
                  values for each column searched. The values for a
                  particular column may take as many lines as needed,
                  but the list for each column must start on a new
                  line.
```

IBUF,ISCAL - Special buffering factors that control the type of data that is output. These are normally omitted from the command. Note: either both values must be present, or both values must be omitted. These items are discussed below under 'Output'.

-DATA- - All data discussed above is entered in VAPEPS free-field format. In interactive mode, the user is prompted for this data.

Notes

The following lines explain the basic search conditions:

MODE 1 (LOW,HIGH) LOW.LE.HIGH

Find all values in input matrix A, such that

```
LOW .LE. A(I,ICOL) .LE. HIGH
```

MODE 2 (HIGH,LOW) LOW.LE.HIGH

Find all values in input matrix A, such that

```
A(I,ICOL) .LE. LOW  .OR. HIGH .LE. A(I,ICOL)
```

ie. find all values outside the exclusive range LOW-HIGH

MODE 3 (NUM)

User inputs a list of values ie: VAL(J) J=1,NUM Find all values in input matrix A such that

```
A(I,ICOL) = VAL(J)  for any J
```

Output

The basic output item is the element N2,EL2,VR2, which is discussed below. In addition, SEARCH keeps track of the number of matches as the variable MCH$. This variable is available following the SEARCH command for use with other commands.

If ICOL > 0, only one column of A will be searched. In the default output mode (IBUF,ISCAL not present), only the row numbers of matches will be returned. These will be output as the vector N2,EL2,VR2.

If ICOL = 0, then all columns will be searched. In the default output mode (no IBUF,ISCAL), 3 items will be returned for each match: ROW #, COL #, VALUE. The output element will be a vector with length = 3*number of matches.

Rows 1,4,7,... contain row numbers

```
   Rows  2,5,8,... contain column numbers
   Rows  3,6,9,... contain values
```

If ICOL < 0, -ICOL columns will be searched. In the default mode, matches are considered on a row by row basis. Thus all columns must match their appropriate condition at a given row in order for that row to match. The matching row numbers will be output as the vector N2,EL2,VR2.

IBUF,ISCAL

The use of these arguments force the output to contain all items (row,col,value) for each match. For the ICOL < 0 case, it changes the match condition so that each column is treated separately. I.e. matches are no longer on a row by row basis. In addition to changing the basic match condition, it limits the maximum row length of the output element to IBUF rows. If this length is too small to contain all matches, additional columns will be used rather than extending the row size.

If ISCAL > 0, then the matching row/col pairs are converted to an integer pointer as IPOINT = IROW*ISCAL + ICOL In this case only two items are output per match, eg IPOINT and VALUE.

SPAR buffs will find this latter form very useful in creating SPAR STRP-type elements -- ie:

```
   *
   * FIRST ZERO LOWER TRIANGULAR HALF
   *
   BUILD  1,TEMP,MAT  &NR,&NC
   IAFJ  0,0,1
   MIX    1,K,MAT  1,1  1,&NC
   DONE
   *
   * NOW SEARCH FOR TERMS WITH ABS-VAL > 1.E-6
   *
   SET  FILT=1.E-6
   SEARCH  1,TEMP,MAT  1,K,STRP  0  &FILT,-&FILT  1000,10000
```

Search and change mode

SEARCH may also be used to find all values within a certain range, and change them to a new value. The command SEACHG may be used to initiate this mode:

```
   SEACHG  N1,EL1,VR1  N2,EL2,VR2  0  LOW,HIGH   NEW
```

Note that ICOL must be a zero. N2,EL2,VR2 will have the same size as N1,EL1,VR1. All values of N1,EL1,VR1 between LOW and HIGH will be reset to NEW, before being output to N2,EL2,VR2. The variable MCH$ is set to indicate the number of values changed.

3.146. RUN SEND

RUN=SEND Command

Purpose

Finds all the type 5 and 6 elements specified and prints them.

Command

```
RUN=SEND  NU,EL1,VR1,IC1,IC2,IDAT,IDS$
```

Where

```
NU,EL1,VR1,IC1,IC2 - DAL unit, element, version and cycles of elements
                     to print.  DAL unit must be specified.  Element,
                     version and cycle numbers can be MASKed.
IDAT               - All elements created before this date are
                     ignored.  (Default = 0)
IDS$               - ?????
```

Notes

If NU is input as negative, then the element

```
-&NU,&EL1,VR1
```

is assumed to contain the elements that are to be sent furthermore, this element is the transposed version of normal 'FINES' elements. The elements themselves are assumed to be on unit IC1.

3.147. SEQUENCE

SEQUENCE Command

Purpose

Generate an equal spaced linear or logarithmic vector given number of points, starting value and increment or final value.

Command

```
SEQUENCE  N1,EL1,VR1  NPTS  START  FINC
or
SEQM      N1,EL1,VR1  NPTS  START  FINC
```

Where

```
N1,EL1,VR1 - Name of new vector.
NPTS       - Length of new vector.
START      - Starting value.
FINC       - Increment.
```

Notes

The FORTRAN code for SEQ is

```
        DO 100 I=1,NPTS
100     V(I) = START + (I-1)*FINC
```

If NPTS < 0, then the value input after START is really the final desired value and FINC is calculated internally as

```
FINC = (FINC-START)/(-NPTS-1)
```

For logarithmic vectors, use SEQM instead of SEQ, which results in the following code:

```
        DO 100 I=1,NPTS
        V(I) = START
100     START = START*FINC
```

If NPTS < 0, FINC is recalculated as

```
FINC = (FINC/START)**(1/(-NPTS-1))
```

3.148. RUN SGET

RUN=SGET Command

Purpose

This command is used to extract SEMOD models from an event. It can also list the names of semod models in an event.

Command

```
RUN=SGET  N1,IVNT  (N2,NAME)
```

Where

```
N1,IVNT - DAL unit and version name of database event.
N2,NAME - DAL unit number of an empty DAL file and name of the
          model to be extracted.  If omitted the names of the models
          on the event are listed out.
```

3.149. RUN SHIF

RUN=SHIFT command

Purpose

This runstream takes a model generated by SEMOD and shifts the valid frequency range for frequency dependant parameters up or down by the specified amount.

Command

```
RUN=SHIFT  N1,NVR1  (N2,NVR2  (SHIFT))
```

Where

```
N1,NVR1 - DAL unit and version name of your model.
N2,NVR2 - DAL unit and version name of output shifted model.
SHIFT   - Fraction of an octave to shift frequency ranges by.  (Default
          = +1. octave)
```

Related Commands

RUN=SAVG

3.150. SHOCK

SHOCK Command

Purpose

Compute absolute or relative shock spectra for equal spaced time transients. Output frequency spectrum need not be equal spaced.

Command

```
SHOCK  N1,EL1,VR1  N2,EL2,VR2  (ISV(,IREL))
-DATA-
```

Where

```
N1,EL1,VR1 - Input transient array, one column per transient. #rows =
             #time points. All transients must correspond to the same
             equal spaced time history.
N2,EL2,VR2 - Output shock spectra. #rows = #freqs There will be one or
             two columns per transient, depending on ISV.
ISV        - Option flag.
           = 0 - or omitted. Do not save maximum response times. Output
                 only maximum values at each frequency (one column
                 output per transient).
           = 1 - Save response times as well as response values. Output
                 two columns per transient. First column contains
                 spectra, second contains times.
IREL       - Option flag.
           = 0 - or omitted. Compute absolute shock spectra.
           = 1 - Compute relative shock spectra.
```

Note

ISV and IREL are optional, but if IREL is present, ISV must be included even if it is zero.

Data

The following card of data is read in VAPEPS free format.

```
Q,F1,F2,NF,T0,DT(,MAX,COND)
```

Where

```
Q    - Gain factor = 1./(2.*critical damping)
F1   - Starting frequency
```

F2 - Final frequency
NF - # of frequency points
T0 - Time corresponding to first values of transients
DT - Time increment of transients
MAX - Maximum number of time points to use in computation. If MAX > # input time points, the transient is assumed constant at the last transient value. If omitted, MAX is set to 10000.
COND - Value of input transients prior to time = T0. If omitted, COND is set to the value of each transient at time = T0.

Note

MAX and COND are optional, but if COND is used, MAX must be included even if MAX = 10000

Uneven Frequencies

The above data card will yield spectra for an equal spaced frequency domain. If desired, the user may create his own frequency vector and replace

```
F1,F2,NF
```

above with

```
N3,EL3,VR3
```

where N3,EL3,VR3 is the element containing the desired frequency vector. In this case the output array will have as many rows as this input element.

Notes

The SHOCK command uses the same Z-transform technique as SHOCK2 and its derivatives. Lagrange six point interpolation is used to pinpoint maximums to DT/5. This is done whenever the maximum lies between time points 4 and MAX-3 inclusive.

At present the actual response time history at each frequency is not available to the user.

3.151. RUN SMAX

RUN=SMAX Command

Purpose

Find the principal stresses and maximum shear given a matrix containg the x,y and shear stresses.

Command

```
RUN=SMAX  N1,NEL1,NVR1  N2,NEL2,NVR2
```

Where

```
N1,NEL1,NVR1 - Input matrix containing stresses.
               Column 1 - Contains the x stress.
               Column 2 - Contains the y stress.
               Column 3 - Contains the shear stress.
N2,NEL2,NVR2 - Output matrix containing stresses.
               Column 1 - Minimum principal stress.
               Column 2 - Maximum principal stress.
               Column 3 - Maximum shear stress.
               Column 4 - Orientation of principal axes, relative
                          to x and y. (Degrees)
```

3.152. RUN SNDP

RUN=SNDP Command

Purpose

Finds all elements that have been changed since IDAT and sends them. Note that elements are grouped according to the first cycle number. Each group is prefaced by a list of elements in that group.

Command

```
RUN=SNDP  NU,IDAT
```

Where

```
NU   - DAL unit to find elements on.
IDAT - All element created before this date are ignored.
```

Notes

If IDAT is input as a negative number, the element 28,TOC,LIST is assumed to already exist. In this case, only those elements contained in the list will be sent.

3.153. RUN SNPV

RUN=SNPV Command

Purpose

This command is used to calculate the percentile value for responses from a SEMOD model.

Command

```
RUN=SNPV  N1,NAM1  N2,NAM2  (PERC  (IOP))
```

Where

```
N1,NAM1 - DAL unit and name of your model.
N2,NAM2 - DAL unit and name to output data to.
PERC    - Percentile value.  Default = .95
IOP     - Option to input Standard deviations or skew factors.
        = 'SDO'  - Prompts for a vector of values for the standard
                   deviation.  One value for each frequency.
        = 'SFO'  - Prompts for a vector of values of the skew factor.
                   One value for each frequency.  It can be input as a
                   single number which will be used for the entire
                   frequency range.
        = 'SDSF' - Prompts for both the skew factor and the standard
                   deviation.
          If IOP is omitted then the Standard deviation is calculated
          based on Std. Dev / Mean = 1.57  The skew factor is set to
          1.225
```

Input

The following elements must be present.

```
N1,'RESP',NAM1 = Response matrix output from TPRD.
N1,'DBFG',NAM1 = Conversion flag matrix output from CFAC.
N1,'IDES',NAM1 = Descriptor matrix output from ATACO.
N1,'FREQ',NAM1 = Frequency vector output from MDENS.
N1,'MNAM',NAM1 = Model name matrix output from ELNAME.
```

Output

The following elements are output.

```
N2,'RESP',NAM2 = Percentile value of prediction.
N2,'DBFG',NAM2 = Conversion flag matrix.
```

```
N2,'FREQ',NAM2 = Frequency vector.
N2,'MNAM',NAM2 = Model name matrix.
```

This output is setup so that you can use LIST in SEMOD or RUN=PRSP to plot your results.

Notes

This runstream uses the RUN=TPVL command to calculated percentile values.

Related Commands

RUN=TPVL, RUN=STAS

3.154. RUN SPIL

RUN=SPILL Command

Purpose

Spill all events from the specified DAL unit to a symbolic form which is suitable for transmittal to other sites.

Command

```
RUN=SPILL  NU  NOUT,NAME,NAMV,IC1,IC2
```

Where

```
NU                     - DAL unit to get events from.
NOUT,NAME,NAMV,IC1,IC2 - DAL unit, element, version and cycle  numbers
                         of output symbolic element.
```

Transferring the VAPEPS data base

This document contains instructions on

1) How to create a VAPEPS SPILL tape from your system

2) Notes on how the destination site may read the tape.

Creating the SPILL tape

Creation of the tape involves two main steps.

1) Using the RUN=SPILL command to convert binary data base files to DAL symbolic datasets. One symbolic dataset is produced per data base file.

2) Using WTAPE to write the symbolic datasets to the tape.

Assume there are three files to be copied to tape. The first step is to define these files as DAL units 1,2, and 3 respectively. (Any DAL units may be used, 1-3 are used here as an example)

```
    Univac                          CDC

@ASG,A DATABASE*ONE.          ATTACH,DAL001=DBONE.
```

```
@USE DAL001.,DATABASE*ONE.         ATTACH,DAL002=DBTWO.
@ASG,A DATABASE*TWO.               ATTACH,DAL003=DBTHREE.
@USE DAL002.,DATABASE*TWO.
@ASG,A DATABASE*THREE.                      VAX
@USE DAL003.,DATABASE*THREE.
                                   $DEFINE DAL001 [DATABASE]ONE
                                   $DEFINE DAL002 [DATABASE]TWO
                                   $DEFINE DAL003 [DATABASE]THREE
```

The next step is to SPILL each file onto one temporary DAL file.

```
execute VAPEPS  ( @XQT V.VAPEPS, VAPEPS, or $VAPEPS)

   *
   *  SET BLOCKING INFORMATION (# RECORDS/BLOCK, # CHARS/RECORD)
   *                                  60                80
   *
   SET NRB$=60, NCR$=80
   *
   *  SPILL each file to a different data set name
   *
   RUN=SPILL  1,20,SPIL,DECK,1,0
   RUN=SPILL  2,20,SPIL,DECK,2,0
   RUN=SPILL  3,20,SPIL,DECK,3,0
   *
   *  Copy these files to tape. Note that a tape unit must be
   *  assigned prior to the following commands. They assume tape
   *  unit 4 is being used. (VAX -- TAPE04, CDC -- TAPE04,
   *                          UNIVAC -- 4 )
   *
   CYCLE 1,0
   WTAPE  20,SPIL,DECK  4,1
   CYCLE 2,0
   WTAPE  20,SPIL,DECK  4,2
   CYCLE 3,0
   WTAPE  20,SPIL,DECK  4,3
   END
```

Note

As written above, the tape will contain info in whatever language your system uses. Eg a VAX will produce an ASCII tape. CDC will produce CYBER data, Univac will produce FIELDATA info. The documentation below indicates that the tapes are in ASCII, so you may use CRYPT before writing the tape if you want.

a) If you have a CDC, and are transmitting to another CDC, you might as well leave the code in CYBER. You should amend the following documentation accordingly.

b) If you want to convert your data to ASCII, CRYPT may be used:

```
* UNIVAC
CYCLE 1,0
CRYPT 20,SPIL,DECK  20,NEW,DECK  6,9  0,63
= 30,FLDT,ASCI

* CDC
CYCLE 1,0
CRYPT 20,SPIL,DECK  20,NEW,DECK 6,8  0,63
= 30,CYBR,ASCI
```

Installing the data base from tape

Once VAPEPS has been successfully loaded on your system, you can start loading up the data base. Typically you will have one or more 'spill' tapes that contain events from other sites. Each tape may contain several different events.

The data on tape consists of symbolic records that comprise a PREP runstream. The runstream may be 'added' to VAPEPS and will create a DAL data file. The naming of the DAL file is left up to the receiving site. Once the DAL files are created, the ADMIN processor must be used to save the files on the master DAL file.

Tape Formats

In general, you may receive data base 'spill' tapes from any other site that uses VAPEPS. The standard tape is unlabeled with blocked fixed-length records. Normally there will be 80 characters per record, and 60 records per block. The standard encoding language is Ascii.

The tape may be read with the TDUMPS command in VAPEPS. On some systems it will be necessary to convert the ASCII characters to a native code. The CRYPT command may be used to perform this task.

VAX stream to read tape

The following runstream will read the tape on a VAX.

```
$@VAPEPS:INIT            ! Add standard VAPEPS initialization
$MOUNT  MTA0:/FOREIGN
$DEFINE  TAPE04  MTA0:
$SET DEFAULT [wherever.the.DAL.files.are.to.be.created]
$VAPEPS
*
*  FIRST, READ ALL FILES FROM TAPE TO A TEMPORARY FILE
```

```
*
TDUMPS  4,1,100  28,TAPE,DUMP  5000 6
*
*  THEN, EXIT VAPEPS, AND FREE TAPE
*
END
$DISMOUNT MTA0:
$!
$! NOW, ASSIGN A NAME FOR EACH OUTPUT FILE
$!
$DEFINE DAL001 name-for-first-file
$DEFINE DAL002 name-for-second-file
...
$VAPEPS
*
*  FIRST FILE FROM TAPE
*
CYCLE 1,1
CHANGJ  28,TAPE,DUMP  60      @ 60 RECORDS PER BLOCK
SET NPRP = 1                  @ DEFINE OUTPUT DAL UNIT
##READ  28,TAPE,DUMP          @ ADD'S THE CARDS
*
*  SECOND FILE FROM TAPE
*
CYCLE 2,1
CHANGJ  28,TAPE,DUMP  60
SET NPRP = 2
##READ  28,TAPE,DUMP
*
*  AND SO ON
*
*  THE FILES MAY BE ADDED TO MASTER FILE BY USING THE SAVE RUNSTREAM
*  NOTE THAT YOU MUST HAVE WRITE ACCESS TO THE MASTER FILE IN ORDER
*  TO 'SAVE' THE NEW EVENTS.
*
RUN=SAVE  1
RUN=SAVE  2
END
```

UNIVAC stream to read tape

The following will read a tape on the UNIVAC.

```
@ADD VAPEPS*VAPEPS.INIT
@ASG,T  TAPE.,U9,reel number
@USE    4.,TAPE.
@XQT  V.VAPEPS
*
* FIRST, READ ALL FILES FROM TAPE
*
```

```
TDUMPS  4,1,100  28,TAPE,DUMP  5000 6
*
*  EXIT VAPEPS, AND FREE TAPE
*  THEN PREPARE AN OUTPUT FILE FOR EACH FILE READ FROM TAPE
*
END
@FREE 4.
@ASG,UP  NAME-OF-FIRST*OUTPUT-FILE.
@USE  DAL001.,NAME-OF-FIRST*OUTPUT-FILE.
@ASG,UP  NAME-OF-SECOND*OUTPUT-FILE.
@USE  DAL002.,NAME-OF-SECOND*OUTPUT-FILE.
@XQT V.VAPEPS
*
*  NOW, LOOP ON FILES
*
*  FIRST FILE FROM TAPE
*
CYCLE 1,1
CHANGJ  28,TAPE,DUMP  60      @ 60 RECORDS PER BLOCK
*
*  CONVERT TO FIELDATA
*
CRYPT   28,TAPE,DUMP  28,FLDT,DUMP  8,6  0,127
= 30,ASCI,FLDT
*
*  NOW, ADD FILE
*
SET NPRP = 1                  @ DEFINE OUTPUT DAL UNIT
##READ  28,FLDT,DUMP          @ ADD'S THE CARDS
*
*  SECOND FILE FROM TAPE
*
CYCLE 2,1
CHANGJ  28,TAPE,DUMP  60
*
*  CONVERT TO FIELDATA
*
CRYPT   28,TAPE,DUMP  28,FLDT,DUMP  8,6  0,127
= 30,ASCI,FLDT
*
*  NOW, ADD FILE
*
SET NPRP = 2
##READ  28,FLDT,DUMP
*
*  AND SO ON
*
*  THE FILES MAY BE ADDED TO MASTER FILE BY USING THE SAVE RUNSTREAM
*  NOTE THAT YOU MUST HAVE WRITE ACCESS TO THE MASTER FILE IN ORDER
*  TO 'SAVE' THE NEW EVENTS.
```

```
*
RUN=SAVE  1
RUN=SAVE  2
END
```

CDC stream to read tape

The control cards needed on a CDC system will depend on the particular system you have. In some cases you will have to stage the tape, in other cases, you can access the tape directly. In either case, TDUMPS may be used to read the tape (it uses BUFFER IN)

The following stream may be used once in VAPEPS

```
*
* FIRST, READ ALL FILES FROM TAPE
*
TDUMPS  4,1,100  28,TAPE,DUMP  5000 6
*
*  EXIT VAPEPS, AND FREE TAPE
*  THEN, RE-ENTER VAPEPS
*
END
*
*  NOW, LOOP ON FILES
*
*  FIRST FILE FROM TAPE
*
CYCLE 1,1
CHANGJ  28,TAPE,DUMP  60      @ 60 RECORDS PER BLOCK
*
*  CONVERT TO CYBER CODE
*
CRYPT   28,TAPE,DUMP  28,CYBR,DUMP  8,6  0,127
= 30,ASCI,CYBR
*
*  NOW, ADD FILE
*
SET NPRP = 1                   @ DEFINE OUTPUT DAL UNIT
FNAME  1,'FILENAM'             @ GIVE THE FILE A NAME
##READ  28,CYBR,DUMP           @ ADD'S THE CARDS
*
*  SECOND FILE FROM TAPE
*
CYCLE 2,1
CHANGJ  28,TAPE,DUMP  60
*
*  CONVERT TO CYBER
*
CRYPT   28,TAPE,DUMP  28,CYBR,DUMP  8,6  0,127
```

```
= 30,ASCI,CYBR
*
*  NOW, ADD FILE
*
SET NPRP = 2
FNAME 2,'NEWNAME'
##READ  28,CYBR,DUMP
*
*  AND SO ON
*
*  THE FILES MAY BE ADDED TO MASTER FILE BY USING THE SAVE RUNSTREAM
*  NOTE THAT YOU MUST HAVE WRITE ACCESS TO THE MASTER FILE IN ORDER
*  TO 'SAVE' THE NEW EVENTS.
*
RUN=SAVE  1
RUN=SAVE  2
END
```

After the second execution of VAPEPS, you should include appropriate control cards that will save the files as permanent files, or write the created DAL files to a save tape.

3.155. RUN SPIN

RUN=SPIN Command

Purpose

Finds all elements and writes them in a manner they can be reentered into VAPEPS.

Command

```
RUN=SPIN  NU,IL1,IV1,IC1,IC2 NOUT
```

Where

```
NU,IL1,IV1,IC1,IC2 - DAL unit, element, version and cycles of  elements
                     to be written.  DAL unit must be specified.
                     Element, version and cycles can be MASKed.
NOUT               - Fortran unit to direct output.
```

Notes

If NU is negative, the element 28,IL1,IV1,IC1,IC2 is assumed to contain the result of a FIND operation transposed.

3.156. RUN SPRE

RUN=SPREP Command

Purpose

This command is used to put a SEMOD model into an already existing event. It also associates channels with each element in the SEMOD model.

Command

```
RUN=SPREP  N1,IVNT  N2,NAME
```

Where

```
N1,IVNT  - DAL unit and event name of PREPed event.
N2,NAME  - DAL unit and model name of SEMOD model.
```

Notes

Only the information input in ELNAME, PATHNAME, FPAR, FREQUENCY, SETEXC, and TEXT is saved.

3.157. SREAD

SREAD Command

Purpose

Read data images from file. Records may be up to 132 characters long. Increments may be input which enable the reading of upper and lower triangular matrices. The only restriction of SREAD is that matrices must be read a column at a time.

Command

```
SREAD  N1,EL1,VR1  NR,NC  NU  IST,IFN,INC  JST,JFN,JNC
(FORMAT)
```

Where

```
N1,EL1,VR1  - Name of output element.
NR          - Number of rows in output element.
NC          - Number of columns of output element.
NU          - FORTRAN unit where card images are read.
            > 0  - Unit is rewound after read.
            < 0  - Unit is not repositioned after read.
IST,IFN,INC - Starting row, final row and increment for first column.
JST,JFN,JNC - Increments of above quantities for each successive
              column.
FORMAT      - Format to be used when reading file. The format must be
              enclosed in parentheses.
```

Description

SREAD always reads column by column. The default is to produce a real type dataset (type 1). Alternate forms may be used for integer or double precision:

```
SREADD - Matrix will be double precision (type 2) Be sure to use
         FORTRAN 'D' type format.
SREADI - Matrix will be integer (type 0). On most systems you may input
         mixed hollerith, real and integer using this type.
```

The following FORTRAN statements indicate exactly how the matrix is read.

```
DIMENSION A(NR,NC)
CALL MOVER(0,0,A,1,NR*NC)
DO 100 J=1,NC
READ(NU,1000) (A(I,J),I=IST,IFN,INC)
```

```
      IST = IST + JST
      IFN = IFN + JFN
      INC = INC + JNC
  100 CONTINUE
 1000 FORMAT(FORMAT)
```

If you are unsure about how to set IST,IFN,INC,JST,JFN,JNC simply sketch out a FORTRAN program which would read the data you have. If your data is really set up to be read row by row, then you may still use this command, but the resulting matrix should be transposed.

You may not skip over any of the command items, but only the first six quantities are essential. The remaining items have preset values and need not be included unless items later in the list need to be changed.

Preset values

```
NU  = 5  (records may not exceed 80 chars.)
IST = 1
IFN = NR
INC = 1
JST = 0
JFN = 0
JNC = 0
```

3.158. SSTRNG

SSTRNG Command

Purpose

Given a list of integers, find all columns of an input matrix that contain all of the integers. Output a list of column numbers and optionally a bit map for each matched column, which indicates which rows were matched.

Command

```
SSTRNG  N1,EL1,VR1  N2,EL2,VR2  N3,EL3,VR3  N4,EL4,VR4
```

Where

```
N1,EL1,VR1 - Input integer vector  (NR,1)
N2,EL2,VR2 - Input integer matrix to be searched  (MR,1)
N3,EL3,VR3 - Output integer vector of column numbers  (M$,1)
N4,EL4,VR4 - Optional output bit map vector  (M$*LB,1)
M$         - # of matching columns
LB         - length of each bit map = (MR-1)/30 + 1
```

The number of matches is returned as the variable M$.

No output elements are produced if there are no matches.

3.159. RUN STAC

RUN=STACK Command

Purpose

Print element from DAL unit 1 to a Fortran unit in such a way that they can be transmitted to another machine and reread into VAPEPS.

Command

```
RUN=STACK  N1,EL1,VR1  NOUT
```

Where

```
N1,EL1,VR1 - Input matrix containing element names -- size = (N,4)
             Column 1 = Element name.
                    2 = Version name.
                    3 = First cycle number.
                    4 = Second cycle number.
NOUT       - Output Fortran unit (Default = 1)
```

Related Commands

RUN=SPIN

3.160. RUN STAS

RUN=STAS Command

Purpose

RUN=STAS calculates the following statistical quantities for each row of your input matrix:

Normal distribution	Log-normal distribution
-mean,	-mean,
-standard deviation,	-standard deviation,
-ninety-fifth percentile,	-fiftieth-percentile,
-and maximum (envelope).	-ninety-fifth percentile,
	-and maximum (envelope).

The command also displays a tabular listing of the results.

Command

```
RUN=STAS N1,EL1,VR1 N2,VR2 (PRIN)
```

Where

```
N1,EL1,VR1 - Input data element, size: (#rows x #columns).
N2,VR2     - Dal unit and version name of the OUTPUT elements
             containing the statistical results, see OUTPUT.
PRIN       - (optional) Print option.
           = 6 - (default) tables of normal and log-normal statistics
                 are printed on your terminal's screen.
           = 0 - Tables are not printed.
           > 0 - Tables are printed to FORTRAN unit PRIN.
```

Output

N2,STAN,VR2 - Element containing normal distribution statistics, size: (#rows x 4), arranged as follows:

Col	Statistic
1	- arithmetic mean
2	- standard deviation
3	- ninety-fifth percentile
4	- maximum (envelope)

N2,STAL,VR2 - Element containing log-normal distribution statistics, size: (#rows x 5), arranged as follows:

Col	Statistic
1	- mean, same as STAN
2	- fiftieth percentile
3	- standard deviation
4	- ninety-fifth percentile
5	- maximum, same as STAN

Equations

For normal distribution statistics:

```
m = (sum of x)/n
sn = ((for i=1,n: (x - m)**2)/(n-1))**.5
n95% = m+(sn*1.645/gf)
gf = (1.0212-((.89447*n)**(-.5848))-.0033sin(pi*log,base10,n) )
```

For log-normal distribution statistics:

```
lm = (sum of logx)/n
sl = ((for i=1,n: (logx - lm)**2)/(n-1))**.5
ln50% = antilog(lm)
lns = antilog(sl)
ln95% = antilog(lm+(sl*1.645/gf))
```

Where

n - Number of samples.

x - Sample value.

m - Arithmetic mean (part of OUTPUT).

sn - Standard deviation, normal distribution (part of OUTPUT).

n95% - Ninety-fifth percentile (50% confidence), normal distribution (part of OUTPUT).

gf - Gumbel factor (corrects standard deviation for small sample sizes).

lm - Mean of logged data.

sl - Standard deviation of logged data.

ln50% - Fiftieth-percentile in original units, log-normal distribution (part of OUTPUT).

lns - Standard deviation of logged data in original units (part of OUTPUT).

ln95% - Ninety-fifth percentile (50% confidence) in original units, log-normal distribution (part of OUTPUT).

3.161. STAT

STAT Command

Purpose

Compute the mean and standard deviation of a set of numbers. Stat will also compute mean + or - 3 sigma values for a series of maxmin arrays.

Command

```
STAT  N1,EL1,VR1  N2,EL2,VR2  NCASE,SIG,ICASE
```

Where

```
N1,EL1,VR1 - Name of input element(s)
N2,EL2,VR2 - Name of output element
NCASE      - Number of cases considered
SIG        - Option flag.
             <= 0 - Compute mean and standard deviation (Default = 0)
             >  0 - Compute mean + or - SIG*sigma
               SIG must be a real number (ie., Use decimal point)
ICASE      - Optional starting case number for evaluation of max-min
               tables (default = 1).
```

Notes

STAT may be used in two completely different instances. The first is when you have one array or vector to be analyzed. Here your sample size is equal to the number of rows, and you evaluate the mean and standard deviation for each column. The number of rows output will equal the number of columns input. There will be two columns output, for means and standard deviations respectively. To use this mode, set NCASE equal to 1 and SIG equal to zero.

The second type of problem involves the statistical analysis of a group of max-min tables. Here NCASE is set to the number of max-min tables involved. Your sample size is equal to the number of cases. The output element will have as many rows as there were in the input elements. Note that all input elements must have the same dimensions. The number of input columns will usually be four, but STAT can handle two columns as well. In fact, if your arrays have four columns, STAT assumes the second and fourth columns are times and ignores them.

All elements input will have the name N1,EL1,VR1. They are distinguished by the presence of a non-zero first cycle number. The second cycle number must be 0. It is important to note that the input cycles as set by the

CYCLE command have no effect on the reading of these elements. If NCASE is 10 (for example), then ten elements will be read. These elements will have first cycle numbers of 1 - 10 resp. If ICASE were entered as 4, then cycles 4 - 13 would be accessed.

Now that the reading of these elements is taken care of, consider what the output element will look like. This depends on the value of SIG.

If SIG is .LE. 0, Then four columns are output. The first will contain the means of the input maximums and the second will contain the standard deviations of the input maximums. The next two columns will contain the same quantities for the input minimums.

If SIG > 0, then only two columns will be output. The first column will be mean plus SIG times the standard deviation of the maximums, and the second will be the mean minus SIG times the standard deviation of the input minimums. Ie: If SIG = 3, then mean + 3 sigma and mean - 3 sigma values are calculated for maximums and minimums respectively.

The following example should help those who are lost (count me in this group):

```
   STAT 1 XNA RUN1 1 SIGA RUN1 27 3
```

would read 27 elements and produce one element called 'SIGA'/ 'RUN1' on DAL unit 1. This element would have two columns and as many rows as there were on the 27 'XNA'/'RUN1' elements.

There are two main ways of evaluating the standard deviation of a set of data. One method (biased) involves the division by the square root of the number of samples. The other method (unbiased) divides by the square root of n-1. Because the latter gives a larger (more conservative) value for the standard deviation, it is used in this command.

3.162. RUN STAT

RUN=STAT Command

Purpose

Calculate statistics on input data sets.

Command

```
RUN=STAT NORM NU1,IV1 NU2,IV2 NU3,NM3,IV3 F1,F2 IOUT,ITYP,ITIT,IRU
```

Where

```
NORM         - Indicates type of data in NU1,IV1.  If 'NORM' = 0,
               NU1,IV1 is assumed to refer to PREPed data.  If 'NORM' >
               0, NU1,IV1 is assumed to refer to output from RUN=NORM.
NU1,IV1      - DAL unit and event name.
NU2,IV2      - Unit and version for output.
NU3,NM3,IV3  - Element containing channel names.
F1,F2        - Frequency range.
IOUT         - Fortran output unit.  Printed output is suppressed if
               IOUT is not specified.
ITYP         - Element name for output elements.
ITIT         - 'EXC' or 'RESP' for excitation data and response data,
               respectively.
IRU          - Type of data. 1 for acoustic, 2 for random vibration.
```

Related Commands

RUN=STTS, RUN=STAL, RUN=NORM

3.163. STORE

STORE Command

Purpose

Modify the value of one term in an array.

Command

```
STORE   VAL  N1,EL1,VR1  IROW,ICOL
STORE   VAL  N1,EL1,VR1  IC1,IC2  IROW,ICOL
STORI  IVAL  N1,EL1,VR1  IROW,ICOL
STORI  IVAL  N1,EL1,VR1  IC1,IC2  IROW,ICOL
```

Where

```
VAL        - Floating point number.
IVAL       - Integer.
N1,EL1,VR1 - DAL element to be modified.
IC1,IC2    - Optional cycle numbers of element to be modified.  If
             omitted, IC1 and IC2 default to last input cycles defined
             through CYCLE command.
IROW,ICOL  - Array location to be modified.
```

Notes

Store is 'dumb' in that it does not know the type of value you are inserting. Thus if the value is a floating point number, you must use STORE. If the value is an integer, you must use STORI.

If STORE is used and the array is real, the value is stored without conversion. If the array is not real, a real to integer conversion is done on VAL prior to insertion.

If STORI is used and the array is real, a conversion takes place. Otherwise there is no conversion.

3.164. STRESS

STRESS Command

Purpose

This command calculates a theoretical spatial average mean-square stress for flat plate, curved plate, cylinder or a user defined ratio of mean-square stress to mean-square displacement based on an input mean-square acceleration response.

Command

```
STRESS N1,NAME F1,F2 IOP
```

Where

```
N1,NAME - DAL unit and version name of stress model.
F1,F2   - Frequency range.
IOP     - Case from the following:
        = 1 - Simply supported isotropic flat plate.
        = 2 - Simply supported isotropic cylinder.
        = 3 - User defined ratio of mean-square stress to mean  square
              displacement.
        = 4 - Simply supported isotropic curved plate.
```

Input

The following elements are required for input.

```
N1,'RESP',NAME - Mean-square acceleration response. Required for  all
                 options.  Units of acceleration must be consistent with
                 the desired output units of stress. eg. acceleration
                 units = (IN/SEC**2)**2 stress units = (PSI)**2 Use the
                 CONVERT command to convert accelerations to the
                 desired units.
N1,'FRMN',NAME - Modal frequencies. Required for IOP=3 only. These
                 modes must be in sequential order.  Use the ORDER
                 command to sort the modes if needed.
N1,'RATO',NAME - Ratio of mean-square stress to mean-square  response
                 as defined by user. Required for IOP=3 only. The
                 ratio's  are to be in the same order as the modal
                 frequencies.
```

Output

The following elements are output.

N1,'STRS',NAME - Stress at each band.
N1,'NMPB',NAME - Number of modes in each band.

Notes

The calculation of mean-square stress is based on a derived value of the ratio of mean-square stress to mean square displacement. The following three cases are provided. In case 1), 2), and 4) the ratio is calculated for you. In case 3) you provide the ratio at each modal frequency and the stress is averaged over each band.

1) The calculation of the ratio of spatial average mean-square stress to spatial average mean square displacement for a plate and associated frequencies over a frequency band.

2) The calculation of the ratio of spatial average mean-square stress to spatial average mean square displacement for a cylinder and associated frequencies over a frequency band.

3) The calculation of the mean-square stress over the frequency band based on input of a user defined ratio of spatial average mean-square stress to spatial average mean-square displacement and associated frequencies.

4) The calculation of the ratio of spatial average mean-square stress to spatial average mean square displacement for a curved plate and associated frequencies over a frequency band.

The cases 1), 2), and 4) prompt for parameter input from the following list of parameters:

Type 1	Type 2	Type 4	Description
E	E	E	Equivalent Young's modulus.
NU	NU	NU	Poisson's ratio.
H	H	H	Equivalent thickness.
L	L	L	Length of plate or cylinder.
W			Width of plate.
	R	R	Radius of cylinder.
RHOS	RHOS	RHOS	Equivalent surface mass density.
EA	EA	EA	Actual Young's modulus of material at the location of the stress.
ZC	ZC	ZC	Distance from the centroid to the desired stress location.
		THETA0	Angle subtended by panel in radians.

No parameters are required for case 3.

If other than 1/3 octave frequencies are desired use the FREQUENCY/SET command.

Refer to NASA Contractor Report 180783 "VAPEPS IMPROVEMENT WITH STRESS ESTIMATION AND PROGRESSIVE WAVE EXCITATION", Y. A. Lee, D. Crowe, and W. Henricks, June 1987, for more information including derivation of equations and verification.

3.165. RUN STTS

RUN=STTS Command

Purpose

Calls RUN=STAT to produce a printed listing of assorted statistical information and create output elements required to compute statistical information for predicted responses.

Command

```
RUN=STTS NORM N1,IV1 N2,IV2 F1,F2 IOUT MODE IXM N3,NM3,IV3 IRM N4,NM4,IV4,IV5
```

Where

NORM - Integer code indicating source of data.
= 0 - Data will be extracted directly from the event file (no normalization).
> 0 - Output from run=norm will be used to compute statistics.
< 0 - User input data set. Will prompt for input element name.

N1,IV1 - DAL unit and version of the element containing 'prepped' data for 'NORM'=0, or DAL unit and version containing output from RUN=NORM for 'NORM' > 0.

N2,IV2 - DAL unit and version for output elts. If MODE=3, IV2 is the version for excitation output elements.

F1,F2 - Lower and upper 1/3-octave frequencies.

IOUT - Fortran unit for printed output. Printed output is suppressed if out=0.

MODE - Mode of operation.
= 1 - Excitation data only.
= 2 - Response data only.
= 3 - Both excitation and response data.

IXM - Type of data in channels specified in N3,NM3,IV3.
= 1 - Acoustic data.
= 2 - Random vibration data.

N3,NM3,IV3 - DAL unit, name and version of element containing channel names. For MODE=1 or MODE=3, these are excitation channels. For MODE=2, these are response channels. If NORM=1, this element is the CHAN element created by RUN=NORM. Using any other source for the channel names could result in improper identification of the channels in the listing.

IRM - Type of data in channels specified by names in N4,NM4,IV4. Required only if MODE=3.

= 1 - Acoustic data.
= 2 - Random vibration data.

N4,NM4,IV4 - Required only if MODE=3. Then, N4,NM4,IV4 specify the DAL unit, name and version of the element con- taining the names of the response channels.

IV5 - Required only if MODE=3. IV5 is the version for response output elements when both excitation and response data are specified.

Output

N1,'FREQ',IV1 - Contains 1/3 octave frequencies. (#freqs X 1)
N1,'STAT',IV1 - Contains four columns as follows: (#freqs X 4)
1 - Average of data - g**2/Hz.
2 - Antilog of average of logs of data.
3 - Antilog of 95th percentile level of logs of data.
4 - Maximum values of data.
N1,'SOVR',IV1 - Overall levels of columns 1-4 of 'STAT' (4X1)
N1,'PROB',IV1 - Ordered data (low to high) (#accels X #freqs)
When IV5 is input there is another set of data the same as above with an output version of IV5.

Interactive mode

If no arguments are input then the user is prompted for input.

3.166. RUN SUBC

RUN=SUBC Programmer's Command

Purpose

Allows you to remove commands in a setcom subroutine

Command

```
RUN=SUBC  OLD NEW  NIN NOUT
```

Where

```
OLD = Version name of existing ABV, CALL, and EPS matrices
NEW = Version name for new elements created here
NIN = Unit number for input (default = 1)
NOUT= Unit number for output (default = 2)
```

Related Commands

RUN=ADDC, RUN=BSC, RUN=GCLS

3.167. RUN SUM

```
RUN=SUM N1,EL1,VR1  N2,EL2,VR2  ITYP

RUNSTREAM COMPUTES THE SUMMATION OF A GROUP OF DATA BY ROWS OR COLUMNS.
DEFAULT MODE IS AN SUMMATION OF COLUMNS.

N1,EL1,VR1 = INPUT ARRAY (NR,NC)
N2,EL2,VR2 = OUTPUT VECTOR OF SUMMED NUMBERS
             (NR,1) FOR ITYP='ROW' (NC,1) FOR ITYP='COL'
ITYP       = CHARACTER STRING INDICATING WHETHER TO SUM
             THE ROWS ('ROW') OR THE COLUMNS ('COL')
             ITYP DEFAULTS TO 'COL'
```

AUTHOR: D M PARK 9/9/83

3.168. SYMIN

SYMIN Command

Purpose

Write card images to a type 5 or type 6 DAL element.

Command

```
SYMIN  NU,ELN,VER  IC1,IC2  NOUT
CARD 1
CARD 2
.
.
.
LAST CARD
SEOF
```

Where

```
NU,ELN,VER - Output DAL element.
IC1,IC2    - Cycle number.
NOUT       - FORTRAN unit for reading card images. (Default NOUT = 5)
```

Notes

Cards are read until either an EOF is sensed, or until the word SEOF is read as the first four characters of a line. In some cases (embedded SYMIN's) it is desirable to use a different word than SEOF to terminate the read. The command form SYMEND allows the specification of any name as follows:

```
SYMEND  NU,ELN,VER  IC1,IC2  NOUT  NAME
```

Where

NAME - Is the new terminator word. The convention is that the last word entered on the command is taken as the terminator, and the command is essentially reissued. Thus any or all of IC1,IC2 NOUT may be omitted prior to NAME.

DAL element types (5 or 6)

SYMIN may create either type 5 or type 6 DAL elements. The type created depends on the value of the variables NCR$ and NRB$. If these variables are not set, SYMIN will produce a type 5 element (variable record length).

If set, NCR$ = number of characters per record (ie 80) and NRB$ = number of records per block (ie 60), and SYMIN will produce a fixed record length element (type 6) according to the defined conventions.

Editor (SYMED)

SYMIN may be used as a batch type editor, by using the command

```
SYMED  N1,EL1,VR1
- correction cards
SEOF
```

Correction cards are of the form

```
-2,3      (delete lines 2 through 3)
NEW LINE  (insert after original line 1)
-6
NEW LINE  (insert after original line 6)
--*    (change correction indicator to *)
*7,7  (delete original line 7)
```

Notes

Correction cards must be specified in order. You cannot go back to the top of the element.

3.169. RUN TBLP

RUN=TBLP Command

Purpose

Calculate the power input for an SEA model which has aerodynamic noise caused by the unsteady pressure fluctuations in the turbulent boundary layer (TBL) which evelopes a cylinder or flat plate.

Command

```
RUN=TBLP  N1,NAME  N2,NEL2,NVR2  N3,NEL3,NVR3 ITYPE
```

Where

```
N1,NAME       - The DAL unit and name of a SEMOD model.
                The only required input matrix is &N1,'FREQ',&NAME.
N2,NEL2,NVR2  - DAL unit, element and version of excitation vector.
                Units must be consistent to parameters.
N3,NEL3,NVR3  - Output DAL unit, element and version which contains
                the power input.
ITYPE         - Type of structure.
              = CYLINDER - Cylinder.
              = PLATE    - Flat plate.
```

Parameters

Cylinder

RHO	Mass density of cylinder.
H	Thickness of cylinder.
R	Radius of cylinder.
AREA	Area of turbulent boundary layer.
CL	Longitudinal wave speed of cylinder.
VEL	Free stream velocity.
TBLT	Turbulent Boundary Layer Thickness.
RATU	Ratio of turbulent boundary layer velocity to free free stream velocity. Default = .75

Flat plate

RHO	Mass density of plate
H	Thickness of plate

AREA	Area of turbulent boundary layer.
CL	Longitudinal wave speed of plate
VEL	Free stream velocity.
TBLT	Turbulent Boundary Layer Thickness.
RATU	Ratio of turbulent boundary layer velocity to free free stream velocity. Default = .75

3.170. RUN TCPY

RUN=TCPY Command

Purpose

Copy all enabled elements from the specified DAL unit to the specified tape unit.

Command

```
RUN=TCPY  NU,NTAP
```

Where

```
NU   - DAL unit to copy from.
NTAP - Tape unit to copy to.
```

Notes

The first file on tape is a table of contents for all transferred elements. This TOC is a matrix (NELT,13)

This runstream does not do any conversion between machines.

3.171. TDUMP

TDUMP Command

Purpose

Dump a tape (or portion thereof) to a DAL file. All records need not be the same length, and the record lengths and the number of records do not have to be known before the read.

Command

```
TDUMP   NTAPE,IF,JF  NU,ELN,VER  MXLENG,ITYPE
```

Where

```
NTAPE  - Logical unit number of the tape.
IF     - First file to be read.
JF     - Last file to be read.
NU     - DAL unit for output.
ELN    - Output element name.
VER    - Output version name.
MXLENG - Maximum record length. MXLENG is default to 1000 words, and
         may be omitted if this is acceptable.
ITYPE  - Type of matrix that will be produced. The default is ITYPE = 0
```

Notes

The tape is first rewound and positioned to the starting file (IF). Then, a binary read is done of MXLENG words. The actual number of words transmitted will be less than or equal to MXLENG. A DAL element is opened on DAL unit NU with ELN,VER as the element/version name. The first cycle number is set to the file number, and the second cycle number is set to 1 . The number of rows in this element is equal to the number of words transmitted, ie the actual record length. The data that was read is written to the first column of the element, and another binary read is performed. If the new number of words read is the same as before, this data is written to the second column. If a different number of words is read, then the first element is closed, and a new element is opened. The name of the new element will be the same as the first, except the second cycle number is incremented by one. This process is repeated until an end of file is encountered.

When an end of file is sensed, the current element is closed. If the next file is to be read, the first cycle number is incremented and the second cycle number is reset to 1 before repeating the above process.

When the command is over, you will have at least one element per tape file. If a given tape file has uneven record lengths, then an additional element is opened every time the record length changes.

Warning

On the UNIVAC system, caution should be used when using TDUMP to read offsite tapes. Since a UNIVAC word is 36 bits long, it is not evenly divisible by the 8 bit bytes that normally are read from 9 track tapes. In some instances, TDUMP will drop a trailing byte. This can be avoided by assigning your tape with a hardware translator -- eg

```
@ASG,T  TAPE.,U9////FLDATA/EBCDIC/6,tape #
or
@ASG,T  TAPE.,U9//////Q,tape #
```

In the latter case, you may have to translate the characters yourself (ie via CRYPT). If so, remember that there are now 9 bits per character.

Alternate forms

```
TDUMPS   NTAPE,IF,JF  NU,ELN,VER  MXLENG,ITYPE
```

Allows each file on tape to have a 'short' record as the last record in the file. This trailing record is included in the output element for the file as opposed to the normal method of creating a new element for the short record. Blocked symbolic decks (ie 80 characters per record, 60 records per block) may be read from tape using TDUMPS and ITYPE=6. Once read, they may be printed directly via the PRINT command.

```
TDUMPR   NTAPE,IF,IR  NU,ELN,VER  MXLENG,MXREC
```

Allows a tape to be dumped for a specified range of records, rather than doing a whole file. In this case, IF is still the starting file number, but JF is set to IF and is not read (only one file may be accessed in R mode). The variable IR is used in its place. IR is the record number in file IF where reading is to start. ITYPE is no longer input. A type of 0 will always be used for R mode. The variable MXREC is input instead of ITYPE. MXREC indicates the maximum number of records to read. The default is 100 .

```
TDUMPR  4,3,5  1,REC,DUMP  5000  10
```

will read tape unit 4, starting at file 3, record 5. A maximum of 10 records will be read. The output element will be 1,REC,DUMP,3,5 . Note that cycle numbers correspond to file number and starting record number respectively. One additional note: Reading stops when an end-of-file is encountered, the record length changes, or when MXREC records have been read.

3.172. RUN TEMP

RUN=TEMPLATE command

Purpose

This command applies the specified SEMOD template to the specified model.

Command

```
RUN=TEMPLATE=TMPL ARG1,ARG2,...
```

Where

```
TMPL            - Is the name of a template.
ARG1,ARG2,... - The input arguments required by the template.
```

3.173. RUN TGLS

RUN=TGLS Command

Purpose

Prepare index from KEYS/DECK.

Command

```
RUN=TGLS  N1,EL1,VR1  NOUT

Where

   N1,EL1,VR1 - DAL unit, element and version of input KEYS/DECK.
   NOUT       - Fortran unit to direct output.
```

3.174. TMUL

TMUL Command

Purpose

Perform the matrix multiplication

$$A^{t} B = C$$

without having to transpose A.

Command

```
TMUL  N1,EL1,VR1  N2,EL2,VR2  N3,EL3,VR3  (FAC)
```

Where

```
N1,EL1,VR1 - Input matrix   (A).
N2,EL2,VR2 - Input matrix   (B).
N3,EL3,VR3 - Output matrix  (C).
FAC        - An optional multiplication factor that is carried along at
             no extra cost.
```

Notes

If the dimensions of A are (NR,NC) and the dimensions of B are (MR,MC), then NR must equal MR Dimensions of C = (NC,MC)

TMUL will handle matrices of virtually any size, but for matrices larger than 300 by 300, batch mode is advised due to larger allowable core size.

TMUL works with single or double precision matrices as well as complex. Single matrices may be mixed with either double or complex, but double may not be multiplied with complex. If any matrices are double, output is double. If any matrices are complex, output is complex.

If the FAC option is used in complex multiplication, FAC is still considered as a real number, multiplying both real and imaginary portions.

3.175. TOC

TOC Command

Purpose

Print the table of contents of DAL unit NU for sequence numbers I1 througl I2.

Command

```
TOC   NU  I1,I2
```

Where

```
NU    = DAL unit.
I1,I2 = Sequence numbers. I1 and I2 may be omitted, in which case the
        entire TOC will be printed.
```

Example

Get the table of contents for DAL unit 1.

```
TOC 1
```

Related Commands

FIND, LAST, TTYP

3.176. RUN TOCO

RUN=TOCO Command

Purpose

Order a TOC listing by cycle numbers then by element and version name. Also creates a READ/ELTS element if cycle numbers are present.

Command

```
RUN=TOCO  N1,EL1,VR1  N2,EL2,VR2 (IC1,IC2)
```

Where

```
N1,EL1,VR1 - Input TOC listing from FINES command.
N2,EL2,VR2 - Output ordered TOC listing in same format.  If IC1 and
             IC2 are present this is in a READ/ELTS format. (ie. ##READ
             nu, eln, vrn ic1 ic2)
```

Related Commands

RUN=OTOC, RUN=GRLT, TOC, FIND

3.177. RUN TPVL

RUN=TPVL Command

Purpose

Calculate a theoretical percentile value for a Log Normal Distribution given the linear mean value and standard deviation.

Command

```
RUN=TPVL N1,IEL1,IVR1  NU1,IVS1  PERC  N2,IEL2,IVR2  N3,(IEL3,IVR3)
```

Where

N1,IEL1,IVR1 - Input matrix of linear mean values.

NU1,IVS1 - DAL unit and version of output.

PERC - Desired Percentile Value (e.g. 0.95).

N2,IEL2,IVR2 - Input matrix of Standard Deviation values. (Default use standard deviation values calculated based on Variance / Mean Square = 4.1)

N3,IEL3,IVR3 - Input matrix of Skew Factors or if only N3 is present a single Skew Factor. (Default N3 = 1.225)

NU1,'TPVL',IVS1 - Output Theoretical Percentile Values.

Equations

$$P^{x\%} = [LN(m) - 0.5LN((sd/m)\ast\ast 2 + 1)] + SF(K^{x\%})(LN((sd/m)\ast\ast 2 + 1))\ast\ast 0.5$$

Where

LN - Natural Log

m - Linear Mean

sd - Standard deviation for normal distribution

SF - Skew Factor

$K^{x\%}$ - Constant relating the desired X percentile value to a multiple of the standard deviation for a normal distribution.

$P^{x\%}$ - Log(base e) of X percentile value.

The output vector (NU1,'TPVL',IVS1) gives the X percentile value.

3.178. TRANSPOSE

TRANSPOSE Command

Purpose

The TRANSPOSE command performs matrix transposition. As an option, you may 'chop' the matrix prior to the transpose, either by specifying a range of rows/columns or by providing an explicit list of rows and columns.

Command

```
TRANSPOSE N1,EL1,VR1  N2,EL2,VR2  I1,I2  J1,J2
or
TRANSPOSE  N1,EL1,VR1  N2,EL2,VR2  NR,NC
row numbers
column numbers
```

Where

```
N1,EL1,VR1 - Matrix to be transposed.
N2,EL2,VR2 - Transpose.
I1,I2      - Row limits.
J1,J2      - Column limits.
NR,NC      - Number of rows and number of columns to be selected prior
             to transpose.
```

Notes

If I1,I2,J1,J2 are not entered, TRANSPOSE will do a normal matrix transposition. If only I1,I2 are entered, they are assumed to be NR and NC, and TRANSPOSE will then prompt you to enter the desired row and column numbers.

If I1,I2,J1,J2 are all entered, only a portion of the input matrix is considered. Thus the input matrix is reduced in size and then transposed.

TRANSPOSE will work for virtually any size matrix, however for large matrices (>200x200), batch mode is recommended. If you are transposing a time history array just so you can use MAXMIN, don't bother. MAXMIN has been modified to operate on either rows or columns.

Example

1) Input matrix is real 57 by 33

```
   TRANSPOSE  1,INPUT,MATRIX  1,OUTPUT,MATRIX  14,37  1,10
```

Chops matrix to a 24 by 10 using rows 14-37 and cols 1-10 then transposes to a 10 by 24

3.179. TRDAL

TRDAL Command

Purpose

TRDAL converts a binary DAL file from one system to another. Currently this conversion is limited to

```
 1)  UNIVAC to host
-1)  Old DALPRO type UNIVAC to host
 2)  VAX to host
 4)  CRAY to host
 5)  FPS to host (VAX host only)
```

where host may be UNIVAC,VAX,CRAY or FPS.

Command

```
TRDAL  NOUT, NIN,LENBUF,LAST  INTYPE
or
TRDAL  NOUT, N1,EL1,VR1  INTYPE
```

Where

```
NOUT   - Output DAL unit. Upon completion of the command you should be
         able to do a TOC NOUT, etc. Eg TRDAL produces a normal DAL
         file.
INTYPE - Type of input binary file. The following types are  currently
         supported:
            -1  --  Old DALPRO type from UNIVAC
             1  --  New DALPRO type from UNIVAC
             2  --  VAX
             4  --  CRAY
             5  --  FPS
```

and N1,EL1,VR1 or NIN,LENBUF,LENGTH are two different ways of specifying the input binary file.

```
N1,EL1,VR1 - The binary file is in the form of a DAL dataset. Typically
             this is from tape, via TDUMPS command:

                TDUMPS  NTAP,IFL,JFL  N1,EL1,VR1  5000 9
                CYCLE 1,1 (eg)
                TRDAL   NOUT  N1,EL1,VR1  INTYPE
```

NIN,LENBUF,LENGTH

- The binary file is read directly. Use this method when translating files that have been transmitted across a network. It may also be used when a file was created from a COPYFI operation.

NIN - DAL unit number of the binary file. Note that NIN is NOT a legitimate DAL file. The DAL unit number is used only to access the file.

LENBUF - Approximate block length to use when reading the input file. Originally the user had to input the exact block length. Currently TRDAL (ala DALPRO version 5.0+) will compute the correct length that is nearest LENBUF. If you are using an old version of DALPRO the discussion under MAGIC may be of some use.

LENGTH - Number of sectors in the input file. If 0 is used, the length is computed internally.

Notes

TRDAL uses the type codes from the input DAL file to determine what kind of conversion is necessary on an element by element basis. It is important that the type codes are correct, and that each element only contains one type of binary data. Ie: DON'T MIX REALS WITH INTEGERS in the same element, etc.

There are some exceptions to the above rule, although they do not necessarily work with all types of systems.

1) Generally you can mix integers and hollerith, as long as the type code is 4 (hollerith).

2) For CRAY translation, you can mix real and integers as long as the type code is 1 or -1. This exception will probably be extended to the UNIVAC form as well, but it is not available yet. It will never apply to the translation of VAX files.

Options

Several options are available for TRDAL

TRDD Will allow TRDAL to translate disabled datasets.

TRDP Writes a TOC line for each dataset as it is converted. Each line will always come out as a TTYP 2 toc-line.

TRDS Lets you specify the sector length that was used on the foreign machine. On some systems, DAL files may exist in two forms, a DAL form and an EAL form. TRDAL assumes that you are translating a DAL file. If this isn't the case, use the S option, which requires an extra value on the command line. This value is the sector length of the binary file as it existed on the original machine. For Cray EAL files, the sector length is 512. For Vax EAL files, the length is 128.

TRDX Allows you to limit the translation of datasets to those that you specify. The X option requires input immediately after the TRDX command. Up to 100 lines of the form

```
ELN,VER,IC1,IC2
```

may be entered. A blank line will terminate input. If fewer than 4 items are specified on a line, the remaining items will be set to MASK which is an automatic match. You may also use the form 'A***' which would match an item that starts with A. Note that these items must be enclosed in single quotes, and that an * must appear for each character.

TRDZ Writes the TOC line before conversion, then asks if you want to skip it. You may enter Y to skip, N to process, G to process it and all remaining datasets, or Q to quit.

Up to three options may be used at a time. Eg TRDZSD would turn on the Z,S and D options.

More notes - variable sensitivity - IBF$, LNG$, IZP$

IBF$

TRDAL is sensitive to the value of the variable IBF$. The value represents the number of bits to skip at the start of each block of input data. It was designed to read @COPY,G tapes from the Univac (when someone accidently used @COPY,GM instead of @COPY,M). In the @COPY,G case, IBF$ should be set to 72 prior to invoking TRDAL. This skips the two words that COPY,G adds.

LNG$

TRDAL reads the input binary unit in blocks. The block length is always given in terms of an integral number of host words. From this, the number of foreign words per block is calculated as LENG*JBPW/IBPW where LENG is the block length, JBPW is the number of bits per host word and IBPW is the number of bits per foreign word. LNG$ is used to override this calculation. As an example, consider a VAX generated DAL file that has been COPY'ed to tape. The DCL command COPY uses a block length of 512 words. 512*32 = 16384 bits, which is 455 and 1/9th Univac words. When you read the tape on the Sperry, it either reads 455 words and discards the extra 4 bits (which is what TDUMP defaults to) or it will read all the info and produce 456 words. The calculation 456*36/32 comes out to 513, which is wrong. To cure this, you must SET LNG$=512 prior to running TRDAL. (For more info on the TDUMP problem, see TDUMP)

IZP$

Turns on the debug mode for TRDAL. Basically it prints a status of all requests that are made to the translator. IZP$ defaults to 0, if set non-zero the debug mode is turned on.

3.179.1. MAGIC

LENBUF refers to the number of HOST words that will be read at a time. If the file came from tape, then LENBUF must be set to the number of HOST words per tape block. TRDAL reads the input file as a sector addressable file. Thus LENBUF should be divisible by the number of host words in a host sector. In addition, LENBUF 'host' words must parlay into an integral number of 'foreign' words. Eg if your host has N bits per word and the foreign machine has M bits per word, then LENBUF*N must be some multiple of M. (On really old versions of DALPRO (prior to 4.0) the number had to parlay into an integral number of foreign sectors) Some MAGIC numbers are as follows (you can use any multiple. The larger the number the fewer I/O operations required):

Foreign	Host	Smallest LENBUF
CRAY DAL	VAX any	128
CRAY EAL	VAX any	1024
CRAY DAL	UNIVAC	7168
CRAY EAL	UNIVAC	57344
UNIVAC	CRAY DAL	4032
UNIVAC	CRAY EAL	32256
UNIVAC	VAX DAL	2016
UNIVAC	VAX EAL	8064
VAX DAL	UNIVAC	1792
VAX EAL	UNIVAC	7168
VAX any	CRAY DAL	64
VAX any	CRAY EAL	512

3.179.2. HASP

A special mode of TRDAL was created especially for VAX/CRAY interfaces, where CRAY binary files were shipped down a communications line using HASP protocol. The mode is no longer used, but still is an available option for TRDAL. The command form is

```
TRDAL  NOUT  NIN  4
```

(only on VAX, and input must be from CRAY). HASP creates binary files that contain embedded byte-counts. TRDAL reads unit NIN (11 = DAL011) and creates the element 28,CRAY,DUMP. It then continues as if the command TRDAL NOUT 28,CRAY,DUMP 4 had been issued. Note that NIN is not a legitimate DAL file although a DEFINE, ASSIGN statement must be used to attach it to a DAL unit number. Eg

```
$DEFINE  DAL003  CRAY.PUN      (where CRAY.PUN is the HASP generated
                                binary punch file)
$DEFINE  DAL001  CRAY.DAL      (desired output DAL file)
$DALPRO
TRDAL  1,3,4
TOC 1
END
```

3.180. RUN TRMT

RUN=TRMT command

Purpose

This command is used to keep track of tapes transmitted to other sites for DALPRO, VAPEPS etc.

Command

```
RUN=TRMT  NU option
```

Where

```
NU     - DAL unit of file containing transmittal information.
option - Is one of the following:
       = LIST - lists out data.
```

This runstream will prompt for the appropriate information.

3.181. RUN TSTA

```
RUN=TSTAT  N1,EL1,VR1  N2,EL2,VR2

TRANSPOSE VERSION OF STAT COMMAND

    N1,EL1,VR1 = INPUT MATRIX  (NR,NC)
    N2,EL2,VR2 = OUTPUT MEAN AND STANDARD DEVIATION (NR,2)

 FIRST OUTPUT COLUMN IS MEAN. SECOND IS SIGMA
```

3.182. TTYP

TTYP Command

Purpose

Selects the type of table of contents listing. TTYP=1 is for NR-NC TYPE, TTYP=2 is for NI-NJ TYPE.

Command

```
TTYP N
```

Where

```
N - Type code.
  = 1 - Displays NWDS,NR,NC in TOC listings.
  = 2 - Displays NWDS,NJ,NINJ in TOC listings.
```

Notes

TOC listings include the table of contents as listed by TOC, and the one-liners output by LOCAT, FIND, and LAST.

3.183. RUN UPDS

RUN=UPDS Command

Purpose

Copy all symbolic elements from a DAL unit to a Fortran unit.

Command

```
RUN=UPDS  NIN,NOUT
or
RUN=UPDS  NIN NOUT,ELN,VER
```

Where

```
NIN            - Input DAL unit.
NOUT           - Output Fortran unit.
NOUT,ELN,VER   - DAL file to write output to instead of Fortran unit. NOUT
                 may not be the same as NIN or 28.
```

Notes

This command uses a marking technique to transfer only those elements that have been changed since the last update. As part of the command, the user is prompted for one line of symbolic information that is meant to identify the transfer.

3.184. RUN VFIX

RUN=VFIX Command

Purpose

This runstream is used to develop a test spec for a component based on a test fixture vibration survey. The survey should include triaxial accelerometer measurements at each component mounting location. The survey should be performed with the component mounted on the fixture and with all wiring harnesses attached. The survey should be performed 6dB down from the spec level. This runstream computes a test spec for each surveyed location such that the average response of the fixture at all surveyed locations is equal to the spec level, with the following exceptions: The envelope of the normal test specs, as calculated, will not be allowed to exceed the the upper tolerance, as defined by the user, and the envelope of the lateral responses will not be allowed to exceed the spec level for the lateral axes.

Command

```
RUN=VFIX  N1,V1  N2,V2  N3,L3,V3  N4,L4,V4  N5,L5,V5  N6
```

Where

N1,V1 - Unit and version of output from RUN=NORM for the normal accels. If you don't use RUN=NORM, then use 'NMGS' as the element name when you create this element. It should contain the data from the nor- mal accels, one row per frequency and one column per accel. The units of the data are G**2/HZ, in 1/3 octaves.

N2,V2 - This is the same as N1,V1 except that this element contains data for the lateral accels. All other comments from N1,V1 apply.

N3,L3,V3 - This element contains the spec level (qual or acceptance) for the axis normal to the fixture. RUN=LTRP may be used to create this vector. If you don't use RUN=LTRP, create a vector containing the spec level at the 1/3-octave center frequencies which correspond to the frequencies in the data elements. The units of the spec level values must be G**2/HZ.

N4,L4,V4 - This element contains the upper tolerance for the spec level in the axis normal to the fixture. RUN=LTRP may be used to create this element. otherwise, see comments with N3,L3,V3.

N5,L5,V5 - This element contains the lateral spec level in 1/3 octave bands. Use RUN=LTRP to create this element or create it yourself following the guidelines presented under N3,L3,V3.

N6 - This is the unit for output elements. Three elements will be created. The first, N6,NMGS,NORM, will contain the new test spec at each accel position. each location is in the same column as in N1,V1. the second element, N6,NMGS,LAT, contains the lateral response at each accel position which will result from testing to the new test spec. Again, each channel is in the same column as it was in N2,V2. The third, N6,NMGS,AVG, contains the average of the computed test specs.

3.185. RUN WLSQ

RUN=WLSQ Command

Purpose

Weighted Least SQuares. Solves the equation

```
D = Z*C
```

for C, where D and Z are known.

Command

```
RUN=WLSQ  N1,EL1,VR1  N2,EL2,VR2  N3,EL3,VR3  N4,EL4,VR4
```

Where

```
N1,EL1,VR1 - input Z matrix  (N,M)
N2,EL2,VR2 - input D vector  (N,1)
N3,EL3,VR3 - output C vector  (M,1)
N4,EL4,VR4 - Optional input weighing matrix  (N,1)
```

Choleski Decomp. is used for inversion. If this is not desirable, SET ICD$=0 prior to issuing command.

3.186. WTAPE

WTAPE Command

Purpose

Writes element to tape.

Command

```
WTAPE  N1,EL1,VR1  NTAP  IFL  IREC
```

Where

```
N1,EL1,VR1 - Input DAL element, size=NR,NC.
NTAP       - Tape unit.
IFL        - File on tape where writing is to start.
IREC       - Record where writing is to start.
```

Notes

Writing is done in 'NC' blocks of 'NR' words each IFL and IREC will default to 1 if not input.

End of File (EOF)

After the write, two EOFs are written after the data, and the tape is backspaced one EOF. This marking may be turned off by using the command:

```
WTAPNE  N1,EL1,VR1  NTAP,IFL,IREC
```

4. PRINTING AND PLOTTING

This chapter gives the commands used in displaying output. These include tabular as well as graphical displays.

4.1. PLOT

PLOT Command

Purpose

PLOT linear, semi-log, or log-log graphs of ordered data contained in DAL elements. This command is only available on the UNIVAC and VAX.

Command

```
PLOT   NX,ELX,VRX  NY,ELY,VRY  IFLG (I1,I2) J,K
```

Where

```
NX,ELX,ELY - Abscissa array
NY,ELY,VRY - Ordinate array
IFLG       - Option flag for TITLES, GRIDS, SCALES, etc.
I1,I2      - Row limits for both abscissa and ordinate. If omitted,
             I1=1, and I2=minimum row dimension of abscissa or ordinate
             array.
J          - Ordinate column to be plotted. If PLOTC is used instead of
             PLOT, both columns J and J+1 will be plotted on the same
             frame. Column J will be a solid curve, J+1 a dashed curve.
K          - Abscissa column to be used as abscissa. Only one column is
             used, even in the case of PLOTC.
```

```
IFLG     IFLG is in general a 5 digit number, where each digit
         represents a different data flag. The meaning of each digit is
         explained below:
-----------------------------------------------------------------------
ONES     XXXX0 - Linear plot
         XXXX1 - Semi-log plot (Logarithmic X)
         XXXX2 - Semi-log plot (Logarithmic Y)
         XXXX3 - Log-log plot
         XXXX4 - Linear plot, with the axes crossing at (0.,0.)
         XXXX5 - Linear plot, axes cross at (0.,0.), and equal X and Y
                 axis lengths, according to whichever has the larger
                 range.

TENS     XXX0X - Use previous abscissa and ordinate labels, or use
                 element names if labels have not been read.
         XXX1X - Read abscissa label
         XXX2X - Read ordinate label
         XXX3X - Read abscissa and ordinate label respectively.
```

100 XX0XX - Use previous abscissa and ordinate limits or scale the data automatically if limits have never been read.
XX1XX - Read abscissa limits as XMIN,XMAX these limits remain fixed until they are reset. To resume automatic scaling, use limits of 0.,0.
XX2XX - Read ordinate limits as YMIN,YMAX The same conditions apply as above.
XX3XX - Read abscissa and ordinate limits respectively.
XMIN,XMAX / YMIN,YMAX (Two cards)

1000 X0XXX - Use previous title. Initial title is blank.
X1XXX - Read title
X2XXX - Read two titles
X3XXX - Read three titles

10000 0XXXX - Use previous grid information. The initial value is no grids.
1XXXX - Enter IX,IY as number of grids per major X axis and Y axis respectively.
-1XXXX - Same as above, but use dashed grid lines instead of solid ones.

Any data that is required from the flags above is entered in the order they are described above. As an example, the flag 11230 would require the following data:

```
THIS IS THE ABSCISSA LABEL
THIS IS THE ORDINATE LABEL
0.,100.       @ THIS IS YMIN,YMAX
THIS IS THE TITLE
1,2           @ ONE GRID PER MAJOR X, TWO FOR Y
```

Plot types

The PLOT command may be used to generate COMP80 plots, TEKTRONIX plots, VERSATEC plots, or plots on other devices. Device selection is accomplished through the command PSET. Please see the documentation on PSET for more information.

VAPEPS uses DISSPLA as the heart of its plotting software. If your system does not have DISSPLA your plots will be severely limited. This will eventually be improved.

Command variations

```
PLOMxx  N1,EL1,VR1  N2,EL2,VR2  IFLAG,...
I1,I2,ICOD  I1,I2,ICOD  I1,I2,ICOD,...
```

xx - Pair of integers (03,05,10,etc) indicating the number of distinct curves to draw for this plot. Each curve is taken from a particular block of values from the input X and Y vectors. The values I1,I2,ICOD must be input for each curve. Thus exactly NCURVE*3 values must be input immediately following the command.

xx - XX if you actually use PLOMXX, then the plot command will look for the variable NPL$, and use that value to get the number of curves. Eg

```
SET NPL$=3
PLOMXX .................
```

would do the same as

```
PLOM03 ............
```

I1,I2 - Row limits for this curve. ie 1,100 mean that rows 1 through 100 will be used to draw this curve. The row limits apply to both the X and Y vectors.

```
ICOD  - Draw code
      = 0  use a solid line to connect points.
      = -I Don't connect points, but mark every Ith point.
      = +I Connect points with solid line and mark every Ith point.
      = 10000 Connect points with a dashed line.
```

Example

```
PLOM03  1,X,VEC  1,Y,VEC
1,100,10  101,200,10
201,300,10
```

Would draw three curves. Each would be a solid line but would have every tenth point marked with a symbol. A different symbol is used for each curve. Note: A maximum of twenty curves can be plotted on one plot. Curve one would consist of points 1-100, two would consist of points 101-200, etc. This implies that both X and Y elements must have at least 300 rows.

```
PLOPxx  N1,EL1,VR1  N2,EL2,VR2  .....
ICOD  ICOD  ICOD ...
```

This is essentially the same as PLOM, except that each curve is taken from successive columns. This implies that all curves have the same number of points. ICOD has the same meaning as for PLOM. You may .se XX explicitly, and then the number of curves will be taken from the variable NPL$.

Example

```
PLOP03  1,X,VEC  1,Y,MAT  11030  10,50  3
```

```
10,10,10
ABSCISSA LABEL
ORDINATE LABEL
....
```

Would plot columns 3,4 and 5 from 1,Y,MAT versus columns 1,2 and 3 from 1,X,VEC. If 1,X,VEC is really a vector, then column 1 would be used for each curve's abscissa. Note that only rows 10-50 would be used for each curve.

```
PLOTB  N1,EL1,VR1  N2,EL2,VR2  IFLAG,...
WIDTH
```

Would plot a histogram (bar graph). WIDTH is a floating point number that determines the width of the bars. A positive number gives the width in inches, a negative number gives the ratio of bar width to point spacing. See the DISSPLA manual for details.

```
PLOT3D  N1,EL1,VR1  (XV,YV,ZV  XMN,XST,YMN,YST  IX,IY)
```

Produces 3D surface plots of a rectangular array. (VAX only)

```
   XV,YV,ZV - View point in coord. system of resp. axis
              Default = (-1.5,-1.5,-1.5)
   XMN,XST  - X minimum and step size for generating X axis
              Default = 1.,1.
   YMN,YST  - Y minimum and step size for generating Y axis
              Default = 1.,1.
   IX,IY    - Use every IXth row and IYth column.
              Default = (1,1)
```

You can now label each axis by using the command PLO3DL. It will then prompt for three LABEL lines. They are X, Y and Z labels respectively.

You can use the 'trapped' title approach to get titles. The FLAG sub-command in PSET can be used to read the title(s), then use PPF to specify the number of trapped titles you want.

The PPF subcommand (in PSET) can also be used to change the scale of your 3D plot. Eg

```
   PPF  1,0,3
   AX,AY,XO,YO> 1. .9  0.  .05
```

will allow up to three 'trapped' titles, and increase the size of the plot so that it pretty much fills up the page.

Related Commands

PSET

4.2. PRCRN

PRCRN Command

Purpose

Print the corners of a matrix. (Single precision only)

Command

```
PRCRN  NU,ELN,VER
```

Where

NU,ELN,VER - The DAL element to be printed.

Description

Output from PRCRN is to unit 6 and usually is a four by four matrix that includes the first two columns and last two columns of the first two rows and the last two rows. If the matrix under consideration has fewer than four rows or columns, the output is reduced accordingly.

4.3. PRINT

PRINT Command

Purpose

Print all or a portion of a DAL element. The DAL element may be a vector, a matrix, or a symbolic (types 5,6) element. PRINT enables the user to list parts of an element in virtually any format. The PRINT section may be entered via several com- mands. The major difference between commands is the output format.

General form for symbolic elements

```
PRIxxx  NU,ELN,VER  LIN1,LIN2  NOUT
```

Allowable forms: PRINT, PRIND, PRINZ

Formats are not allowed (currently) for printing of symbolic elements. LIN1, LIN2 correspond to range of lines to be printed. If LIN1,LIN2 are omitted, the entire symbolic element is printed. NOUT is optional, an indicates the FORTRAN unit where dataset is to be written. NOUT defaults to 6.

PRINT - Counts lines and writes a header at top of each page. Number of lines per page may be set via '##SET NLP$=xx' prior to issuing command. Lines are numbered in the output listing.

PRIND - Numbers lines, but does not do page ejects nor does it write any header. Suitable for demand terminal usage.

PRINZ - Does not number lines or do any page ejects, etc.

General form for matrix elements

```
PRIxxx  NU,ELN,VER  I1,I2 J1,J2  IC,JC  NOUT
(format) in some cases
```

Allowable forms: PRINT, PRIND, PRINZ, PRINZF, PRIZ*, PRIZF* PRICF*, PRINF*, PRITF*, PRINX**

* Indicates forms which require an input format, note that * should NOT be entered as part of the command

** Indicates forms that require additional data

Matrix commands:

```
PRINT  NU,ELN,VER  I1,I2  J1,J2  IC,JC  NOUT
```

Prints ten cols per line, labels row and col numbers.Vectors are printed with one value per line.

```
PRIND  NU,ELN,VER  I1,I2  J1,J2  IC,JC  NOUT
```

Same as PRINT, except only 6 cols per line. The D stands for Demand (eg 80 column terminal).

```
PRINX  NU,ELN,VER  I1,I2  J1,J2  IC,JC  NOUT
NCPL,IF1,IF2,KR,KC>
FORMAT:  (iff IF1 .NE. 0)
FORMAT:  (iff IF2 .NE. 0)
ROW IDS> (iff KR  .NE. 0)
COL IDS> (iff KC  .NE. 0)
```

Same as PRINT and PRIND except that you specify the number of columns per line, and can modify the format for printing the heading and the format for printing the main body of the dataset. You can even substitute values for the row and column numbers (eg use names and specify A4 for the format, etc)

NCPL - Number of data columns per line (must be specified)
IF1 - If non-zero you are prompted to enter the format for the header (default = 0)
IF2 - If non-zero you are prompted to enter the format for the rest of the printing (default = 0)
KR - If non-zero you are asked for NR IDs that will be used instead of the row numbers. On most systems you may specify integer,real or hollerith data, as long as you provide for it in the format statements.
KC - If non-zero you are asked for NC IDs that will be used instead of the column numbers.

```
PRICF  NU,ELN,VER  I1,I2  J1,J2  IC,JC  NOUT
(FORMAT)
```

Print by columns, one FORTRAN write per column. Matrix is printed according to specified FORMAT. It is convenient to think of PRICF as printing the transpose of a matrix. Eg a vector will be printed across the page, as opposed to one value per line.

```
PRITF  NU,ELN,VER  I1,I2  J1,J2  IC,JC  NOUT
(FORMAT)
```

Print entire matrix by rows using one FORTRAN write statement. Matrix is printed according to specified FORMAT.

```
PRINF  NU,ELN,VER  I1,I2  J1,J2  IC,JC  NOUT
(FORMAT)
```

Print by rows, one FORTRAN write per row. Matrix is printed according to specified FORMAT.

PRINZ NU,ELN,VER I1,I2 J1,J2 NOUT
Print non-zero terms only. The format is (3(2I4,G15.7))

PRINZF NU,ELN,VER I1,I2 J1,J2 NOUT FAC
Print terms whose absolute values exceed FAC. The same format is used as with PRINZ

PRIZ NU,ELN,VER I1,I2 J1,J2 NOUT
(format)
Print non-zero terms only. The format must be input, and must be of the form (3(2Ix,Fxx.x)) where F may be replaced by E, G or D. The x fields should be replaced with digits of your choice.

PRIZF NU,ELN,VER I1,I2 J1,J2 NOUT FAC
(format)
Print terms whose absolute values exceed FAC. Format requirements are the same as PRIZ above.

General Notes

IC, JC indicate row and col increment, respectively. If IC, JC are not given, default is 1,1. If NOUT is specified, print is to FORTRAN unit NOUT. If NOUT is input as positive, FORTRAN unit is rewound after write. If negative, unit is left open (for later writes)

Some of the above forms require input of a format. This FORMAT must be enclosed by parenthesis and may be up to 60 words long. On the UNIVAC, 60 words gives you 360 characters. On the VAX, you may use 240 characters. The format reader is smart, in that it keeps reading format lines until the parentheses are balanced or until you exceed the maximum character count. Parentheses within hollerith fields are ignored (eg. '((' or 2H(( )

A special format feature is available, as follows: Assume you input a format as xx(10I5) where xx is a two digit number. Later in the execution you may just type xx and the (10I5) format will be recalled from an internal table.

If matrix size is NR by NC, then the following are default values:

I1,I2	1,NR
J1,J2	1,NC
IC,JC	1,1
NOUT	6

If J1 is specified, then I1,I2,J2 must also be specified If I1 is specified, then I2 must also be specified. Etc.

The FORTRAN logic for PRICF is:

```
      DO 10 J=1,NC
      WRITE(NOUT,FORMAT)  (A(I,J),I=1,NR)
   10 CONTINUE
```

PRINF prints according to the following logic:

```
      DO 10 I=1,NR
      WRITE(NOUT,FORMAT)  (A(I,J)J=1,NC)
   10 CONTINUE
```

PRITF does not use a do-loop, but prints according to:

```
   WRITE(NOUT,FORMAT) ((A(I,J),J=1,NC),I=1,NR)
```

Advanced notes

Some elements (eg JLOC BTAB) are special in that they use NJ to logically break each column into sub-records. The above print commands pretty much ignore this field. You can make the print commands pay attention to the field, by use of the variable NJ$.

```
    If NJ$ = 0 - Ignore NJ. (default)
           = 1 - Read NJ, and treat input row limits as NI limits,
                 treating each new logical record as a new column.
           = 2 - Mode shape type elements. Use NJ as for NJ$=1, but treat
                 each new physical column (eg mode shape) as a new block.
```

If you don't understand the above paragraph, SET NJ$=x prior to some prints (of elements with NJ.ne.1) and see what happens.

4.4. PSET

PSET Command

Purpose

PSET is the plot interface to VAPEPS. It controls the plot device selection and contains many sub-commands associated with plotting.

Introduction

VAPEPS can be mapped in one of three plot configurations: No plots, primitive plots or DISSPLA plots. The latter is by far the nicest although all systems may not have access to DISSPLA software. This system dependence is also seen in the number and type of allowable plot devices. A partial list of devices is contained at the end of this description. Hopefully your devices are contained in this list.

Command

```
PSET
```

Enters PSET and prompts for sub-commands. PSET exits on receipt of DONE or a blank record.

```
PSET  DTYPE  ARG1,ARG2,..
```

Sets the device type. This is the same as

```
PSET
PSET> DEVICE  DTYPE  ARG1,ARG2,..
PSET> DONE
```

```
PSET  0
```

Closes current plot frame. It is used when less than N*M plots are desired with PPF N,M. This usage is equivalent to

```
PSET
PSET> EPL
PSET> DONE
```

4.4.1. COLOR

Color plots (DISSPLA only)

Colors may be used on some devices (eg Tek 4105). See the "CURVE COLOR" command for more info.

4.4.2. COPY

COPY Command

Purpose

Turns on automatic copy mode for Tektronix devices. It may or may not work for your system.

Command

```
COPY
```

4.4.3. CURVE

CURVE MARKERS Command (DISSPLA only)

Purpose

Sets marker symbols that will be used in multiple curve plots. Up to 15 integers may be specified (M1-M15). Currently this is available only for DISSPLA plots. See DISSPLA manual (A4.6) for table relating numbers to symbols.

Command

```
CURVE  MARKERS   M1,M2,M3,M4,...
```

Where

M1,M2,M3,M4,...
- Curve marker numbers as defined in the DISSPLA manual.

CURVE COLORS Command (DISSPLA only)

Purpose

Turns on the color mode. From 0 to 15 colors may be used. Zero colors turns off the mode. The first color is used for the first curve on all plots. For multiple curves per plot (PLOMxx or PLOPxx), the curves are drawn in the colors specified. If you have more curves than specified colors, WHITE will be used. BLACK is invisible (at least on a Tek 4105) and probably shouldn't be used.

Command

```
CURVE  COLORS  RED,BLUE,GREEN,YELLOW,MAGENTA,CYAN,WHITE,BLACK
```

Where

RED,BLUE,...
- Color of each curve. The order above is the default.

CURVE SHADES Command (DISSPLA only)

Purpose

Allows you to use different shade patterns for each curve in a multiple curve plot. The pattern numbers correspond to DISSPLA's call to SHDPAT. (Note that shades will only be used if the SHADE command is also called).

Command

```
CURVE  SHADES  IPAT1,IPAT2,IPAT3,...
```

Where

IPAT1,IPAT2,...
- The pattern numbers to be used for shading as defined in the DISSPLA Manual.

4.4.4. DELAY

DELAY Command

Purpose

Sets a delay factor (in milliseconds) for the Tektronix copy mode. On older scopes, plot information can be lost if sent to the terminal while a copy is in progress. The delay is initiated after a copy, so that nothing is sent to the terminal during this period.

Command

```
DELAY  MSEC
```

Where

MSEC - Is the delay time in milliseconds.

4.4.5. DEVICE

DEVICE Command

Purpose

Sets the device to be used for plotting. This command must precede the use of any plot command. The legal devices will vary from site to site. The current devices (for your system) can be listed by using DEVICE HELP . A list of devices and their arguments appears later in this documentation.

Command

```
DEVICE  DTYPE  (ARG1,ARG2,ARG3,...)
```

Where

DTYPE - Is four characters specifying the device type.
(ARG1,ARG2,ARG3,...)
- Optional arguments for some devices.

Notes

DEVICE may be executed as a one liner from regular VAPEPS mode by using the command PSET DTYPE where DTYPE is the device type desired.

DEVICE implicitly causes an IPP (Initialize Plot Parameters) This means that any PSET sub-commands that were issued prior to DEVICE may be overwritten (eg. PPF,PAGE,COPY,etc) Thus it is a good idea to use DEVICE as the first PSET sub-command.

4.4.5.1. LIST

Devices

The following is a partial list of devices.

--Tektronix--

```
DEVICE  TEK ITYP,ICPS,IOPT,IRES,IADDR

    ITYP  - Tektronix type (4014,4051,...)
    ICPS  - Characters per second (default=120)
    IOPT  - Option (use 0, which is default)
    IRES  - Resolution (default = 0, small screen) use 1 for large
            screen.
    IADDR - Plot station (use 0, which is default)
```

--Megatek--

```
DEVICE  MEGA  IDEV

    IDEV = device number (1 or 2, 1 is default)
```

--Regis--

```
DEVICE REGIS

    Uses REGIS commands for output to VT240's, etc.
```

--Save--

```
DEVICE  SAVE

    Uses DISSPLA's COMPRS routine to write plots to PLT2.DAT
```

--Versatec--

```
DEVICE  VERS
```

For systems without DISSPLA the above list may be severly truncated.

Normally only TEK plots are supported.

4.4.6. DONE

DONE Command

Purpose

Terminates the PSET processor and returns to regular VAPEPS control mode. A carriage return will serve the same purpose.

Command

```
DONE
```

4.4.7. EPL

EPL Command

Purpose

End plot directive. If you are plotting several plots per frame (ie using PPF), you can terminate the current frame by issuing the EPL sub-command. For example if PPF 2,2 was used, you are plotting 4 plots per frame. If you only have 3 plots (or only want 3 plots on a particular frame) you should end the frame with EPL. The command PSET 0 can be issued directly from VAPEPS control mode to accomplish the same purpose. This one-liner mode will leave you in VAPEPS rather than continuing with PSET.

Command

```
EPL
```

4.4.8. FLAG

FLAG Command

Purpose

Activates the label reader. XXXXX is a five digit integer. Each digit pertains to a particular function, as follows:

Command

```
FLAG  XXXXX
-DATA-
```

Where

XXXXX - Option flag from following list:

XXXX0 - linear X, linear Y
XXXX1 - log X, linear Y
XXXX2 - linear X, log Y
XXXX3 - log X, log Y

XXX0X - use previous labels for X and Y axis. Default = no axis.
XXX1X - read new X label. Use old Y label.
XXX2X - read new Y label. Use old X label.
XXX3X - read new X label, then read new Y label.

XX0XX - use previous maxmins for X and Y. (Default = automatic scaling)
XX1XX - read XMIN,XMAX for abscissa scale. Note: 0,0 implies that automatic scaling should be turned on.
XX2XX - read YMIN,YMAX for ordinate scale.
XX3XX - read XMIN,XMAX then read YMIN,YMAX

X0XXX - use previous title(s). Default = no titles.
X1XXX - read one title
X2XXX - read two titles
X3XXX - read three titles

0XXXX - use previous grid definition (default = no grids)
1XXXX - read IGX,IGY to define grids.
-1XXXX - same as above except use dashed lines for grids.

-DATA-

Flag reads the appropriate data immediately following the flag command. The order for reads corresponds to the order the items are discussed above. The input items are stored in an internal table. Subsequent plot commands can use this data or overwrite portions through individual use of XXXXX if it is available with the particular plot command being used.

4.4.9. GRACE

GRACE Command

Purpose

Sets the grace margin for the plot. The default margin is 0, which means that if a data point (or part of a symbol) falls off of the plot, it will not be plotted. GMARG is the distance in inches that a point may be off the plot and still be plotted.

Command

```
GRACE  GMARG
```

Where

GMARG - Distance in inches that a point may be off the plot and still be plotted.

4.4.10. HGT

HGT Command

Purpose

Sets the scale factor for plotting characters in DISSPLA. Scale must be a positive real number (decimal point) that is less than or equal to 1.

Command

```
HGT  SCALE
```

Where

SCALE - A positive real number that is less than or equal to 1.

4.4.11. IPP

IPP Command

Purpose

Initialize Plot Parameters. This resets all parameters to their default values.

Command

```
IPP
```

4.4.12. LEGEND

LEGEND Command (DISSPLA Only)

Purpose

Provides for specification of legends for multiple curve plots. Legends appear on right side of plot area. Default location allows room for up to 6 characters.

Command

```
LEGEND  ICURVE, 'Legend'
or
LEGEND  -LINES,MAXCH,X0,Y0
```

Where

```
ICURVE   - Curve number.
'Legend' - Text to be plotted next to legend.
-LINES   - Set the maximum number of legend lines to LINES.
MAXCH    - Maximum number of characters per legend.
X0,Y0    - X and Y location of legend.
```

Notes on Option 2

Sets maximum number of legend lines, and maximum characters per legend. (For some reason, the size calculation doesn't work to well, so be generous on max chars per line) Defaults are 10 lines of 20 characters each. X0,Y0 control location of legend. X0,Y0 are expressed in terms of graphs size. Defaults are 1.01 and .25 which places legend just to the right of the plot, starting 1/4 of the way down the plot.

4.4.13. MIXALF

MIXALF Command (DISSPLA Only)

Purpose

Sets up mixed alphabets in DISSPLA. Up to 6 character sets may be in use at one time. ISET is the set number (1 - 6). CHAR is the character that triggers a particular alphabet (should be enclosed in single quotes). TYPE indicates the particular alphabet and may be STAND,L/CSTD,GREEK,INSTR, etc. See DISSPLA manual (under MXALF) for details. Most forms can be abbreviated to four characters. If more than 4 characters are used, or the TYPE name includes a slash, you must enclose TYPE in single quotes.

Command

```
MIXALF  ISET 'CHAR'  'TYPE'
```

Where

```
ISET   - Character set number.
'CHAR' - Character that triggers the particular alphabet.
'TYPE' - Alphabet.
```

4.4.14. PAGE

PAGE Command

Purpose

Sets the page size. A border is drawn around each frame. PX gives the length (inches) of the horizontal lines. PY gives the length (inches) of the vertical lines. The default is 11.,8.5

Command

```
PAGE  PX,PY
```

Where

PX,PY - Page size. PX give length in inches of horizontal lines.
PY give length in inches of vertical lines.

4.4.15. PPF

PPF Command

Purpose

Plots Per Frame. NX = number of plots on a horizontal line. NY = number of plots on a vertical line. NX*NY = total number of plots per frame. NTT = number of trapped titles. A trapped title is essentially a frame title. NTT=1 means that the first title for your plot will be trapped. If the current plot is the first on the frame, the title will be used as the frame title. If the current plot is not the first on the frame, the title will be completely ignored. NTT may be 0,1,2,3 (omitted=0), but should not exceed the number of titles (ie as set through flag). If it exceeds the number of titles, all titles will be considered frame titles.

Command

```
PPF  NX,NY,NTT
```

Where

NX,NY - Number of plots in X and Y directions.
NTT - Number of trapped title lines.

The plots are plotted top-to-bottom, left-to-right, starting in the upper left hand corner. If you want them left-to-right and then top-to-bottom, use a negative number for either NX or NY (not both).

A special case exists where you can position your own plots. PPF 3,0,&NTT tells PPF that you want 3 plots per frame, but that you wish to place them yourself. It will then prompt for AX,AY,X0,Y0 for each plot.

AX,AY - Axis length for X and Y respectively. These are in non-dimensional units. A value of 1. sets the length equal to the page length for that direction. A value of .5 Sets it to half the page length, etc.
X0,Y0 - Offset to lower left corner of plot. These are also input as non-dimensional values with respect to the page lengths.

Note that the non-dimensional form allows you to set your plot locations and then change the page size without messing everything up.

4.4.16. ROT

ROT Command

Purpose

Rotate plots.

Command

```
ROT  IOP
```

Where

```
IOP - Rotation specification.
    = 0     - No rotation of plot fram
    = 1     - Rotate frame 90 degrees
    = MOVIE - Y direction corresponds with paper direction
    = COMIC - X direction corresponds with paper direction
    = AUTO  - Biggest axes placed in direction with most room.
```

4.4.17. SHADE

SHADE Command (DISSPLA Only)

Purpose

Turn shading on or off.

Command

```
SHADE   IPATTERN  METHOD1 METHOD2
```

Where

IPATTERN - Sets the default shade pattern. It is an integer number that corresponds to DISSPLA's patterns. (See a DISSPLA manual under SHADPAT) A pattern of zero turns off shading. If IPATTERN is entered as a negative number, then the pattern for each curve is taken from the CURVE SHADE list. (See documentation under PSET's CURVE command)

METHOD1 - Sets the shading method for the first curve on each plot. Allowable methods are

ABOVE, BELOW, LEFT, RIGHT, XAXIS, or YAXIS

METHOD2 - If entered as non-zero, will cause shading to occur between curve 1 and 2, 2 and 3, etc. for as many curves as there are (eg PLOP and PLOM commands only) Note that METHOD2 defaults to zero, meaning that all shading will be done according to METHOD1.

4.4.18. STYLE

STYLE Command (DISSPLA Only)

Purpose

Sets default style for DISSPLA plots. Allowable styles are SIMPLX, COMPLX, DUPLX, GOTHIC. This option is only available on VAX DISSPLA installations. TOP-STYLE indicates style to be used for top title. SUB-STYLE indicates style for titles 2 and 3, if present. LABEL-STYLE indicates style for labels.

Command

```
STYLE   TOP-STYLE   SUB-STYLE   LABEL-STYLE
```

Where

TOP-STYLE - Style for top title line.
SUB-STYLE - Style for title lines 2 and 3.
LABEL-STYLE - Style for labels.

4.4.19. WAIT

WAIT Command

Purpose

Turns off the automatic copy mode for Tektronix.

Command

```
WAIT
```

4.4.20. ADVANCED USAGE

One of the driving forces behind X-Y plotting in VAPEPS is to make plotting easy. As you become more familiar with the plot capability you will probably get pickier with the plot layouts. With some combinations of N,M plots per frame and page orientations and character heights, you will get less than desirable layouts. The truly adventurous users will place each sub-plot individually, using PPF -N,0,NTRAP but quite often it is sufficient to use equal sized sub-plot areas, and simply tweek some of the sub-plot parameters. The following variables may be set prior to using PPF or prior to PLOTting to influence some of the layout dimensions.

The 'page' is a rectangle with dimensions PX and PY (from PSET's PAGE command) All other dimensions are scaled by these values. Eg the sub-plot Y axis length may be specified as SYY$, but the true size in inches will be SYY$*PY.

PPF parameters

DX$,DY$ - Dimensions of each sub-plot area. Eg for a PPF 2,1 setup, the upper plot is DY$*PY inches above the lower plot.
SXX$,SYY$ - Axis lengths.
XO$,YO$ - Offsets from lower left of sub-plot area to lower left of axis corner.

PLOT parameters

Trapped titles are plotted by defining a sub-plot area. The length of the Y axis of this imaginary sub-plot determines the Y offset to the trapped title. This length may be set by setting SYT$ (which defaults to .7) The character height of the trapped titles are controlled by the variable HGT$ which defaults to 1.

4.5. RITE

RITE Command

Purpose

Print a matrix along with a vector that identifies the rows. This is useful for printing functions of time or responses as a function of vehicle station, etc.

Command

```
RITE  N1,EL1,VR1  N2,EL2,VR2  ITH  NAME  N3,EL3,VR3
```

Where

```
N1,EL1,VR1 - Input abscissa vector. It may be a matrix, but only the
             first column will be printed. This may be of real, integer
             or hollerith type.
N2,EL2,VR2 - Input matrix to be printed. This matrix must be of real
             type (-1 or 1).
ITH        - Print every ITH row. Note that if the elements have a
             different number of rows, the smaller dimension will be
             used. Also, if every ITH value doesn't come out even, the
             last row will still be printed.
NAME       - Optional hollerith name (4 chars) that will be used as the
             heading for the abscissa vector (ie: N1,EL1,VR1)
N3,EL3,VR3 - Optional heading vector (or matrix). The first column must
             contain one hollerith word for each column of N2,EL2,VR2
             that is to be printed. If this vector has fewer rows than
             N2,EL2,VR2 has columns, the lesser dimension will be used.
             If N3,EL3,VR3 has 2 columns, the second column must
             contain integers which indicate which columns of
             N2,EL2,VR2 will be listed.
```

Notes

The purpose of RITE is to print a matrix in a minimum amount of space and with minimum effort. If you are on a terminal it will print 72 characters per line. If not, it will use all 132 characters. Either mode can be forced, by using the command RITER for 132 chars or RITED for 72 chars.

Examples

For a quick and dirty listing of time responses use:

```
RITE  1,TIME,VEC  1,RESP,MAT  10
```

and every 10th time step will be listed. If you want to label responses, use

```
ECOL  1,LABL,VEC  6,1
X,Y,Z,TX,TY,TZ
RITE  1,TIME,VEC  1,RESP,MAT  10  TIME  1,LABL,VEC
```

If you want to select particular columns to list, use

```
ECOL  1,LABL,MAT  6,2
X1,X2,X3,X4,X5,X6
1/6/6
RITE  1,TIME,VEC  1,RESP,MAT  10  TIME  1,LABL,MAT
```

For Vibroacoustic use, you can list all channels in a 'PREP'ed event by:

```
RITE  30,FREQ,FREQ  1,DBDT,VENT  1  FREQ  1,PCHN,VENT
```

where VENT is the name of the event on the local event file.

4.6. TABLE

TABLE Command

Purpose

Provide report type listings of many types of DAL elements.

Command

```
TABLE  N1,EL1,VR1  N2,EL2,VR2  NLPP,NPG,ITAB
-DATA-
or
TABLE  IFLAG  N2,EL2,VR2  NLPP,NPG,ITAB
-DATA-
```

Where

```
N1,EL1,VR1 - First DAL element  (size MR,MC)
N2,EL2,VR2 - Second DAL element  (size NR,NC)
NLPP       - #data lines per output page
NPG        - Starting page number
ITAB       - Integer table number
IFLAG      - Option flag from the following:
           = 'NONE' - Set MC to zero. Print only second element.
           = 'RSEQ' - Set MC to one and build a vector NR long composed
                      of real numbers 1. though NR.
           = 'ISEQ' - Set MC to one and build a vector NR long composed
                      of integer numbers from 1 to NR
-DATA-     - Five lines of data as follows:
```

1. APP,DATE,PROJ

   Where

   APP - 4 character word to be placed infront of page number.
   DATE - 8 character date field.
   PROJ - 16 character project name.

   This line is read using fortran format. This means that the first 4 columns contain APP the next 8 columns contain DATE and the next 16 columns contain PROJ. Spaces and/or commnas are treated as part of the input text.

2. TITLE

- Title of the table. 72 characters maximum.

3. HEAD1
   - First header line. 72 characters maximum.

4. HEAD2
   - Second header line. 72 characters maximum.

5. FORMAT
   - FORTRAN format for each column in the output table.

Description

First the two DAL elements are read into core, and one matrix is formed with NR rows and MC + NC columns. If the first element is specified, MR must be greater than or equal to NR. Typically the first element will contain only one column, giving station descriptors for the rows in the second element. This element may be omitted by replacing N1,EL1,VR1 by IFLG as described above.

After the full matrix is formed, a header is written (see example). Then the first part of the matrix is written using the following FORTRAN WRITE statement:

```
WRITE(6,FORMAT) ((A(I,J),J=1,MC+NC),I=1,NLPP)
```

If NR > NLPP a new page is started with the same heading, except the page is incremented by one. This continues with NLPP lines printed per page until the entire matrix has been listed.

(The above example shows a FORTRAN write to unit 6. You may have TABLE write to unit 7 instead, by using a negative value for NPG. You may override these by setting the variable NTB$ equal to any other FORTRAN unit. In this last case, you may set NTB$ to a negative number which will leave the unit open and positioned for subsequent writes. The default for unit 7 (NPG < 0) is to close the file.

TABTDY

If the command TABTDY is used instead, the date specified in the first data card is ignored (although it must still be specified), and the present date is substituted.

Notes

NLPP,NPG,ITAB are optional and default to 42,1,1 respectively. If one is included, they all should be.

The data cards are read as hollerith, and no decoding is performed. Therefore variables may not be used in these five lines unless the '$$'

notation is used. See the PLOT command write-up for details of '$$'.

Although not indicated above, the format statement may consist of up to 4 lines. Lines are read until the last non blank character is a ')'.

Examples

Print MAX-MIN tables of acceleration for 15 stations. MAX-MIN values are on 'XNA' 'S734' on DAL unit 3, dimensioned 15 by 4

```
TABLE ISEQ 3 XNA S734
  A- 8/23/76EM SSD NO. 734
             ACCELERATION MAX-MINS CASE 734
     STATION   MAX          TIME          MAX          TIME
       NO.     VALUE                      VALUE
 (I10,4E13.7)
```

yields

```
                                                          EM SSD NO. 734
                                                          DATE  8/23/76
                                                          PAGE   A-1
   TABLE     1
         ACCELERATION MAX-MINS CASE 734

 STATION    MAX         TIME          MAX          TIME
   NO.     VALUE                     VALUE

     1  .1577612+01  .6380000+00 -.4750813+01  .2600000-01
     2  .7838695-01  .3740000+00 -.6513848-01  .4980000+00
     3  .2901360-02  .4000000+00 -.3321168-02  .3620000+00
     4  .2008276+00  .4880000+00 -.2122634+00  .4540000+00
     5  .1612680-02  .7180000+00 -.1584313-02  .6840000+00
     6  .4509538+01  .2000000-02  .9846635-01  .7380000+00
     7  .1882007+00  .4880000+00 -.1946676+00  .4560000+00
     8  .1384610-02  .7180000+00 -.1411408-02  .6860000+00
     9  .4510252+01  .4000000-02  .8885011-01  .7380000+00
    10  .6775055-05  .6700000+00 -.5264160-05  .6840000+00
    11  .8695986-03  .4500000+00 -.7007983-03  .5920000+00
    12  .8953631-04  .1000000+01 -.9314436-04  .9700000+00
    13  .1359705+00  .4920000+00 -.1412006+00  .4600000+00
    14  .1196683-02  .6600000+00 -.1074888-02  .6900000+00
    15  .4511631+01  .4000000-02  .7622536-01  .7380000+00
```

If stations were available on DAL002 as 'PXID' 'M734', then change format to F10.2 (instead of I10), and

```
TABLE 2 PXID M734 3 XNA S734
  A- 8/23/76EM SSD NO. 734
```

```
                 ACCELERATION MAX-MINS CASE 734
         STATION    MAX        TIME          MAX          TIME
           NO.     VALUE                    VALUE
     (I10,4E13.7)
```

yields

```
                                                   EM SSD NO. 734
                                                   DATE  8/23/76
                                                   PAGE   A-1
      TABLE    1
           ACCELERATION MAX-MINS CASE 734

     STATION    MAX         TIME           MAX           TIME
       NO.     VALUE                      VALUE

      1.00  .1577612+01  .6380000+00 -.4750813+01  .2600000-01
      2.00  .7838695-01  .3740000+00 -.6513848-01  .4980000+00
      3.00  .2901360-02  .4000000+00 -.3321168-02  .3620000+00
      1.10  .2008276+00  .4880000+00 -.2122634+00  .4540000+00
      2.20  .1612680-02  .7180000+00 -.1584313-02  .6840000+00
      3.30  .4509538+01  .2000000-02  .9846635-01  .7380000+00
     13.10  .1882007+00  .4880000+00 -.1946676+00  .4560000+00
     14.20  .1384610-02  .7180000+00 -.1411408-02  .6860000+00
     15.30  .4510252+01  .4000000-02  .8885011-01  .7380000+00
     16.40  .6775055-05  .6700000+00 -.5264160-05  .6840000+00
     17.50  .8695986-03  .4500000+00 -.7007983-03  .5920000+00
     18.60  .8953631-04  .1000000+01 -.9314436-04  .9700000+00
     31.10  .1359705+00  .4920000+00 -.1412006+00  .4600000+00
     32.20  .1196683-02  .6600000+00 -.1074888-02  .6900000+00
     33.30  .4511631+01  .4000000-02  .7622536-01  .7380000+00
```

Advanced notes ----- 132 character wide tables ----

With the advent of the XEROX printer, it is very convenient to get report quality listings 8.5X11 that have a full 132 characters per line. As documented above, the TABLE command will always start the PROJECT, DATE and PAGE headings in column 57. Similarly, the table heading will always start in column 9. It is possible to alter these tabs, by including two more arguments on the command.

```
TABLE  N1,EL1,VR1  N2,EL2,VR2  NLPP,NPG,ITAB  ISP,JSP
```

Where

ISP = Number of spaces to skip before 'TABLE' (line 4)
JSP = Number of spaces to skip before 'DATE', 'PAGE', etc.

Their default values are 7 and 55 respectively. Ie, there is one space for carriage control, and then 7 (or 55) spaces, which puts the tabs at 9 (and

57, resp).

In addition, the above documentation limits the titles to 72 characters apiece. If ISP is entered as a negative number, two cards of 60 characters each are read for each title. Ie, data lines 2,3 and 4 now become 2A,2B,3A,3C,4A,4B, With only 60 characters per line (ie (10A6)). Note that the titles always begin immediately after the carriage control space, ie column 2. Also note that the format you input must include the carriage control space as well as any other spacing you may have in mind.

4.7. WRITE

WRITE Command

Purpose

Formatted FORTRAN write of variables and/or constants.

Command

```
WRITE  NUNIT  VAL1,VAL2,&VAR1,&NVAR,.....
(FORMAT)
```

Where

```
NUNIT     - Fortran unit for write
          > 0 - Rewind file after write (eg. so it can be read)
          < 0 - Leave file open so that subsequent writes will follow
                it.
VAL1,VAL2 - Constants in free-field format
VAR1,NVAR - VAPEPS variables
FORMAT    - Legal Fortran format, enclosed in parenthesis ().
```

Example

```
WRITE  6,  1.,2.,&FAC,&I,NAME
(1H1,3F10.5,I5,3XA4)
```

Note that 1., 2. and NAME are constants. FAC and I are variables.

5. DATA ENTRY

ENTER Processor

Purpose

The ENTER command reads the following vibroacoustic test event information into the database:

* Event description.
* Frequency analysis information.
* Channel descriptions.
* Acoustic and/or vibration spectral data.

The information is stored in DAL elements in the file you specify.

5.1. FEATURES OF ENTER

Overview

ENTER stores your event's information on a DAL file. This enables the data to be operated on by other VAPEPS commands. Your event's data must be processed by ENTER before you can put it into the standardized VAPEPS database with the PREP processor.

Features

* You can change the name of the data channels to new ones. The new names will be used to identify the channels in the database (e.g. if a channel is labelled 001 in your spectral data, you can change it to MIC1 for the database).

* ENTER will accept data processed in non-standard frequency bands.

* If you are working with card data and you get the cards out of order, ENTER will be able to unscramble and read it.

* If your data is not available all at once but you want to input what you have, or you want it divided into sections (e.g. one for microphones, one for accelerometers), ENTER allows you to input the data in sections.

Restrictions

ENTER can only read spectral data in fixed field FORTRAN format.

Each line of input data must contain data as well as a channel identifier and a sequence number.

5.2. EXAMPLE

A simple ENTER session is shown below. Some general comments about this example are:

- * The output is to DAL unit 1 and SAMP is the database event name.
- * Data channels: 2 microphones, and 1 accelerometer.
- * Data frequency analysis: 4 bands, bandwidth = 10 Hertz (Hz), starting center frequency = 50 Hz.
- * A line of comment will be input for each channel.
- * Frequencies are embedded in the spectral data.

Example

```
--------------------------------------------------------------------------------
  Prompt                    Input                          Description
--------------------------------------------------------------------------------
?                         ENTER 1 SAMP                    Command.

DESC:                     MASS LOAD EFFECT ON EQUIP.      Event
DESC:                     MOUNTING STRUCTURE. CASE 1:     Description.
DESC:                     20#/FT**2. TEST CONFIG:
DESC:                     MASS & C.G. SIMULATORS.

NCHN,NFRQ,ITYP,ITCH>      3,4,0,-2                        Data
NDV,IFOR,IFRQ,ISEQ,IOVR>  2,1,1,0,0                       Parameters.

FORMAT:                   (2E12.4,A4,I4)                  FORTRAN Format.

START,DF>                 20.,10.                         Freq. Definition.

ID,TYPE,UNITS,RMS:        M1,MIC,DBSP                     Channel
DESC:                     REVERBERANT CHAMBER MIC.        Description.
ID,TYPE,UNITS,RMS:        M2,MIC,DBSP
DESC:                     REVERBERANT CHAMBER MIC.
ID,TYPE,UNITS,RMS:        V1,NNAC,G2PH
DESC:                     X AXIS ACCEL, MOUNTING FOOT 1

                           2.0000E+01  1.1000E+02M1     1  Spectral Data.
                           3.0000E+01  1.1700E+02M1     2
                           4.0000E+01  1.2000E+02M1     3
                           5.0000E+01  1.2700E+02M1     4
                           2.0000E+01  1.0900E+02M2     1
                           3.0000E+01  1.2000E+02M2     2
                           4.0000E+01  1.2100E+02M2     3
                           5.0000E+01  1.2800E+02M2     4
```

```
2.0000E+01  7.5000E-04V1     1
3.0000E+01  5.2300E-02V1     2
4.0000E+01  7.2200E-01V1     3
5.0000E+01  9.2300E-01V1     4
    -blank line-
```

A more complex ENTER example is shown in the VAPEPS SAMPLE PROBLEMS manual.

5.3. STEPS TO PERFORM ENTER

Before you start

a. Initialize VAPEPS

b. Assign a permanent mass storage file to the DAL unit specified for the output.

c. Execute VAPEPS.

Perform ENTER

d. Issue the ENTER command.

e. Input four lines of Event Description.

f. Input two lines of Data Parameters which describe the frequency, channel, and spectral data information to follow.

g. Input the FORTRAN Format of the Spectral Data.

h. Input the Frequency Definition information.

i. Input the data Channel Descriptions.

j. Input the Spectral Data.

You're done. Now you can:

- terminate the execution,

- use VAPEPS commands and runstreams to manipulate and display the data,

- continue the database process by using the processor PREP to standardize and further describe the event.

5.4. EXECUTION

The ENTER process is performed in two ways:

a. Fully interactive - once you issue the ENTER command, prompts appear for the required input, you respond from the keyboard. This mode is normally used to get a feel for the way ENTER works. It is not the recommended mode due to the high probability of typographical errors, and the need to redo the entire process if an error is encountered.

b. Batch (recommended) - you create a runstream file on you host system and place in it the input required to execute VAPEPS and perform ENTER. You perform the ENTER session by 'adding' the runstream file to the system. If errors are encountered, you edit the runstream file, correct you mistakes, and 'add' it again - you do this until the ENTER session completes successfully.

5.5. COMMAND FORMAT

Command

```
ENTER NU,VENT(,ISEC NIN,IPNT,IFAT)
-DATA-
```

Where

```
NU   - DAL unit in which to store the output.
VENT - Event name (up to four characters) which you use to identify the
       event in the database; all DAL elements created by the ENTER
       session will have the version name VENT.
ISEC -\
NIN  - \ (Optional) - see more information.
IPNT - /
IFAT -/
-DATA-  Six sections of input that follow the ENTER command line.
        They are input in the order show below.

           1) DESC.............Event Description.
           2) PARAMETERS ......Data Parameters.
           3) FORMAT...........FORTRAN Format.
           4) FREQUENCY........Frequency Definition.
           5) CHANNEL..........Channel Description.
           6) SPECTRAL.........Spectral Data.
        - see more information.
```

5.5.1. ISEC (OPTIONAL ARGUMENT)

ISEC - Section number. All DAL elements created by ENTER will have cycle numbers (C1,C2): ISEC,0. If ISEC = 0 (default) the event is processed as a single section and you cannot add any additional data to it in the future.

If you want to ENTER the data in parts (that is, your event's data is not available all at one time or the data set is too large to be ENTERed in one section), each part must be ENTERed with a unique section number; the first part of your data being entered with section number ISEC = 1, the following parts with section numbers ISEC = 2,3 etc. You must specify the section numbers starting at 1 and continuing in increments of 1 in order for the DAL elements output by ENTER to be successfully processed by PREP.

5.5.2. NIN (OPTIONAL ARGUMENT)

NIN - FORTRAN unit number from which the Channel Description and/or the Spectral Data will be read. Default: NIN=5; FORTRAN unit 5 usually being your terminal, card reader, etc.

- > 0 - all inputs required by ENTER are read from FORTRAN unit NIN.
- < 0 - all inputs required by ENTER are read from FORTRAN unit 5 except the spectral data which is read from FORTRAN unit -NIN.

5.5.3. IPNT (OPTIONAL ARGUMENT)

IPNT - Output print option.

- = 0 - all specifications and general event description will be printed out.
- = 1 - (default) same as IPNT = 0 plus channel and frequency summaries.
- = 2 - same as IPNT = 1 plus a tabular listing of the spectral data.

5.5.4. IFAT (OPTIONAL ARGUMENT)

IFAT - Fatal error option flag.

- = 0 - ENTER will save those channels that are without error. Channels that are noted as being in error would have to be corrected and processed by ENTER as another section.
- = 1 - (default) data will not be saved if any errors are encountered.

5.5.5. EVENT DESCRIPTION

Input four lines of Event Description following the ENTER command line. The Event Description should include details of the event such that other users can understand its general nature. Each line may be up to 72 characters long.

Example

```
desc:   Purpose: To measure the effect of different mass loading
desc:   conditions on xxx vehicles equipment mounting structure.
desc:   Case one: mass load = 20#/ft**2.  Test config:  All up
desc:   dynamic test vehicle with mass and c.g. simulators.
```

5.5.6. DATA PARAMETERS

Input two lines of Data Parameters following the Event Description. This section defines the characteristics of the remaining input.

Parameters

```
NCHAN,NFRQ,ITYP,ITCH
NDVPC,IFORM,IFREQ,ISEQ,IOVER
```

5.5.6.1. NCHAN (DATA PARAMETER)

NCHAN - The number of channels (accelerometers and/or microphones) in the Spectral Data section.

Example

```
NCHAN,nfrq,ityp,itch
ndvpc,iform,ifreq,iseq,iover

NCHAN = 2

   ... for this sample data ...

      data    ID   seq
```

```
1.1000E+02 M1     1  Spectral Data.
1.1700E+02 M1     2
1.0900E+02 M2     1
1.2000E+02 M2     2
```

5.5.6.2. NFRQ (DATA PARAMETER)

NFRQ - The number of frequency bands used to analyze the data.

Example

```
nchan,NFRQ,ityp,itch
ndvpc,iform,ifreq,iseq,iover

NFRQ = 2

    ... for this sample data ...

       data     ID   seq

    1.1000E+02 M1     1  Spectral Data.
    1.1700E+02 M1     2
```

5.5.6.3. ITYPE (DATA PARAMETER)

```
ITYP  - The type of frequency band used to analyze the data.
      = 3 - 1/3 octave (ANSI standard).
      = 1 - whole octave (ANSI standard).
      = 0 - constant bandwidth.
      < 0 - user specified bands.
```

Example

```
nchan,nfrq,ITYP,itch
ndvpc,iform,ifreq,iseq,iover

ITYP = 0

    ... for this sample data ...

      freq        data      ID   seq
```

```
2.0000E+01  1.1000E+02 M1     1  Spectral Data.
3.0000E+01  1.1700E+02 M1     2
4.0000E+01  1.2000E+02 M1     3
```

5.5.6.4. ITCH (DATA PARAMETER)

ITCH - The number of lines of description per channel to be input in the Channel Description section.

- = 0 - input one line describing the measurement type and units to apply to all channels. Note: Use this option only when all data channels are the same in this respect.
- = 1 - input one line per channel of data describing the measurement type and units, and rename the channel.
- = -1 - same as ITCH = 1, except that the channels will not be renamed.
- = 2 - input two lines per channel: the first being the same as ITCH=1, the other is a line of comment.
- = -2 - same as ITCH = 2, except the channels will not be renamed.

Example

```
nchan,nfrq,ityp,ITCH
ndvpc,iform,ifreq,iseq,iover

ITCH = -2

   ... for this sample description ...

   M1,MIC,DBSP                      Channel
   REVERBERANT CHAMBER MIC.         Description.
```

5.5.6.5. NDVPC (DATA PARAMETER)

NDVPC - The number of data values per line in the Spectral Data section. Data values can include the spectral data, the overall level, and the frequencies. The channel ID, and sequence number are not considered data values.

Example

```
nchan,nfrq,ityp,itch
```

```
NDVPC,iform,ifreq,iseq,iover

NDVPC = 2

    ... for this sample data ...

      freq        data       ID   seq

    2.0000E+01  1.1000E+02 M1     1  Spectral Data.
    3.0000E+01  1.1700E+02 M1     2
```

5.5.6.6. IFORM (DATA PARAMETER)

IFORM - The order the information on each line in the Spectral Data section.

IFORM	order
1 -	DATA,ID,SEQUENCE
2 -	DATA,SEQUENCE,ID
3 -	ID,DATA,SEQUENCE
4 -	ID,SEQUENCE,DATA
5 -	SEQUENCE,DATA,ID
6 -	SEQUENCE,ID,DATA

Example

```
nchan,nfrq,ityp,itch
ndvpc,IFORM,ifreq,iseq,iover

IFORM = 1

    ... for this sample data ...

      data     ID   seq

    1.1000E+02 M1     1  Spectral Data.
    1.1700E+02 M1     2
```

5.5.6.7. IFREQ (DATA PARAMETER)

```
IFREQ - The pattern of frequency vs spectral data as it  appears in the
        Spectral Data section.
      =  0 - frequencies are not embedded in the  data.
      =  1 - frequencies and data alternate:  FREQ,DATA,FREQ,DATA,....
      = -1 - data and frequencies alternate: DATA,FREQ,DATA,FREQ,....
```

Example

```
nchan,nfrq,ityp,itch
ndvpc,iform,IFREQ,iseq,iover

IFREQ = 1

   ... for this sample data ...

     freq        data       ID   seq

   2.0000E+01  1.1000E+02 M1      1  Spectral Data.
   3.0000E+01  1.1700E+02 M1      2
```

5.5.6.8. ISEQ (DATA PARAMETER)

```
ISEQ  - Indicates the starting value of the sequence number for each
        channel in the spectral data.
      = 0 - sequence number begins at 1 and continues  sequentially.
      > 0 - sequence number begins at ISEQ and continues  sequentially.
```

Example

```
nchan,nfrq,ityp,itch
ndvpc,iform,ifreq,ISEQ,iover

ISEQ = 0

   ... for this sample data ...

      data     ID   seq

   1.1000E+02 M1      1  Spectral Data.
   1.1700E+02 M1      2
```

5.5.6.9. IOVER (DATA PARAMETER)

IOVER - Indicates where the overall level for each channel appears in the spectral data if at all.

- < 0 - no overall level is included in the data , but it is input as part of the channel description - ITCH cannot be zero.
- = 0 - no overall level is included in the data.
- = 1 - the overall level is included in the data as the first data value for each channel.
- = 2 - the overall level is included in the data as the last data value for each channel.

Example

```
nchan,nfrq,ityp,itch
ndvpc,iform,ifreq,iseq,IOVER

IOVER = 2

   ... for this sample data ...

      data     ID   seq

   1.1000E+02 M1     1  Spectral Data.
   1.1700E+02 M1     2
   1.1779E+02 M1     3 <= overall
```

5.5.7. FORTRAN FORMAT

Input a one line FORTRAN format following the Data Parameters. The FORTRAN format is used to read the spectral data. The format must be enclosed by parenthesis. Refer to the table below for allowable format types.

Data type	Format
Data values........	E,F
Channel ID.........	A,I
Sequence number....	I

where

A - alphanumeric, I - integer, E - floating point, and F - fixed point.

Example

FORMAT: (2E12.4,1X,A4,I4)

... for this sample data ...

```
        data              ID   seq

2.0000E+01  1.1000E+02 M1     1  Spectral Data.
3.0000E+01  1.1700E+02 M1     2
```

5.5.8. FREQUENCY DEFINITION

The frequency definition section of ENTER follows after the FORTRAN format. It defines the frequency bands used for the spectral analysis.

The input for this section is controlled by the parameter ITYP. All frequencies are in units of Hertz.

Input

* For standard whole or 1/3 octaves (ITYP = 1 or 3):

  START - starting center frequency.

* For constant bandwidths (ITYP = 0):

  START,DF - starting center frequency, bandwidth.

* For other bands (ITYP < 0):

  CF(1),....,CF(NFRQ) - center frequencies NFRQ number of values.
  BE(1),....,BE(NFRQ) - beginning (lower) band edges.
  TL(1),....,TL(NFRQ) - trailing (upper) band edges.

Note: CF, BE, and TL values can be entered on multiple lines in free-field format.

5.5.8.1. FREQUENCY DEFINITION EXAMPLE

- Third octave data. Frequency range 50 to 2000 Hertz:

```
ITYP= 3
NFRQ= 17
   START> 50.
```

- Constant 10 Hertz band data. Frequency range 50 to 2000 Hertz:

```
ITYP= 0
NFRQ= 1996
   START,DF> 50.,10.
```

- Non-standard band data. Frequency range 50 to 2000 Hertz:

```
ITYP= -1
NFRQ= 8
   CF= 50.,80.,140.,220.
   CF= 400.,700.,1200.,2000.

   BE= 40.,65.,110.,180.,310.
   BE= 550.,950.,1600.

   TL= 65.,110.,180.,310.,550.,950.
   TL= 1600.,2850.
```

5.5.9. CHANNEL DESCRIPTION

The channel description section of ENTER follows the frequency definition section. The input to this section is controlled by the ITCH parameter.

Input types

- If ITCH = 0, one line is input:

```
TYPE,UNITS
```

- If ITCH = + or - 1, one line is input per channel:

```
ID,TYPE,UNITS(,RMS)       - if ITCH = -1
or
ID,NAME,TYPE,UNITS(,RMS) - if ITCH =  1
```

* If ITCH = + or - 2, two lines are input per channel:

```
ID,TYPE,UNITS(,RMS)
DESC                       - if ITCH = -2
or
ID,NAME,TYPE,UNITS(,RMS)
DESC                       - if ITCH =  2
```

5.5.9.1. ID (CHANNEL DESCRIPTION PARAMETER)

ID - The channel ID is it appears in the spectral data. If the channel is not renamed, then the ID will be used for any post-ENTER references to the channel. Restriction: if the channel ID is an integer, the channel must be renamed to a hollerith name.

Example

```
ID,type,units: M1,MIC,DBSP

   ... for this sample data ...

       data    ID   seq

   1.1000E+02 M1    1  Spectral Data.
   1.1700E+02 M1    2
```

5.5.9.2. NAME (CHANNEL DESCRIPTION PARAMETER)

NAME - New channel ID. A hollerith word (up to four characters long) that will be used for any post-ENTER references to the channel.

Example

```
id,NAME,type,units: 001,M1,MIC,DBSP

   ... for this sample data ...

       data    ID   seq

   1.1000E+02 001   1  Spectral Data.
   1.1700E+02 001   2
```

5.5.9.3. TYPE (CHANNEL DESCRIPTION PARAMETER)

```
TYPE  - Measurement type:
      = MIC  - microphone
      = SMIC - surface microphone
      = NSMC - not surface microphone
      = ACC  - accelerometer
      = VIB  -      "
      = NACC - normal accelerometer
      = NNAC - not normal accelerometer
```

Example

```
id,TYPE,units: M1,MIC,DBSP

   ... for this sample data ...

       data    ID    seq

   1.1000E+02 M1     1  Spectral Data.
   1.1700E+02 M1     2
```

5.5.9.4. UNITS (CHANNEL DESCRIPTION PARAMETER)

```
UNITS - Units of measurement:
      = P2PH - psi**2/Hz
      = PA2H - Pa**2/Hz
      = G2PH - g**2/Hz
      = IA2H - (in/(sec**2))**2/Hz
      = FA2H - (ft/(sec**2))**2/Hz
      = MA2H - (m/(sec**2))**2/Hz
      = DBSP - 10*log(psi**2/8.4144E-18)
      = VLSP - 10*log(g**2/8.4144E-18
      = DBVL - 10*log((m/(sec**2))**2/1.0E-10) or equivalent
```

Example

```
id,type,UNITS: M1,MIC,DBSP

   ... for this sample data ...

       data    ID    seq

   1.1000E+02 M1     1  Spectral Data.
```

```
    1.1700E+02 M1    2
```

5.5.9.5. RMS (CHANNEL DESCRIPTION PARAMETER)

RMS - (optional) Root mean square or overall level. RMS is specified only if the parameter IOVER < 0.

Example

```
IOVER= -1

id,type,units,RMS: M1,MIC,DBSP,117.79

   ... for this sample data ...

       data    ID   seq

   1.1000E+02 M1    1  Spectral Data.
   1.1700E+02 M1    2
```

5.5.9.6. CHANNEL DESCRIPTION

DESC - Channel description (up to 72 characters) used identify location, etc.

Example

```
ITCH= -2

id,type,units: M1,MIC,DBSP
DESC: REVERBERANT CHAMBER MICROPHONE, QUADRANT 2
```

5.5.9.7. NOTES

Note

The order of the descriptions does not have to match the order of the channels in the spectral data section.

(optional) This section can reside, along with the spectral data, on a mass storage file designated as a FORTRAN unit. The choice is controlled by the NIN argument of the ENTER command line.

5.5.10. SPECTRAL DATA

The spectral data section follows the channel description section and is the last part of the input to ENTER. The contents, internal form, and arrangement of the information in this section has been specified in the previous sections.

Each line in this section must contain three types of data:
- -an integer card sequence number,
- -a channel ID, and
- -numeric data.

Numeric data can include spectral, frequency, and overall values.

The last line of the spectral data section must be a blank (a carriage return will do if the user is performing ENTER interactively).

Example

```
      data           ID    seq

2.0000E+01  1.1000E+02M1    1
3.0000E+01  1.1700E+02M1    2
2.0000E+01  1.0900E+02M2    1
3.0000E+01  1.2000E+02M2    2
      -blank line-
```

... for these sample specifications ...

```
NCHN,NFRQ,ityp,itch> 2,2,0,-2
NDVPC,IFORM,IFREQ,ISEQ,iovr> 2,1,1,0,0
FORMAT: (2E12.4,A4,I4)
```

Note

(optional) This section can reside on a mass storage file designated as a FORTRAN unit. The choice is controlled by the NIN argument of the ENTER command line.

5.6. OUTPUT DAL ELEMENTS

unit,element,version - description (size - rows x columns)

NU,EVNT,NAME - Event description. The element is one column containing the four 72 character lines of event description (288 x 1).

NU,FREQ,NAME - Frequency table. The element is a three column matrix; each column has one value per input frequency band; the first column contains the center frequency of the band; the second and third contain the upper and lower band limits respectively (#frequencies x 3).

NU,CHAN,NAME - Channel description. The element is a matrix with one column per channel; each column contains the channel name, channel type, units type, and 72 characters of description (23 x #channels).

NU,SPDT,NAME - Spectral data. The element is a matrix with one column per channel and one row per frequency band; each row contains the spectral data value for the corresponding frequency band (#frequencies x #channels).

NU,RMSI,NAME - RMS or overall data - created only if this information was input during ENTER. The element is a vector containing one value per channel (#channels x 1).

A more detailed description of the output of ENTER can be found in the VAPEPS PROGRAMMERS MANUAL.

6. DATABASE PREPARATION

PREP Processor

Purpose

The PREP processor is used to define a database event and standardize raw data (input from the ENTER processor) associated with an event. PREP is divided into sections which are accessed by issuing subcommands within PREP. The sections of PREP allow the user to input or modify book-keeping information (see the BOOK subcommand), channel descriptions (see the CHANnel subcommand), a configuration tree (see the CONFiguration subcommand), and data modules (see the MODUles subcommand) as well as perform data standardization (see the CHECK subcommand). PREP is the phase of the database entry process where key words and physical parameters are input that will later be searched on by users operating in the database query mode.

Command

```
PREP NU,VENT,(NSEC)
```

Where

```
NU   - The DAL unit where a previously PREPed event resides or where
       raw data read in during the ENTER processor resides.
VENT - The event name defined during the ENTER process.
NSEC - Optional flag indicating the status of the raw data read in
       during the ENTER process.  This argument need only be specified
       when the PREP command is issued for the first time following the
       processing of an event by ENTER. For a description of the
       conversions performed on the raw data from ENTER, see the CHECK
       subcommand.
     > 0 - Indicates that 1 through NSEC sections of raw data were
           created during the ENTER process.
     = 0 - Only one section (section 0) was input during ENTER.
     < 0 - (default) The event has been previously processed by PREP
```

6.1. BOOK

BOOK Subprocessor

Purpose

The book-keeping section of the PREP is entered by issuing the BOOK subcommand. This section allows for the entry into the database of basic book-keeping information for the event being PREPed. The entry of book-keeping information has been limited to fields describing key aspects the event. The descriptive fields are input through the specification of subcommands within BOOK. Subcommands to input, list, and edit these fields are described in the additional HELP information available below.

Command

```
BOOK
```

6.1.1. APROPOS

APROPOS Command

Purpose

This command displays the words which have been used in the BOOK section in previous events. It also allows you to list the definitions of those words as defined in the Data Dictionary.

Command

```
APROPOS
```

6.1.1.1. WORDS

WORDS Command

Purpose

This command redisplays the list of words.

Command

```
WORDS
```

6.1.1.2. DEFINITIONS

DEFINITIONS Command

Purpose

This command will display a single definition or all the definitions of the words listed by the WORDS command.

Command

```
DEFINITIONS
or
DEFINITION WORD
```

Where

WORD - Is the word for which a definition is desired. This is not limited to the words displayed. It can access any word in the dictionary.

Notes

The use of the DEFINITIONS command is discouraged because it is quite slow.

6.1.2. INIT

INIT Command

Purpose

The INIT subcommand of BOOK has no arguments. It's purpose is to clear out all sections of BOOK.

Command

```
INIT
```

6.1.3. LIST

LIST Command

Purpose

The LIST subcommand of BOOK has no arguments. It's purpose is to list out all sections of BOOK that have been defined.

Command

```
LIST
```

6.1.4. DONE

DONE Command

Purpose

The DONE (or a blank line) subcommand of BOOK has no arguments. It's purpose is to save changes and exit BOOK.

Command

```
DONE
```

A carriage return is equivalent to typing DONE.

6.1.5. OMIT

OMIT Command

Purpose

The OMIT subcommand of BOOK has no arguments. It's purpose is to exit BOOK without saving any changes.

Command

```
OMIT
```

6.1.6. PROCESSING

PROCESSING Command

Purpose

The PROCessing subcommand of BOOK has four arguments (up to twelve characters each) that defines the event as seen by the agency who processed the data. These particular four arguments will be saved as the event global name in the master file by the database administrator and cannot be changed once the event has gone through the ADMINistration operation.

Command

```
PROC AGENCY,PROGRAM,PROJECT,ID
```

Where

```
AGENCY  - Processing agency name.
PROGRAM - Program name as seen by the processing agency.
PROJECT - Project name   "               "            .
ID      - Identifier.  Restrictions:  the first four characters of ID
          must match those of the event name VENT created during the
          ENTER process.
```

Example

```
PROC WYLE,SPACE-TEL,MIRROR,MDTV
```

6.1.7. CONTRACTING

CONTRACTING Command

Purpose

The CONTracting subcommand of BOOK has four arguments (up to twelve characters each) that defines the event as seen by the contracting agency.

Command

```
CONT AGENCY,PROGRAM,PROJECT,ID
```

Where

```
AGENCY  - Contracting agency name.
PROGRAM - Program name as seen by the contracting agency.
PROJECT - Project name   "                  "           .
ID      - Other identifier.
```

Example

```
CONT LMSC,SPACE-TEL,MIRROR,MDTV
```

6.1.8. COGNIZANT

COGNIZANT Command

Purpose

The COGNizant subcommand of BOOK has four arguments (up to twelve characters each) that defines the event as seen by the cognizant agency.

Command

```
COGN AGENCY,PROGRAM,PROJECT,ID
```

Where

```
AGENCY  - Contracting agency name.
PROGRAM - Program name as seen by the contracting agency.
PROJECT - Project name   "                  "           .
ID      - Other identifier.
```

Example

```
COGN NASA,ASTRONOMY,SPACE-TEL,MDTV
```

6.1.9. DATE

DATE Command

Purpose

The DATE subcommand of BOOK has one argument (up to two characters for MM, DD, and YY) that defines the date the event took place.

Command

```
DATE MM/DD/YY
```

Where

```
MM - Month
DD - Day
YY - Year
```

Example

```
Nov,1 1982 = DATE 11/1/82
```

6.1.10. TIME

TIME Command

Purpose

The TIME subcommand of BOOK has one argument (up to two characters for HH, MM, SS, and up to three characters for FRAC) that defines the time the event began place.

Command

```
TIME HH:MM:SS.FRAC
```

Where

```
    HH   - Hour (24 hour clock)
    MM   - Minute
    SS   - Second
    FRAC - Fraction of a second
```

Example

```
1:00:11.0 P.M = TIME 13:00:11.0
```

6.1.11. EVENT

EVENT Command

Purpose

The EVENT subcommand of BOOK has four arguments (up to twelve characters each) that defines the type of event.

Command

```
EVENT CLASS,TYPE,ID1,ID2
```

Where

```
    CLASS - A word to describe the general class of test
            (see recommendations below).
    TYPE  - The type of test (see recommendations below).
    ID1   - Other identifier (see recommendations below).
    ID2   - Other identifier.  A free creation of the user.
```

Recommended words

For the CLASS,TYPE pair:

```
GROUND,REVERBERANT
GROUND,DIRECT
GROUND,PROGRESSIVE
GROUND,CAPTIVE
GROUND,DIR-REV
FLIGHT,LIFTOFF
FLIGHT,LANDING
FLIGHT,ASCENT
FLIGHT,REENTRY
```

For ID1:

```
DEVELOPMENT
QUAL
ACCEPTANCE
PROTOFLIGHT
PROTOTYPE
```

6.1.12. LOCATION

LOCATION Command

Purpose

The location subcommand of BOOK has four arguments (up to twelve characters each) that defines the location where the event took place.

Command

```
LOCA GLOBAL,LOCAL,ID1,ID2
```

Where

```
GLOBAL - Global location (i.e. city name)
LOCAL  - Local    "      (i.e. site name)
ID1    - Other identifier (i.e. test area name)
ID2    - Other identifier
```

Example

```
LOCA GREENBELT GSFC 40KCELL
```

6.1.13. VEHICLE

VEHICLE Command

Purpose

The VEHIcle subcommand of BOOK has four arguments (up to twelve characters each) that defines the vehicle that will fly the item tested.

Command

```
   VEHICLE CLASS,TYPE,CONF,ID

   Where

      CLASS - Vehicle classification. (see recommendations)
      TYPE  -    "    type. (see recommendations)
      CONF  -    "    configuration
      ID2    - Other identifier
```

Recommended words

For the CLASS,TYPE pair:

```
      SHUTTLE,ENTERPRISE
      SHUTTLE,COLUMBIA
      BOOSTER,TITAN
      BOOSTER,ATLAS
      BOOSTER,THOR
      BOOSTER,TAT
      BOOSTER,SCOUT
```

Example

```
   VEHI BOOSTER,TITAN,T3C,SSLV5-17
```

6.1.14. DESC

DESCRIPTION Command

Purpose

The DESCription subcommand of BOOK is an optional subcommand where the user can input four lines of general information (up to 72 characters per line) that will help future users identify the event above and beyond that which has already been described in the other sections of PREP. If DESC is not specified in PREP, then these four lines of description are a copy of those specified in the Event Description section of ENTER.

Command

```
   DESC
   1 line one
   2  "   two
   3  "   three
   4  "   four
```

Where

A descriptive string is entered next to each number 1 through 4

Example

```
DESC
1 Purpose: To determine the effect of different mass loading conditions
2 in bay 5 of the xxx vehicles equipment mounting structure.  This is
3 case one: mass load = 20#/ft**2.  Test config:  All up dynamic test
4 vehicle with mass and c.g. simulators.
```

6.1.15. EXAMPLE

BOOK example with prompts shown in lower case. It demonstrates the use of BOOK subcommands: PROCessing,CONTracting, COGNizant, DATE, TIME, LOCAtion, VEHIcle, DESCription, and DONE. The long names for the subcommands will be used to make the example more readable.

```
 prep>BOOK
 book>PROCESSING  WYLE,SPACE-TEL,MIRROR,MDTV
 book>CONTRACTING LMSC,SPACE-TEL,MIRROR,MDTV
 book>COGNIZANT NASA,ASTRONOMY,SPACE-TEL,MDTV
 book>DATE 11/1/82
 book>TIME 13:00:11.0
 book>LOCATION GREENBELT GSFC 40KCELL
 book>VEHICLE BOOSTER,TITAN,T3C,SSLV5-17
 book>DESC
 desc>1 Purpose: To determine the effect of different mass loading condition
 desc>2 in bay 5 of the xxx vehicles equipment mounting structure.  This is
 desc>3 case one: mass load = 20#/ft**2.  Test config:  All up dynamic test
 desc>4 vehicle with mass and c.g. simulators.  Duration 60 sec.
 book> DONE
book data saved
 prep>
```

6.2. CHECK

CHECK Command

The CHECK subcommand of PREP collects and converts data and descriptor DAL elements created by the ENTER process to create a set of DAL elements containing data of a standard format with consistent pressure and acceleration units over fixed frequency bands. Standard units are defined in comment section below. The CHECK process is performed automatically the first time an event is PREPed and is not normally executed as a separate subcommand. CHECK is only executed if a modification or an additional section has been added to the ENTER information for an event that has already been PREPed.

Command

```
CHECK NSEC
```

Where

```
NSEC - The number of ENTER sections for the event.
     = 0 implies only one section (section 0)
     > 0 implies sections 1 through NSEC.
```

Comments

Standard units that data from ENTER is converted to are:

```
1) Acoustics - DBSP (10log(psi**2/PREF))
               PREF = 8.4144E-18 psi**2

2) Vibration - VLSP (10log(g**2/GREF))
               GREF = 8.4144E-18 g**2

3) Frequency - 1/3 Octave bands from 10 to 10,000 Hertz
```

6.3. CHANNEL

CHANNEL Subprocessor

Purpose

The channel section of PREP is entered by issuing the CHANnel subcommand. This section allows for additions and/or modifications to the event's channel table. The channel table contains information for each channel such as the channel name, the type of transducer, the valid frequency range of the data from that channel, etc. Part of the the channel table includes descriptions originally input during the ENTER process. The table is created when the PREP command is executed for the first time for a new event. The table is updated each time the CHECK subcommand of PREP is issued. Descriptive information in addition to that input during the ENTER process can be added in the channel section. Subcommands within CHANnel allow for the definition of coordinates, for changing a channels type, and for changing the valid frequency range for each channel in the channel table. The full description for a given transducer in the channel table looks like this:

```
CHAN,ITYPE,FLOW,FHIGH,X,Y,Z,SYSTEM
DESC
```

Where

```
CHAN   - A channel name which exists in the channel table.
ITYPE  - An integer code indicating the following transducer type:
       = -1 : any microphone
       = -2 : surface microphone
       = -3 : not surface microphone
       =  1 : any accelerometer
       =  2 : normal accelerometer
       =  3 : not normal accelerometer
FLOW,FHIGH
       - Center frequencies defining the range of valid data.
X,Y,Z  - Coordinates for the transducer (real numbers).
SYSTEM - Reference system for the coordinates (up to 4 characters ).
DESC   - One line of general description (72 characters max).
```

The fields CHAN,ITYPE,FLOW,FHIGH and DESC in the description for a given transducer are originally created from descriptions input for each transducer during the ENTER process. These descriptions were collected into the channel table when the event is first PREPed. The remaining fields X,Y,Z,SYSTEM of the channel description can only be input in this section.

Command

CHAN

6.3.1. LIST

LIST Command

Purpose

The LIST subcommand of CHAN allows for listing of all or part of an event's channel table. The forms and functions of LIST are shown below.

Command

LIST ALL (DESC)

Lists the channel table. (Optional) If DESCription is specified, the general channel descriptions are included in the listing.

LIST DESC

Lists the channel descriptors.

LIST RANGE I1,I2 (DESC)

Lists items I1 through I2 of the channel table. (Optional) DESCription is as described above.

LIST TYPE ITEM (DESC)

Lists the channel table for all channels of type ITEM. (Optional) DESCription is as described above.

6.3.2. CHANGE

CHANGE Command

Purpose

The CHANGE subcommand of CHAN allows the user to change specific descriptive fields for specified transducers in the channel table.

Command

CHANGE ALL

Change all values in the channel table by accepting input following the command. The input lines must be of the form:

```
CHAN,ITYPE,FLOW,FHIGH,X,Y,Z,SYSTEM
```

CHANGE TYPE

Change the ITYPE field in the channel table. Input must follow the subcommand and have the form:

```
CHAN,ITYPE
```

CHANGE FREQuency

Change the FLOW,FHIGH fields in the channel table. Input must follow the subcommand and have the form:

```
CHAN FLOW,FHIGH
```

CHANGE COORdinates

Change the X,Y,Z,SYSTEM fields in the channel table. Input must follow the subcommand and have the form:

```
CHAN X,Y,Z,SYSTEM
```

Notes

The channel name CHAN must match a channel name already in the table. Input is accepted until a blank card is encountered or until a card with only one item (such as the word 'END') is read.

6.3.3. DONE

DONE Command

Purpose

The DONE (or a blank line) subcommand of CHANnel has no arguments. It's purpose is to exit CHANnel.

Command

```
DONE
```

6.3.4. OMIT

OMIT Command

Purpose

The OMIT subcommand of CHANnel has no arguments. It's purpose is to exit CHANnel without saving any changes.

Command

```
OMIT
```

6.3.5. EXAMPLE

CHANnel example with prompts shown in lower case. The subcommands LIST, CHANGE, and DONE will be demonstrated. The commands are shown in their long form for clarity.

```
prep>CHANNEL
chan>LIST ALL DESCRIPTION

CHANNEL  TYPE      FLOW    FHIGH         X          Y          Z        SYST

 1 M5     -1       25.0  10000.0       0.00       0.00       0.00
         M5   INTERNAL MIC   BETWEEN LIGHT SHIELD AND OTA  FWD
 2 A3      1       25.0  10000.0       0.00       0.00       0.00
```

```
         A3   PRIMARY MIRROR
 3 A4      1        25.0  10000.0        0.00      0.00      0.00
         A4   PRIMARY MIRROR ON MIRROR

chan>LIST ALL

CHANNEL  TYPE      FLOW    FHIGH         X         Y         Z       SYST

 1 M5     -1        25.0  10000.0        0.00      0.00      0.00
 2 A3      1        25.0  10000.0        0.00      0.00      0.00
 3 A4      1        25.0  10000.0        0.00      0.00      0.00

chan>LIST DESCRIPTION

 1 M5    M5   INTERNAL MIC   BETWEEN LIGHT SHIELD AND OTA  FWD
 2 A3    A3   PRIMARY MIRROR
 3 A4    A4   PRIMARY MIRROR ON MIRROR

chan>LIST RANGE 1 2

CHANNEL  TYPE      FLOW    FHIGH         X         Y         Z       SYST

 1 M5     -1        25.0  10000.0        0.00      0.00      0.00
 2 A3      1        25.0  10000.0        0.00      0.00      0.00

chan>LIST TYPE -1

CHANNEL  TYPE      FLOW    FHIGH         X         Y         Z       SYST

 1 M5     -1        25.0  10000.0        0.00      0.00      0.00

chan>CHANGE ALL
M5,-1,100.,2000.,10.,0.,0.,BOOS
END
chan>CHANGE TYPE
A3,2
END
chan>CHANGE FREQUENCY
A3,100.,2000.
END
chan>CHANGE COORDINATES
A3,11.,17.,10.,BNEW
A4,11.,12.5,0.,BNEW
END
chan>LIST ALL

CHANNEL  TYPE      FLOW    FHIGH         X         Y         Z       SYST

 1 M5     -1       100.0   2000.0       10.00      0.00      0.00     BOOS
 2 A3      2       100.0   2000.0       11.00     17.00     10.00     BNEW
 3 A4        1         25.0  10000.0       11.00       12.50       0.00       BNEW

chan>DONE
prep>
```

6.4. CONFIGURATION

CONFIGURATION Subprocessor

The vehicle configuration section of PREP is entered by issuing the CONFiguration subcommand. This section allows for the entry of a vehicle configuration tree to define the hierarchy of subsystems that make up the vehicle as it was tested or flown. The configuration tree is composed of branches. These branches are related to one another by showing which branch is the parent branch to the current branch. Branches are described by local names and global names. The local names are use to establish the hierarchy amongst the branches. The global names describe the branch. The global names are key words that will be later be searched on by users operating in the database query mode. A branch of the configuration is represented by line made up of six fields as shown below:

```
LAST,THIS,GENERIC,SPECIFIC,(MOUNT),(ENCLOSURE)
```

Where

LAST - Local name (up to 4 characters) of the parent branch of the current branch . Restriction: The first branch of a configuration tree ,the trunk, must be designated as a $.

THIS - Local name (up to 4 characters) for the current branch.

GENERIC - Global generic name for the current branch (up to 12 characters).

SPECIFIC - Global specific name for the current branch (up to 12 characters).

MOUNT - (Optional) The branch (specified by the local branch name) which is connected to the current branch but not in a tree like manner.

ENCLOSURE- (Optional) The branch (specified by the local branch name) which encloses the current branch.

Command

```
CONF (NU,NAME)
```

Where

NU,NAME - (Optional) DAL unit and event name for an existing database event. The configuration tree for that event is read in to this event.

Subcommands to create, delete, and list the branches of the configuration are described in the additional HELP information below.

6.4.1. APROPOS

APROPOS Command

Purpose

This command displays the words which have been used in the CONFIGURATION section in previous events. It also allows you to list the definitions of those words as defined in the Data Dictionary.

Command

```
APROPOS
```

6.4.1.1. WORDS

WORDS Command

Purpose

This command redisplays the list of words.

Command

```
WORDS
```

6.4.1.2. DEFINITIONS

DEFINITIONS Command

Purpose

This command will display a single definition or all the definitions of the words listed by the WORDS command.

Command

```
DEFINITIONS
or
DEFINITION WORD
```

Where

WORD - Is the word for which a definition is desired. This is not limited to the words displayed. It can access any word in the dictionary.

Notes

The use of the DEFINITIONS command is discouraged because it is quite slow.

6.4.2. INPUT

INPUT Command

Purpose

The INPUT subcommand of CONFiguration allows the user to create or add branches to the event's configuration tree by accepting input following the command. The lines following INPUT must have the form:

```
LAST,THIS,GENERIC,SPECIFIC,(MOUNT),(ENCLOSURE)
```

Command

```
INPUT
branch one
branch two
.
.
END
```

Notes

Input is accepted until a blank card is encountered or until a card with one item (such as the word 'END') is read.

Example

```
INPUT
$,BOOS,BOOSTER,T34D
BOOS,BSTG,BASESTAGE,T34D
BOOS,USTG,UPPERSTAGE,CENTAUR
USTG,FAIR,FAIRING,ISOGRID
USTG,ADAP,ADAPTER,TYPE3,*,FAIR
```

```
USTG,PAYL,PAYLOAD,CSAT,ADAP,FAIR
END
```

6.4.3. LIST

LIST Command

Purpose

The LIST subcommand of CONFiguration allows for listing of all or part of an event's configuration tree.

Command

```
LIST
```

Lists out the entire configuration tree.

```
LIST I1
```

Lists line I1 of the configuration tree.

```
LIST I1,I2
```

Lists lines I1 through I2 of the configuration tree.

6.4.4. DISABLE

DISABLE Command

Purpose

The DISAble subcommand of CONFiguration allows the user to disable (mark for deletion) branches of an event's configuration tree. Once a branch has been disabled it will be deleted upon exit from the CONFiguration section. A disabled branch can also be deleted from the configuration tree by the PACK subcommand.

Command

```
DISA I1
```

Disables the branch in the configuration tree listing with line number I1.

```
DISA I1,I2
```

Disables branches in the configuration tree listing with line numbers I1 through I2.

6.4.5. ENABLE

ENABLE Command

Purpose

The ENABle subcommand of CONFiguration allows the user to enable branches of an event's configuration tree that have become disabled (marked for deletion) by the subcommand DISAble or by the input of branches with the same local name.

Command

```
ENAB I1
```

Enables the branch in the configuration tree listing with line number I1.

```
ENAB I1,I2
```

Enables branches in the configuration tree listing with line numbers I1 through I2.

6.4.6. PACK

PACK Command

Purpose

The PACK subcommand of CONFiguration physically removes all disabled configuration tree branches.

Command

```
PACK
```

6.4.7. READ

READ Command

Purpose

The READ subcommand of CONFiguration allows the user to add an entire configuration tree from another database event to the tree being built.

Command

```
READ NU,NAME
```

Where

NU,NAME - DAL unit and event name for an existing database event with the desired configuration tree.

6.4.8. CHECK

CHECK Command

Purpose

The CHECK subcommand of CONFiguration examines the current configuration tree for errors. CHECK will look for errors such as:

1) The local name of the configuration trees trunk (LAST=$) occuring more than once).
2) A parent branch local name LAST that was never defined by a current branch local name THIS.
3) A MOUNT or ENCLOSURE that was never defined by a current branch local name THIS.

CHECK will also check the global names GENERIC and SPECIFIC against the database master file and warn the user if global names used for the current tree are not in the master file. That means these names have never previously been used to describe branches of a configuration tree and the user should make sure that these names are what he really wants to use.

Command

```
CHECK
```

6.4.9. DONE

DONE Command

Purpose

The DONE (or a blank line) subcommand of CONFiguration has no arguments. It's purpose is to check for errors (see the CHECK subcommand), eliminate any branches that were marked for deletion (see the PACK subcommand), save changes, and exit CONFiguration.

Command

```
DONE
```

6.4.10. OMIT

OMIT Command

Purpose

The OMIT subcommand of CONFiguration has no arguments. It's purpose is to exit CONFiguration without saving any changes.

Command

```
OMIT
```

6.4.11. EXAMPLE

CONFiguration example with prompts shown in lower case. A configuration tree will be built for one event (event ONE) and then a tree will be built for a second event (event TWO) starting with the tree from the first event. The subcommands INPUT, LIST, PACK, and DONE will be demonstrated. All commands are shown in their long form for clarity.

```
?PREP 1,ONE
prep>CONFIGURATION
conf>INPUT
ltgsme>$,BOOS,BOOSTER,T34D
ltgsme>BOOS,BSTG,BASESTAGE,T34D
ltgsme>BOOS,USTG,UPPERSTAGE,CENTAUR
ltgsme>USTG,FAIR,FAIRING,ISOGRID
ltgsme>USTG,ADAP,ADAPTER,TYPE3,USTG,FAIR
ltgsme>USTG,PAYL,PAYLOAD,CSAT,ADAP,FAIR
ltgsme>END
conf>LIST
 1:  $     BOOS  BOOSTER      T34D
 2:  BOOS  BSTG  BASESTAGE    T34D
 3:  BOOS  USTG  UPPERSTAGE   CENTAUR
 4:  USTG  FAIR  FAIRING      ISOGRID
 5:  USTG  ADAP  ADAPTER      TYPE3      USTG  FAIR
 6:  USTG  PAYL  PAYLOAD      CSAT       ADAP  FAIR
conf>DONE
prep>DONE
?PREP 1,TWO
prep>CONFIGURATION 1,ONE
conf>LIST
 1:  $     BOOS  BOOSTER      T34D
 2:  BOOS  BSTG  BASESTAGE    T34D
 3:  BOOS  USTG  UPPERSTAGE   CENTAUR
 4:  USTG  FAIR  FAIRING      ISOGRID
 5:  USTG  ADAP  ADAPTER      TYPE3      USTG  FAIR
 6:  USTG  PAYL  PAYLOAD      CSAT       ADAP  FAIR
conf>INPUT
ltgsme>USTG,ADAP,ADAPTER,TYPE2,USTG,FAIR
ltgsme>USTG,PAYL,PAYLOAD,LANDSAT,ADAP,FAIR
ltgsme>PAYL,CAM,SUBSYSTEM,CAMERA
ltgsme>END
conf>LIST
 1:  $     BOOS  BOOSTER      T34D
 2:  BOOS  BSTG  BASESTAGE    T34D
 3:  BOOS  USTG  UPPERSTAGE   CENTAUR
 4:  USTG  FAIR  FAIRING      ISOGRID
-5:  USTG  ADAP  ADAPTER      TYPE3      USTG  FAIR
-6:  USTG  PAYL  PAYLOAD      CSAT       ADAP  FAIR
 7:  USTG  ADAP  ADAPTER      TYPE2      USTG  FAIR
 8:  USTG  PAYL  PAYLOAD      LANDSAT    ADAP  FAIR
 9:  PAYL  CAM   SUBSYSTEM    CAMERA
conf>PACK
conf>LIST
 1:  $     BOOS  BOOSTER      T34D
 2:  BOOS  BSTG  BASESTAGE    T34D
 3:  BOOS  USTG  UPPERSTAGE   CENTAUR
 4:  USTG  FAIR  FAIRING      ISOGRID
 5:  USTG  ADAP  ADAPTER      TYPE2      USTG  FAIR
 6:  USTG  PAYL  PAYLOAD      LANDSAT    ADAP  FAIR
 7:  PAYL  CAM   SUBSYSTEM    CAMERA
conf>DONE
prep>
```

6.5. MODULES

MODULES Subprocessor

Purpose

The MODUles subcommand of PREP allows the user to create data modules which are used to further define a branch (subsystem) of an event's configuration tree (see the CONF subcommand for information on configuration trees). Data modules can be assembled to represent one or more subsystems within the general system' that was tested. Data modules are stored in a module table.

Data modules define a subsystem by allowing inputs that create collections of transducer names, key descriptions, and acoustic/structural parameters. The transducer names in the data module are the specific channels used for the instrumentation of the subsystem (the transducer names are a subset of those contained in the event's channel table - see the CHANnel subcommand of PREP). The physical characteristics of the subsystem are defined by the input of key descriptions and acoustic/structural parameters. The acoustic/structural parameters form a statistical energy analysis (SEA) model.

By including SEA models with acoustic/structural parameters in the database information for an event, predictions can be made on new systems by extrapolating from systems saved in the database. The user can find an appropriate data module in database by using the SERCH command, extract the data module (establishing a baseline model with corresponding empirical data), model the new system, and predict the response of the new system by extrapolating from the baseline system (see the PREDICTION section of VAPEPS).

Subcommands of MODUles allow the user to build data modules (see the DAMO subcommand), associate data modules with the appropriate branch of the configuration tree (see the ATTACH subcommand), manipulate the module table (see the PACK, SAVE, REMOVE, DISABLE, ENABLE, CHANGE, CHECK, LIST, and DUMP subcommands), manipulate individual modules (see the EDIT, and READ subcommands), and exit the modules section (see OMIT, and DONE subcommands).

A data module appears in the module table in this form:

```
MODULE,CONF,ZONE(,SYSTEM,X,Y,Z)
```

Where

MODULE - Local module name (up to 4 characters) created by the DAMO

```
                subcommand
     CONF     - Local branch of the event's configuration associated with
                this module (the branch it is attached to).
     ZONE     - Other name (up to 4 characters) created by the ATTACH
                subcommand which intended to indicate the general area in
                the configuration tree associated with this module.
     SYSTEM   - (Optional)  Other name designating a global coordinate
                system or zone associated with this module.
     X,Y,Z    - (Optional)  Integer coordinates (in the coordinate system:
                SYSTEM) to define the position of this module.
```

Command

```
MODU (MXMOD,MXCT,MXCPS)
```

Where

```
   MXMOD - (Optional)  The maximum number of modules for this event.
           Default = 20, or 5 more than currently exist, whichever is
           greater.
   MXCT  - (Optional)  The maximum number of channels used for all modules
           for this event.  Default = twice the number of channels in the
           event's channel table, or 20 more than are currently attached,
           whichever is greater.
   MXCPS - (Optional)  The maximum number of channels per SEA element.
           Default = 15.
```

6.5.1. APROPOS

APROPOS Command

Purpose

This command displays the words which have been used in the MODULES section in previous events. It also allows you to list the definitions of those words as defined in the Data Dictionary.

Command

```
APROPOS
```

6.5.1.1. WORDS

WORDS Command

Purpose

This command redisplays the list of words.

Command

```
WORDS
```

6.5.1.2. DEFINITIONS

DEFINITIONS Command

Purpose

This command will display a single definition or all the definitions of the words listed by the WORDS command.

Command

```
DEFINITIONS
or
DEFINITION WORD
```

Where

WORD - Is the word for which a definition is desired. This is not limited to the words displayed. It can access any word in the dictionary.

Notes

The use of the DEFINITIONS command is discouraged because it is quite slow.

6.5.2. DAMO

DAMO Subprocessor

Purpose

The DAMO subcommand of MODUles creates a data module in the event's module table. Subcommands within DAMO allow the user to describe the module (see the DESCription subcommand), define up to six types of elements to to form a SEA model (see the EXTA, SKIN, INTA, MONT, INST and FRAM subcommands), input structural parameters (see the P* subcommand), associate channels with the module (see the C* and REMOVE subcommands), list the module (see the LIST subcommand), read other modules (see the READ subcommand), save a module to DAL elements not associated with the event (see the SAVE subcommand), check a module (see the CHECK subcommand), and exit DAMO (see the DONE subcommand).

Command

```
DAMO (NU,NAME,)NEW
```

Where

NU,NAME - (Optional) DAL unit and name of a DAL elements containing a module previously created by the SAVE subcommand of DAMO or by DAMO issued from VAPEPS. If NU,NAME is omitted, the module is built from scratch. If NU,NAME is specified and NEW is omitted the local module name will be NAME.

NEW - Local data module name

6.5.2.1. DESC

DESCRIPTION Command

Purpose

The DESCription subcommand of DAMO has three arguments (up to 12 characters each) that describe the module as a whole. These arguments will be saved in the database master file as part of the descriptions that can be searched on by users operating in the database query mode.

Command

```
DESC NAME1,NAME2,NAME3
```

6.5.2.2. EXTA

EXTA Command

EXTA is a SEA element used to physically describe an acoustic volume that communicates with the SEA elements SKIN (acoustic/structural transmission path) and INTA (non-resonant transmission path through SKIN).

The EXTA subcommand of DAMO has three optional arguments (up to 12 each) that describe this SEA element. These arguments will be saved in the database master file as part of the descriptions that can be searched on by users operating in the database query mode. Once the EXTA subcommand is issued other subcommands can be issued to input, modify, delete, list, and save the contents of the SEA element as well as move over to another SEA element or exit DAMO. These subcommands are: P*, C*, REMOVE, LIST, CHECK, READ, SAVE, DONE, DESC, EXTA, SKIN, INTA, MONT, INST, and FRAM.

Command

```
EXTA (NAME1,NAME2,NAME3)
```

6.5.2.3. SKIN

SKIN Command

Purpose

SKIN is a SEA element used to physically describe a structure that communicates with the SEA elements EXTA (structural/acoustic transmission path), INTA (structural/acoustic transmission path), and MONT (structural transmission path).

The SKIN subcommand of DAMO has three optional arguments (up to 12 each) that describe this SEA element. These arguments will be saved in the database master file as part of the descriptions that can be searched on by users operating in the database query mode. Once the SKIN subcommand is issued other subcommands can be issued to input, modify, delete, list, and save the contents of the SEA element as well as move over to another SEA element or exit DAMO. These subcommands are: P*, C*, REMOVE, LIST, CHECK, READ, SAVE, DONE, DESC, EXTA, SKIN, INTA, MONT, INST, and FRAM.

Command

```
SKIN (NAME1,NAME2,NAME3)
```

6.5.2.4. INTA

INTA Command

Purpose

INTA is a SEA element used to physically describe an acoustic volume that communicates with the SEA elements SKIN (acoustic/structural transmission path), EXTA (non-resonant transmission path through SKIN), MONT (acoustic/structural transmission path - when MONT has significant surface area), and INST (acoustic/structural transmission path).

The INTA subcommand of DAMO has three optional arguments (up to 12 each) that describe this SEA element. These arguments will be saved in the database master file as part of the descriptions that can be searched on by users operating in the database query mode. Once the INTA subcommand is issued other subcommands can be issued to input, modify, delete, list, and save the contents of the SEA element as well as move over to another SEA element or exit DAMO. These subcommands are: P*, C*, REMOVE, LIST, CHECK, READ, SAVE, DONE, DESC, EXTA, SKIN, INTA, MONT, INST, and FRAM.

Command

```
INTA (NAME1,NAME2,NAME3)
```

6.5.2.5. MONT

MONT Command

Purpose

MONT is a SEA element used to physically describe a structure that communicates with the SEA elements INTA (structural/acoustic transmission path), SKIN (structural transmission path), and INST (structural transmission path).

The MONT subcommand of DAMO has three optional arguments (up to 12 each) that describe this SEA element. These arguments will be saved in the database master file as part of the descriptions that can be searched on by users operating in the database query mode. Once the MONT subcommand is issued other subcommands can be issued to input, modify, delete, list, and save the contents of the SEA element as well as move over to another SEA element or exit DAMO. These subcommands are: P*, C*, REMOVE, LIST, CHECK, READ, SAVE, DONE, DESC, EXTA, SKIN, INTA, MONT, INST, and FRAM.

Command

```
MONT (NAME1,NAME2,NAME3)
```

6.5.2.6. INST

INST Command

Purpose

INST is a SEA element used to physically describe a structure that communicates with the SEA elements INTA (structural/acoustic transmission path) and MONT (structural transmission path).

The INST subcommand of DAMO has three optional arguments (up to 12 each) that describe this SEA element. These arguments will be saved in the database master file as part of the descriptions that can be searched on by users operating in the database query mode. Once the INST subcommand is issued other subcommands can be issued to input, modify, delete, list, and save the contents of the SEA element as well as move over to another SEA element or exit DAMO. These subcommands are: P*, C*, REMOVE, LIST, CHECK, READ, SAVE, DONE, DESC, EXTA, SKIN, INTA, MONT, INST, and FRAM.

Command

```
INST (NAME1,NAME2,NAME3)
```

6.5.2.7. FRAM

FRAM Command

Purpose

FRAM is an optional structure use to define the physical connection between SKIN and MONT (only useful if MONT is a truss).

The FRAM subcommand of DAMO has three optional arguments (up to 12 each) that describe the structure. These arguments will be saved in the database master file as part of the descriptions that can be searched on by users operating in the database query mode. Once the FRAM subcommand is issued other subcommands can be issued to input, modify, delete, list, and save the contents of FRAM as well as move over to a SEA element or exit DAMO. These subcommands are: P*, C*, REMOVE, LIST, CHECK, READ, SAVE, DONE, DESC, EXTA, SKIN, INTA, MONT, INST, and FRAM.

Command

```
FRAM (NAME1,NAME2,NAME3)
```

6.5.2.8. P*

P* Command

Purpose

The P* subcommand of DAMO (actually of EXTA, SKIN, INTA, MONT, INST or FRAM only) is followed by arguments that set the values of named physical parameters which are used to describe the SEA element. See the PREDICTION section of the VAPEPS USER'S MANUAL for parameter lists and descriptions of the allowable types of SEA elements.

Command

```
P* NAME=VALUE,NAME=VALUE
```

Where

NAME - An allowable parameter name.
VALUE - The value of that parameter. The value must be represented as a real number (eg. 1042.02 or 1.04202E+3 or 1.04202+3)

6.5.2.9. C*

C* Command

The C* subcommand of DAMO (actually of EXTA, SKIN, INTA, MONT, INST or FRAM) is followed by arguments that define channel names to be associated with the SEA element. These channel names must exist in the channel table of the event (see the CHANNEL section for information on and event's channel table).

Command

```
C* C1,C2,C3....,Cn
```

Where

C1,C2... - Channel names.

6.5.2.10. REMOVE

REMOVE Command

Purpose

The REMOVE subcommand of DAMO (actually of EXTA, SKIN, INTA, MONT, INST or FRAM only) is followed by arguments that define channel names (input by the C* subcommand) to be deassociated with the SEA element.

Command

```
REMOVE C1,C2,C3....,Cn
```

Where

C1,C2... - Channel names.

6.5.2.11. LIST

LIST Command

Purpose

The LIST subcommand of DAMO, EXTA, SKIN, INTA, MONT, INST, or FRAM allows the user to list out the contents of the module/SEA element.

Command

```
LIST (PARA)
or
LIST (CHAN)
```

Where

PARA - (Optional) List out parameters only.

CHAN - (Optional) List out channels only.

6.5.2.12. READ

READ Command

Purpose

The READ subcommand of DAMO, EXTA, SKIN, INTA, MONT, INST and FRAM allows for the user to read in a data module that has been previously stored in DAL elements by the subcommand SAVE or by DAMO issued from VAPEPS.

Command

```
READ  NU,VER(,ZONE)
```

Where

NU,VER - DAL unit and version of the DAL elements which contain the module.

ZONE - (Optional) SEA element to read in. Otherwise READ will read in all SEA elements.

6.5.2.13. SAVE

SAVE Command

Purpose

The SAVE subcommand of DAMO, EXTA, SKIN, INTA, MONT, INST, or FRAM allows the user to save the data module in DAL elements not associated with the event.

Command

```
SAVE NU,NAME
```

Where

NU,NAME - DAL element and version name of the DAL elements where the module will be stored. The names of these elements will be:

in all SEA elements.

6.5.2.13. SAVE

SAVE Command

Purpose

The SAVE subcommand of DAMO, EXTA, SKIN, INTA, MONT, INST, or FRAM allows the user to save the data module in DAL elements not associated with the event.

Command

```
SAVE NU,NAME
```

Where

NU,NAME - DAL element and version name of the DAL elements where the module will be stored. The names of these elements will be:

(

els only.

TA, SKIN, INTA, MONT, INST and FRAM allows
module that has been previously stored in
SAVE or by DAMO issued from VAPEPS.

ersion of the DAL elements which contain the

EA element to read in. Otherwise READ will read

```
C* C1,C2,C3....,Cn
```

Where

C1,C2... - Channel names.

6.5.2.10. REMOVE

REMOVE Command

Purpose

The REMOVE subcommand of DAMO (actually of EXTA, SKIN, INTA, MONT, INST or FRAM only) is followed by arguments that define channel names (input by the C* subcommand) to be deassociated with the SEA element.

Command

```
REMOVE C1,C2,C3....,Cn
```

Where

C1,C2... - Channel names.

6.5.2.11. LIST

LIST Command

Purpose

The LIST subcommand of DAMO, EXTA, SKIN, INTA, MONT, INST, or FRAM allows the user to list out the contents of the module/SEA element.

Command

```
LIST (PARA)
or
LIST (CHAN)
```

Where

PARA - (Optional) List out parameters only.

CHAN - (Optional) List out channels only.

6.5.2.12. READ

READ Command

Purpose

The READ subcommand of DAMO, EXTA, SKIN, INTA, MONT, INST and FRAM allows for the user to read in a data module that has been previously stored in DAL elements by the subcommand SAVE or by DAMO issued from VAPEPS.

Command

```
READ  NU,VER(,ZONE)
```

Where

```
NU,VER - DAL unit and version of the DAL elements which contain the
         module.
ZONE   - (Optional)  SEA element to read in.  Otherwise READ will read
         in all SEA elements.
```

6.5.2.13. SAVE

SAVE Command

Purpose

The SAVE subcommand of DAMO, EXTA, SKIN, INTA, MONT, INST, or FRAM allows the user to save the data module in DAL elements not associated with the event.

Command

```
SAVE NU,NAME
```

Where

```
NU,NAME - DAL element and version name of the DAL elements where the
          module will be stored.  The names of these elements will be:
```

```
NU,PARA,NAME - Physical parameters.
NU,DCHN,NAME - Channels associated with the module.
NU,DMDC,NAME - Module descriptions.
```

6.5.2.14. CHECK

CHECK Command

Purpose

The CHECK subcommand of DAMO, EXTA, SKIN, INTA, MONT, INST or FRAM checks over the data module for errors, calculates non-dimensional parameters, and lists out a summary of the module. CHECK will look for errors such as:

1) Incomplete physical parameters specified for given types of SEA elements.
2) Channels associated with a module that do not exist in the event's channel table.

Command

```
CHECK
```

6.5.2.15. DONE

DONE Command

Purpose

The DONE (or a blank line) subcommand of DAMO, EXTA, SKIN, INTA, MONT INST, or FRAM checks over the module (see the CHECK subcommand) and exits DAMO.

Command

```
DONE
```

6.5.3. ATTACH

ATTACH Command

Purpose

The ATTAch subcommand of MODUles creates an entry in the module table. Modules formed by the DAMO operation must be associated with a branch of the event's configuration tree. The ATTAch subcommand defines this relationship. ATTAch also allows for associating a coordinate system and integer coordinates with the module. The arguments of the ATTAch subcommand are discussed in MODUles.

Command

```
ATTACH MODULE,CONF,ZONE,SYSTEM,X,Y,Z
```

Each entry in the module table created by ATTAch must have a unique MODULE,CONF,ZONE name. If not, other modules with the same three name combination will be disabled (marked for deletion) and will be lost is a PACK or DONE subcommand is given from MODUles.

6.5.4. LIST

LIST Command

Purpose

The LIST subcommand of MODUles allows for listing of all or part of an event's module table.

Command

```
LIST
```

Lists out the entire module table.

```
LIST I1
```

Lists line I1 of the module table.

```
LIST I1,I2
```

Lists lines I1 through I2 of the module table.

6.5.5. DUMP

DUMP Command

Purpose

The DUMP subcommand of MODUles produces a long listing of all or part of an event's module table.

Command

DUMP

Lists out the entire module table.

DUMP I1

Lists line I1 of the module table.

DUMP I1,I2

Lists lines I1 through I2 of the module table.

6.5.6. EDIT

EDIT Command

Purpose

The EDIT subcommand of MODUles allows the user to edit existing modules or copy existing modules into new modules and then edit. EDIT puts the user on the DAMO level.

Command

EDIT NUM (short hand mode)

Where

NUM - The module number as listed in the module table.

When EDIT is executed in the short hand mode, the three argument module name is not changed in the module table.

Other form

```
EDIT MODULE,CONF,ZONE (NEW)
```

Where

```
MODULE,CONF,ZONE - Full three argument module name as it appears in
                   the event's module table.  If edit is issued in
                   this form with out the optional new name NEW, the
                   three argument module name in the module table is
                   not changed.
NEW              - (Optional) Specifies a new local module name where
                   the resulting module output of the edit session
                   will reside.  To become a part of the module table,
                   the new module NEW must be associated with a branch
                   of the configuration tree (see the ATTA
                   subcommand).
```

6.5.7. CHANGE

CHANGE Command

Purpose

The CHANGE subcommand of MODUles allows the user to modify the name of a data module as it is listed in the module table.

Command

```
CHANGE MODULE,CONF,ZONE NEWMOD,NEWCONF,NEWZONE,NEWSYSTEM,NEWX,NEWY,NEWZ
```

Where

```
MODULE,CONF,ZONE                 - Three argument module name.
NEWMOD,NEWCONF,NEWZONE
,NEWSYSTEM,NEWX,NEWY,NEWZ - (Optional)  New names and values of the
                                   module.  The list of new names and values
                                   may be truncated at any point, leaving the
                                   remaining names and values unchanged.
```

6.5.8. READ

READ Command

Purpose

The READ subcommand of MODUles allows the user to add an entire module table (this includes the data module's contents) from another database event to the module table for this event.

Command

```
READ NU,NAME
```

Where

NU,NAME - DAL unit and event name for an existing database event.

6.5.9. CHECK

CHECK Command

Purpose

The CHECK subcommand of MODUles examines the modules in the module table and checks for errors. CHECK will look for errors such as:

1) Modules associated with non-existent branches of the event's configuration tree.
2) Channels associated with modules that are not in the event's channel table (see the CHANNEL section of PREP).

Command

```
CHECK
```

6.5.10. REMOVE

REMOVE Command

Purpose

The REMOVE subcommand of MODUles allows the user to disable (mark for deletion) a module in the module table.

Command

```
REMOVE MODULE,CONF,ZONE
```

Where

MODULE,CONF,ZONE - The three argument module name as it appears in the module table listing.

6.5.11. DISABLE

DISABLE Command

Purpose

The DISAble subcommand of MODUles allows the user to disable (mark for deletion) modules of an event's module table. Once a module has been disabled it will be deleted upon exit from the modules section. A disabled module can also be deleted from the module table by the PACK subcommand.

Command

```
DISA I1
```

Disables the module in the module table listing with line number I1.

```
DISA I1,I2
```

Disables modules in the module table listing with line numbers I1 through I2.

6.5.12. ENABLE

ENABLE Command

Purpose

The ENABle subcommand of MODUles allows the user to enable modules in an event's module table that have become disabled (marked for deletion) by the subcommand DISA or by the attachment of modules with the same three argument module name.

Command

```
ENAB I1
```

Enables the module in the module table listing with line number I1.

```
ENAB I1,I2
```

Enables modules in the module table listing with line numbers I1 through I2.

6.5.13. PACK

PACK Command

Purpose

The PACK subcommand of MODUles physically removes all disabled modules in the module table.

Command

```
PACK
```

6.5.14. SAVE

SAVE Command

Purpose

The SAVE subcommand of MODUles saves the current module table (unpacked) on the current event DAL file using the current event name. This operation is automatically performed as part of the DONE subcommand.

Command

```
SAVE
```

6.5.15. DONE

DONE Command

Purpose

The DONE (or a blank line) subcommand of MODUles has no arguments. It's purpose is to exit MODUles while saving any changes. The operations described for the PACK, CHECK, and SAVE subcommands of MODUle are also performed when DONE is issued.

Command

```
DONE
```

6.5.16. OMIT

OMIT Command

Purpose

The OMIT subcommand of MODUles has no arguments. It's purpose is to exit MODUles without saving any changes.

Command

OMIT

7. SEARCHING THE DATABASE

SERCH Processor

Purpose

The SERCH Processor is used to interrogate the VAPEPS data base. Through various sub-commands and sub-processors you may locate all events and/or data modules that satisfy certain criteria.

Command

SERCH

This command has no arguments.

Overview

There are three basic types of sub-commands.

Searching commands - These form lists of events/data modules that match input conditions.

BOOK,EVENT,CONFIG,MODULE,GENERAL

Printing commands - These write the contents of the lists or portions of the data base to the screen.

LIST

List operations - These perform operations on the lists. Operations include logical functions AND,OR,NOT as well as the ability to save lists on file and restore them.

FIND,NAME,SAVE,RESTORE,STATUS

7.1. NAME

NAME command

Purpose

The purpose of the name command is to set what the input and output event number lists are set to.

Command

```
NAME  IN,OUT (CIRC)
```

Where

```
IN,OUT - Input and output list names.  Default is 'ALL','OUT'
CIRC   - Is the optional string 'CIRC' which causes the output list to
         become the input list.
```

Overview

While in SERCH there will always be an input list name and an output list name. Originally these are set to 'ALL' and 'OUT'. When using a primary command, the searching constraints are applied to all events contained in the input list. Those events that match are then placed in the output list.

The basic form of primary commands is

```
BOOK  INPUT  OUTPUT  (CIRC)
```

where INPUT is the name of desired input list and OUTPUT is the name of the desired output list. Both names may be omitted in which case the INPUT and OUTPUT names are not changed. To change only the output name, a $ may be used as a the INPUT name.

Each subsequent search operation will examine events in the INPUT list placing matching events in the output list. The second such operation will overwrite the current output list, unless the output list name is changed. Each primary section provides the NAME sub command (NAME INPUT OUTPUT CIRC) to enable the names to be changed between operations.

The CIRC argument may be included to set the 'circular' mode. When this mode is set, the input list no longer remains fixed from operation to operation. After each operation, the output list from that operation becomes the input list for the next operation.

7.2. BOOK

BOOK command

The BOOK sub-command provides the interrogation of the BOOK section. It is entered via the command:

Command

```
BOOK INPUT,OUTPUT,CIRC
```

Where

```
INPUT  - The name of the input list.
OUTPUT - Name of the output list.
CIRC   - Keyword which forces the output list to become the input list
         after a successful search.
```

7.2.1. AND

AND Command

Purpose

Finds all events from the list of input events that have **all** of the specified specs in the appropriate book section, and writes the event numbers to the output list.

Command

```
AND  BOOKSEC  SPEC1,SPEC2,...
or
AND BETWEEN TIMEDATE START END
```

Where

```
BOOKSEC  - Section of BOOK to search.
         = PROCESSING  - Searches the processing agency field.
         = CONTRACTING - Searches the contracting agency field.
         = COGNIZANT   - Searches the cognizant agency field.
         = VEHICLE     - Searches the vehicle field.
         = EVENT       - Searches the event field.
         = LOCATION    - Searches the location field.
         = TIME        - Searches for the specified time.
```

```
          = DATE       - Searches for the specified date.
    SPEC1,SPEC2,...
          - List of descriptions to search for.  All descriptions must be
            found in the particular field in order for an event to match.
            Each SPEC may contain up to 12 characters. The SPECs may
            contain one or more question marks (?) which are used as wild
            cards. (?AB?D? would match ABCDE or XXABYYD)

    BETWEEN  - Keyword that triggers a search for a specified time or date
               range.
    TIMEDATE - Either TIME or DATE as described under BOOKSEC.
    START    - Starting time or date.
    END      - Ending time or date.  All events that have DATE or TIME values
               between START and END will be found.
```

Example

Search for all events between the dates 10/03/65 and 05/15/81.

```
AND BETWEEN DATE 10/03/65 05/15/81
```

Search for all events between the times 12:30:00 and 15:45:00.

```
AND BETWEEN TIME 12:30:00 15:45:00
```

Find all events whose cognizant agency is NASA.

```
AND COGNIZANT NASA
```

7.2.2. OR

OR Command

Purpose

Finds all events from the list of input events that have **at least one** of the specified specs in the appropriate book section, and writes the event numbers to the output list.

Command

```
OR  BOOKSEC  SPEC1,SPEC2,...
or
OR BETWEEN TIMEDATE START END
```

Where

BOOKSEC - Section of BOOK to search.
= PROCESSING - Searches the processing agency field.
= CONTRACTING - Searches the contracting agency field.
= COGNIZANT - Searches the cognizant agency field.
= VEHICLE - Searches the vehicle field.
= LOCATION - Searches the location field.
= TIME - Searches for the specified time.
= DATE - Searches for the specified date.

SPEC1,SPEC2,...
- List of descriptions to search for. **At least one** description must be found in the particular field in order for an event to match. Each SPEC may contain up to 12 characters. The SPECs may contain one or more question marks (?) which are used as wild cards. (?AB?D? would match ABCDE or XXABYYD)

BETWEEN - Keyword that triggers a search for a specified time or date range.
TIMEDATE - Either TIME or DATE as described under BOOKSEC.
START - Starting time or date.
END - Ending time or date. All events that have DATE or TIME values between START and END will be found.

Example

Search for all events between the dates 10/03/65 and 05/15/81.

```
OR BETWEEN DATE 10/03/65 05/15/81
```

Search for all events between the times 12:30:00 and 15:45:00.

```
OR BETWEEN TIME 12:30:00 15:45:00
```

Find all events whose cognizant agency is NASA or AEROSPACE.

```
OR COGNIZANT NASA AEROSPACE
```

7.2.3. NOT

NOT Command

Purpose

Finds all events from the list of input events that **do not contain any** of the specified specs in the appropriate book section, and writes the event numbers to the output list.

Command

```
NOT  BOOKSEC  SPEC1,SPEC2...
Where

    BOOKSEC  - Section of BOOK to search.
             = PROCESSING  - Searches the processing agency field.
             = CONTRACTING - Searches the contracting agency field.
             = COGNIZANT   - Searches the cognizant agency field.
             = VEHICLE     - Searches the vehicle field.
             = LOCATION    - Searches the location field.
             = TIME        - Searches for the specified time.
             = DATE        - Searches for the specified date.
    SPEC1,SPEC2,...
             - List of descriptions to search for.  None of the descriptions
               must  be found in the particular field in order for an event to
               match. Each SPEC may contain up to 12 characters. The SPECs may
               contain one or more question marks (?) which are used as wild
               cards. (?AB?D? would match ABCDE or XXABYYD)

    BETWEEN  - Keyword that triggers a search for a specified time or date
               range.
    TIMEDATE - Either TIME or DATE as described under BOOKSEC.
    START    - Starting time or date.
    END      - Ending time or date.  All events that have DATE or TIME values
               not between START and END will be found.
```

Example

Search for all events not between the dates 10/03/65 and 05/15/81.

```
NOT BETWEEN DATE 10/03/65 05/15/81
```

Search for all events not between the times 12:30:00 and 15:45:00.

```
NOT BETWEEN TIME 12:30:00 15:45:00
```

Find all events whose cognizant agency is not NASA or AEROSPACE.

NOT COGNIZANT NASA AEROSPACE

7.2.4. NAME

NAME Command

Purpose

Changes the input and output list names. If CIRC is specified, the circular mode is enabled.

Command

```
NAME  INPUT,OUTPUT  (CIRC)
```

Where

```
INPUT  - The name of the input list.
OUTPUT - Name of the output list.
CIRC   - Keyword which forces the output list to become the input list
         after a successful search.
```

7.2.5. INTERSECT

INTERSECT Command

Purpose

Forms intersection of current input list with the named list, and writes the resulting list to the output list.

Command

```
INTERSECT  NAME
```

Where

```
NAME - Is the name of the list to be intersected with the input list.
```

7.2.6. INCLUDE

INCLUDE Command

Purpose

Combines the input list with the named list, and writes the result to the output list.

Command

```
INCLUDE  NAME
```

Where

NAME - Is the name of the list to be combined with the input list.

7.2.7. EXCLUDE

EXCLUDE Command

Purpose

Finds all events on the input list which are not present in the named list, and writes those event numbers to the output list.

Command

```
EXCLUDE NAME
```

Where

NAME - Is the name of the list to remove from the input list.

7.2.8. SYNONYMS

SYNONYMS Command

Purpose

This command turn on and off synonym matching from the data dictionary.

Command

```
SYNONYMS  (IOP)

Where

   IOP - Option to turn on or off synonym matching.
       = ON  - Turns synonym matching on.
       = OFF - Turns synonym matching off.
         If not input then the current status is displayed.
```

General

With this option on all input words are expanded to include their synonyms. The search logic works something like this:

```
Given the following command:
   AND  A,B,C
The following is implied:
   A.OR.(One of it's synonyms).AND.
   B.OR.(One of it's synonyms).AND.
   C.OR.(One of it's synonyms)

Given the following command:
   OR  A,B,C
The following is implied:
   A.OR.(One of it's synonyms).OR.
   B.OR.(One of it's synonyms).OR.
   C.OR.(One of it's synonyms)

Given the following command:
   NOT  A,B,C
The following is implied:
   .NOT.(A.OR.(One of it's synonyms)).AND.
   .NOT.(B.OR.(One of it's synonyms)).AND.
   .NOT.(C.OR.(One of it's synonyms))
```

This process can become quite time consuming if the combined number of words and synonyms gets very large. It can also have some undesirable side effects if a synonym that you don't wish to use has been input in the data dictionary. On the other hand it can be quite useful if you are unsure of exact word for which you are looking.

7.2.9. DONE

DONE Command

Purpose

Terminate execution of the current processor and returns control to the main level of SERCH.

Command

DONE
or
Carriage return.

There are no arguments to this command. A carriage return is equivalent to the command DONE.

7.3. EVENT

EVENT Command

Purpose

The EVENT section is used to search the PROC portion of the BOOK data. The major difference between EVENT searches and BOOK searches are that searches are done on 4 character words, as opposed to the full 12 character words that are used in BOOK.

Command

```
EVENT INPUT,OUTPUT,CIRC
```

Where

```
INPUT  - The name of the input list.
OUTPUT - Name of the output list.
CIRC   - Keyword which forces the output list to become the input list
         after a successful search.
```

Search command

There is only one search command in this section. It is an implied OR command. Wild cards (?'s) are not supported in this section. The OR is not specified as part of the command.

Command

```
SPEC1,SPEC2,....
```

Where

```
SPEC1,SPEC2,... - List of descriptions in the PROCESSING agency section
                  of book.  If an event contains at least on of the
                  descriptions then a match occurs.
```

7.3.1. NAME

NAME Command

Purpose

Changes the input and output list names. If CIRC is specified, the circular mode is enabled.

Command

```
NAME  INPUT,OUTPUT  (CIRC)
```

Where

```
INPUT  - The name of the input list.
OUTPUT - Name of the output list.
CIRC   - Keyword which forces the output list to become the input list
         after a successful search.
```

7.3.2. INTERSECT

INTERSECT Command

Purpose

Forms intersection of current input list with the named list, and writes the resulting list to the output list.

Command

```
INTERSECT  NAME
```

Where

```
NAME - Is the name of the list to be intersected with the input list.
```

7.3.3. INCLUDE

INCLUDE Command

Purpose

Combines the input list with the named list, and writes the result to the output list.

Command

```
INCLUDE  NAME
```

Where

NAME - Is the name of the list to be combined with the input list.

7.3.4. EXCLUDE

EXCLUDE Command

Purpose

Finds all events on the input list which are not present in the named list, and writes those event numbers to the output list.

Command

```
EXCLUDE NAME
```

Where

NAME - Is the name of the list to remove from the input list.

7.3.5. DONE

DONE Command

Purpose

Terminate execution of the current processor and returns control to the main level of SERCH.

Command

DONE
or
Carriage return.

There are no arguments to this command. A carriage return is equivalent to the command DONE.

7.4. CONFIGURATION

CONFIGURATION Command

Purpose

The CONF section is used to search the configuration trees of events listed in the input list. The target of the search is the generic and specific qualifiers that may be used to identify each branch in the tree.

Command

```
CONFIGURATION INPUT,OUTPUT,CIRC
```

Where

```
INPUT  - The name of the input list.
OUTPUT - Name of the output list.
CIRC   - Keyword which forces the output list to become the input list
         after a successful search.
```

7.4.1. MODE

MODE Command

Purpose

Set one of three search modes.

Command

```
MODE  MODSPEC
```

Where

```
MODSPEC - Search mode.
        = GENERIC  - Only generic names are searched. (default mode)
        = SPECIFIC - Only specific names are searched.
        = BOTH     - Both generic and specific names are searched.
```

7.4.2. AND

AND Command

Find all events which **have all** of the specified descriptions. The search is performed as specified by the MODE command.

Command

```
AND SPEC1,SPEC2,...
```

Where

```
SPEC1,SPEC2,... - List of names in either the generic or specific
                  sections or in both.  See the MODE command for
                  details on how to set this.
```

For generic mode, the SPECs are generic names (12 chars max), for specific mode, the SPECs are specific names (12 chars max), and for BOTH, the SPECs must be entered in pairs, as

```
AND  GEN1,SPEC1  GEN2,SPEC2
```

NODE Command

A special form of the AND command is NODE:

```
NODE  SPEC1,SPEC2,....
```

where a given event must have a branch that contains all of the listed specs in order to match. (Remember that for AND, the specs may exist anywhere in the tree)

7.4.3. OR

OR Command

Find all events which **have at least one** of the specified descriptions. The search is performed as specified by the MODE command.

Command

```
OR SPEC1,SPEC2,...
```

Where

```
   SPEC1,SPEC2,... - List of names in either the generic or specific
                     sections or in both.  See the MODE command for
                     details on how to set this.
```

For generic mode, the SPECs are generic names (12 chars max), for specific mode, the SPECs are specific names (12 chars max), and for BOTH, the SPECs must be entered in pairs, as

```
   OR GEN1,SPEC1  GEN2,SPEC2
```

7.4.4. NOT

NOT Command

Find all events which **do not have any** of the specified descriptions. The search is performed as specified by the MODE command.

Command

```
NOT SPEC1,SPEC2,...
```

Where

```
   SPEC1,SPEC2,... - List of names in either the generic or specific
                     sections or in both.  See the MODE command for
                     details on how to set this.
```

For generic mode, the SPECs are generic names (12 chars max), for specific mode, the SPECs are specific names (12 chars max), and for BOTH, the SPECs must be entered in pairs, as

```
   NOT  GEN1,SPEC1  GEN2,SPEC2
```

7.4.5. TREE

TREE Command

Purpose

List the configuration tree for the specified event.

Command

```
TREE  IVENT
```

Where

IVENT - Absolute number of the event. Normally you look for some item on a tree to get a list, then list the contents of the list to get the event numbers. These numbers may be used in the TREE command.

7.4.6. SAVE

SAVE Command

Purpose

Save a specified list on a DAL file so that it can be accessed later.

Command

```
SAVE LST,NU,NAME,VER
```

Where

```
LST          - Name of the list to save.
NU,NAME,VER  - DAL unit, element and version where list is to be  saved.
```

7.4.7. RESTORE

RESTORE Command

Purpose

Restore a previously SAVEd list from the specified DAL file.

Command

```
RESTORE LST,NU,NAME,VER
```

Where

```
LST          - Name of the list to create.
NU,NAME,VER  - DAL unit, element and version where list was saved.
```

7.4.8. LIST

LIST Command

Purpose

List events contained in the specified lists.

Command

```
LIST LST1,LST2, ... LST14
```

Where

```
LST1,LST2,...LST14
          - Up to 14 list names.  If none are specified then the  output
            list used.
```

Notes

The entry for each event includes all of the information from the PROC section of the BOOK information, plus the name of the file which contains the data for the event. If no lists are specified, the current output list is listed.

7.4.9. STATUS

STATUS Command

Purpose

This command will return the names of the current input and output lists. If CIRC was set, STATUS will so inform the user.

Command

```
STATUS
```

7.4.10. NAME

NAME Command

Purpose

Changes the input and output list names. If CIRC is specified, the circular mode is enabled.

Command

```
NAME  INPUT,OUTPUT  (CIRC)
```

Where

```
INPUT  - The name of the input list.
OUTPUT - Name of the output list.
CIRC   - Keyword which forces the output list to become the input list
         after a successful search.
```

7.4.11. DONE

DONE Command

Purpose

Terminate execution of the current processor and returns control to the main level of SERCH.

Command

DONE
or
Carriage return.

There are no arguments to this command. A carriage return is equivalent to the command DONE.

7.5. MODULE

MODULE Command

Purpose

Search the modules section of the data base for the specified items.

Command

```
MODULES INPUT,OUTPUT,CIRC
```

where INPUT,OUTPUT, and CIRC control the list names for subsequent operations. These names may be omitted, in which case they retain their current values.

7.5.1. MODE SETTING COMMANDS

Purpose

Turn on or off certain modes of searching.

Command

```
ON  ITEM
OFF ITEM
```

Where

```
    ITEM - Mode from the following list:
         = PARAMETERS     - Turn on or off parameter searching mode.   See
                            element searching for information on it's
                            affect.
         = DESCRIPTIONS   - Turn on or off description searching mode.  This
                            has the opposite effect of the PARAMETERS mode.
         = CHANNELS       - Turn on of off channel searching mode.  When  on
                            only those modules with channels will be
                            searched.  When off all modules will be
                            searched.
```

7.5.2. MODULE SEARCHING SUBCOMMANDS

SYSTEM Command

Purpose

Find all modules from the input list that have the indicated coordinate system and lie within the indicated coordinate range.

Command

```
SYSTEM SYST X=IX,JX Y=IY,JY Z=IZ,JZ
```

Where

```
SYST     - Name of the coordinate system desired.  Must always be input.
           A dollar sign ($) may be entered to match all coordinate
           systems.
X=IX,JX Y=IY,JY Z=IZ,JZ
         - X, Y, and Z refer to the three axes of a three dimensional
           cartesian coordinate system.  (IX,JX), (IY,JY) and (IZ,JZ)
           are the lower and upper bounds respectively in each axis.
           Only modules that fall within these limits will be considered
           a match.  Any or all of the ranges may be omitted.
```

Example

Find all systems which have a Y coordinate range from 1 to 10.

```
SYSTEM $ Y=1,10
```

Notes

This command is not affected by the DESCRIPTIONS/PARAMETERS mode. The CHANNELS mode does effect it. If the CHANNELS mode is on then at least one element in a model must have channels associated with it before the module will be considered in the search.

DESC Command

Purpose

Search the module descriptions.

Command

```
DESC NAME1(,NAME2(,NAME3))
```

Where

NAME1(,NAME2(,NAME3))
- Up to three descriptions to search for. All name specified must be in a module for a match to occur.

Notes

This command is not affected by the DESCRIPTIONS/PARAMETERS mode. The CHANNELS mode does effect it. If the CHANNELS mode is on then at least one element in a model must have channels associated with it before the module will be considered in the search.

7.5.3. ELEMENT SEARCHING COMMANDS

The element searching commands are similar to the module searching commands, except that the search is performed on individual SEA elements within the modules, rather that on the modules as a whole.

EXTA, SKIN, INTA, MONT, INST, FRAM Commands

Purpose

Search for information in one of the six elements of a 5 element SEA model.

Command

```
SEANAM NAME1(,NAME2(,NAME3))
or
SEANAM ITYPE,A=IA,JA B=IB,JB C=IC,JC D=ID,JD
```

Where

SEANAM - Is one of EXTA, SKIN, INTA, MONT, INST or FRAM. This designates what element is to be searched.

NAME1(,NAME2(,NAME3))
- Up to three descriptions to search for. All name specified must be in a module for a match to occur. This mode works only if DESCRIPTIONS is on.

ITYPE - Type of SEA element to search for. A dollar sign ($) will match all types.

A=IA,JA B=IB,JB C=IC,JC D=ID,JD
- Indicates which nondimensional parameter (A, B, C or D) to search on and the range in which to search for each (IA,JA

IB,JB ...). Any or all of these may be omitted. This mode works only when PARAMETERS mode is on.

7.5.4. MISCELLANEOUS COMMANDS

LIST Command

Purpose

Lists the modules in the OUTPUT list.

Command

```
LIST
```

DUMP Command

Purpose

Produces a long list of the specified module or the output list.

Command

```
DUMP
or
DUMP NUM
```

Where

NUM - Is an absolute module number to be listed. If omitted then all modules in the output list will be listed.

NAME Command

Purpose

Changes the input and output list names. If CIRC is specified, the circular mode is enabled.

Command

```
NAME  INPUT,OUTPUT  (CIRC)
```

DONE Command

Purpose

Exit the MODULES subprocessor.

Command

```
DONE
or
Carriage return
```

7.6. GENERAL

GENERAL Search processor

Purpose

This processor searches the entire Master File for the input list of words.

Command

```
GENERAL  IN,OUT (CIRC)
```

Where

```
IN,OUT - Input and output lists.  Default is 'ALL','OUT'.
CIRC   - Is the optional string 'CIRC' which causes the output list
         to become the input list.
```

7.6.1. LIST

LIST Command

Purpose

List events contained in the specified lists.

Command

```
LIST LST1,LST2, ... LST14
```

Where

```
LST1,LST2,...LST14
          - Up to 14 list names.  If none are specified then the  output
            list used.
```

Notes

The entry for each event includes all of the information from the PROC section of the BOOK information, plus the name of the file which contains the data for the event. If no lists are specified, the current output list is listed.

7.6.2. SAVE

SAVE Command

Purpose

Save a specified list on a DAL file so that it can be accessed later.

Command

```
SAVE LST,NU,NAME,VER
```

Where

```
LST          - Name of the list to save.
NU,NAME,VER  - DAL unit, element and version where list is to be  saved.
```

7.6.3. RESTORE

RESTORE Command

Purpose

Restore a previously SAVEd list from the specified DAL file.

Command

```
RESTORE LST,NU,NAME,VER
```

Where

```
LST          - Name of the list to create.
NU,NAME,VER  - DAL unit, element and version where list was saved.
```

7.6.4. STATUS

STATUS Command

Purpose

This command will return the names of the current input and output lists. If CIRC was set, STATUS will so inform the user.

Command

```
STATUS
```

7.6.5. NAME

NAME command

Purpose

The purpose of the name command is to set what the input and output event number lists are set to.

Command

```
NAME  IN,OUT (CIRC)
```

Where

```
IN,OUT - Input and output list names.  Default is 'ALL','OUT'
CIRC   - Is the optional string 'CIRC' which causes the output
         list to become the input list.
```

See the SERCH subtopic NAME for more information on this command.

7.6.6. INTERSECT

INTERSECT Command

Purpose

Forms intersection of current input list with the named list, and writes the resulting list to the output list.

Command

```
INTERSECT  NAME
```

Where

NAME - Is the name of the list to be intersected with the input list.

7.6.7. INCLUDE

INCLUDE Command

Purpose

Combines the input list with the named list, and writes the result to the output list.

Command

```
INCLUDE  NAME
```

Where

NAME - Is the name of the list to be combined with the input list.

7.6.8. EXCLUDE

EXCLUDE Command

Purpose

Finds all events on the input list which are not present in the named list, and writes those event numbers to the output list.

Command

```
EXCLUDE NAME
```

Where

NAME - Is the name of the list to remove from the input list.

7.6.9. AND

AND Command

Purpose

This command searches for the input words with the condition that all of the input words must be in each event for a match to occur.

Command

```
AND  WORD1(,WORD2(,...(,WORDN)))
```

Where

WORD... - Is the input list of word to be searched on.

Note

Special conditions apply if synonym matching is used. See the SYNONYM command for more information.

7.6.10. OR

OR Command

Purpose

This command searches for the input words with the condition that at least on of the input words must be in each event for a match to occur.

Command

```
OR  WORD1(,WORD2(,...(,WORDN)))
```

Where

WORD... - Is the input list of word to be searched on.

Note

Special conditions apply if synonym matching is used. See the SYNONYM command for more information.

7.6.11. NOT

NOT Command

Purpose

This command searches for the input words with the condition that none of the input words can be in an event for a match to occur.

Command

```
NOT  WORD1(,WORD2(,...(,WORDN)))
```

Where

WORD... - Is the input list of word to be searched on.

Note

Special conditions apply if synonym matching is used. See the SYNONYM command for more information.

7.6.12. SYNONYMS

SYNONYMS Command

Purpose

This command turn on and off synonym matching from the data dictionary.

Command

```
SYNONYMS  (IOP)

Where

   IOP - Option to turn on or off synonym matching.
       = ON  - Turns synonym matching on.
       = OFF - Turns synonym matching off.
         If not input then the current status is displayed.
```

General

With this option on all input words are expanded to include their synonyms. The search logic works something like this:

```
Given the following command:
   AND  A,B,C
The following is implied:
   A.OR.(One of it's synonyms).AND.
   B.OR.(One of it's synonyms).AND.
   C.OR.(One of it's synonyms)

Given the following command:
   OR  A,B,C
The following is implied:
   A.OR.(One of it's synonyms).OR.
   B.OR.(One of it's synonyms).OR.
   C.OR.(One of it's synonyms)

Given the following command:
   NOT  A,B,C
The following is implied:
   .NOT.(A.OR.(One of it's synonyms)).AND.
   .NOT.(B.OR.(One of it's synonyms)).AND.
   .NOT.(C.OR.(One of it's synonyms))
```

This process can become quite time consuming if the combined number of words and synonyms gets very large. It can also have some undesirable side effects if a synonym that you don't wish to use has been input in the data dictionary. On the other hand it can be quite useful if you are unsure of exact word for which you are looking.

7.6.13. DONE

DONE Command

Purpose

Terminate execution of the current processor and returns control to the main level of SERCH.

Command

DONE
 or
Carriage return.

There are no arguments to this command. A carriage return is equivalent to the command DONE.

7.7. FIND

FIND Command

Purpose

The FIND command does not actually perform any searches. It is used to make logical comparisons between lists.

Command

```
FIND INPUT,OUTPUT,CIRC
```

Where

```
INPUT  - The name of the input list.
OUTPUT - Name of the output list.
CIRC   - Keyword which forces the output list to become the input list
         after a successful search.
```

LOGOP LISTS

7.7.1. AND

AND Command

Purpose

Create a list which consists of the intersection of the lists specified and the input list. The output list will contain all the events which are common to all the lists including the input list. If INPUT = ALL then the input list is not included.

Command

```
AND LIST1,LIST2,...LIST14
```

Where

```
LIST1,LIST2,...LIST14 - Are up to 14 input lists to be operated  on.
```

7.7.2. OR

OR Command

Purpose

Create a list which consists of the union of the lists specified and the input list. The output list will contain all the events which are common to all the lists including the input list. If INPUT = ALL then the input list is not included.

Command

```
OR LIST1,LIST2,...LIST14
```

Where

LIST1,LIST2,...LIST14 - Are up to 14 input lists to be operated on.

7.7.3. NOT

NOT Command

Purpose

When INPUT = ALL, a list is created which contains all events which are not in any of the lists specified. When INPUT not equal to ALL a list is created which contains all of the event in the input list which do not appear in any of the lists specified.

Command

```
NOT LIST1,LIST2,...LIST14
```

Where

LIST1,LIST2,...LIST14 - Are up to 14 input lists to be operated on.

7.7.4. NAME

NAME command

Purpose

The purpose of the name command is to set what the input and output event number lists are set to.

Command

```
NAME  IN,OUT (CIRC)
```

Where

```
IN,OUT - Input and output list names.  Default is 'ALL','OUT'
CIRC   - Is the optional string 'CIRC' which causes the output
         list to become the input list.
```

See the SERCH subtopic NAME for more information on this command.

7.7.5. DONE

DONE Command

Purpose

Terminate execution of the current processor and returns control to the main level of SERCH.

Command

```
DONE
or
Carriage return.
```

There are no arguments to this command. A carriage return is equivalent to the command DONE.

7.8. LIST

LIST Command

Purpose

List events contained in the specified lists.

Command

```
LIST LST1,LST2, ... LST14
```

Where

```
LST1,LST2,...LST14
          - Up to 14 list names.  If none are specified then the  output
            list used.
```

Notes

The entry for each event includes all of the information from the PROC section of the BOOK information, plus the name of the file which contains the data for the event. If no lists are specified, the current output list is listed.

7.9. SAVE

SAVE Command

Purpose

Save a specified list on a DAL file so that it can be accessed later.

Command

```
SAVE LST,NU,NAME,VER
```

Where

```
LST          - Name of the list to save.
NU,NAME,VER - DAL unit, element and version where list is to be  saved.
```

7.10. RESTORE

RESTORE Command

Purpose

Restore a previously SAVEd list from the specified DAL file.

Command

```
RESTORE LST,NU,NAME,VER
```

Where

```
LST          - Name of the list to create.
NU,NAME,VER  - DAL unit, element and version where list was saved.
```

7.11. STATUS

STATUS Command

Purpose

This command will return the names of the current input and output lists. If CIRC was set, STATUS will so inform the user.

Command

```
STATUS
```

8. DATABASE ADMINISTRATION

ADMINISTRATION Command

Purpose

The ADMINISTRATION processor performs the administrative functions associated with the VAPEPS data base. These functions include:

1) Saving portions of event files in the master file.
2) Spilling an event (or events) into a symbolic file as card images. These card images are suitable for transmittal to other sites.

Command

ADMINISTRATION MU

Where

MU - The unit number of the master file. MU defaults to 13 if not specified. MU will set the master file unit number for the duration of the run unless reset by re-entering the ADMINISTRATION processor with a different value. Under normal circumstances MU should always be equal to the default value of 13.

8.1. PREPARATION OF DATA

The following steps must be done to the data before it is saved on the master file.

1) Enter the data with the ENTER command or it's equivalent. Consult Chapter 5 for more information.

2) Prepare the data with the PREP command. Consult Chapter 6 for more information.

3) Set up word lists with the RUN=PADMIN command. Consult Chapter 3 RUN=PADMIN command for more information.

The remaining steps must be executed by the database administrator to insure that proper naming conventions are used. The master file and dictionary file normally will be write protected so that only the database administrator will be able to access them.

4) Define all words with the DICTIONARY/PREADMIN command. Consult Chapter 9 for more information.

At this point if all the words are properly defined the event can be saved on the master file.

8.2. SAVE

SAVE Command

Purpose

SAVE is used to write portions of an event file to the master file. Once an event has been saved it is available for searching, etc.

Command

```
SAVE NU,VENT
```

Where

NU,VENT - The DAL unit and event name of the event to be saved.

Notes

Note that SAVE writes into the MASTER file. Thus you must have write access on the MASTER file in order to use this command. Typically only the VAPEPS Administrator has write access on the MASTER file.

Data Dictionary

If an authorization list, prepared by the DICTIONARY/PREADMIN command, is not present then the event will not be saved on the MASTER file.

8.3. SPILL

SPILL Command

Purpose

The SPILL sub-command is used to dump an entire event in symbolic form so that it can be transmitted to other sites. An optional form of the SPILL sub-command may be used to dump only the PREP deck for an event.

Command

```
SPILL NU,VENT NOUT,NELT,NVER
```

Where

NU,VENT - The DAL unit and version of the event for which card images are to be produced.

NOUT - Either a FORTRAN unit number or a DAL unit number. If NELT and NVER are left off of the sub-command, NOUT is the FORTRAN unit to which the card images will be written. If NELT and NVER are specified, NOUT is the DAL unit to which card images will be written.

NELT,NVER - The element name and version on DAL unit NOUT to which card images will be written. Note that NELT and NVER are optional. If not specified, card images will be written to FORTRAN unit NOUT instead of DAL element NOUT,NELT,NVER.

In this form, SPILL will write the entire event (spectral data, book-keeping info, configuration tree and data modules) in symbolic form. An alternate form of the SPILL sub-command will write all of the above mentioned information except the spectral data. This is accom- plished by specifying the value of NU as a negative number. In this case the spectral data would be transmitted separately. See the desc- ription of the RUN=ESAV runstream in chapter 13 for details on creating spectral data decks for transmission to other sites.

There is a runstream RUN=SPIL, which enters ADMINISTRATION, and spills all events in a given file.

8.4. FLIP

FLIP Command

Purpose

The FLIP command transposes several of the files in the MASTER file. Once transposed, the BOOK info (for example) may be interrogated using general VAPEPS commands as opposed to the more specific SERCH command. Since this usage requires a fair amount of experience with VAPEPS, FLIP is not normally used.

9. DIRECT DICTIONARY INTERFACE

DICTIONARY Command

Purpose

This command enters the Direct Dictionary Interface processor which allows the maintenance of the Data Dictionary. It also has some specialized functions for adding new events to the Data Base.

Command

```
DICTIONARY
or
DICTIONARY/PREADMIN  N1,EVENT
```

Where

N1,EVENT - DAL unit and event name for event which is to have it's words added to the Data Dictionary.

Notes

The Data Dictionary resides on DAL unit 15.

9.1. DEFINE

DEFINE command

Purpose

This command allows a definition to be associated with a word.

Command

```
DEFINE  WORD
-definition-
or
DEFINE WORD NDEF
```

Where

WORD - Is the word to be defined.

-definition- - Is the text that comprises the definition. It can be as many lines as desired. It is terminated by a blank line. The prompt for this will be the word input.

NDEF - Is the definition number which is to be associated with the word as defined in PREP. This only works when the PREADMIN option is used. This defaults to 1 so that where only one definition is used it is not necessary to use this command.

Notes

The definition is formated by making line breaks between words so there is no need to worry about making it appear well formatted on input. More than one definition can be input for a single word. Each definition has an integer number associated with it. The numbering is automatic staring at 1 and incrementing by 1 with each new definition added to a word. When a definition is deleted all higher numbered definition numbers are decremented by 1. This could cause problems later on. In most cases there will be only 1 definition.

9.2. DELETE

DELETE Command

Purpose

This command is used to delete a word, definition, or synonym.

Command

```
DELETE  WORD
or
DELETE WORD NDEF
or
DELETESYN WORD SYNONYM
```

Where

```
WORD    - Is the word on which to operate.
NDEF    - Is the number of the definition for WORD which is to be
          deleted.
SYNONYM - Is the synonym of WORD which is to be deleted.
```

Notes

When only WORD is specified all reference to that word is deleted including definitions and synonyms. This command is a little slow in that it has to search all of the word lists to delete other words pointers to this word.

9.3. LIST

LIST Command

Purpose

This command allows the definitions and synonyms of words to be displayed in an easily readable format.

Command

```
LIST
or
LIST WORD
or
LISTS SECTION
```

Where

WORD - Is a word for which the definitions and synonyms are to be displayed.
SECTION - Is a section for which words are to be displayed. (Advanced usage only.)

Notes

When no options are input the entire Data Dictionary is listed out in alphabetic order. A hardcopy of this listing is useful when filling out PREP worksheets.

Wild Card Character (?)

The character ? (question mark) can be used in the list command to match zero or more characters in a string.

Example: LIST ?LMSC?
This will match any word which has the string LMSC in it and list it out.

9.4. SPILL

SPILL Command

Purpose

This command is used to 'spill' out the contents of the Data Dictionary in a manner in which the computer can readily read it back in.

Command

```
SPILL
or
SPILLWORD WORD
or
SPILLS SECTION
```

Where

```
WORD    - Is a word for which all information is to be spilled.
SECTION - Is a section for which all information is to be spilled.
```

Notes

When no options are input the entire Data Dictionary is spilled. At this time all output is directed to fortran unit 6. This limitation will be removed in later versions.

Wild Card Character (?)

The character ? (question mark) can be used in the SPILL command to match zero or more characters in a string.

Example: SPILL ?LMSC?
This will match any word which has the string LMSC in it and spill it out.

9.5. SYNONYM

SYNONYM Command

Purpose

This command creates associates another word in the Data Dictionary to the word specified. This association means that the two words are synonymous.

Command

```
SYNONYM WORD SYNONYM1 (SYNONYM2 (SYNONYM3 ... (SYNONYMn)))
```

Where

```
WORD      - Is the word for which SYNONYM is a synonym.
SYNONYMn  - Word which is a synonym of WORD. As many synonyms as will
            fit on one line can be input for WORD.
```

Notes

Synonyms can be used by the SERCH command to expand the scope of a search such that is will find all events which have words similar in meaning to the word(s) which were specified in the search command. See SERCH for more information.

Allowed synonyms

Synonyms must be a previously defined word. A word cannot be a synonym of itself. Multiple instances of the same synonym for a word are not allowed.

9.6. PREADMIN

PREADMIN Option

Purpose

This option is used to interface the PREPed event with the Data Dictionary and the Master File.

Notes

The following things happen when this option is used:

1) All the words in a list prepared by the RUN=PADMIN runstream which are associated with a particular event are read.

2) Those words not appearing in the Data Dictionary are displayed.

3) The Direct Dictionary Interface is entered.

4) Words which were not defined are defined. This can be done all at once or in separate executions of the DICTIONARY/PREADMIN command.

5) When all the words are adequately defined an authorization vector is written to the event file. This tells ADMIN that all the words in the event have been properly entered into the Data Dictionary. ADMIN will not allow an event to be saved unless this step is finished.

6) The matrix BOOK LST$ is updated on the Master File. This is what controls the warning in the BOOK section of PREP which tells you that certain items do not follow the convention.

10. STATISTICAL ENERGY MODELER

Introduction

This section contains descriptions of commands that can be used for performing a variety of acoustic and structural predictions. Besides providing for strictly analytical predictions, prediction tools are also provided that make use of the VAPEPS database of vibroacoustic data and structural parameters to predict the environment for payload components of new spacecraft systems; such predictions are empirical and are referred to as either extrapolated or scaled predictions depending on the analytical details around which the prediction tool was developed. To be considered by the analyst in the selection of a particular prediction tool, is the familiarity of the analyst with the underlying concepts upon which the prediction tool was developed -- a very important consideration for the successful application of the prediction tool to practical problems. The following type of predictions are available:

1) Theoretical predictions. These predictions are all based on Statistical Energy Analysis (SEA) concepts. SEA treats the various acoustic/structural components of a spacecraft system as energy storage elements; the primary dynamic variable is then energy from which other dynamic variables such as sound pressure, acceleration, etc. are derived. A SEA model is made up of a series of discrete energy storage elements (SEA elements), interconnected by a network of energy paths. The Statistical Energy MODeler (SEMOD) is used to construct the SEA model. To use this prediction technique effectively the analyst must be familiar with SEA theory; the low frequency limitations of SEA must be understood, as well as, the assumptions made with respect to the manner in which SEA elements couple together to receive and transmit energy -- this interchange of energy is characterized by parameters referred to as coupling loss factors.

2) Extrapolated predictions (EXTRAP I). This empirical prediction is used to make response predictions of a new system based on the measured data of a baseline system. Two SEA models (one for the new system and the other for the baseline system) are used. Data from a baseline model -- selected from the VAPEPS database -- is used to obtain a transfer function. This transfer function is then corrected for acoustic/structural differences between the database spacecraft and the new spacecraft based on the SEA models. The SEMOD processor is again used to build the appropriate models; the command PREDICT/EXTRAP/I is used to make the prediction.

3) Scaling predictions (EXTRAP II). This is another empirical prediction technique which captures the essential features of techniques that have been previously developed and employed by the aerospace community (eg. Condos-Butler, et. al.). Corrections, based on scaling, are made for

differences in structural parameters and differences in acoustic excitation between the baseline and new spacecraft systems. This method employs the SEMOD/EXTRAP/II command to build the required models and the PREDICT/EXTRAP/II command to make the prediction.

The extrapolated prediction EXTRAP I provides for the modeling of systems which are more complex than provided for by the scaling prediction EXTRAP II. Since EXTRAP I is based on SEA concepts it has low frequency limitations; EXTRAP II, however, is not based on SEA theory and has no such frequency limitations as long as the database contains data obtained from a spacecraft system that can be considered dynamically similar to the new system for which a prediction is to made. The method for establishing a baseline system from the VAPEPS database to have the desired dynamic characteristics is described in this section -- refer to the paragraphs entitled NON-DIMENSIONAL PARAMETERS.

4) Mean-squared spatial averaged stress. This method predicts the mean-square spatial averaged stress of a flat plate, curved plate, cylinder or user defined ratio of spatial average mean-square stress to mean-square displacement. The STRESS command is used to do this prediction (refer to Chapter 3).

5) Other predictions can be made by combining the general commands into a runstream.

What is SEMOD?

SEMOD (Statistical Energy MODeler) is an interactive/batch processor used to create Statistical Energy Analysis (SEA) models. The idea is to model a spacecraft system using a series of discrete SEA elements interconnected by a network of energy paths; each SEA element and energy path is labeled with a type number and assigned the appropriate acoustic/structural properties.

For example: if you have a cylinder in a reverberant acoustic chamber you would want to model the acoustic chamber as a reverberant field (element type 1), the cylinder as a cylinder (element type 4) and the internal acoustic space of the cylinder as a reverberant field (element type 1). You would then connect the external space to the cylinder (connection type 2) and the internal space to the cylinder (connection type 2). The non-resonant "mass law" connection from the external space to the internal space is modeled with connection type 5. The element and connection type numbers have been preassigned and are described in this section - refer to the paragraphs entitled PARAMETERS.

SEMOD allows an analyst almost unlimited flexibility in modeling, by using a variety of elements and connection types, the analyst can model nearly any type of structure desired either as (1) consistent with the predefined element and element connection types or (2) the analyst can exercise an option of utilizing independently derived connection types - the connection types are developed from the SEA coupling loss factors, refer to the paragraphs entitled ATACALC and PATHNAME of this section.

ATTENTION

SEMOD is designed to be easy to use from the standpoint of creating SEA models. However, this may lead the the deception that SEA modeling is easy. **SEA modeling is not easy.** It takes a lot of experience and a good understanding of the structural details with respect to the spacecraft being modeled, as well as, an understanding of the assumptions associated with SEA theory and the limitations of the coupling loss factors being used to characterize the interconnecting network or energy paths to be able to create a model that will yield rational predictions. Successful application of the SEA concept to practical problems presents a challenge that can be met only with substantial effort on the part of the analyst to appropriately model the spacecraft structure using the abstract elements and element interconnections with which SEA theory deals.

It is recommended that theoretical predictions be checked against data--retrieved from the VAPEPS database--from structures perceived to be dynamically similar to the structure for which a prediction is being made. If the analyst is uncertain about performing theoretical predictions, the use of the scaling prediction, EXTRAP II, is advised.

Input requirements.

Input is divided up into several steps. The following steps are normally required to create a model. (Command names appear in parenthesis).

1 Determine how the structure is to be modeled by examining structural drawings.

2 Obtain physical parameters of the structure to be modeled.

3 Input the elements and parameters for the model. (ELNAME)

4 Input the connections and connection dependant parameters. (PATHNAME)

5 Define the excitation element(s). (SETEXC)

6 Step through the prediction sequence. (MDENS, ATACALC, ATACO, CFAC, and, TPRD)

7 Display results. Tables and plots. (LIST, and RUN=PRSP)

Execution

The SEMOD processor is entered from the upper level of VAPEPS (question mark prompt) by use of the command SEMOD. The SEMOD command and sub-commands are described in the following documentation.

As you may have noticed there are quite a few commands in SEMOD. To the beginner this may lead to confusion. It would appear that several of these commands could be combined into a single command. This is true for the majority of simple models. The reason that these commands are broken up in this manner is to allow greater flexibility in modeling.

For example: If a user wishes to input his own damping loss factors this could be done between the ATACALC and ATACO commands.

This type of prediction modification allows a great degree of flexibility for an experienced SEMOD modeler. For the novice it means more typing but it also points out that the potential is there for user modification of the prediction process.

References

The following list of references is included for those interested in learning more about SEA. The list is far from complete but it is felt that these references give a good start for more research into SEA:

R. H. Lyon and G. Maidanik, "Power Flow Between Linearly Coupled Oscillators" J. Acoust. Soc. Am., Vol. 34, No. 5, pp. 623-639, (May 1962)

P. W. Smith, Jr., "Response and Radiation of Structural Modes Excited by Sound", J. Acoust. Soc. Am., Vol. 34, No. 5, pp. 640-647, (May 1962)

G. Maidanik, "Response of Ribbed Panels to Reverbverant Acoustic Fields", J. Acoust. Soc. Am., Vol. 34, No. 6, pp. 809-826, (June 1962)

E. Eichler, "Thermal Circuit Approach to Vibrations in Coupled Systems and the Noise Reuction of A Rectangular Box", J. Acoust. Soc. Am., Vol. 37, No. 6, pp. 995-1007, (June 1963)

R. H. Lyon and T. D. Scharton, "Vibrational Energy Transmission in a Three Element Structure", J. Acoust. Soc. Am., Vol. 38, No. 2, pp. 253-261, (August 1965)

R. H. Lyon, Statistical Energy Analysis of Dynamical Systems, MIT Press, Cambridge, MA, 1975

SEMOD processor

Purpose

This processor is used to build and predict responses of Statistical Energy Analysis (SEA) models. These models are created using commands and sub-processors within SEMOD.

Command

```
SEMOD N1,NAME1 (N2,NAME2)
```

Where

```
N1,NAME1 - Input dal unit and version name for model.
N2,NAME2 - Optional output dal unit and version name for model. If
           present, the model N1,NAME1 is copied to N2,NAME2 then all
           updates are made to N2,NAME2. If omitted, updates are made
           to N1,NAME1. Note: This option is not operational yet.
```

Utility commands

```
-----------------------------------------------------------------
DELETE      - Deletes an element or connection.
RENAME*     - Renames an element.
FREQUENCY   - Sets the frequency range.
SETEXC      - Sets the excitation elements.
SETRES      - Sets the response elements.  (EXTRAP I models)
EXCITATION  - Prompts for input excitation.
RESPONSE    - Prompts for input response.  (Baseline EXTRAP I models.)
CRFR        - Extracts appropriate critical and ring frequencies.
              (EXTRAP I models.)
LIST**      - Prints out a summary of your entire model.
CHECK*      - Checks model for inconsistencies.
READ*       - Reads in a previously developed model.
SAVE*       - Saves the current model as a separate model.
DONE        - Exits SEMOD.
QUIT        - Exits SEMOD without updating the model history matrix.
-----------------------------------------------------------------
*  Not yet operational.
** Partially operational.
```

Sub-processors

```
-----------------------------------------------------------------
ELNAME   - Used to define elements.
PATHNAME - Used to define connections.
FPAR     - Used to define frequency dependant parameters.
TEXT     - Used to input descriptive text.
-----------------------------------------------------------------
```

Prediction commands

MDENS	- Calculates modal density.
CFAC	- Calculates conversion factors.
ATACALC	- Calculates coupling loss factors.
ATACO	- Does inversion and calculates transfer functions.
TPRD	- Does theoretical prediction.
POWER	- Calculates the power flow.

10.1. ORDER OF EXECUTION

Some commands or processors must be executed in a certain order when building a model. The following list shows what commands must be executed before continuing on to any following commands.

ELNAME	Must be executed first to define elements
PATHNAME	Can be executed after ELNAME.
SETEXC	Can be executed after ELNAME.
FREQUENCY	Can be executed at any time.
EXCITATIO	Can be executed after SETEXC and FREQUENCY.
FPAR*	Can be executed after PATHNAME.
MDENS	Can be executed after ELNAME.
ATACALC	Can be executed after MDENS and PATHNAME.
ATACO	Can be executed after ATACALC and SETEXC.
CFAC	Can be executed after ELNAME.
TPRD	Can be executed after CFAC and ATACO.
POWER*	Can be executed after TPRD.

* Optional

Any other commands can be executed at any time.

10.2. EXAMPLE

```
VAP4.0 1-04/10/86-15:38:17     67200
*
*
*  This problem uses the data from the event SP01 in the VAPEPS database.
*
*
*
*
*  Enter SEMOD to build model.
 ?SEMOD 1,SAMP
Creating new model SAMP



*
*  Enter descriptive text.
SEMOD > TEXT WXYZ
 >Prediction of event SP01 using SEMOD prediction software.
 >
 >This model consists of a cone in a reverberant chamber.  The elements to
 >be modeled are as follows:
 >
 >EXTA - Reverberant chamber.
 >SKIN - Cylinder.
 >INTA - Internal acoustic space.
 >
 >Three connections are to be modeled as follows:
 >
 >EXTA,SKIN      - Reverberant field to cylinder.
 >INTA,SKIN      - Reverberant field to cylinder.
 >EXTA,SKIN,INTA - Reverberant field to cylinder to reverberant field.
 >
 >The excitation element is EXTA.
 >
 >The valid frequency range is 25. to 10000. Hz.
 >
 >WXYZ
Saving text...
*
*  Define elements.
SEMOD > ELNAME
*
*  There are three elements  EXTA,SKIN,INTA
Input element name > EXTA,1
EXTA > RHO=.112E-6,CO=.132E+5,V=.34E+9,AP=.314E+7,AAC=.01
EXTA > DESCRIPTION='External acoustic space (excitation).'
EXTA > DONE
OK
Input element name > SKIN,CYLN
SKIN > RHO=.471E-4,CL=.2E+6,H=.4,AP=.186E+6,BL=.494E+3,ALX=.16E+2,ALY=.188E+3
SKIN > D=.12E+3,DLF=.04,E=.182E+7,PATA=.29E+5,RHOS=.188E-4,ASMS=0,CO=.132E+5
SKIN > DESCRIPTION='Cylinder'
SKIN > DONE
OK
Input element name > INTA,1
INTA > RHO=.112E-6,CO=.132E+5,V=.559E+7,AP=.2090E+6,AAC=.01
INTA > DESCRIPTION='Internal acoustic space.'
INTA > DONE
OK
Input element name > LIST


EXTA
DESCRIPT=External acoustic space (excitation).
TYPE    =         1
RHO     = 1.120E-07 CO      = 1.320E+04 VOLUME  = 3.400E+08 AP      = 3.140E+06
AAC     = 1.000E-02

SKIN
DESCRIPT=Cylinder
TYPE    =         4
RHO     = 4.710E-05 CL      = 2.000E+05 H       = 4.000E-01 AP      = 1.860E+05
BL      = 4.940E+02 ALX     = 1.600E+01 ALY     = 1.880E+02 D       = 1.200E+02
DLF     = 4.000E-02 E       = 1.820E+06 PATA    = 2.900E+04 RHOS    = 1.880E-05
ASMS    = 0.000E+00 CO      = 1.320E+04

INTA
DESCRIPT=Internal acoustic space.
TYPE    =         1
RHO     = 1.120E-07 CO      = 1.320E+04 VOLUME  = 5.590E+06 AP      = 2.090E+05
AAC     = 1.000E-02
Input element name > DONE
OK
*
*  Define connections.
SEMOD > PATHNAME
*
*  There are three connections.
*  EXTA,SKIN
*  SKIN,INTA
*  EXTA,SKIN,INTA - non-resonant.
Input connection > EXTA,SKIN,2
Creating new path.
*  No parameters are required for this connection.
EXTA,SKIN > DONE
OK
```

```
Input connection > SKIN,INTA,2
Creating new path.
*  No parameters are required for this connection.
INTA,SKIN > DONE
OK
Input connection > EXTA,SKIN,INTA,5
Creating new path.
*  No parameters are required for this connection.
EXTA,SKIN,INTA > DONE
OK
Input connection > LIST

EXTA     SKIN
TYPE    =         2

INTA     SKIN
TYPE    =         2

EXTA     SKIN     INTA
TYPE    =         5
Input connection > DONE
OK
*
*  Define the excitation element.
SEMOD > SETEXC EXTA
*
*  Define frequency range.
SEMOD > FREQUENCY=25.,10000.
  FREQ  1:   FREQ   SAMP    0    0, SIZE =    27     1, NJ =    1
*
*  Create excitation vector.
SEMOD > EXCITATION
EXTA    (27) > 103.3,105.8,122.5,132.3,130.5,132.1,131.2,135.4,136.0,135.4,133.4,134.4
EXTA    (15) > 135.3,135.2,137.1,135.9,134.7,135.1,132.4,130.4,129.3,128.1,126.8,124.9
EXTA    ( 3) > 122.4,121.0,122.4
*
*  Calculate modal density and establish frequency range.
SEMOD > MDENS
  DENS  1:   DENS   SAMP    0    0, SIZE =    27     3, NJ =    1
*
*  Calculate ata's.
SEMOD > ATACALC
  ATA   1:   ATA    SAMP    0    0, SIZE =    27     9, NJ =    1
*
*  Calculate coefficient matrix and transfer functions.
SEMOD > ATACO
  CO    1:   CO     SAMP    0    0, SIZE =     9    27, NJ =    1
  TRNF  1:   TRNF   SAMP    0    0, SIZE =    27     2, NJ =    1
*
*  Define conversion factors.
```

```
SEMOD > CFAC 7,1,4
  CONV  1:   CONV   SAMP     0     0, SIZE =     27      3, NJ =     1
*
*  Do theoretical prediction.
SEMOD > TPRD
  RESP  1:   RESP   SAMP     0     0, SIZE =     27      3, NJ =     1
*
*  Print the responses.
SEMOD > LIST RESP
```

```
Excitations and responses for model SAMP

Frequency  EXTA         SKIN         INTA
Hertz      dB           G**2/Hz      dB

     25.0  103.3        3.5678E-03    99.2
     31.5  105.8        4.9831E-03   101.5
     40.0  122.5        1.9384E-01   117.9
     50.0  132.3        1.6785E+00   127.5
     63.0  130.5        8.5578E-01   125.5
     80.0  132.1        1.0307E+00   127.0
    100.0  131.2        7.6262E-01   125.9
    125.0  135.4        1.6679E+00   130.0
    160.0  136.0        1.4390E+00   130.4
    200.0  135.4        1.2311E+00   129.8
    250.0  133.4        6.7551E-01   127.7
    315.0  134.4        7.6424E-01   128.6
    400.0  135.3        6.7900E-01   128.8
    500.0  135.2        9.3572E-01   128.8
    630.0  137.1        1.0232E+00   131.2
    800.0  135.9        6.5679E-01   130.2
   1000.0  134.7        6.5027E-01   130.5
   1250.0  135.1        8.7401E-01   133.0
   1600.0  132.4        1.3734E-01   126.2
   2000.0  130.4        5.5018E-02   122.4
   2500.0  129.3        2.4882E-02   119.5
   3150.0  128.1        1.0652E-02   116.5
   4000.0  126.8        4.7543E-03   113.3
   5000.0  124.9        2.0807E-03   109.6
   6300.0  122.4        6.7243E-04   105.1
   8000.0  121.0        3.0041E-04   101.7
  10000.0  122.4        2.8518E-04   101.2
*
*  Calculate power flow.
SEMOD > POWER
  POWR  1:   POWR   SAMP    0    0, SIZE =    27     9, NJ =    1
  PCRF  1:   PCRF   SAMP    0    0, SIZE =    27    10, NJ =    1
*
*  Print out power flow and percent power flow.
SEMOD > LIST POWER PCRF
```

Power flow and loss for model SAMP

Frequency	EXTA	SKIN	INTA	EXTA	SKIN
Hertz	EXTA	SKIN	INTA	SKIN	EXTA
25.0	9.5496E-01	2.6589E+00	2.4765E-02	4.3612E+00	1.7008E+00
31.5	1.6982E+00	3.9473E+00	4.1594E-02	6.1959E+00	2.2642E+00
40.0	7.9431E+01	1.5316E+02	1.8425E+00	2.3023E+02	7.8650E+01
50.0	7.5856E+02	1.2286E+03	1.6802E+01	1.7776E+03	5.7025E+02
63.0	5.0117E+02	6.7790E+02	1.0629E+01	9.4530E+02	2.8429E+02
80.0	7.2442E+02	8.1440E+02	1.4748E+01	1.0968E+03	3.0894E+02
100.0	5.8883E+02	5.5819E+02	1.1579E+01	7.3030E+02	1.9393E+02
125.0	1.5488E+03	1.2430E+03	2.9519E+01	1.5861E+03	3.9867E+02
160.0	1.7782E+03	1.1969E+03	3.2908E+01	1.4941E+03	3.5632E+02
200.0	1.5488E+03	9.0109E+02	2.8097E+01	1.1103E+03	2.5591E+02
250.0	9.7721E+02	5.0342E+02	1.7582E+01	6.1849E+02	1.4112E+02
315.0	1.2302E+03	6.0538E+02	2.1364E+01	7.3833E+02	1.6279E+02
400.0	1.5135E+03	5.3652E+02	2.2599E+01	5.9175E+02	9.3988E+01
500.0	1.4791E+03	6.8489E+02	2.2406E+01	7.9589E+02	1.4607E+02
630.0	2.2908E+03	8.1053E+02	3.9311E+01	9.2200E+02	1.7459E+02
800.0	1.7377E+03	5.1897E+02	3.0935E+01	6.5338E+02	1.5458E+02
1000.0	1.3182E+03	4.7596E+02	3.3129E+01	7.9900E+02	3.1236E+02
1250.0	1.4454E+03	6.5135E+02	5.8884E+01	1.9289E+03	1.2291E+03
1600.0	7.7623E+02	1.1424E+02	1.2387E+01	1.6414E+02	4.4627E+01
2000.0	4.8976E+02	4.0270E+01	5.1257E+00	5.1985E+01	9.9447E+00
2500.0	3.8018E+02	1.8543E+01	2.6798E+00	2.2577E+01	3.2124E+00
3150.0	2.8840E+02	8.4376E+00	1.3426E+00	9.8733E+00	1.0631E+00
4000.0	2.1379E+02	3.7566E+00	6.3942E-01	4.2660E+00	3.5052E-01
5000.0	1.3803E+02	1.5229E+00	2.7008E-01	1.6914E+00	1.0909E-01
6300.0	7.7622E+01	5.3266E-01	9.7165E-02	5.8042E-01	2.9342E-02
8000.0	5.6233E+01	2.3737E-01	4.4210E-02	2.5447E-01	1.0049E-02
10000.0	7.7622E+01	2.0873E-01	3.9507E-02	2.2094E-01	6.9474E-03

```
Power flow and loss for model SAMP

Frequency     INTA        SKIN        EXTA        INTA
Hertz         SKIN        INTA        SKIN        SKIN
                                      INTA        EXTA

    25.0   1.6992E+00  1.7008E+00  3.8011E-02  1.4810E-02
    31.5   2.2800E+00  2.2642E+00  9.0709E-02  3.3380E-02
    40.0   8.0235E+01  7.8650E+01  5.2609E+00  1.8335E+00
    50.0   5.9153E+02  5.7025E+02  5.7061E+01  1.8989E+01
    63.0   3.0119E+02  2.8429E+02  4.0397E+01  1.2871E+01
    80.0   3.3548E+02  3.0894E+02  5.9477E+01  1.8192E+01
   100.0   2.1576E+02  1.9393E+02  4.7409E+01  1.4006E+01
   125.0   4.5419E+02  3.9867E+02  1.1916E+02  3.4123E+01
   160.0   4.1540E+02  3.5632E+02  1.2742E+02  3.5428E+01
   200.0   3.0262E+02  2.5591E+02  1.0284E+02  2.8030E+01
   250.0   1.6719E+02  1.4112E+02  5.9813E+01  1.6168E+01
   315.0   1.9264E+02  1.6279E+02  6.9284E+01  1.8077E+01
   400.0   1.3275E+02  9.3988E+01  7.9104E+01  1.7745E+01
   500.0   1.8114E+02  1.4607E+02  7.4408E+01  1.6935E+01
   630.0   2.3771E+02  1.7459E+02  1.3800E+02  3.5579E+01
   800.0   1.7475E+02  1.5458E+02  6.9764E+01  1.8659E+01
  1000.0   3.0168E+02  3.1236E+02  3.6062E+01  1.3616E+01
  1250.0   1.1806E+03  1.2291E+03  2.6843E+01  1.6429E+01
  1600.0   3.9353E+01  4.4627E+01  9.3552E+00  2.2429E+00
  2000.0   8.1739E+00  9.9447E+00  3.9808E+00  6.2592E-01
  2500.0   2.3910E+00  3.2124E+00  2.0785E+00  2.2012E-01
  3150.0   6.9058E-01  1.0631E+00  1.0430E+00  7.2954E-02
  4000.0   1.9169E-01  3.5052E-01  5.0321E-01  2.2612E-02
  5000.0   4.9719E-02  1.0909E-01  2.1708E-01  6.3813E-03
  6300.0   1.0916E-02  2.9342E-02  8.0248E-02  1.5092E-03
  8000.0   3.0057E-03  1.0049E-02  3.7611E-02  4.4425E-04
 10000.0   1.6894E-03  6.9474E-03  3.4513E-02  2.6391E-04
```

Percent power flow per path for element SKIN in model SAMP

Frequency	SKIN	EXTA	SKIN	INTA	SKIN
Hertz	SKIN	SKIN	EXTA	SKIN	INTA
25.0	-43.87%	71.96%	-28.06%	28.04%	-28.06%
31.5	-46.57%	73.10%	-26.71%	26.90%	-26.71%
40.0	-49.33%	74.16%	-25.33%	25.84%	-25.33%
50.0	-51.86%	75.03%	-24.07%	24.97%	-24.07%
63.0	-54.38%	75.84%	-22.81%	24.16%	-22.81%
80.0	-56.86%	76.58%	-21.57%	23.42%	-21.57%
100.0	-59.00%	77.19%	-20.50%	22.81%	-20.50%
125.0	-60.92%	77.74%	-19.54%	22.26%	-19.54%
160.0	-62.68%	78.25%	-18.66%	21.75%	-18.66%
200.0	-63.78%	78.58%	-18.11%	21.42%	-18.11%
250.0	-64.08%	78.72%	-17.96%	21.28%	-17.96%
315.0	-65.03%	79.31%	-17.49%	20.69%	-17.49%
400.0	-74.05%	81.68%	-12.97%	18.32%	-12.97%
500.0	-70.10%	81.46%	-14.95%	18.54%	-14.95%
630.0	-69.89%	79.50%	-15.05%	20.50%	-15.05%
800.0	-62.67%	78.90%	-18.67%	21.10%	-18.67%
1000.0	-43.24%	72.59%	-28.38%	27.41%	-28.38%
1250.0	-20.95%	62.03%	-39.53%	37.97%	-39.53%
1600.0	-56.14%	80.66%	-21.93%	19.34%	-21.93%
2000.0	-66.94%	86.41%	-16.53%	13.59%	-16.53%
2500.0	-74.27%	90.42%	-12.87%	9.58%	-12.87%
3150.0	-79.87%	93.46%	-10.06%	6.54%	-10.06%
4000.0	-84.27%	95.70%	-7.86%	4.30%	-7.86%
5000.0	-87.47%	97.14%	-6.27%	2.86%	-6.27%
6300.0	-90.08%	98.15%	-4.96%	1.85%	-4.96%
8000.0	-92.19%	98.83%	-3.90%	1.17%	-3.90%
10000.0	-93.76%	99.24%	-3.12%	0.76%	-3.12%

```
Percent power flow per path for element INTA     in model SAMP

Frequency     INTA        INTA        SKIN        EXTA        INTA
Hertz         INTA        SKIN        INTA        SKIN        SKIN
                                                  INTA        EXTA

    25.0        -1.42%     -97.72%      97.81%       2.19%      -0.85%
    31.5        -1.77%     -96.82%      96.15%       3.85%      -1.42%
    40.0        -2.20%     -95.62%      93.73%       6.27%      -2.18%
    50.0        -2.68%     -94.29%      90.90%       9.10%      -3.03%
    63.0        -3.27%     -92.76%      87.56%      12.44%      -3.96%
    80.0        -4.00%     -91.06%      83.86%      16.14%      -4.94%
   100.0        -4.80%     -89.40%      80.36%      19.64%      -5.80%
   125.0        -5.70%     -87.71%      76.99%      23.01%      -6.59%
   160.0        -6.80%     -85.87%      73.66%      26.34%      -7.32%
   200.0        -7.83%     -84.35%      71.33%      28.67%      -7.81%
   250.0        -8.75%     -83.20%      70.23%      29.77%      -8.05%
   315.0        -9.21%     -83.01%      70.15%      29.85%      -7.79%
   400.0       -13.06%     -76.69%      54.30%      45.70%     -10.25%
   500.0       -10.16%     -82.16%      66.25%      33.75%      -7.68%
   630.0       -12.58%     -76.04%      55.85%      44.15%     -11.38%
   800.0       -13.79%     -77.89%      68.90%      31.10%      -8.32%
  1000.0        -9.51%     -86.58%      89.65%      10.35%      -3.91%
  1250.0        -4.69%     -94.00%      97.86%       2.14%      -1.31%
  1600.0       -22.95%     -72.90%      82.67%      17.33%      -4.15%
  2000.0       -36.81%     -58.70%      71.41%      28.59%      -4.49%
  2500.0       -50.65%     -45.19%      60.72%      39.28%      -4.16%
  3150.0       -63.75%     -32.79%      50.48%      49.52%      -3.46%
  4000.0       -74.90%     -22.45%      41.06%      58.94%      -2.65%
  5000.0       -82.80%     -15.24%      33.45%      66.55%      -1.96%
  6300.0       -88.66%      -9.96%      26.77%      73.23%      -1.38%
  8000.0       -92.76%      -6.31%      21.08%      78.92%      -0.93%
 10000.0       -95.29%      -4.07%      16.76%      83.24%      -0.64%
SEMOD > DONE
 ?END
CPU: DELTA, EXECUTION, RUN =      11.25       11.25      403.90
***CIAU***
```

10.3. ATACALC

ATACALC command

Purpose

This command computes the coupling loss and damping loss factors for the entire model.

Command

ATACALC (IOP)

Where

IOP - Option to control status output.
= 0 - No status output. (Default)
= 1 - The path number of the calculation currently being made is displayed.
= 2 - The cpu time is displayed as well as the output for IOP=1.

Input

The following are the input matrices:

N1,'ELPH',NAME - Number of elements and paths. (ELNAME and PATHNAME)
N1,'PIPM',NAME - Path independent variable control list. (ELNAME, PATHNAME, and FPAR)
N1,'PDPM',NAME - Path dependant variable control list. (ELNAME, PATHNAME, and FPAR)
N1,'MNAM',NAME - Element name list. (ELNAME)
N1,'DENS',NAME - Modal densities. (MDENS)
N1,'FREQ',NAME - Frequency vector. (MDENS)
N1,'PIPS',NAME - String control list. (ELNAME) Optional.
N1,'PISL',NAME - String list. (ELNAME) Optional.
N1,'PIRL',NAME - Real list. (ELNAME,PATHNAME, and FPAR)
N1,'PIIL',NAME - Integer list. (ELNAME,PATHNAME, and FPAR)
N1,'FPAR',NAME - Frequency dependant control list. (FPAR) Optional.
N1,'FRAN',NAME - Frequency range list. (FPAR) Optional.

Output

The following matrices are output:

N1,'PDES',NAME - Path descriptor vector. (#ATA's,1)
N1,'ATA ',NAME - Ata matrix. (#Frequencies,#ATA's)

The following two matrices are only output if plates, cylinders, or cones are in the model:

N1,'CDES',NAME - Critical frequency descriptor vector.
(#Critical frequencies,1)
N1,'CRFR',NAME - Critical frequency matrix.
(#Frequencies,#Critical frequencies)

The following two matrices are only output if cylinders or cones are in the model.

N1,'RDES',NAME - Ring frequency descriptor vector.
(#Ring frequencies,1)
N1,'RING',NAME - Ring frequency matrix.
(#Frequencies,#Ring frequencies)

Comments

This command can take a while to run depending on the number of paths. The execution time is proportional to the number of paths. For large models (greater than 50 paths) execution in a batch mode is recommended.

The amount of time to calculate each connection type varies greatly so it is difficult to predict exactly how long a particular model will take to execute. In general the execution time is proportional to the number of connections. If you are curious as to how long each connection type takes to execute use IOP=2.

The PDES matrix identifies the coupling loss or damping loss factor in question with an integer string. The integers in this string point to row locations in the MNAM vector. The following example will illustrate this:

The following MNAM vector exists:

ROW	
1	EXTA
2	SKIN
3	INTA

The code 001002 refers to the path EXTA,SKIN.
The code 001002003 refers to the path EXTA,SKIN,INTA.

Using this scheme a maximum of 999 elements can be used in the model. There is a practical limit based on the capacity of the computer which is, in most cases, much lower than this.

The commands RUN=CLFGET and RUN=CLFPUT can be used to extract or replace coupling or damping loss factors in the ATA matrix.

10.4. ATACO

ATACO command

Purpose

This command will calculate the coefficient matrix, inverse reduced coefficient matrix, and the transfer functions required to make a theoretical prediction on any size model.

Command

```
ATACO
```

Input

The following matrices are required as input:

```
N1,'ATA' ,NAME = Ata matrix.  #Freq's X #Ata's.
N1,'PDES',NAME = Descriptor matrix #Ata's X 1. These two matrices are
                 output by the ATACALC command.
N1,'ELPH',NAME = Number of elements and connections. (2,1)
N1,'EXEL',NAME = List of excitation elements.
```

Output

The following matrices are output:

```
N1,'IDES',NAME = Descriptor matrix for use in the theoretical
                 prediction.
N1,'CO'  ,NAME = Full coefficient matrix.  Includes excitation terms.
N1,'INCO',NAME = Inverse reduced coefficient matrix.
N1,'TRNF',NAME = Transfer function matrix.
```

Comments

The cpu time on this calculation depends on the time required to invert the reduced coefficient matrix. Since the dimensions of the reduced coefficient matrix are the same as the number of response elements in the model the cpu time for each inversion is dependant on the number of excitation elements in the model. There is one inversion for each frequency so the total cpu time is the time required for one inversion times the number of frequencies.

This command has the potential for taking a large amount of cpu time when very large models are used, since the time for each inversion is not linearly increasing with the size of the model. For moderately large models it has been found that the time required for this command is small compared for the cpu time required for the ATACALC command.

10.5. CFAC

CFAC command

Purpose

This command is used to calculate conversion factors for the model.

Command

```
CFAC  ACC,SPL,PARA
```

Where

```
ACC  - The flag for input and output units of acceleration. (See
       ACC_FLAGS.)
SPL  - The flag for input and output units of sound pressure. (See
       SPL_FLAGS.)
PARA - The flag for units of parameters.  (See PARA_FLAGS.)
```

Input

The following are the input matrices for CFAC:

```
N1,'PIPM',NAME - Path independent variable control list.  (ELNAME,
                 PATHNAME, and FPAR)
N1,'MNAM',NAME - Element name list.  (ELNAME)
N1,'FREQ',NAME - Frequency vector.  (MDENS)
N1,'PIPS',NAME - String control list.  (ELNAME)  Optional.
N1,'PISL',NAME - String list.  (ELNAME)  Optional.
N1,'PIRL',NAME - Real list.  (ELNAME,PATHNAME, and FPAR)
N1,'PIIL',NAME - Integer list. (ELNAME,PATHNAME, and FPAR)
N1,'FPAR',NAME - Frequency dependant control list. (FPAR)  Optional.
N1,'FRAN',NAME - Frequency range list.  (FPAR)  Optional.
```

Output

There are three matrices output by CFAC.

```
N1,'CONV',NAME - This matrix contains the conversion factor for mean
                 square response in MKS units to energy in MKS units.
N1,'CPOW',NAME - This is a 1 X 1 matrix containing the conversion
                 factor from energy in the user's units to energy in
                 MKS units.
N1,'DBFG',NAME - This is a vector containing a flag for the unit type
                 for each element in the model.
```

10.5.1. ACC_FLAGS

Valid flags for ACCeleration units.

Flag	Units
1	dB
2	G**2
3	(M/S**2)**2
4	(CM/S**2)**2
5	(FT/S**2)**2
6	(IN/S**2)**2
7	G**2/HZ
8	(M/S**2)**2/HZ
9	(CM/S**2)**2/HZ
10	(FT/S**2)**2/HZ
11	(IN/S**2)**2/HZ
12	G
13	M/S**2
14	CM/S**2
15	FT/S**2
16	IN/S**2

10.5.2. SPL_FLAGS

Valid flags Sound Pressure Level units.

Flag	Units
1	dB
2	(N/M**2)**2
3	(DYNES/CM**2)**2
4	(LBF/FT**2)**2
5	(LBF/IN**2)**2
6	(N/M**2)**2/HZ
7	(DYNES/CM)**2)**2/HZ
8	(LBF/FT**2)**2/HZ
9	(LBF/IN**2)**2/HZ
10	N/M**2
11	DYNES/CM**2
12	LBF/FT**2
13	LBF/IN**2

10.5.3. PARA_FLAGS

Valid flags for PARAmeter units.

Flag	Units
1	Meter, Kilogram, second. (MKS)
2	Centimeter, Gram, second. (CGS)
3	FT, Slug (LBF-SEC**2/FT), second.
4	IN, Snail (LBF-SEC**2/IN), second.

10.6. CHECK

CHECK command (Not yet operational)

Purpose

This command is used to check your model for missing parameters and any other discrepancies such as elements which are not connected to any other elements.

Command

```
CHECK
```

There are no arguments for this command.

10.7. CRFR

CRFR Command

Purpose

Extract critical and ring frequencies for an EXTRAP I model.

Command

```
CRFR
```

This command has no arguments.

Notes

This command looks at the output of ATACALC and finds the appropriate critical and ring frequencies for the current EXTRAP I model and then creates a 2 x 1 matrix for input into the PREDICT/EXTRAP/II command.

10.8. DELETE

DELETE command

Purpose

This command deletes an element and all connections to that element from the model or deletes a single connection from the model.

Command

```
DELETE NAME
or
DELETE NAME1,NAME2,(NAME3,...)
```

Where

CASE I - Deleting an element

NAME - The name of an element. This form deletes the named element and all connections to that element from the model.

CASE II - Deleting a path

NAME1,NAME2,NAME3,...
These are the names of elements in a path. (See PATHNAME) The connection defined by these elements is then deleted from the model

BUGS

Does not update excitation elements(s). This will require the use of the SETEXCN command to update the excitation elements.

Does not get rid of unused string variable space. This doesn't affect the prediction.

10.9. DONE

DONE command

Purpose

This command checks your model, updates the model history matrix and then exits the SEMOD processor.

Command

```
DONE
or
carriage return
```

10.10. ELNAME

ELNAME sub-processor

Purpose

This sub-processor is used to input SEA element names and their parameters.

Command

```
ELNAME (MXELTS,(MXVAR,(MXSTR)))
```

Where

MXELTS - The maximum number of elements allowed for this model. Default = 50 if this is a new model or twice the number of elements if this is an existing model. (optional)

MXVAR - The maximum number of parameters allowed for this model. Default = MXELTS*20 if this is a new model or twice the number of parameters if this is an existing model. (optional)

MXSTR - The maximum amount of string storage space allowed for this model. Default = MXELTS*20 words if this is a new model or twice the existing string space if this is an existing model. (optional)

Note

It is not normally required to input these parameters because when a model starts to get close to exceeding the defaults it will increase the default amount the next time the model is modified. That means that if you exceed one of these parameters all you have to do is exit ELNAME (see DONE command) and then get back into ELNAME.

Usage

Upon entry to this sub-processor you will receive the prompt:

```
Input element name >
```

At this point you have the option of inputing an SEA element name and type or issuing one of the following commands:

```
DONE     QUIT     LIST     CHECK     RENAME     DELETE
```

If you are inputing a new element the command is of the form:

```
NAME,TYPE
```

Where

NAME - The name of the element. Up to 8 characters.
TYPE - The type of the element. Either a number or a 4 character word from the following list:

#	Word	Description
1	REVB	- Reverberant acoustic space.
2	NREV	- Non-reverberant acoustic space. (not supported)
3	PLAT	- Flat plate.
4	CYLN	- Cylinder.
5	CONE	- Cone (Treated as an equivalent cylinder.)
6	BEMR	- Beam or truss rectangular cross section.
7	BEMC	- Beam or truss circular cross section.
8	BEMT	- Beam or truss tubular cross section.
9	BEMO	- Beam or truss other cross section.

If you are changing the type of an element then the input is the same as above. The values of parameters common to both the new and old type will be retained.

If you are updating an element just put in the element name. If you put in the type it will be treated as if you were changing the type if the type you put in is not the same as the type that already exists.

10.10.1. PARAMETERS

Reverberant and non-reverberant acoustic space

Type = 1,2 (REVB,NREV)

Type=1	Parameter 2*	Description
RHO	RHO	Mass density of acoustic medium.
CO	CO	Speed of sound in medium.
VOLUME	VOLUME	Volume of acoustic space.
AP	AP	Surface area of acoustic space.
AAC	AAC	Acoustic absorbtion coefficient.
	VISC	Kinematic viscosity of fluid.
	VEL	Velocity of flying body.
DESCRIPT	DESCRIPT	Optional description.

* Non-reverberant type is not supported for predictions.

Notes

DENSITY UNITS ARE IN MASS NOT WEIGHT
Parameters VOLUME, AP, and AAC don't affect this element when it is used as an excitation element. These parameters still need to be input. It is suggested that good values for these parameters be used so that the model is better documented.

Skin elements

Flat plate, cylinder, cone.

Type = 3,4,5 (PLAT,CYLN,CONE)

Type=3	4	5	Parameters Description
RHO	RHO	RHO	Mass density
CL	CL	CL	Longitudinal wave speed.
H	H	H	Equivalent thickness.
AP	AP	AP	Surface area.
	BL	BL	Length of cylinder or cone.
ALX	ALX	ALX	Sub-panel dimension X direction.
ALY	ALY	ALY	Sub-panel dimension Y direction.
	D	D	Diameter of cylinder or base of cone.
DLF	DLF	DLF	Damping loss factor.
E	E	E	Young's modulus.
PATA	PATA	PATA	Total length of discontinuity.
		BETA	Half apex angle of cone.
RHOS	RHOS	RHOS	Surface mass density.
ASMS	ASMS	ASMS	Non-structural mass.
	CO	CO	Speed of sound in medium surrounding cone or cylinder.
PIVOTFRQ	PIVOTFRQ	PIVOTFRQ	Frequency to pivot functionalized damping at.*
DESCRIPT	DESCRIPT	DESCRIPT	Optional description.

* Optional - If this factor is input then the damping loss factor is calculated as described in the following section.

Notes

DENSITY UNITS ARE IN MASS NOT WEIGHT

Non-structural Mass (ASMS)

This factor provides a means for adding a rigid mass to a model. This method does not smear the mass into the model. This mass does not affect the calculation of coupling loss factors or damping loss factors. It is used in the conversion from energy to response of an element. This means that the response of adjacent elements is not effected by changing this parameter.

Total Length of Discontinuity (PATA)

This factor accounts for edge mode effects and does not affect the model above the critical frequency. For a clamped plate the length of the perimeter should be included. For free-free or simply supported panels the length of the perimeter should not be included. For other cases a portion of the perimeter may be included. When a beam bisects the panel then the length of each edge of the beam that is on the panel is to be included. For most cases this would be twice the length of the beam.

Damping Loss Factor

If PIVOTFRQ is input then DLF is calculated as follows:

```
DLFi = K/Fi
```

Where

```
Fi   = The center frequency.
K    = DLFo*PIVOTFRQ
DLFo = The DLF parameter that you input.
```

It has been found that a good value of PIVOTFRQ is 250 Hz. For electronic boxes the following formula has been suggested:

```
DLFi = K/sqrt(Fi)
```

The parameters are the same as above. This may be calculated using general computational commands and input using the RUN=CLFPUT command.

Trusses and beams

Type = 6,7,8,9 (BEMR,BEMC,BEMT,BEMO)

			Parameters	
Type=6	7	8	9	Description.
RHO	RHO	RHO	RHO	Mass density.
ASMS	ASMS	ASMS	ASMS	Non-structural mass.
H				Thickness.
B				Width.
BL	BL	BL	BL	Length of truss members.
		DI		Inside diameter.
	DO	DO		Outside diameter.
			RGF	Flexural radius of gyration.
			A	Cross section area.
			PJ	Polar moment of inertia.
E	E	E	E	Young's modulus.
G	G	G	G	Shear modulus.
			T	Torsional stiffness.
DLF	DLF	DLF	DLF	Damping loss factor.
CL	CL	CL	CL	Longitudinal wave speed.
CNT	CNT	CNT	CNT	Number of truss members.
DESCRIPT	DESCRIPT	DESCRIPT	DESCRIPT	Optional description.

Notes

DENSITY UNITS ARE IN MASS NOT WEIGHT

Non-structural Mass (ASMS)

This factor provides a means for adding a rigid mass to a model. This method does not smear the mass into the model. This mass does not affect the calculation of coupling loss factors or damping loss factors. It is used in the conversion from energy to response of an element. This means that the response of adjacent elements is not effected by changing this parameter.

10.10.2. CHECK

CHECK command

Purpose

This command checks the parameters and tells you which ones have not been input yet.

Command

```
CHECK
```

Note

If no parameters are missing then OK is printed out.

10.10.3. DELETE

DELETE command

Purpose

This command deletes an element and all connections to that element from the model or deletes a single connection from the model.

Command

```
DELETE NAME
or
DELETE NAME1,NAME2,(NAME3,...)
```

Where

CASE I - Deleting an element

NAME - The name of an element. This form deletes the named element and all connections to that element from the model.

CASE II - Deleting a path

NAME1,NAME2,NAME3,...

These are the names of elements in a path. (See PATHNAME) The connection defined by these elements is then deleted from the model

Notes

This command saves any changes you have mode to the model before deleting. This means that after a delete doing a quit will not undo the delete or any changes up to the delete. It is equivalent to typing:

```
DONE
DELETE ...
ELNAME
```

BUGS

Does not update excitation elements(s). This will require the use of the SETEXCN command to update the excitation elements.

Does not get rid of unused string variable space. This doesn't affect the prediction.

10.10.4. DONE

DONE command

Purpose

This command checks the elements for missing parameters, saves the elements and exits ELNAME.

Command

```
DONE
or
carriage return
```

10.10.5. LIST

LIST command

Purpose

This command is used to list out the parameters for every element in the model or to list out the names of the elements.

Command

```
LIST (NAME)
```

Where

NAME - Is the four character string NAME. This causes only the names of the elements to be listed. (optional)

Note

If NAME is left off then all of the parameters are listed as well as the names of each element.

10.10.6. PARAMETER INPUT

The Parameter Input Processor

Purpose

This sub-processor is initiated when you type in an element name. It allows for the input of parameters for that element.

```
Commands
-------------------------------------------------------------------
CHECK - Checks the parameters for the current element.
DONE  - Exits sub-processor with check.
EXIT  - Exits sub-processor without a check.
LIST  - Lists out the parameters for the current element.
P*    - Input list of parameters.
-------------------------------------------------------------------
```

10.10.6.1. CHECK

CHECK command

Purpose

This command checks to see if all required parameters have been input for the current element.

Command

```
CHECK
```

Note

The required parameters which have not been input will be displayed.

10.10.6.2. DONE

DONE command

Purpose

This command checks for missing parameters for the current element then exits the parameter input processor.

Command

```
DONE
or
carriage return
```

Note

See CHECK for information on checking.

This command is recommended for exiting this subprocessor when it is executed in an interactive mode. Use EXIT in a batch mode.

10.10.6.3. EXIT

EXIT command

Purpose

This command exits the parameter input processor without checking.

Command

```
EXIT
```

Note

This command is recommended for exiting this subprocessor when it is executed in a batch mode. Use DONE in an interactive mode. Using exit results in faster execution because the checking is then only done when you exit ELNAME.

10.10.6.4. LIST

LIST command

Purpose

This command provides a list of the parameters and their values for the current element.

Command

```
LIST
```

Note

Parameters not input will be flagged.

10.10.6.5. P*

P* command

Purpose

This command is used to input parameters.

Command

```
P* VAR1=VAL1,VAR2=VAL2,VAR3=VAL3....
or
VAR1=VAL1,VAR2=VAL2,VAR3=VAL3....
```

Where

VAR1=VAL1,VAR2=VAL2,....
This is the list of parameters and their associated values. If a string is to be input, it must be enclosed in single quotes (').

Example

```
DESCRIPT='This is a string.'
```

If it is desired to use ' in the string use ''.

Example

```
DESCRIPT='I''m using a string.'
Would result in: I'm using a string.
```

10.10.7. QUIT

QUIT command

Purpose

This command exits ELNAME without saving or checking your elements.

Command

```
QUIT
```

Note

This command should only be used if you don't want to save any changes that you have made to the model.

10.10.8. RENAME

RENAME command

Purpose

This command is used to change the name of an element.

Command

```
RENAME OLDNAME NEWNAME
```

Where

OLDNAME - The existing name of an element that you wish to change.
NEWNAME - The new name to give to the element.

Note

The names associated with each path are also changed.

Example

```
RENAME EXTA EXTERNAL
```

Now to refer to the old path EXTA,SKIN you would now type EXTERNAL,SKIN.

10.11. EXCITATION

EXCITATION command

Purpose

This command allows the input of an excitation for the prediction.

Command

```
EXCITATION
```

This command has no arguments.

The user is then prompted to input excitations for each frequency of each excitation element as was specified by the SETEXC and FREQUENCY commands.

10.11.1. EXAMPLE

EXCITATION example

```
SEMOD > SETEXC EXTA
SEMOD > FREQUENCY 50.,100.
SEMOD > EXCITATION
EXTA    ( 4) > 100.,110.,110.,100.
SEMOD >
```

This case shows input for the frequency range 50. to 100. hz for the excitation element EXTA.

10.12. FPAR

FPAR subprocessor

Purpose

This subprocessor is used to input frequency dependant parameters for either elements or connections.

Command

```
FPAR  F1,F2
or
FPAR RNUM
or
FPARL
```

Where

```
CASE I   - FPAR F1,F2

   F1,F2 - Frequency range for parameters that will be input.

CASE II  - FPAR RNUM

   RNUM  - The range number of a previously specified range.

CASE III - FPARL
```

This form lists out the ranges already specified then returns to the SEMOD level.

Note

See RANGE sub-command for more information on setting frequency ranges.

10.12.1. LIST

LIST command

Purpose

This command lists out all the frequency dependant parameters.

Command

```
LIST
```

10.12.2. PARAMETER INPUT

The Parameter Input Processor

Purpose

This sub-processor allows the input of parameters for the particular connection or element specified in FPAR.

Commands

```
-------------------------------------------------------------------
CHECK - Checks the parameters for the current element.
DONE  - Exits sub-processor with check.
EXIT  - Exits sub-processor without a check.
LIST  - Lists out the parameters for the current element.
P*    - Input list of parameters.
-------------------------------------------------------------------
```

10.12.2.1. CHECK

CHECK command

Purpose

This command checks to see if all required parameters have been input for the current element.

Command

```
CHECK
```

Note

The required parameters which have not been input will be displayed.

10.12.2.2. DONE

DONE command

Purpose

This command checks for missing parameters for the current element then exits the parameter input processor.

Command

```
DONE
or
carriage return
```

Note

See CHECK for information on checking.

This command is recommended for exiting this subprocessor when it is executed in an interactive mode. Use EXIT in a batch mode.

10.12.2.3. EXIT

EXIT command

Purpose

This command exits the parameter input processor without checking.

Command

```
EXIT
```

Note

This command is recommended for exiting this subprocessor when it is executed in a batch mode. Use DONE in an interactive mode. Using exit results in faster execution because the checking is then only done when you exit FPAR.

10.12.2.4. LIST

LIST command

Purpose

This command provides a list of the parameters and their values for the connection or element being operated on in FPAR.

Command

```
LIST
```

Note

Parameters not input will be flagged.

10.12.2.5. P*

P* command

Purpose

This command is used to input parameters.

Command

```
P* VAR1=VAL1,VAR2=VAL2,VAR3=VAL3....
or
VAR1=VAL1,VAR2=VAL2,VAR3=VAL3....
```

Where

VAR1=VAL1,VAR2=VAL2,....
This is the list of parameters and their associated values. If a string is to be input, it must be enclosed in single quotes

(').

Note

Only integer and real parameters are allowed to be frequency dependant.

10.12.3. RANGE

RANGE - command

Purpose

This command is used to set or change the frequency range for a given set of parameters input in FPAR.

Command

```
RANGE (F1,F2)
or
RANGE (RNUM)
or
RANGEL
or
RANGEC RNUM F1,F2
```

Where

CASE I - RANGE (F1,F2)

F1,F2 - The frequency range for which subsequently input parameters is valid. If no range is specified the current frequency range is displayed.

CASE II - RANGE (RNUM)

RNUM - The frequency range number of a previously input frequency range. See RANGEL for range numbers.

CASE III - RANGEL

This case displays a list of the frequency ranges already input along with the range number.

CASE IV - RANGEC RNUM F1,F2

RNUM F1,F2 - These are the same as described above. This case will replace range number RNUM with the frequency range F1,F2.

Notes

The frequency range is not actually entered in the RANGE list until parameters have been input for that range. This means that if you do a RANGEList before you input any parameters for that range then that range will not be displayed.

10.13. FREQUENCY

FREQUENCY command

Purpose

This command inputs the frequency range for the model.

Command

```
FREQUENCY  FL,FH
```

Where

FL,FH - Frequency range desired.

10.13.1. EXAMPLE

FREQUENCY example

```
FREQUENCY=10.,10000.
```

This sets the frequency range at 10. to 10000. hz.

10.14. LIST

LIST command (Limited version)

Purpose

This command provides a summary of all the elements, connections, parameters and various other things associated with your model. This summary is intended to tell everything about the model so that it could be recreated from the listing. It can also be used as a tool to hand check your model and could be used as an appendix to a report.

Command

```
LIST  ARG1,ARG2,...
```

Where

ARG1,ARG2,... - List of operations. The following operations can be performed.

ATA - Lists the damping loss and coupling factors.
CRIT - Lists the critical frequencies.
CONV - Lists the conversion factor matrix.
DENS - Lists the modal densities.
POWER - Lists power flow. See POWER command.
PCRF - Lists percentage power flow. See POWER command. This can result in many pages of output.
RESP - Lists the responses.
RING - Lists the ring frequencies.

Output modifiers:
WIDE - Formats output for 132 column display. Default is for 80 column display.
FOR=NU - Lists output to fortran unit NU. Default = 6.

10.15. MDENS

MDENS command

Purpose

This command calculates the modal density and establishes the frequency range for the model.

Command

```
MDENS (F1,F2)
```

Where

F1,F2 - Frequency range for prediction.

Input

The following are the input matrices for MDENS:

N1,'PIPM',NAME - Path independent variable control list. (ELNAME, PATHNAME, and FPAR)
N1,'MNAM',NAME - Element name list. (ELNAME)
N1,'PIPS',NAME - String control list. (ELNAME) Optional.
N1,'PISL',NAME - String list. (ELNAME) Optional.
N1,'PIRL',NAME - Real list. (ELNAME,PATHNAME, and FPAR)
N1,'PIIL',NAME - Integer list. (ELNAME,PATHNAME, and FPAR)
N1,'FPAR',NAME - Frequency dependant control list. (FPAR) Optional.
N1,'FRAN',NAME - Frequency range list. (FPAR) Optional.

Output

The following are the output matrices of MDENS:

N1,'FREQ',NAME - Band center frequencies. Output only if F1, and F2 are input.
N1,'DENS',NAME - Modal densities.

10.16. NON-DIMENSIONAL PARAMETERS

VAPEPS computes values of non-dimensional parameters for each SEA element stored in the VAPEPS database. These non-dimensional parameters are used for searching out existing models. The SERCH processor, however, searches for integer non-dimensional parameter values. Thus, the values of the non-dimensional parameters are actually computed as integer codes which represent the real value of the non-dimensional parameters. This is best explained by way of example.

Example 1

Suppose that the real value of a non-dimensional parameter was 0.0015983. VAPEPS would convert this real number to the integer code -5159. The sign and thousands digit indicate how many places and which direction to move the decimal point. The hundreds, tens and ones digits of the code are the three most significant digits of the real value. So the integer code 05159 represents the real number 159E-5 or .00159.

Example 2

What is the integer code for the real value 15.933.

-1159 (159E-1)

It may seem that this unnecessarily complicates the situation, but it should be noted that this procedure actually simplifies some tasks. Number one, integer searches are more reliable than real searches. And two, using integer codes reduces mass storage requirements. So while this may seem like an inconvenience, keep in mind that it is a necessary one. It should also be noted that the error in retaining only the three most significant digits is always less than one percent, which is insignificant when compared with the uncertainties associated with the parameters themselves.

Each SEA element has a maximum of four non-dimensional parameters which may be calculated. The actual number of non-dimensional parameters which will be calculated for each SEA element will be between one and four.

The following table presents the formulae used to compute the values of the non-dimensional parameters for each SEA element in an EXTRAP I model.

Non-Dimensional Parameters

ELEMENT	TYPE	A	B	C	D
EXTA	Acoustic Space	AP**.5/ (V**(1/3))			
	Fluid Pressure	VEL/CO	FBL*VEL/VISC		
SKIN and INST*	Panel-like	H/(AP**.5)	RHO*AP**.5/RHOS	CO**2*AP**.5/ H*CL**2	
	Cylinder	H/(AP**.5)	RHO*AP**.5/RHOS	CO**2*AP**.5/ H*CL**2	AP**.5/ D
INTA	Acoustic Space	AP**.5/ (V**(1/3))			
MONT and FRAM	(Truss) Rectangular	H/BL	H*(E/(72*RHO))**.5/ (B*CL)		
	Circular	H/BL	(E/(2*RHO))**.5/CL	DO/BL	
	Tubular	H/BL	(E/(2*RHO))**.5/CL	DO/BL	
	Any section	H/BL	(E/(2*RHO))**.5/CL		
MONT	Plate and Cone	H/BL	(E/(2*RHO))**.5/CL		

* INST does not require RHOS to be input so non-dimensional parameter B is not computed.

10.17. PATHNAME

PATHNAME sub-processor

Purpose

This sub-processor is used to input connections and their associated parameters.

Command

```
PATHNAME (MXPATH,(MXVAR))
```

Where

MXPATH - Approximate maximum number of paths allowed for this model. Default = Number of elements * 10 if this is a new model or twice the number of already existing paths for an old model.

MXVAR - Approximate maximum variable storage allowed for connection variables. Default = MXPATH * 7 for a new model or twice the existing storage for an old model.

Note

It is not normally required to input these parameters because when a model starts to get close to exceeding the defaults it will increase the default amount the next time the model is modified. That means that if you exceed one of these parameters all you have to do is exit PATHNAME (see DONE command) and then get back into PATHNAME.

The above values are approximate because each different path types can have different numbers of variables. The variable storage has a header for each path so it is decreased by 1 for every path. The best bet is, if you feel that you are getting close to exceeding the storage, to allocate more than you think you need.

Usage

Upon entry to this sub-processor you will receive the prompt

```
Input connection >
```

At this point you have the option of inputing a connection and type or issuing one of the following commands:

CHECK   DELETE   DONE   LIST   QUIT

If you are inputing a new connection the command is of the form:

NAME1,NAME2,...,NAMEn,TYPE

Where

NAME1,NAME2,...,NAMEn
- The names of the elements associated with this connection.

TYPE - The type of the element. Types are as in the following list:

Type Description

---

1 - Acoustic space to flat plate.
2 - Acoustic space to cylinder.
3 - Acoustic space to cone.
4 - Acoustic space to flat plate to acoustic space.
5 - Acoustic space to cylinder to acoustic space.
6 - Acoustic space to cone to acoustic space.
7 - Flat plate to flat plate. "L" connection.
8 - Flat plate to flat plate. "T" connection of any angle.
9 - Flat plate to cylinder. "T" connection of any angle.
10 - Flat plate to cone. "T" connection of any angle. Cone angle is used for connection angle.
11 - Cylinder to flat plate. "T" connection of any angle.
12 - Cone to flat plate. "T" connection of any angle.
13 - Unused.
14 - Unused.
15 - Flat plate* to truss with equivalent frame. Truss extends from the middle of the plate at some angle.
16 - Flat plate* to truss with user supplied rectangular frame.
17 - Flat plate* to truss with user supplied circular frame. Truss extends from the middle of the plate at some angle.
18 - Flat plate* to truss with user supplied tubular frame. Truss extends from the middle of the plate at some angle.
19 - Flat plate* to truss with user supplied other frame. Truss extends from the middle of the plate at some angle.
20 - Flat plate* to truss with equivalent frame and attenuation factor. Truss extends from the middle of the plate at some angle.
21 - Flat plate* to truss with user supplied rectangular frame and attenuation factor. Truss extends from the middle of the plate at some angle.
22 - Flat plate* to truss with user supplied circular frame and attenuation factor. Truss extends from the middle of the plate at some angle.
23 - Flat plate* to truss with user supplied tubular frame and attenuation factor. Truss extends from the middle of the plate at some angle.
24 - Flat plate* to truss with user supplied other frame and

attenuation factor. Truss extends from the middle of the plate at some angle.

25 - Flat plate* to truss. Truss extends from the edge of the plate at an acute angle.

26 - Flat plate* to truss with attenuation factor. Truss extends from the edge of the plate at an acute angle.

27 - Flat plate* to truss. Truss extends from the edge of the plate at an oblique angle.

28 - Flat plate* to truss with attenuation factor. Truss extends from the edge of the plate at an oblique angle.

29 - Flat plate* to beam. Beam is perpendicular to the plate.

30-39 - Non-resonant beam cases. (Not yet operational)

40 - Flat plate* to parallel beam. Point connection.
41 - Flat plate* to perpendicular flat plate*. Point connection.
42 - Flat plate* to parallel flat plate*. Point connection.
43 - Butt connection between two plates.**
44 - "T" connection between three plates.**
45 - Beam to parallel beam. Point connection.
46 - "+" Connection between four plates.**

---

* You can substitute a cylinder or cone for the flat plate in these connections.

** Same as above except some limitations may apply.

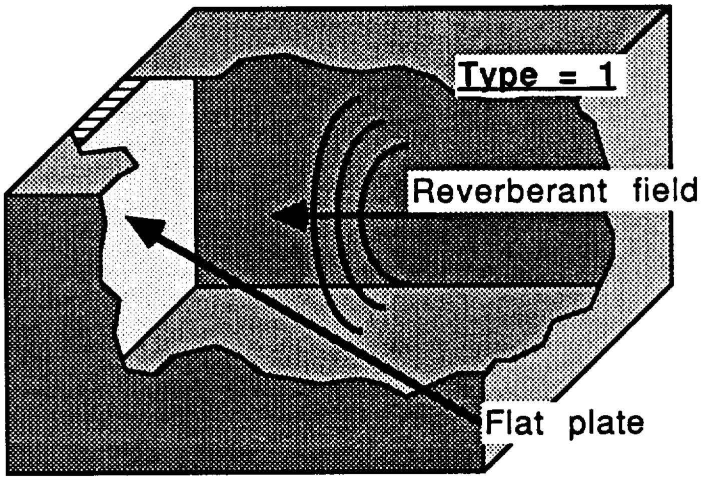
Notes

The following are general notes to the limitations of the SEA software.

1) Units are in terms of MASS not weight, length and time.
2) When the number of modes per band (Modal density x band width) is small (< 6 has been suggested), the uncertainty could become excessive. Therefore the results may be questionable.

If you are changing the type of a connection then the input is the same as above. The values of parameters common to both the new and old type will be retained.

If you are updating a connection just put in the connection name. If you put in the type it will be treated as if you were changing the type, if the type you put in is not the same as the type that already exists.



Parameters

SCALEFAC - Optional factor to scale ATA's by.

Figure 10.1 Type = 1: Acoustic space to flat plate.

Notes

The following notes apply to connection type 1:

1/2 Wavelength Restriction

For a panel excited by an acoustic filed below the frequency,

F < CO/(2*L)

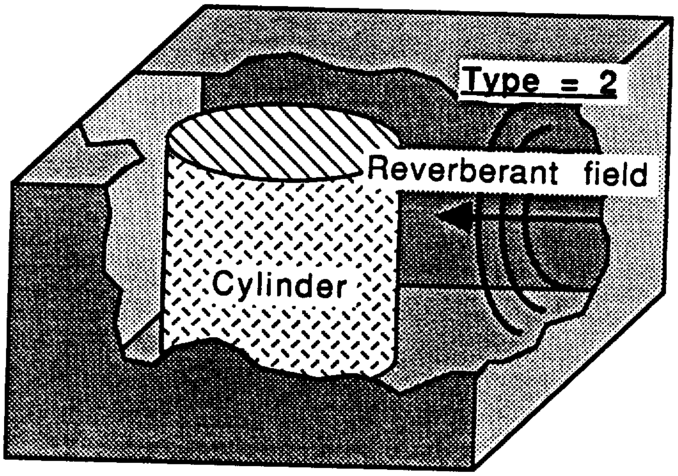
Where

CO = Speed of sound in acoustic field.
L = Typical dimension of a panel.

the calculated response of that panel may be questionable. Also adjoining elements may be affected.

Acoustic Short Circuit Limitation

For a panel excited on both sides by the same acoustic field, a short circuit of the acoustic filed may occur, and the calculated response may be overly conservative.



Parameters

SCALEFAC - Optional factor to scale ATA's by.

Figure 10.2 Type = 2: Acoustic space to cylinder.

Notes

The following notes apply to connection type 2:

Ring Frequency > Critical Frequency

When the ring frequency

```
Fr = CL/(2*PI*R)
```

Where

```
CL = Longitudinal wave speed.
R  = Radius of cylinder.
```

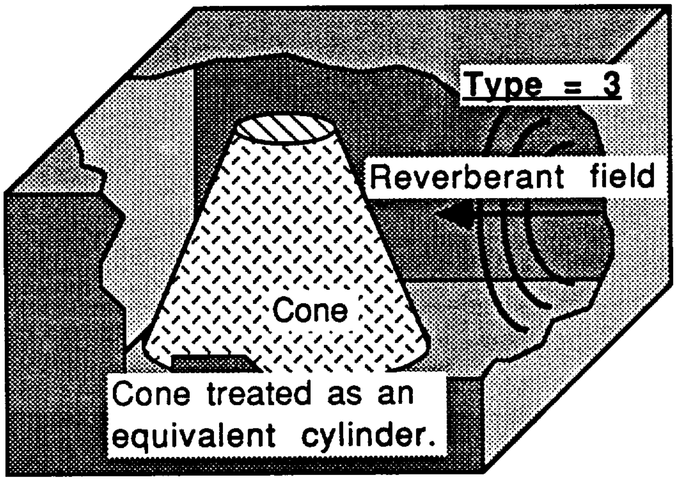
is greater than the critical frequency

```
Fc = CO**2/(K*CL)
```

Where

```
CO = Speed of sound in acoustic medium surrounding cylinder.
K  = Radius of gyration.
CL = Longitudinal wave speed.
```

the coupling loss factor is calculated using a somewhat questionable method and the results should be used cautiously.



Parameters

SCALEFAC - Optional factor to scale ATA's by.

Figure 10.3 Type = 3: Acoustic space to cone.

Notes

The following notes apply to connection type 3:

Cone Treated as Equivalent Cylinder

The cone is treated as a cylinder whose diameter is equal to the average diameter of the cone.

Ring Frequency > Critical Frequency

When the ring frequency

Fr = CL/(2*PI*R)

Where

CL = Longitudinal wave speed.
R = Radius of cylinder.

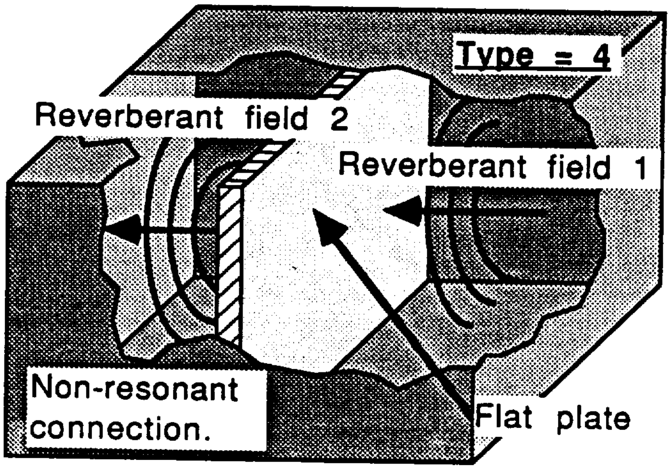
is greater than the critical frequency

```
Fc = CO**2/(K*CL)
```

Where

CO = Speed of sound in acoustic medium surrounding cylinder.
K = Radius of gyration.
CL = Longitudinal wave speed.

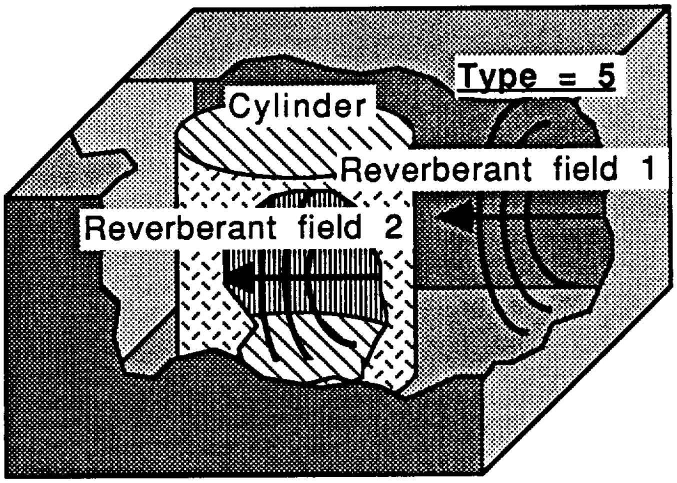
the coupling loss factor is not calculated and is set to zero. This will cause the results to be invalid.



Parameters

SCALEFAC - Optional factor to scale ATA's by.

Figure 10.4 Type = 4: Acoustic space to flat plate to acoustic space.



Parameters

SCALEFAC - Optional factor to scale ATA's by.

Figure 10.5 Type = 5: Acoustic space to cylinder to acoustic space.

Ring Frequency > Critical Frequency

When the ring frequency

Fr = CL/(2*PI*R)

Where

CL = Longitudinal wave speed.
R = Radius of cylinder.

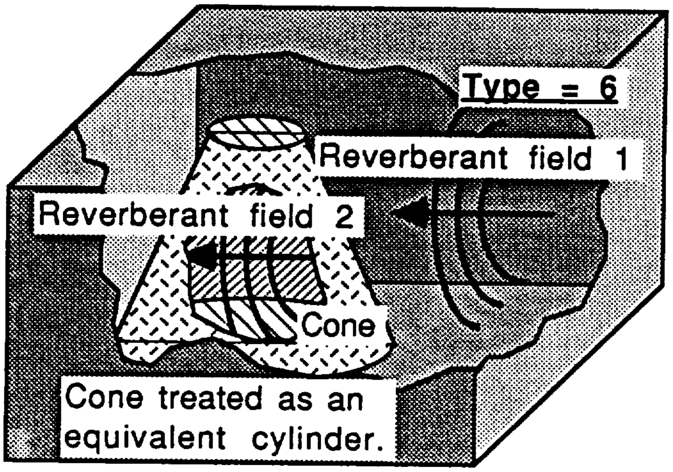
is greater than the critical frequency

Fc = CO**2/(K*CL)

Where

CO = Speed of sound in acoustic medium surrounding cylinder.
K = Radius of gyration.
CL = Longitudinal wave speed.

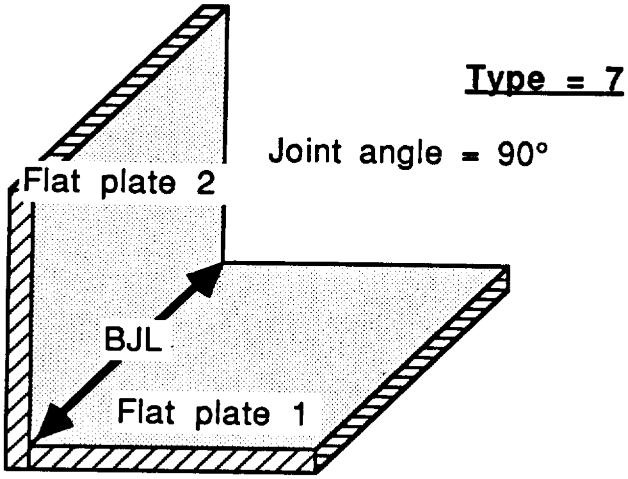
the coupling loss factor is calculated using a somewhat questionable method and the results should be used cautiously.



Parameters

SCALEFAC - Optional factor to scale ATA's by.

Figure 10.6 Type = 6: Acoustic space to cone to acoustic space.



Parameters

BJL - Joint length.

Figure 10.7 Type = 7: Flat plate to flat plate. "L" connection.

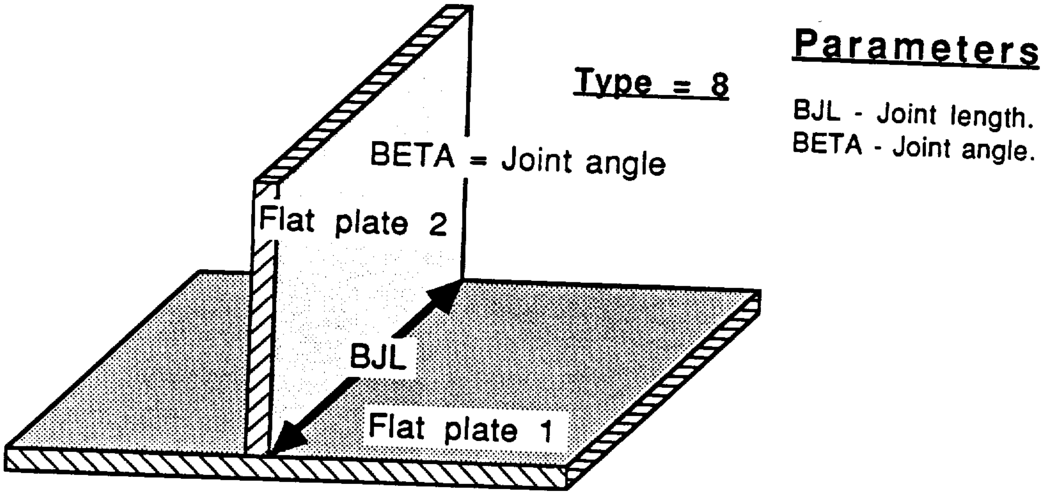
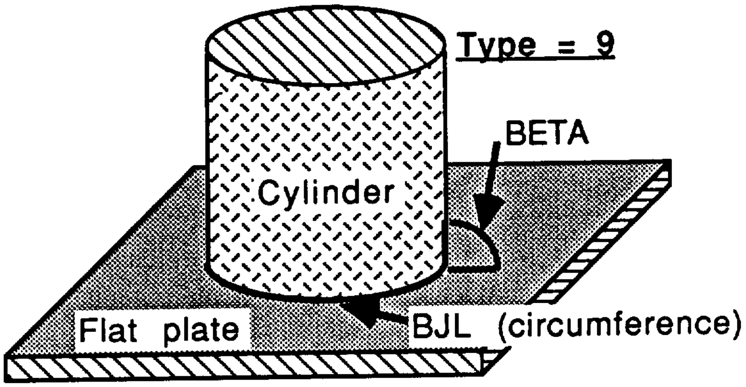


Figure 10.8 Type = 8: Flat plate to flat plate. "T" connection.



Parameters

BJL - Joint length.
BETA - Joint angle.

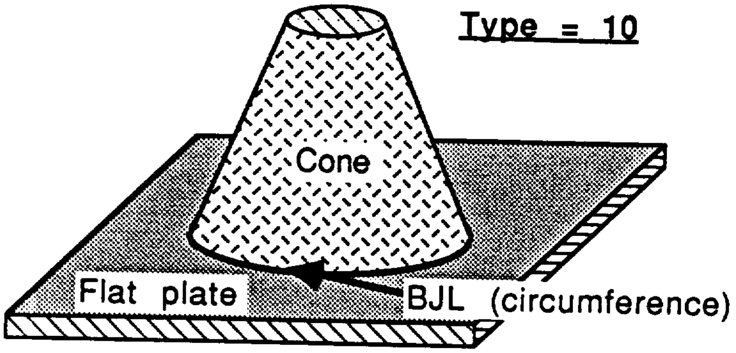
Figure 10.9 Type = 9: Flat plate to cylinder. "T" connection.

Notes

The following notes apply to connection type 9.

Cylinder Treated as Flat Plate

This connection type treats the cylinder as a flat plate for the coupling loss factor calculation.



Parameters

BJL - Joint length.

Figure 10.10 Type = 10: Flat plate to cone. "T" connection.

Notes

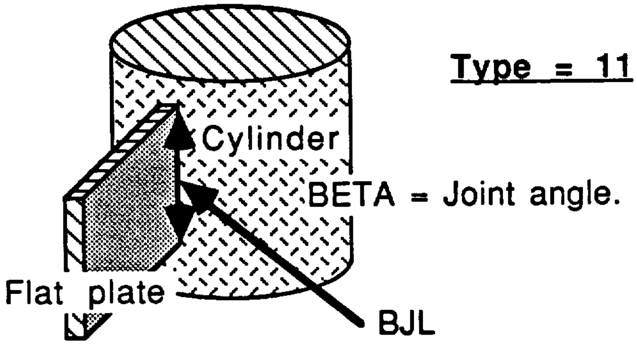
The following notes apply to connection type 10.

Connection angle

The cone angle is used for connection angle.

Cone Treated as Flat Plate

This connection type treats the cone as a flat plate for the coupling loss factor calculation.



Parameters

BJL - Joint length.
BETA - Joint angle.

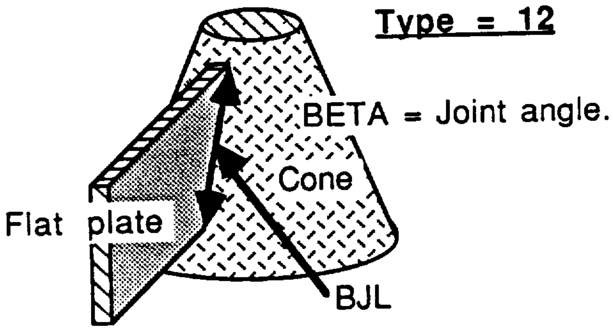
Figure 10.11 Type = 11: Cylinder to flat plate. "T" connection.

Notes

The following notes apply to connection type 11.

Cylinder Treated as Flat Plate

This connection type treats the cylinder as a flat plate for the coupling loss factor calculation.



Parameters

BJL - Joint length.
BETA - Joint angle.

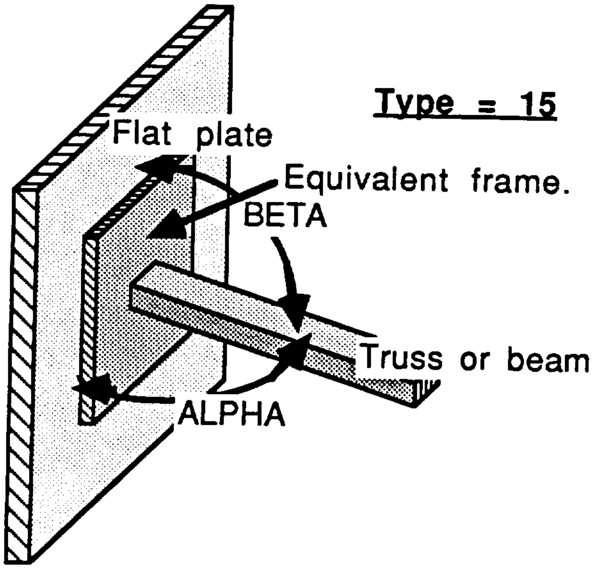
Figure 10.12 Type = 12: Cone to flat plate. "T" connection.

Notes

The following notes apply to connection type 10.

Cone Treated as Flat Plate

This connection type treats the cone as a flat plate for the coupling loss factor calculation.



Parameters

ALPHA - Orientation angle.
BETA - Orientation angle.

Figure 10.13 Type = 15: Flat plate to truss with equivalent frame.

Notes

The following notes apply to connection type 15.

Angle

The truss extends from middle of plate at the specified angle.

Only Moment Coupling Used

Note only moment coupling is assumed. Force coupling is assumed to be negligible. This assumption is not valid for certain types of joints (eg. a pinned joint).

Cylinder or Cone May be used

A cylinder or cone may be used in place of the flat plate but will be treated as a flat plate.

Truss Types Supported

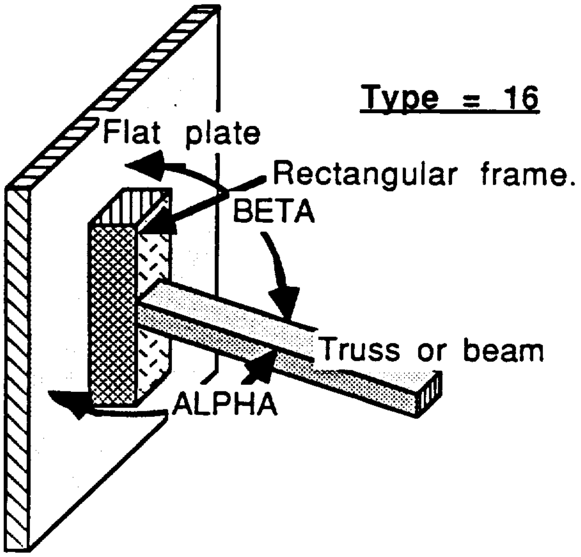
The truss may be of type BEMR, BEMC, BEMT or BEMO.

Non-Resonant Coupling

This type does not account for non-resonant coupling.

Equivalent Frame

The equivalent frame is calculated using the panel properties for a strip the width of the truss.



Parameters

ALPHA - Orientation angle.
BETA - Orientation angle.

Frame Parameters

RHO	- Mass density.
H	- Thickness.
B	- Width.
BL	- Length.
CL	- Wave speed.

Figure 10.14 Type = 16: Flat plate to truss with rectangular frame.

Notes

The following notes apply to connection type 16.

Angle

The truss extends from middle of plate at the specified angle.

Only Moment Coupling Used

Note only moment coupling is assumed. Force coupling is assumed to be negligible. This assumption is not valid for certain types of joints (eg. a pinned joint).

Cylinder or Cone May be used

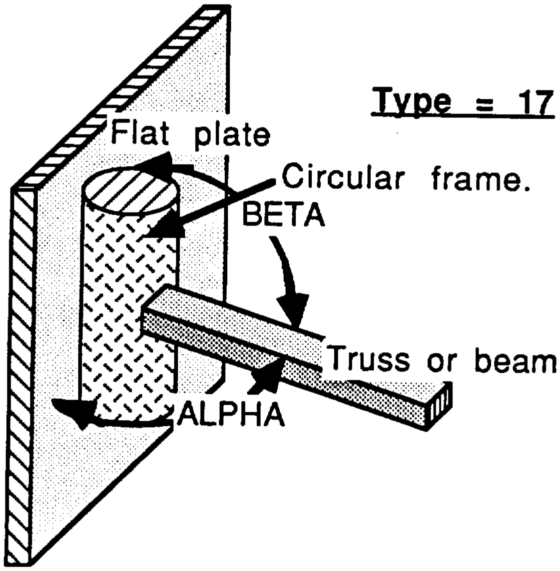
A cylinder or cone may be used in place of the flat plate but will be treated as a flat plate.

Truss Types Supported

The truss may be of type BEMR, BEMC, BEMT or BEMO.

Non-Resonant Coupling

This type does not account for non-resonant coupling.



Parameters

ALPHA - Orientation angle.
BETA - Orientation angle.

Frame Parameters

RHO	- Mass density.
DO	- Outer diameter.
BL	- Length.
CL	- Wave speed.

Figure 10.15 Type = 17: Flat plate to truss with circular frame.

Notes

The following notes apply to connection type 17.

Angle

The truss extends from middle of plate at the specified angle.

Only Moment Coupling Used

Note only moment coupling is assumed. Force coupling is assumed to be negligible. This assumption is not valid for certain types of joints (eg. a pinned joint).

Cylinder or Cone May be used

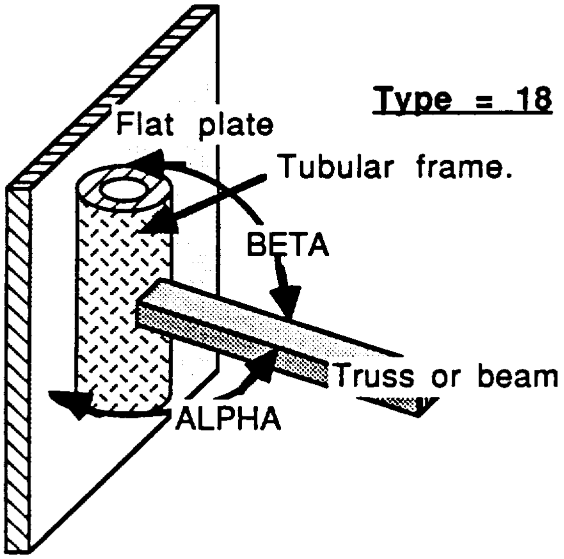
A cylinder or cone may be used in place of the flat plate but will be treated as a flat plate.

Truss Types Supported

The truss may be of type BEMR, BEMC, BEMT or BEMO.

Non-Resonant Coupling

This type does not account for non-resonant coupling.



Parameters

ALPHA - Orientation angle.
BETA - Orientation angle.

Frame Parameters

RHO - Mass density.
DO - Outer diameter.
DI - Inner diameter.
BL - Length.
CL - Wave speed.

Figure 10.16 Type = 18: Flat plate to truss with tubular frame.

Notes

The following notes apply to connection type 18.

Angle

The truss extends from middle of plate at the specified angle.

Only Moment Coupling Used

Note only moment coupling is assumed. Force coupling is assumed to be negligible. This assumption is not valid for certain types of joints (eg. a pinned joint).

Cylinder or Cone May be used

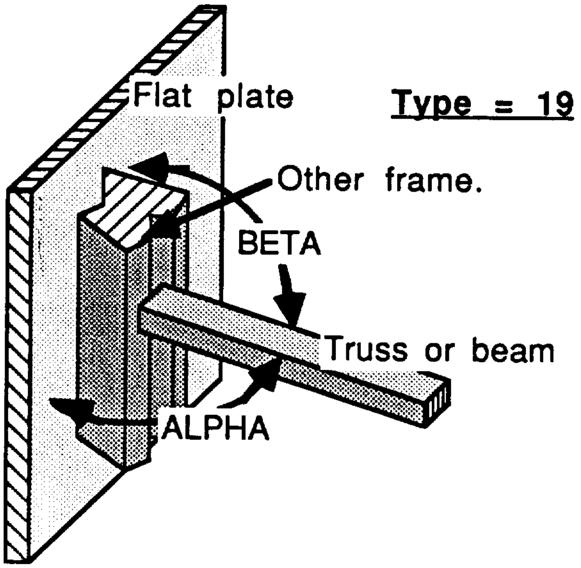
A cylinder or cone may be used in place of the flat plate but will be treated as a flat plate.

Truss Types Supported

The truss may be of type BEMR, BEMC, BEMT or BEMO.

Non-Resonant Coupling

This type does not account for non-resonant coupling.



Parameters

ALPHA - Orientation angle.
BETA - Orientation angle.

Frame Parameters

RHO	- Mass density.
BL	- Length.
RGF	- Flexural radius of gyration.
A	- Cross section area.
CL	- Wave speed.

Figure 10.17 Type = 19: Flat plate to truss with general frame.

Notes

The following notes apply to connection type 19.

Angle

The truss extends from middle of plate at the specified angle.

Only Moment Coupling Used

Note only moment coupling is assumed. Force coupling is assumed to be negligible. This assumption is not valid for certain types of joints (eg. a pinned joint).

Cylinder or Cone May be used

A cylinder or cone may be used in place of the flat plate but will be treated as a flat plate.

Truss Types Supported

The truss may be of type BEMR, BEMC, BEMT or BEMO.

Non-Resonant Coupling

This type does not account for non-resonant coupling.

Attenuated Truss Connections

Truss types 20-24 correspond to truss types 15-19 respectively. They are the same except for the addition of an attenuation factor which is specified with the following two parameters:

RATE	Attenuation rate. dB/octave.
CFRQ	Center frequency at which attenuation begins

Type = 20

Flat plate to truss with equivalent frame and attenuation factor. Corresponds to type 15.

Notes

For notes on connection type 20, refer to connection type 15.

Type = 21

Flat plate to truss with user supplied rectangular frame. and attenuation factor. Corresponds to type 16.

Notes

For notes on connection type 21, refer to connection type 16.

Type = 22

Flat plate to truss with user supplied circular frame. and attenuation factor. Corresponds to type 17.

Notes

For notes on connection type 22, refer to connection type 17.

Type = 23

Flat plate to truss with user supplied tubular frame. and attenuation factor. Corresponds to type 18.

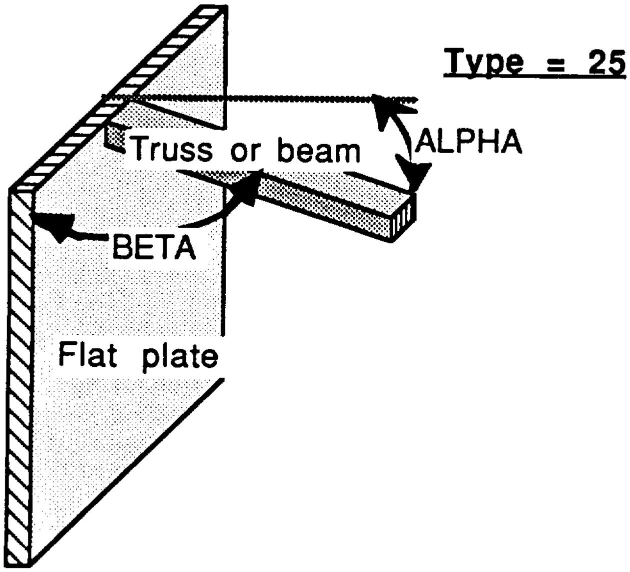
Notes

For notes on connection type 23, refer to connection type 18.

Type = 24 - Flat plate to truss with user supplied other frame. and attenuation factor. Corresponds to type 19.

Notes

For notes on connection type 24, refer to connection type 19.



Parameters

ALPHA - Orientation angle.
BETA - Orientation angle.

Figure 10.18 Type = 25: Flat plate to truss no frame.

Notes

The following notes apply to connection type 25.

Frame

There is no frame in this connection.

Angle

The truss extends from the edge of the plate at an acute angle.

Only Moment Coupling Used

Note only moment coupling is assumed. Force coupling is assumed to be negligible. This assumption is not valid for certain types of joints (eg. a pinned joint).

Cylinder or Cone May be used

A cylinder or cone may be used in place of the flat plate but will be treated as a flat plate.

Truss Types Supported

The truss may be of type BEMR, BEMC, BEMT or BEMO.

Non-Resonant Coupling

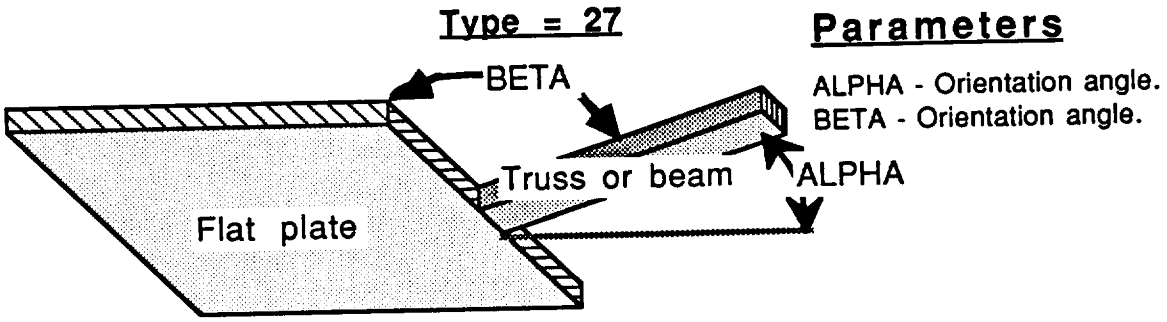
This type does not account for non-resonant coupling.
Attenuated Truss Connection

Type = 26

Flat plate to truss no frame. Extends from edge of plate at an acute angle. With attenuation factor.

Same as type 25 except the following parameters are added:

RATE	Attenuation rate. dB/octave.
CFRQ	Center frequency at which attenuation begins



Parameters

ALPHA - Orientation angle.
BETA - Orientation angle.

Figure 10.19 Type = 27: Flat plate to truss no frame.

Notes

The following notes apply to connection type 27.

Frame

There is no frame in this connection.

Angle

The truss extends from the edge of the plate at an oblique angle.

Only Moment Coupling Used

Note only moment coupling is assumed. Force coupling is assumed to be negligible. This assumption is not valid for certain types of joints (eg. a pinned joint).

Cylinder or Cone May be used

A cylinder or cone may be used in place of the flat plate but will be treated as a flat plate.

Truss Types Supported

The truss may be of type BEMR, BEMC, BEMT or BEMO.

Non-Resonant Coupling

This type does not account for non-resonant coupling.

Attenuated Truss Connection

Type = 28

Flat plate to truss no frame. Extends from edge of plate at an oblique angle. With attenuation factor.

Same as type 27 except the following parameters are added:

RATE	Attenuation rate. dB/octave.
CFRQ	Center frequency at which attenuation begins

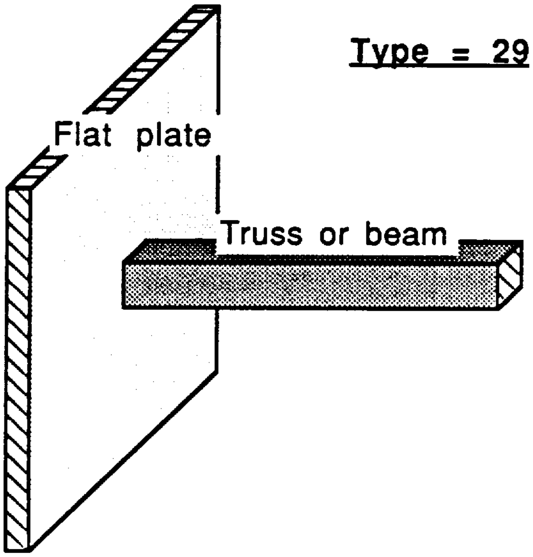


Figure 10.20 Type = 29: Flat plate to beam.

Notes

The following notes apply to connection type 27.

Frame

There is no frame in this connection.

Angle

The beam is perpendicular to the plate.

Only Moment Coupling Used

Note only moment coupling is assumed. Force coupling is assumed to be negligible. This assumption is not valid for certain types of joints (eg. a pinned joint).

Cylinder or Cone May be used

A cylinder or cone may be used in place of the flat plate but will be treated as a flat plate.

Truss Types Supported

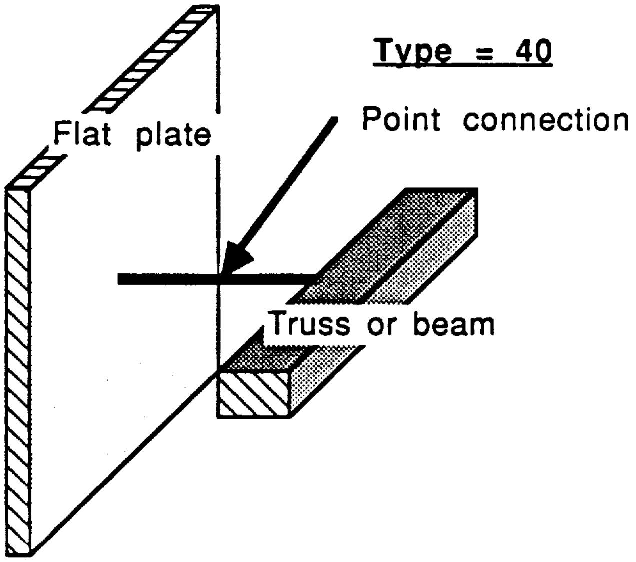
The truss may be of type BEMR, BEMC, BEMT or BEMO.

Non-Resonant Coupling

This type does not account for non-resonant coupling.

Non-Resonant Truss Connections

There are a number of possible non-resonant truss connections. In the EXTRAP I prediction some of these are supported. In the current implementation of SEMOD none of these non-resonant truss connections are supported. The transmission from this type of coupling can be significant so the results from not using these connections can be in error.

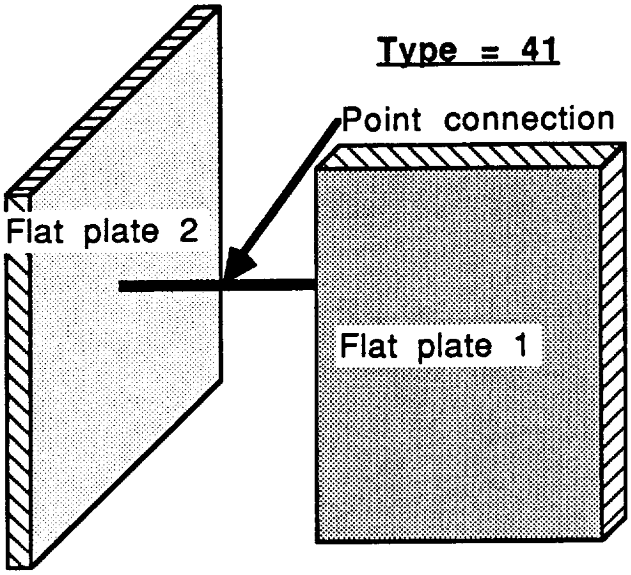


Type = 40 **Parameters**

NB - Number of points connecting beam and plate.

Figure 10.21 Type = 40: Plate to parallel beam. Point connection.

Type = 41

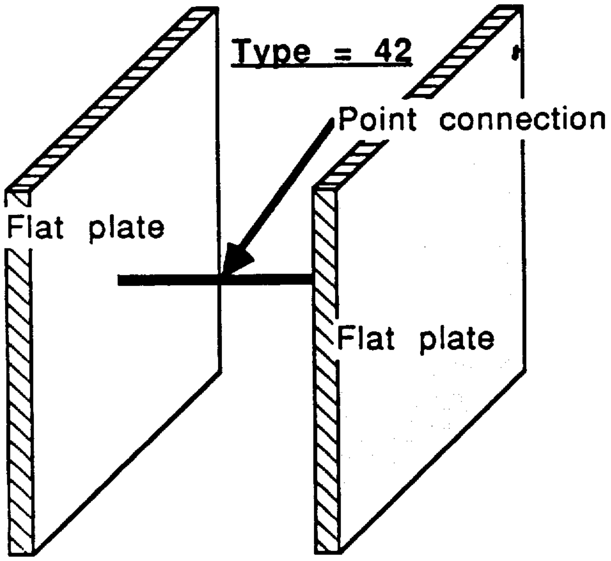


Parameters

NB - Number of points connecting plate 1 to plate 2.

Note the order of specification.

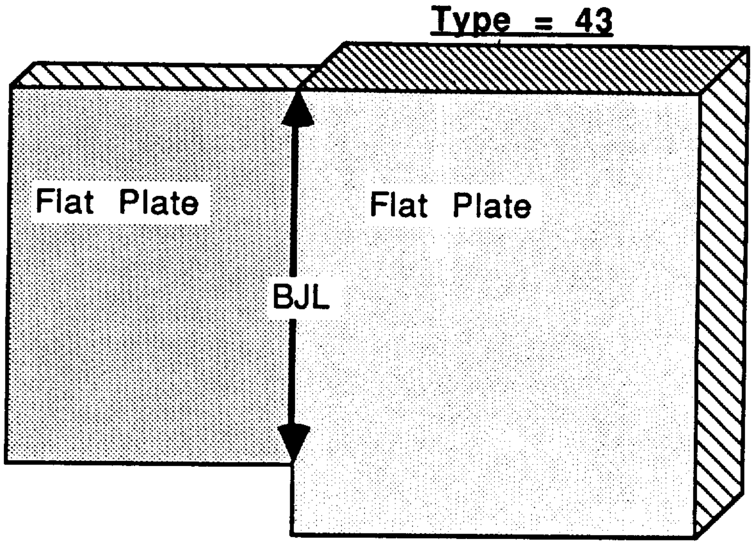
Figure 10.22 Type = 41: Plate to perpendicular plate. Single point.



Parameters

NB - Number of points connecting the two plates.

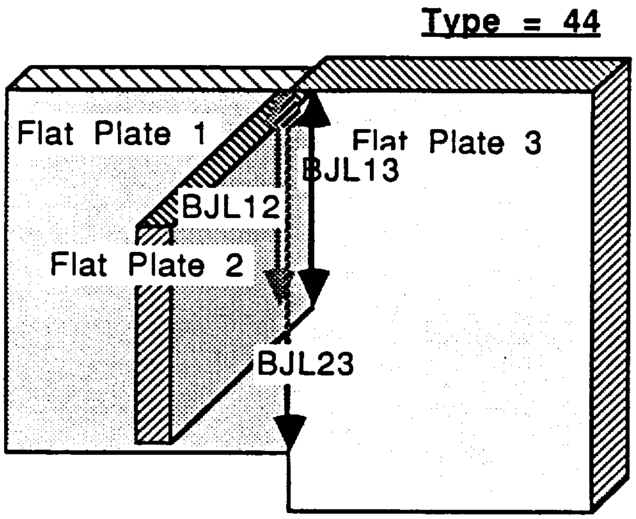
Figure 10.23 Type = 42: Plate to parallel plate. Point connection.



Parameters

BJL - Joint length.

Figure 10.24 Type = 43: Butt connection between two plates.



Parameters

BJL12 - Joint length for plates 1 and 2.

BJL13 - Joint length for plates 1 and 3.

BJL23 - Joint length for plates 2 and 3.

Figure 10.25 Type = 44: "T" connection between three plates.

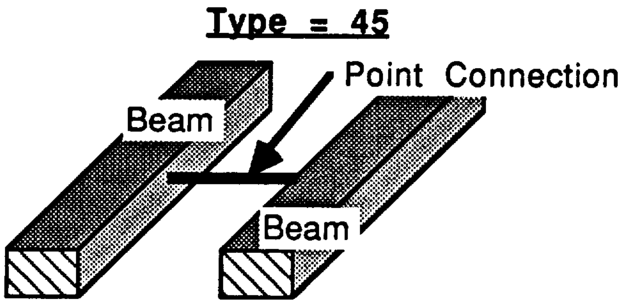
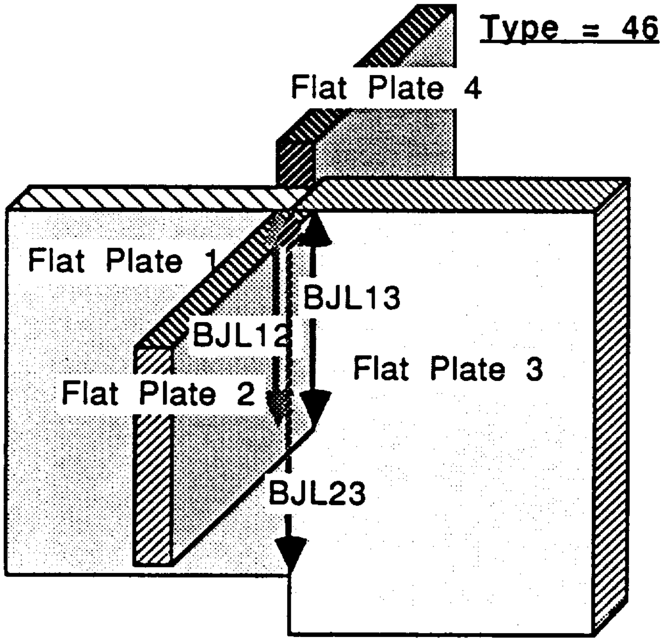


Figure 10.26 Type = 45: Beam to parallel beam. Point connection.

Notes

Not yet implemented.



Parameters

BJL12 - Joint length for plates 1 and 2.
BJL13 - Joint length for plates 1 and 3.
BJL14 - Joint length for plates 1 and 4.
BJL23 - Joint length for plates 2 and 3.
BJL24 - Joint length for plates 2 and 4.
BJL34 - Joint length for plates 3 and 4

Figure 10.27 Type = 46: "+" connection between four plates.

10.17.1. CHECK

CHECK command

Purpose

This command checks the parameters and tells you which ones have not been input yet.

Command

```
CHECK
```

Note

If no parameters are missing then OK is printed out.

10.17.2. DELETE

DELETE command

Purpose

This command deletes an element and all connections to that element from the model or deletes a single connection from the model.

Command

```
DELETE NAME
or
DELETE NAME1,NAME2,(NAME3,...)
```

Where

CASE I - Deleting an element

NAME - The name of an element. This form deletes the named element and all connections to that element from the model.

CASE II - Deleting a path

NAME1,NAME2,NAME3,...
These are the names of elements in a path. (See PATHNAME) The connection defined by these elements is then deleted from the model

Notes

This command saves any changes you have mode to the model before deleting. This means that after a delete doing a quit will not undo the delete or any changes up to the delete. It is equivalent to typing:

```
DONE
DELETE ...
PATHNAME
```

BUGS

Does not update excitation elements(s). This will require the use of the SETEXCN command to update the excitation elements.

Does not get rid of unused string variable space. This doesn't affect the prediction.

10.17.3. DONE

DONE command

Purpose

This command checks the connections for missing parameters, saves the connections and exits the sub-processor.

Command

```
DONE
or
carriage return
```

10.17.4. LIST

LIST command

Purpose

This command is used to list out the parameters for every connection in the model or to list out the names of the connections.

Command

```
LIST (NAME)
```

Where

NAME - Is the four character string NAME. This causes only the names of the connections to be listed. (optional)

Note

If NAME is left off then all of the parameters are listed as well as the names of each connection.

10.17.5. PARAINPUT

Parameter Input processor

Purpose

This sub-processor allows the input of parameters for the particular connection last specified in PATHNAME.

This sub-processor is entered when you specify a path name and/or type in response to the prompt:

```
Input connection >
```

Commands

```
------------------------------------------------------------------
CHECK - Checks the parameters for the current connection.
DONE  - Exits sub-processor with check.
EXIT  - Exits sub-processor without a check.
LIST  - Lists out the parameters for the current connection.
P*    - Input list of parameters.
------------------------------------------------------------------
```

10.17.5.1. CHECK

CHECK command

Purpose

This command checks for parameters which have not been input yet for the connection last specified in PATHNAME.

Command

```
CHECK
```

Note

The parameters which have not been input will be displayed.

10.17.5.2. DONE

DONE command

Purpose

This command checks for missing parameters for the last connection specified in PATHNAME then exits the sub-processor.

Command

```
DONE
or
carriage return
```

Note

See CHECK for information on checking.

This command is recommended for exiting this subprocessor when it is executed in an interactive mode. Use EXIT in a batch mode.

10.17.5.3. EXIT

EXIT command

Purpose

This command exits the sub-processor without checking.

Command

```
EXIT
```

Note

This command is recommended for exiting this subprocessor when it is executed in a batch mode. Use DONE in an interactive mode. Using exit results in faster execution because the checking is then only done when you exit PATHNAME.

10.17.5.4. LIST

LIST command

Purpose

This command provides a list of the parameters and their values for the connection last specified in PATHNAME.

Command

```
LIST
```

Note

Parameters not input will be flagged.

10.17.5.5. P*

P* command

Purpose

This command is used to input parameters.

Command

```
P* VAR1=VAL1,VAR2=VAL2,VAR3=VAL3....
or
VAR1=VAL1,VAR2=VAL2,VAR3=VAL3....
```

Where

VAR1=VAL1,VAR2=VAL2,....
This is the list of parameters and their associated values.

10.17.6. QUIT

QUIT command

Purpose

This command exits the subprocessor without saving or checking your connections.

Command

```
QUIT
```

Note

This command should only be used if you don't want to save any changes that you have made to the model connections.

10.18. POWER

POWER command

Purpose

This command calculates the power flow between each element and also calculates the flow between elements as a percentage of the total power flowing into a specific element.

Command

```
POWER (MAXCON)
```

Where

MAXCON - Is the maximum connectivity in your model. Default = 2* #elements.

Use the LIST POWER and LIST PCRF commands to display output.

10.19. QUIT

QUIT command

Purpose

This command exits SEMOD without updating the model history matrix.

Command

```
QUIT
```

10.20. READ

READ command (Not yet operational)

Purpose

This command is used to include as a part of your model another model which was created at an earlier time. Also a single element of another model can be included.

Command

```
READ N1,NAME1 (ELEMENT)
```

Where

```
N1,NAME1 - The DAL unit and version name for the model to be read.
ELEMENT   - The name of a SEA element in the model to be read that you
            want included in your model. (optional)
```

Conventions

If your model has an element with the same name as one in the model to be read from, then you will be asked for a new name for the element being read. If you don't put in a new name then that element will replace the one that is already in your model.

10.21. RENAME

RENAME command

Purpose

This command is used to change the name of an element.

Command

```
RENAME OLDNAME NEWNAME
```

Where

```
OLDNAME - The existing name of an element that you wish to change.
NEWNAME - The new name to give to the element.
```

Note

The names associated with each path are also changed.

Example

```
RENAME EXTA EXTERNAL
```

Now to refer to the old path EXTA,SKIN you would now type EXTERNAL,SKIN.

BUGS

Does not check to see if you are duplicating another name. This could result in two elements of the same name. If this were the case the first one in the list would be accessed. Another RENAME should correct the problem.

10.22. RESPONSE

RESPONSE command

Purpose

This command allows the input of a response for an EXTRAP I prediction.

Command

```
RESPONSE
```

This command has no arguments.

The user is then prompted to input responses for each frequency of each response element as was specified by the SETRES and FREQUENCY commands.

10.22.1. EXAMPLE

RESPONSE example

```
SEMOD > SETRES INTA
SEMOD > FREQUENCY 50.,100.
SEMOD > RESPONSE
INTA    ( 4) > 100.,110.,110.,100.
SEMOD >
```

This case shows input for the frequency range 50. to 100. hz for the response element INTA.

10.23. SAVE

SAVE command (Not yet operational)

Purpose

This command is used to save your entire model or a single element of your model to a separate location from the model you are working on.

Command

```
SAVE N1,NAME (ELEMENT)
```

Where

```
N1,NAME1 - The dal unit and version name for the model to be saved.
ELEMENT   - The name of a SEA element in the model to be saved.
           (optional)
```

Note

It is not necessary to issue this command to save your model as it is automatically saved as you build it.

10.24. SETEXC

SETEXC command

Purpose

This command is used to specify which elements are excitation elements.

Command

```
SETEXCA NAME1,NAME2,NAME3...
or
SETEXCN NAME1,NAME2,NAME3...
or
SETEXCL
```

Where

NAME1,NAME2,NAME3...
This is the list of elements to designate as excitation elements.

CASE I

SETEXCA - Add elements to the list of excitation elements.

This command will add the input list of elements to an already existing list of excitation elements. This form is the default.

CASE II

SETEXCN - Create new list of excitation elements.

This command will create an entirely new list of excitation elements even if an old list already exists.

CASE III

SETEXCL - List the excitation elements.

The "L" option will list the excitation elements; it can also be used in conjunction with the above two cases.

Notes

If an element is specified more than once a warning will be printed and that element will only be used once.

This command must be issued prior to the ATACO command.

10.25. SETRES

SETRES command

Purpose

This command is used to specify which elements are response elements for EXTRAP I predictions..

Command

```
SETRESA NAME1,NAME2,NAME3...
or
SETRESN NAME1,NAME2,NAME3...
or
SETRESL
```

Where

NAME1,NAME2,NAME3...
This is the list of elements to designate as response elements.

CASE I

SETRESA - Add elements to the list of response elements.

This command will add the input list of elements to an already existing list of response elements. This form is the default.

CASE II

SETRESN - Create new list of response elements.

This command will create an entirely new list of response elements even if an old list already exists.

CASE III

SETRESL - List the response elements.

The "L" option will list the response elements; it can also be used in conjunction with the above two cases.

Notes

If an element is specified more than once a warning will be printed and that element will only be used once.

This command must be issued prior to the ATACO command.

10.26. TEXT

TEXT command

Purpose

This command is used to input a block of descriptive text for your model.

Command

```
TEXT  (EOFL)
Line 1
Line 2
  .
  .
  .
ZZZZ
```

Where

EOFL - Is a flag to terminate input of text. Default is 'ZZZZ'

10.27. TPRD

TPRD command

Purpose

This command calculates the theoretical response for the model.

Command

TPRD

Input

The following matrices are required for input:

N1,'FREQ',NAME - Frequency vector. (MDENS)
N1,'IDES',NAME - Descriptor matrix. (ATACO)
N1,'CONV',NAME - This matrix contains the conversion factor for mean square response in MKS units to energy in MKS units. (CFAC)
N1,'CPOW',NAME - This is a 1 X 1 matrix containing the conversion factor from energy in the user's units to energy in MKS units. (CFAC)
N1,'DBFG',NAME - This is a vector containing a flag for the unit type for each element in the model. (CFAC)
N1,'INCO',NAME - Inverse coefficient matrix. (ATACO)
N1,'TRNF',NAME - Transfer function matrix. (ATACO)
N1,'EXC ',NAME - Excitation matrix. Each column is an excitation corresponding to the element specified in SETEXC. The order must be the same as is specified in SETEXC. The ECOL command is useful for inputting this matrix.
N1,'POWF',NAME - Power input flags. These are the numbers of the elements (same as the row number in the MNAM matrix) for which a power input is to be specified. (#Power inputs,1) Optional
N1,'POWD',NAME - Power inputs. 1 column for each flag specified in the POWF matrix. (#Frequencies,#Power inputs) Optional

Output

The following matrix is output:

N1,'RESP',NAME - The response matrix. One column for each element. Contains both excitations and responses.

Comments

This command generally executes in much less time than the ATACALC and ATACO commands.

The use of power inputs is identical to the usage in the PRDICT command for EXTRAP I predictions.

10.28. PREDICT

PREDICT command

Purpose

This command is used to do EXTRAP I, EXTRAP II and Theoretical predictions.

Command

```
PREDICT/THEORETICAL  N1,NAME
or
PREDICT/EXTRAP/I N1,BASE N2,NEW IBACK
or
PREDICT/EXTRAP/II N1,BASE N2,NEW IUEXC,IURES
```

Where

CASE I: Theoretical prediction.

N1,NAME - DAL unit and version name for SEMOD model to do theoretical prediction on.

CASE II: EXTRAP I prediction.

N1,BASE - DAL unit and version name for SEMOD model to use as a baseline.
N2,NEW - DAL unit and version name for SEMOD model to predict.
IBACK - SEA element number for which to back out a transfer function.

CASE III: EXTRAP II prediction. (Similar to SCALE command)

N1,BASE - DAL unit and version name for SEMOD,S model to use as a baseline.
N2,NEW - DAL unit and version name for SEMOD,S model to predict.
IUEXC - Units of excitation. Same as used in the CONVERT command.
IURES - Units of response. Same as used in the CONVERT command.

Notes

Refer to chapter 11 (Statistical Energy Modeler) for information on building models. The two extrapolation methods are documented ,in part, in the following sections. Some of the commands are documented in chapter 11.

10.28.1. EXTRAP_I

Procedure for EXTRAP I predictions

Overview

This new procedure takes advantage of the SEMOD software for creating SEA models. The two prediction methods, EXTRAP I and EXTRAP II, were modified to use this software. The advantages are:

1) The input of parameters is easier.
2) The flexibility of the predictions are increased.
3) The connections listed for the SEMOD method are used so a different set of connection types is not needed.

There are several disadvantages:

1) The user must now specify the connection types.
2) The interface to the database is not as strong.
3) More input is required.

The basic idea is to set up a set of allowable names and types for the elements and connections in an EXTRAP I model. Then the user enters SEMOD and then begins input of his model using ELNAME and PATHNAME. The set of allowable names and types limits what can be entered. This makes it easy to create an EXTRAP I model using SEMOD . The following is a list of allowed elements and connections and their allowed types (see the SEMOD documentation for descriptions of each type):

Element	Allowed types	Connection	Allowed types
EXTA	1,2	EXTA-SKIN	1:3
SKIN	3:5	EXTA-SKIN-INTA	4:6
INTA	1,2	SKIN-INTA	1:3
MONT	3:9	SKIN-MONT	7:12,15:29,41:43
INST	3:5	SKIN-MONT-INST	30:39
		INTA-MONT	1:3
		INTA-INST	1:3
		MONT-INST	7:12,15:29,41:43

Use the RUN=DCON command to create a symbolic list of the parameters in a DAMO type parameter matrix such as the one returned by the RUN=GMOD command.

10.28.1.1. EXAMPLE

EXTRAP I example.

The following is an example using this new method. This example does not use data from the data base:

```
* VAPEPS demo of EXTRAP I prediction.
*
* 1) Parameter input.
*
TEMPLATE=EXTRAP 2,S772 2,YPAN
SEMOD 2,s772
 ELNAME
  EXTA,1
   rho =.11480e-06,  co  =.13390e+05,  v   =.23000e+07,  ap  =.10600e+06
   aac =.30000e-01
   EXIT
  SKIN,PLAT
   rho =.25700e-04,  cl  =.20200e+06,  h   =.12500e+01,  ap  =.21920e+04
   alx =.67900e+02,  aly =.32290e+02
   dlf =.10000e+00,  e   =.10500e+07,  pata=.20000e+03
   rhos=.32000e-04,  asms=.71980e+00
   EXIT
  LIST
  DONE
 PATHNAME
  EXTA,SKIN,1
   EXIT
  DONE
 FREQUENCY=25.,10000.
 SETEXCL EXTA
 EXCITATION
  74.4, 91.0,111.9,125.6,134.1,135.2,133.6,135.1
 135.6,129.6,126.9,129.9,129.1,127.0,126.7,124.2
 124.2,124.3,123.5,121.0,117.6,115.3,112.6,108.9
 105.4,104.2,101.9
 SETRESL SKIN
 MDENS
 ATACALC
 ATACO
 CFAC 7,1,4
 CRFR
 DONE
SEMOD 2,YPAN
 ELNAME
  EXTA,1
   rho =.11480E-06,  co  =.13390e+05,  v   =.23000e+07,  ap  =.10600e+06
   aac =.30000e-01
   EXIT
  SKIN,PLAT
   rho =.25900e-04,  cl  =.20200e+06,  h   =.12400e+01,  ap  =.21920e+04
   alx =.67900e+02,  aly =.32290e+02
   dlf =.10000e+00,  e   =.10600e+07,  pata=.20000e+03
   rhos=.32000e-04,  asms=.70700e+00
   EXIT
  LIST
  DONE
 PATHNAME
  EXTA,SKIN,1
   EXIT
  DONE
 FREQUENCY=25.,10000.
 SETEXCL EXTA
 EXCITATION
   74.1  88.9 105.4 124.4 135.5 138.7 138.6 139.8
  137.0 133.1 130.7 131.2 132.1 130.1 129.7 128.0
  126.8 126.1 124.8 122.7 120.7 118.2 116.0 113.6
  111.0 110.9 108.8
  SETRESL SKIN
  RESPONSE
  3.9107E-06  2.9200E-06  2.3052E-06  5.3739E-05  2.0972E-03  6.5924E-04
  5.3930E-03  2.4060E-02  4.1195E-03  4.4200E-03  4.5172E-03  6.7789E-03
  1.0011E-02  5.8346E-03  3.2441E-03  2.7141E-03  1.6797E-03  1.0214E-03
  5.5276E-04  5.3567E-04  6.5492E-04  4.2887E-04  1.6892E-04  2.9483E-04
  9.1950E-05  1.3979E-04  3.3978E-05
 MDENS
 ATACALC
 ATACO
 CFAC 7,1,4
 CRFR
 DONE
*
* Do prediction
PREDICT/EXTRAP/I 2,ypan,2,s772,1
*
SEMOD 2,S772
 LIST RESP
 DONE
END
```

10.28.1.2. PROGRAMMING NOTES

Programming Notes.

The storage of the above allowed element and connection names and types is accomplished via 4 matrices.

The first is called the allowed name matrix (ANAM). It is stored as follows:

Column1	Column2
EXTA	' '
SKIN	' '
INTA	' '
MONT	' '
INST	' '

The second is the allowed element type matrix (AETY). It is stored as follows:

Column1	Column2
1	1
1	2
2	3
2	4
2	5
3	1
3	2
etc...	

Column 1 refers to the element number in ANAM
Column 2 refers to the element types allowed.
In this case there are 17 allowed types.

The third is the allowed connection matrix (APTH). It is stored as follows:

Column1	Column2	Column3
1	2	0
1	2	3
2	3	0
2	4	0
2	4	5
3	4	0
3	5	0

Each column represents an element in the connection A 0 means that only 2 elements are involved in the connection.

```
    4        5        0
```

The fourth is the allowed connection type matrix (APTY). It stored in the same way as the allowed element type matrix. The difference is that the columns refer the the connection number and connection type respectively.

Any one of these or all may be omitted except that the element name matrix must be present if any of the others are to be used and the connection matrix must be present if the connection type matrix is to be used.

10.28.2. EXTRAP_II

EXTRAP II predictions

The EXTRAP II prediction method scales a structural element based on some basic parameters. It is basically a 1 element SEA model. The parameters required are different that for theoretical or EXTRAP I SEA predictions.

10.28.2.1. EXAMPLE

The following is an example using the new EXTRAP II method:

```
*
*  Create base line model.
SEMOD,S 1,YPAN
 ELNAME
  YPAN,SCAL
*
* Parameters for baseline model.
   RHO=7.2E-6,RHOS=2.19E-5,ASMS=.707,H=3.04,AP=2123.,DLF=.04
   EXIT
  DONE
*
* Input excitation and frequency range for baseline model.
 SETEXC YPAN
 FREQUENCY=160.,250.
 EXCITATION
  137.0,133.1,130.7
 DONE
*
* Create new model.
SEMOD,S 1,S772
```

```
 ELNAME
  S772,SCAL
*
* Parameters for new model.
   RHO=7.20E-6,RHOS=2.19E-5,ASMS=.720,H=3.04,AP=2123.,DLF=.04
   EXIT
  DONE
*
* Input excitation and frequency range for baseline model.
 SETEXC S772
 FREQUENCY=160.,250.
 EXCITATION
  135.7,129.6,126.9
DONE
*
* Input response for baseline model.
ECOL 1,RESP,YPAN 3,1
 4.12E-3,4.42E-3,4.52E-3
*
* Do prediction.
PREDICT/EXTRAP/II 1,YPAN 1,S772 1,7
*
* Display results.
SEMOD,S 1,S772
 LIST RESP
 DONE
```

11. OLD SEA COMMANDS

This section contains an early version of the prediction commands given in Chapter 10; they are included for the sake of transition from an old method to the new method. The use of the Chapter 10 commands is now recommended.

The use of the commands in this chapter is described under the PRDICT command in this chapter. Briefly the order of use of these commands is:

DAMO	- Input properties.
DENS	- Calculate modal densities.
TRNF	- Calculate coupling loss factors and transfer functions.
CNVRT	- Calculate conversion factors.
PRDICT	- Do the prediction.

11.1. CNVRT

CNVRT Command

Purpose

The arguments of the CNVRT command are used to identify the user's units. CNVRT takes these units and calculates a set of conversion factors which are used by the prediction routines to convert into the units for which these routines were designed.

Command

```
CNVRT NU,VER,ITYP,JTYP,KTYP
```

Where

NU,VER - DAL unit and version name of your model. Output from CNVRT will be stored in the same DAL file with the same version name as the parameter matrix.

ITYP - The units of mean-square acceleration.

= 1 - (M/Sec**2)**2

= 2 - g**2

= 3 - (in/sec**2)**2

= 4 - (ft/sec**2)**2

= 5 - dB (REF = 8.4144E-18)

JTYP - The units of mean-square pressure.

= 1 - (Newton/M**2)**2

= 2 - psi**2

= 3 - dB (REF = 8.4144E-18)

KTYP - The units of the parameters in the parameter matrix.

= 1 - M, KG, SEC

= 2 - CM, GRAM, SEC

= 3 - FT, SLUG (LBF-SEC**2/FT), SEC

= 4 - IN, SNAIL (LBF-SEC**2/IN), SEC

Output

The following matrices are output by CNVRT:

NU,'CONV',VER - Conversion factor from energy to response.

NU,'CPOW',VER - Conversion factor from power in user units to power in MKS units.

NU,'DBFG',VER - Flags for conversion into dB units.

The output of this command is used by the other prediction commands. The prediction command that uses this output is PRDICT. These output data sets should not normally be used except by the PRDICT command.

Related Commands

DENS, DAMO, TRNF, PRDICT

11.2. DAMO

DAMO Command

Purpose

DAMO creates and/or edits data module packets, as described in the MODULE section of the VAPEPS PREP command. It is included as a stand alone command to aid users in forming general SEA modules that may be applied (with minor changes) to many events.

Command

```
DAMO  N1,VR1  N2,VR2  MAXCH
```

Where

N1,VR1 - Input configuration (unit,name) If omitted, the module is built from scratch.
N2,VR2 - Output configuration unit and name.
MAXCH - Maximum number of channels for this module. If omitted, MAXCH defaults to 50.

Sub Commands

DESC NAME1 NAME2 NAME3
Provides up to three 12 character names as a description of this module. These names may later be searched in the SERCH processor.

EXTA NAME1 NAME2 NAME3
Enters the external acoustic zone mode and optionally provides for up to 3 names to describe this zone. If any names are included, all previous EXTA names are removed. If no names are included any previous names are kept.

SKIN NAME1 NAME2 NAME3
Enters the skin zone. The above rules apply here as well.

INTA NAME1 NAME2 NAME3
Interior acoustic zone.

MONT NAME1 NAME2 NAME3
Mounting structure.

INST NAME1 NAME2 NAME3
Installation (usually the item whose response is desired)

FRAM NAME1 NAME2 NAME3

Frame structure. This zone is applicable only in modules where the mounting structure is a truss.

SEA Zone Editting Commands

P* NAM1=VAL1 NAM2=VAL2 NAM3=VAL3 .....
Sets the value of named parameters. See the VAPEPS manual for details.

C* CHAN1,CHAN2,CHAN3,.....
Adds the named channels (4 chars max) to the list of channels for this zone.

REMOVE CHAN1,CHAN2,CHAN3,....
Removes the named channels from the list.

LIST CHANNEL
Will list the channels in the current zone. If no zone has been entered or DESC was the last command, then channels will be listed for all zones.

LIST PARAMETERS
Will list parameters in this zone. If no zone is active parameters will be listed for all zones.

General (any zone) commands

READ NU,VER,ZONE
Reads the specified module. If zone is included (ie. SKIN INTA, etc), then only that zone will be read. Note that a read operation will completely replace the data module or zone.

SAVE NU,VER
Saves the current module and stays in DAMO.

CHECK
Checks the parameters for all zones with non-zero types and computes the appropriate non-dimensional parameters for those sections. These parameters are then listed. Diagnostics are printed for parameters which are needed but have not been specified.

DONE
Performs a CHECK operation (no listing), a SAVE to the original output module name (N2,VR2) and then exits DAMO. A blank line is equivalent to DONE.

11.2.1. PARAMETERS

Note all parameters must be input as real numbers.

SEA Element EXTA.

TYPE = 1.	TYPE = 2.	Description
Reverberant Field	Non Reverberant Field	
TYPE	TYPE	Code describing type of acoustic field.
RHO	RHO	Mass density.
	FBL	Length of flying body.
AP	AP	Surface area of external space.
V	V	Volume of external space.
	VISC	Kinematic viscosity of fluid.
AAC	AAC	Acoustic absorbtion coefficient.
CO	CO	Velocity of sound.
	VEL	Velocity of flying object.

If TYPE = 0. (default condition) then the SEA element is considered to be not present in the model.
TYPE = 2. is not presently supported by the software.

SEA Element SKIN

TYPE = 1.	TYPE = 2.	TYPE = 3.	TYPE = 4.	Description
Flat Plate	Cylinder	Cone	EXTRAP II	
TYPE	TYPE	TYPE	TYPE	Code describing geometry.
RHO	RHO	RHO	RHO	Mass density.
RHOS	RHOS	RHOS	RHOS	Surface mass density.
ASMS	ASMS	ASMS	ASMS	Non structural mass.
H	H	H	H	Equivalent thickness.
	D	D		Diameter.
	BL	BL		Length.
ALX	ALX	ALX		Sub panel dimension X.
ALY	ALY	ALY		Sub panel dimension Y.
PATA	PATA	PATA		Length of discontinuity.
AP	AP	AP	AP	Surface area.
		BETA		Half apex angle of cone.
E	E	E		Young's Modulus.
DLF	DLF	DLF	DLF	Damping loss factor.
CL	CL	CL		Longitudinal wave speed.

If TYPE = 0. (default condition) then the SEA element is considered to be not present in the model.

SEA Element INTA

TYPE = 1.	Description
Reverberant Field	
TYPE	Code describing type of acoustic field.
RHO	Mass density.
AP	Surface area of external space.
V	Volume of external space.
AAC	Acoustic absorbtion coefficient.
CO	Velocity of sound.

If TYPE = 0. (default condition) then the SEA element is considered to be not present in the model.

SEA Element MONT
TYPE = 1., 2. or 6.
Truss Mount A, Truss Mount B or Beam Mount

XTYP = 1.	XTYP = 2.	XTYP = 3.	XTYP = 4.	Description
Rectangular Cross Section	Circular Cross Section	Tubular Cross Section	Other Cross Section	
TYPE	TYPE	TYPE	TYPE	Code describing geometry.
XTYP	XTYP	XTYP	XTYP	Cross section type.
TYPJ	TYPJ	TYPJ	TYPJ	Coupling type.
RHO	RHO	RHO	RHO	Mass density.
ASMS	ASMS	ASMS	ASMS	Non structural mass.
H				Thickness.
B				Width.
BL	BL	BL	BL	Length or truss members.
		DI		Inner diameter.
	DO	DO		Outer diameter.
			RGF	Flexural radius of gyration
			A	Cross section area.
			PJ	Polar moment of inertia.
ALPHA	ALPHA	ALPHA	ALPHA	Orientation angle.
BETA	BETA	BETA	BETA	Orientation angle.
E	E	E	E	Young's Modulus.
G	G	G	G	Shear Modulus.
			T	Torsional stiffness.
DLF	DLF	DLF	DLF	Damping loss factor.
CL	CL	CL	CL	Longitudinal wave speed.
RATE	RATE	RATE	RATE	Attenuation rate, dB/octave
CFRQ	CFRQ	CFRQ	CFRQ	1/3 octave band center freq at which attenuation begins
CNT	CNT	CNT	CNT	Number of truss members.

If TYPE = 0. (default condition) then the SEA element is considered to be not present in the model.
TYPJ is required only if INST is present.
If TYPE = 1. allowable values for TYPJ are 1. and 6. only.
If TYPE = 2. allowable values for TYPJ are 2. and 7. only.
If TYPE = 6. allowable value for TYPJ is 11.
RATE and CFRQ are required fro TYPJ = 6. or 7. only..
ALPHA and BETA are not required if TYPE = 6.

SEA Element MONT
TYPE = 3. or 4.
T Plate Mount or Oblique Plate Mount

	Description
TYPE	Code describing geometry.
TYPJ	Coupling type.
RHO	Mass density.
ASMS	Non structural mass.
H	Equivalent thickness.
ALX	Sub panel dimension X.
ALY	Sub panel dimension Y.
PATA	Length of discontinuity.
BJL	Length of joint.
AP	Surface area.
BETA	Plate to plate angle.
E	Young's Modulus.
DLF	Damping loss factor.
CL	Longitudinal wave speed.

If TYPE = 0. (default condition) then the SEA element is considered to be not present in the model.
BETA is not required fro TYPE = 3.
TYPJ is required only if INST is present.
The allowable values for TYPJ are 3., 4., 5., 9. and 10. only.

SEA Element MONT
TYPE = 5. or -5.
Corner Plate Mount or Line Mount

		Description
TYPE = 5.	TYPE= -5.	Code describing geometry.
TYPJ		Coupling type.
RHO		Mass density.
ASMS		Non structural mass.
H		Equivalent thickness.
ALX		Sub panel dimension X.
ALY		Sub panel dimension Y.
PATA		Length of discontinuity.
BJL	BJL	Length of joint.
AP		Surface area.
BETA	BETA	Plate to plate angle.
E		Young's Modulus.
DLF		Damping loss factor.
CL		Longitudinal wave speed.

If TYPE = 0. (default condition) then the SEA element is considered to be not present in the model.
TYPJ is required only if INST is present.
The allowable values for TYPJ are 3., 4., 5., 9. and 10. only.

SEA Element MONT
TYPE = 7.
Isolation Truss Mount

XTYP = 1.	XTYP = 2.	XTYP = 3.	XTYP = 4.	Description
Rectangular Cross Section	Circular Cross Section	Tubular Cross Section	Other Cross Section	
TYPE	TYPE	TYPE	TYPE	Code describing geometry.
XTYP	XTYP	XTYP	XTYP	Cross section type.
TYPJ	TYPJ	TYPJ	TYPJ	Coupling type.
RHO	RHO	RHO	RHO	Mass density.
ASMS	ASMS	ASMS	ASMS	Non structural mass.
H				Thickness.
B				Width.
BL	BL	BL	BL	Length or truss members.
		DI		Inner diameter.
	DO	DO		Outer diameter.
			RGF	Flexural radius of gyration
			A	Cross section area.
			PJ	Polar moment of inertia.
ALPHA	ALPHA	ALPHA	ALPHA	Orientation angle.
BETA	BETA	BETA	BETA	Orientation angle.
E	E	E	E	Young's Modulus.
G	G	G	G	Shear Modulus.
			T	Torsional stiffness.
DLF	DLF	DLF	DLF	Damping loss factor.
CL	CL	CL	CL	Longitudinal wave speed.
RATE	RATE	RATE	RATE	Attenuation rate, dB/octave
CFRQ	CFRQ	CFRQ	CFRQ	1/3 octave band center freq at which attenuation begins
CNT	CNT	CNT	CNT	Number of truss members.

If TYPE = 0. (default condition) then the SEA element is considered to be not present in the model.
TYPJ is required only if INST is present.
The allowable values for TYPJ are 1., 2., 6. and 7. only.
TYPJ = 6. or 7. indicates vibration isolators between SKIN and MONT and INST. RATE and CFRQ apply to both isolators in this case.

SEA Element MONT
TYPE = 8.
Cone Mount

	Description
TYPE	Code describing geometry.
TYPJ	Coupling type.
RHO	Mass density.
ASMS	Non structural mass.
H	Equivalent thickness.
D	Base diameter.
BL	Height of cone.
ALX	Sub panel dimension X.
ALY	Sub panel dimension Y.
PATA	Length of discontinuity.
BJL	Length of joint.
AP	Surface area.
BETA	Half apex angle of cone.
E	Young's Modulus.
DLF	Damping loss factor.
CL	Longitudinal wave speed.

If TYPE = 0. (default condition) then the SEA element is considered to be not present in the model.
TYPJ is required only if INST is present.
The only allowable value for TYPJ is 8.

SEA Element INST

TYPE = 1.	TYPE = 2.	TYPE = 3.	TYPE = 4.	Description
Flat Plate	Cylinder	Cone	EXTRAP II	
TYPE	TYPE	TYPE	TYPE	Code describing geometry.
RHO	RHO	RHO	RHO	Mass density.
			RHOS	Surface mass density.
ASMS	ASMS	ASMS	ASMS	Non structural mass.
H	H	H	H	Equivalent thickness.
	D	D		Diameter.
	BL	BL		Length.
ALX	ALX	ALX		Sub panel dimension X.
ALY	ALY	ALY		Sub panel dimension Y.
PATA	PATA	PATA		Length of discontinuity.
BJL	BJL	BJL		Joint length.
AP	AP	AP	AP	Surface area.
		BETA		Half apex angle of cone.
GAMA	GAMA	GAMA		Plate to plate angle.
E	E	E		Young's Modulus.
DLF	DLF	DLF	DLF	Damping loss factor.
CL	CL	CL		Longitudinal wave speed.

If TYPE = 0. (default condition) then the SEA element is considered to be not present in the model.
BJL is required for MONT TYPJ = 3., 4., 5., 8., 9. and 10. only.
GAMA is required for MONT TYPJ = 4., 8., and 10. only.

FRAM Structure
TYPE = 1.
Beam Frame
(Required only for MONT TYPE = 1., 2., and 7.)

XTYP = 1.	XTYP = 2.	XTYP = 3.	XTYP = 4.	Description
Rectangular Cross Section	Circular Cross Section	Tubular Cross Section	Other Cross Section	
TYPE	TYPE	TYPE	TYPE	Code describing geometry.
XTYP	XTYP	XTYP	XTYP	Cross section type.
RHO	RHO	RHO	RHO	Mass density.
H				Thickness.
B				Width.
BL	BL	BL	BL	Length or truss members.
		DI		Inner diameter.
	DO	DO		Outer diameter.
			RGF	Flexural radius of gyration
			A	Cross section area.
CL	CL	CL	CL	Longitudinal wave speed.

If TYPE = 0. (default condition) then the SEA element is considered to be not present in the model.

Note

The following figures are shown as they appeared in the original VAPEPS User's Manual, including the original figure numbers used.

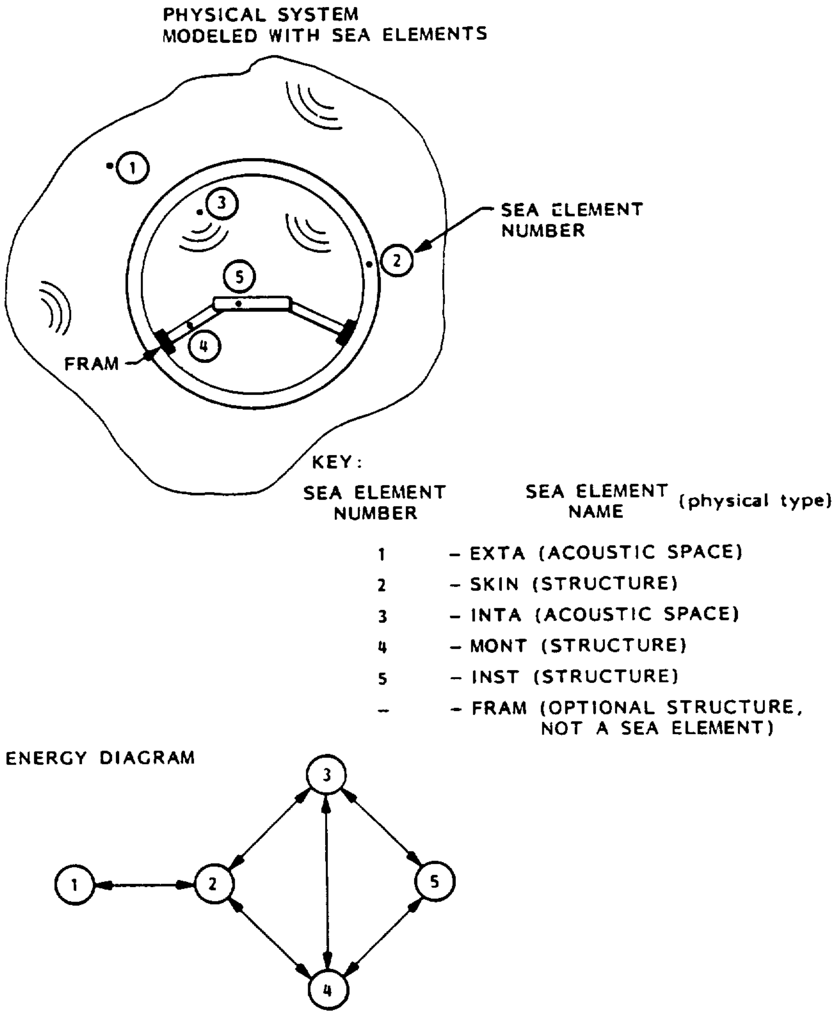


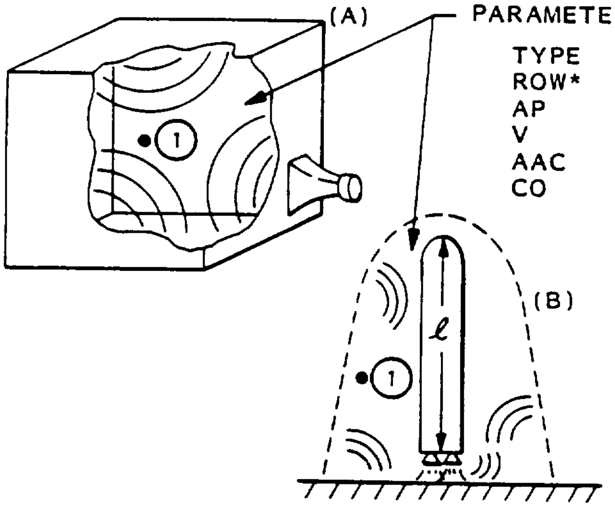
Figure 11-1

Figure 11.1 5 element SEA model.

(1) EXTA (ACOUSTIC SPACE)

TYPE = 1.

REVERBERANT (A)
OR LIFTOFF (B)*

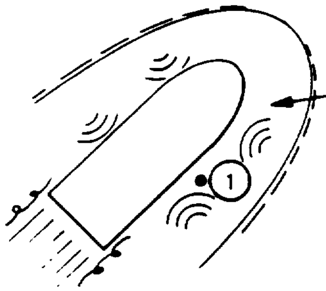


PARAMETERS	[UNITS]
TYPE	[ ]
ROW*	[MASS/(LENGTH**3)]
AP	[LENGTH**2]
V	[LENGTH**3]
AAC	[ ]
CO	[LENGTH/TIME]

* To model the external acoustic environment that exists at liftoff, the volume V should be calculated from the following expression: $V = 0.35 * \ell^3$

= 2.

NOT REVERBERANT Δ



PARAMETERS	[UNITS]
TYPE	[ ]
ROW*	[MASS/(LENGTH**3)]
FBL	[LENGTH]
AP	[LENGTH**2]
V	[LENGTH**3]
VISC	[(LENGTH**2)/TIME]
AAC	[ ]
CO	[LENGTH/TIME]
VEL	[LENGTH/TIME]

Δ Parameters for a TYPE = 2 acoustic space can be entered into the VAPEPS data base, but currently cannot be used in the EXTRAP prediction scheme.

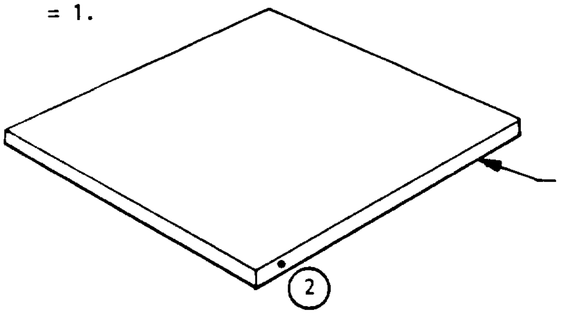
Figure 11-2

Figure 11.2 EXTA SEA acoustic space element.

② SKIN
(STRUCTURE)

TYPE

= 1.

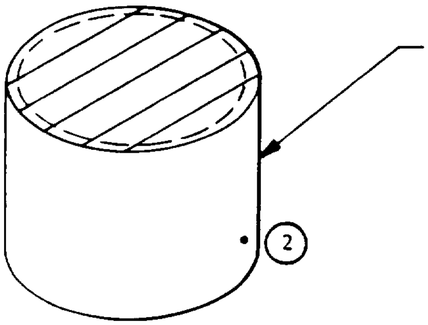


FLAT PLATE

PARAMETERS [UNITS]

TYPE	[ ]
ROW*	[MASS/(LENGTH**3)]
ROWS*	[MASS/(LENGTH**2)]
ASMS*	[MASS]
H	[LENGTH]
ALX*	[LENGTH]
ALY*	[LENGTH]
PATA*	[LENGTH]
AP	[LENGTH**2]
E	[FORCE/(LENGTH**2)]
DLF	[ ]
CL	[LENGTH/TIME]

= 2.



CYLINDER

PARAMETERS [UNITS]

TYPE	[ ]
ROW*	[MASS/(LENGTH**3)]
ROWS*	[MASS/(LENGTH**2)]
ASMS*	[MASS]
H	[LENGTH]
D	[LENGTH]
BL	[LENGTH]
ALX*	[LENGTH]
ALY*	[LENGTH]
PATA*	[LENGTH]
AP	[LENGTH**2]
E	[FORCE/(LENGTH**2)]
DLF	[ ]
CL	[LENGTH/TIME]

*SEE SKIN DETAIL (Figures 11-3c and 11-3d)

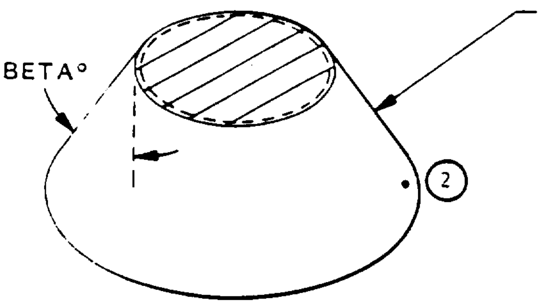
Figure 11-3a

Figure 11.3 SKIN SEA elements, plate and cylinder.

SKIN (CONTINUED)

TYPE

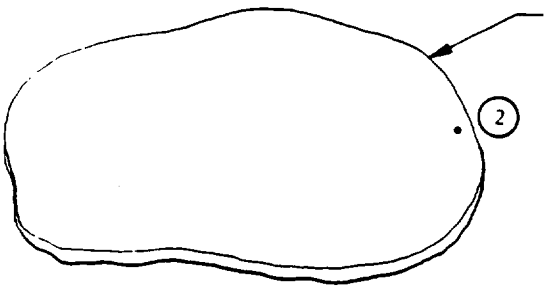
= 3.



PARAMETERS [UNITS]

TYPE	[ ]
ROW*	[MASS/(LENGTH**3)]
ROWS*	[MASS/(LENGTH**2)]
ASMS*	[MASS]
H	[LENGTH]
D	[LENGTH]
BL	[LENGTH]
ALX*	[LENGTH]
ALY*	[LENGTH]
PATA*	[LENGTH]
AP	[LENGTH**2]
BETA	[DEGREES]
E	[FORCE/(LENGTH**2)]
DLF	[ ]
CL	[LENGTH/TIME]

= 4.



PARAMETERS [UNITS]

TYPE	[ ]
ROWS*	[MASS/(LENGTH**2)]
ASMS*	[MASS]
H	[LENGTH]
AP	[LENGTH**2]
DLF	[ ]

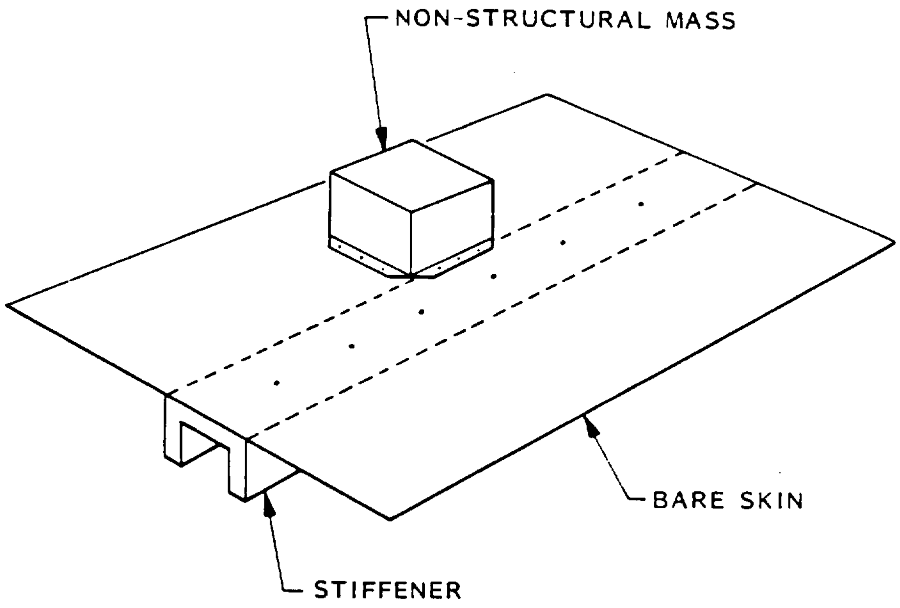
*SEE SKIN DETAIL (Figures 11-3c and 11-3d)

Figure 11-3b

Figure 11.4 SKIN SEA elements, cone and EXTRAP II.

SKIN – DETAIL

TYPICAL SKIN



PARAMETERS

ROW – INCLUDES ALL STRUCTURAL MASSES (I.E., BARE SKIN AND STIFFENER)

ROWS – INCLUDES THE BARE SKIN ONLY

ASMS – INCLUDES ALL ATTACHED EQUIPMENT DEVICES, ETC.

Figure 11-3c

Figure 11.5 Typical SEA SKIN element.

SKIN-DETAIL (CONTINUED)

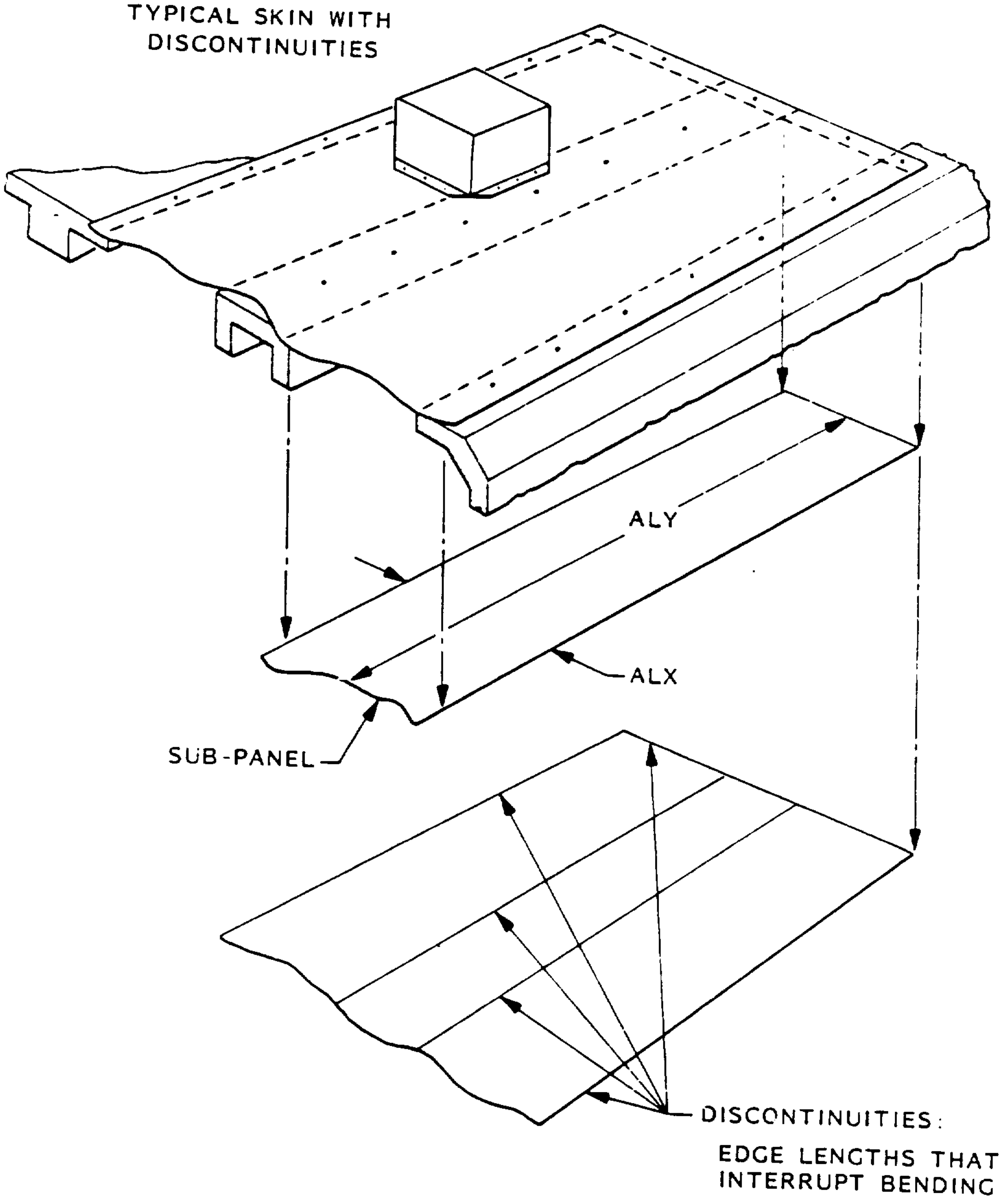


Figure 11-3d PATA = Σ DISCONTINUITIES

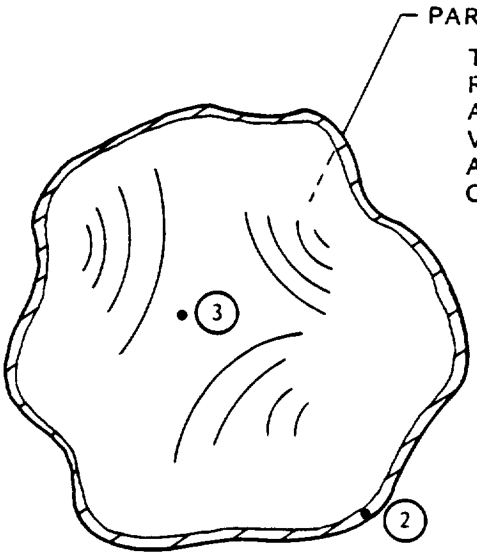
Figure 11.6 Typical SEA SKIN element continued.

③ INTA (ACOUSTIC SPACE)

TYPE = 1. REVERBERANT△

PARAMETERS [UNITS]

TYPE	[ ]
ROW*	[MASS/(LENGTH**3)]
AP	[LENGTH**2]
V	[LENGTH**3]
AAC	[ ]
CO	[LENGTH/TIME]



△ For some prediction models it may be desirable to model INTA as the external acoustic environment (See VAPEPS Volume 1 for details). For an external environment model of a liftoff acoustic space, see the sketch for EXTA in this section.

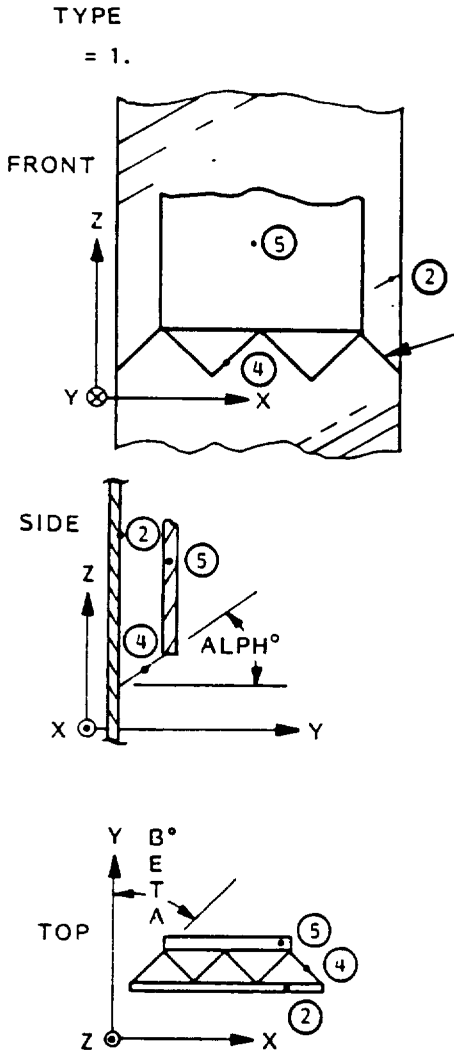
Figure 11-4

Figure 11.7 INTA SEA acoustic space element.

④ MONT (STRUCTURE)

TYPE
= 1.

TRUSS MOUNT CONNECTION
TO SKIN



② PARAMETERS [UNITS]

TYPE	[ ]
XTYP	[ ]
TYPJ†	[ ]
ROW*	[MASS/(LENGTH**3)]
ASMS*	[MASS]
H	[LENGTH]
B	[LENGTH]
BL	[LENGTH]
DI	[LENGTH]
DO	[LENGTH]
RGF	[LENGTH]
A	[LENGTH**2]
PJ	[LENGTH**4]
ALPH	[DEGREES]
BETA	[DEGREES]
E	[FORCE/(LENGTH**2)]
G	[FORCE/(LENGTH**2)]
T	[FORCE*(LENGTH**2)]
DLF	[ ]
CL	[LENGTH/TIME]
RATE	[dB/OCTAVE]
CFRQ	[HERTZ]
CNT	[ ]

† SEE MONT DETAIL (Figures 11-5i through 11-5m)

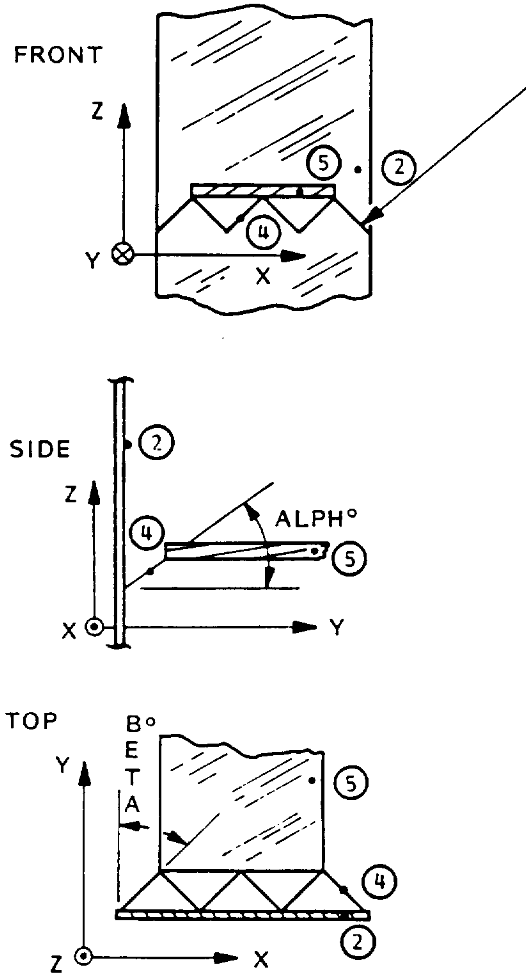
* SEE SKIN DETAIL (Figures 11-3c and 11-3d)

Figure 11-5a

Figure 11.8 MONT SEA element TYPE = 1.

④ MONT (STRUCTURE)

TYPE = 2. TRUSS MOUNT CONNECTION TO SKIN



PARAMETERS	[UNITS]
TYPE	[ ]
XTYP	[ ]
TYPJ†	[ ]
ROW*	[MASS/(LENGTH**3)]
ASMS*	[MASS]
H	[LENGTH]
B	[LENGTH]
BL	[LENGTH]
DI	[LENGTH]
DO	[LENGTH]
RGF	[LENGTH]
A	[LENGTH**2]
PJ	[LENGTH**4]
ALPH	[DEGREES]
BETA	[DEGREES]
E	[FORCE/(LENGTH**2)]
G	[FORCE/(LENGTH**2)]
T	[FORCE*(LENGTH**2)]
DLF	[ ]
CL	[LENGTH/TIME]
RATE	[dB/OCTAVE]
CFRQ	[HERTZ]
CNT	[ ]

† SEE MONT DETAIL (Figures 11-5i through 11-5m)

* SEE SKIN DETAIL (Fiqures 11-3c and 11-3d)

Figure 11-5b

Figure 11.9 MONT SEA element TYPE = 2.

(4) MONT (STRUCTURE)

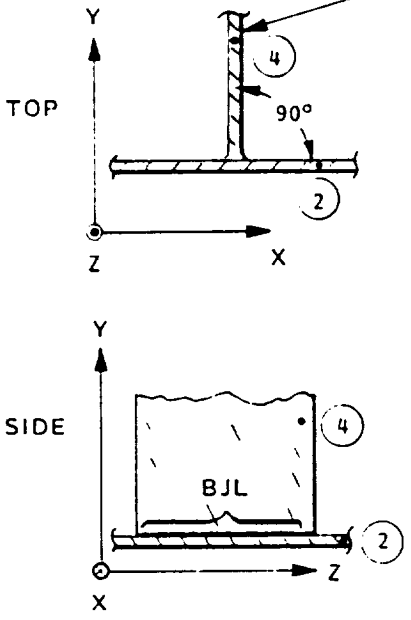
TYPE

= 3.

PLATE MOUNT "T" CONNECTION TO SKIN

PARAMETERS [UNITS]

TYPE	[ ]
TYPJ†	[ ]
ROW*	[MASS/(LENGTH**3)]
ASMS*	[MASS]
H	[LENGTH]
ALX*	[LENGTH]
ALY*	[LENGTH]
PATA*	[LENGTH]
BJL	[LENGTH]
AP	[LENGTH**2]
E	[FORCE/(LENGTH**2)]
DLF	[ ]
CL	[LENGTH/TIME]



† SEE MONT DETAIL (Figures 11-5i through 11-5m)

* SEE SKIN DETAIL (Figures 11-3c and 11-3d)

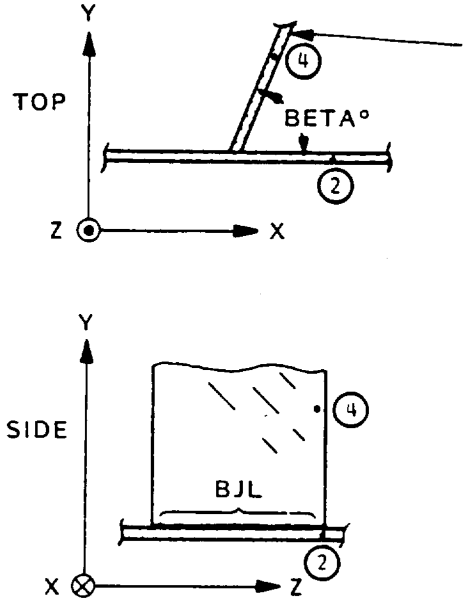
Figure 11-5d

Figure 11.10 MONT SEA element TYPE = 3.

④ MONT (STRUCTURE)

TYPE
= 4.

PLATE MOUNT OBLIQUE CONNECTION TO SKIN



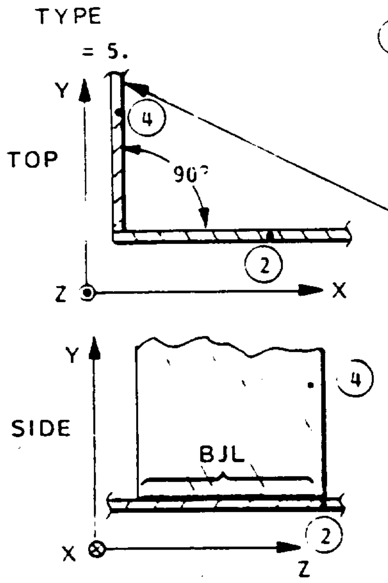
PARAMETERS	[UNITS]
TYPE	[ ]
TYPJ†	[ ]
ROW*	[MASS/(LENGTH**3)]
ASMS*	[MASS]
H	[LENGTH]
ALX*	[LENGTH]
ALY*	[LENGTH]
PATA*	[LENGTH]
BJL	[LENGTH]
AP	[LENGTH**2]
BETA	[DEGREES]
E	[FORCE/(LENGTH**2)]
DLF	[ ]
CL	[LENGTH/TIME]

†SEE MONT DETAIL (Figures 11-5i through 11-5m)

*SEE SKIN DETAIL (Figures 11-3c and 11-3d)

Figure 11-5e

Figure 11.11 MONT SEA element TYPE = 4.

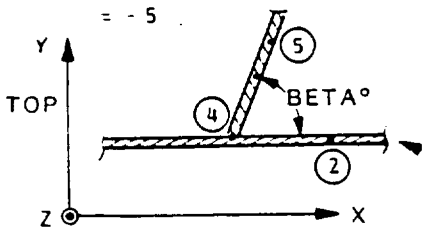


④ MONT (STRUCTURE)

PLATE MOUNT CONNECTION TO SKIN AT CORNER

PARAMETERS [UNITS]

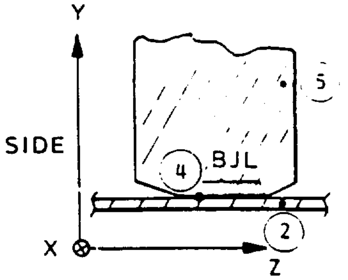
TYPE	[ ]
TYPJ†	[ ]
ROW*	[MASS/(LENGTH**3)]
ASMS*	[MASS]
H	[LENGTH]
ALX*	[LENGTH]
ALY*	[LENGTH]
PATA*	[LENGTH]
BJL	[LENGTH]
AP	[LENGTH**2]
BETA	[DEGREES]
E	[FORCE/(LENGTH**2)]
DLF	[ ]
CL	[LENGTH/TIME]



LINE MOUNT (INACTIVE) CONNECTION BETWEEN SKIN AND INST

PARAMETERS [UNITS]

TYPE	[ ]
BJL	[LENGTH]
BETA	[DEGREES]



† SEE MONT DETAIL (Figures 11-5i through 11-5m)

* SEE SKIN DETAIL (Figures 11-3c and 11-3d)

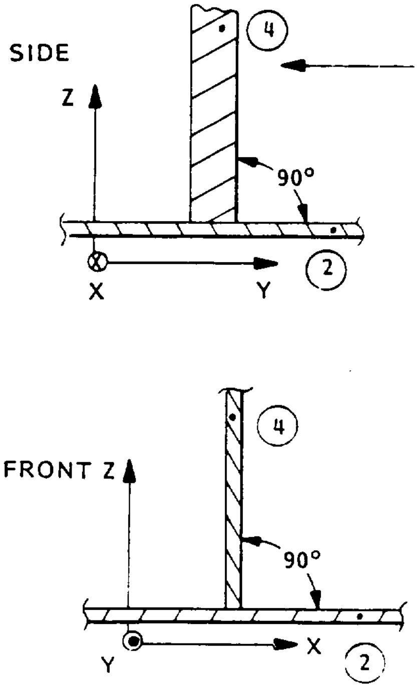
Figure 11-5f

Figure 11.12 MONT SEA element TYPE = 5. and -5.

(4) MONT (STRUCTURE)

TYPE

= 6. BEAM MOUNT CONNECTION TO SKIN



PARAMETERS	[UNITS]
TYPE	[ ]
XTYP	[ ]
TYPJ†	[ ]
ROW*	[MASS/(LENGTH**3)]
ASMS*	[MASS]
H	[LENGTH]
B	[LENGTH]
BL	[LENGTH]
DI	[LENGTH]
DO	[LENGTH]
RGF	[LENGTH]
A	[LENGTH**2]
PJ	[LENGTH**4]
E	[FORCE/(LENGTH**2)]
G	[FORCE/(LENGTH**2)]
T	[FORCE*(LENGTH**2)]
DLF	[ ]
CL	[LENGTH/TIME]
CNT	[ ]

†SEE MONT DETAIL (Figures 11-5i through 11-5m)

*SEE SKIN DETAIL (Figures 11-3c and 11-3d)

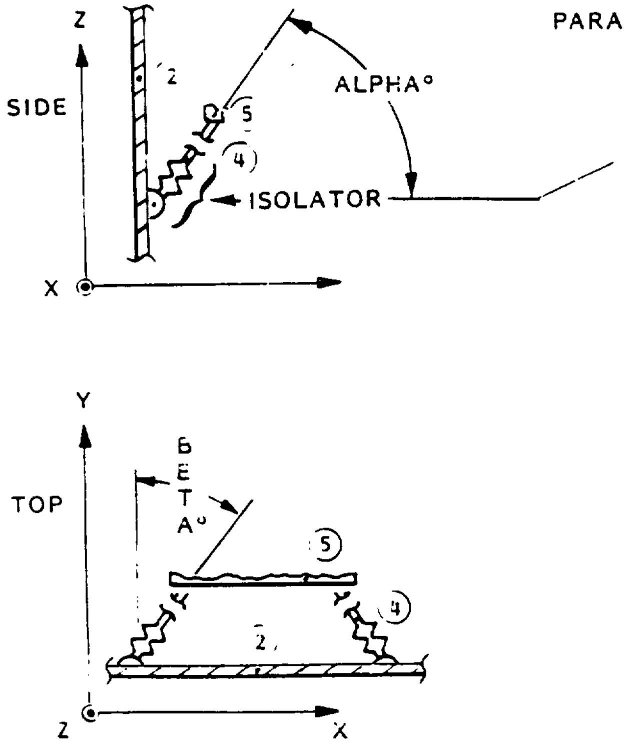
Figure 11-5c

Figure 11.13 MONT SEA element TYPE = 6.

(4) MONT (STRUCTURE)

TYPE

= 7. ISOLATION TRUSS MOUNT CONNECTION TO SKIN



PARAMETERS	[UNIT]
TYPE	[ ]
XTYP	[ ]
TYPJ†	[ ]
ROW*	[MASS/(LENGTH**3)]
ASMS*	[MASS]
H	[LENGTH]
B	[LENGTH]
BL	[LENGTH]
DI	[LENGTH]
DO	[LENGTH]
RGF	[LENGTH]
A	[LENGTH**2]
PJ	[LENGTH**4]
ALPH	[DEGREES]
BETA	[DEGREES]
E	[FORCE/(LENGTH**2)]
G	[FORCE/(LENGTH**2)]
T	[FORCE*(LENGTH**2)]
DLF	[ ]
CL	[LENGTH/TIME]
RATE	[dB/OCTAVE]
CFRQ	[HERTZ]
CNT	[ ]

† SEE MONT DETAIL (Figures 11-5i through 11-5m)

* SEE SKIN DETAIL (Figures 11-3c and 11-3d)

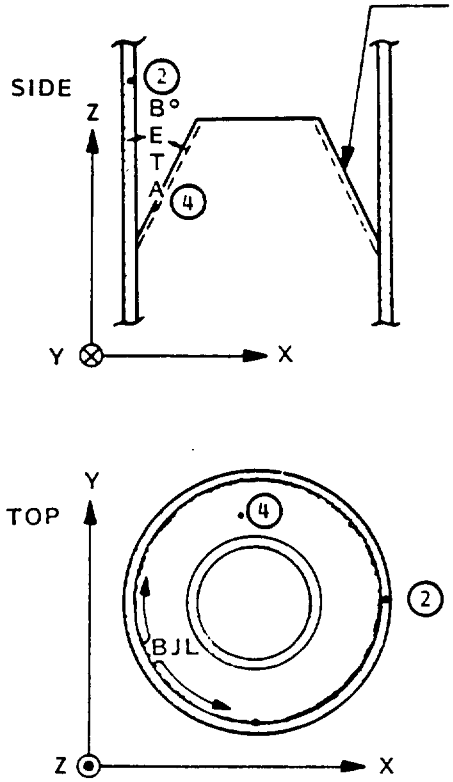
Figure 11-5g

Figure 11.14 MONT SEA element TYPE = 7.

④ MONT (STRUCTURE)

TYPE = 8.

CONE MOUNT CONNECTION TO SKIN



PARAMETERS [UNITS]

TYPE	[ ]
TYPJ†	[ ]
ROW*	[MASS/(LENGTH**3)]
ASMS*	[MASS]
H	[LENGTH]
D	[LENGTH]
BL	[LENGTH]
ALX*	[LENGTH]
ALY*	[LENGTH]
PATA*	[LENGTH]
BJL	[LENGTH]
AP	[LENGTH**2]
BETA	[DEGREES]
E	[FORCE/(LENGTH**2)]
DLF	[ ]
CL	[LENGTH/TIME]

*SEE SKIN DETAIL (Figures 11-3c and 11-3d)

+SEE MONT DETAIL (Figures 11-5i through 11-5m)

Figure 11-5h

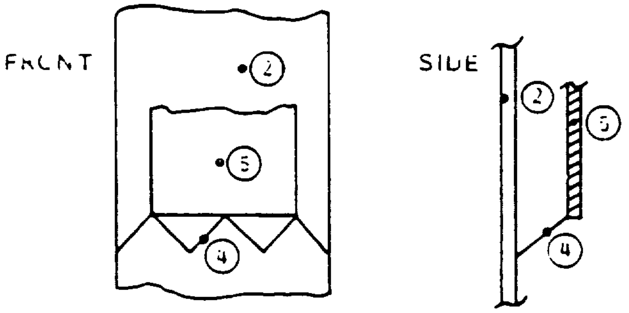
Figure 11.15 MONT SEA element TYPE = 8.

MONT DETAIL

TYPJ

= 1.

TRUSS MOUNT CONNECTION A TO PLATE INSTALLATION (INSTALLATION PARALLEL TO SKIN)



2.

TRUSS MOUNT CONNECTION B TO PLATE INSTALLATION (INSTALLATION NORMAL TO SKIN)

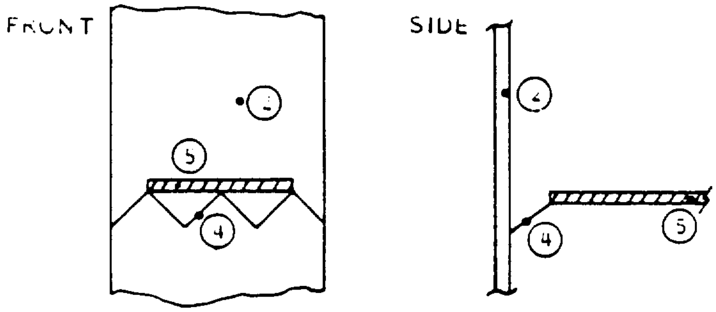


Figure 11-5i

Figure 11.16 MONT SEA element TYPJ = 1. and 2.

TYPJ MONT DETAIL (CONTINUED)

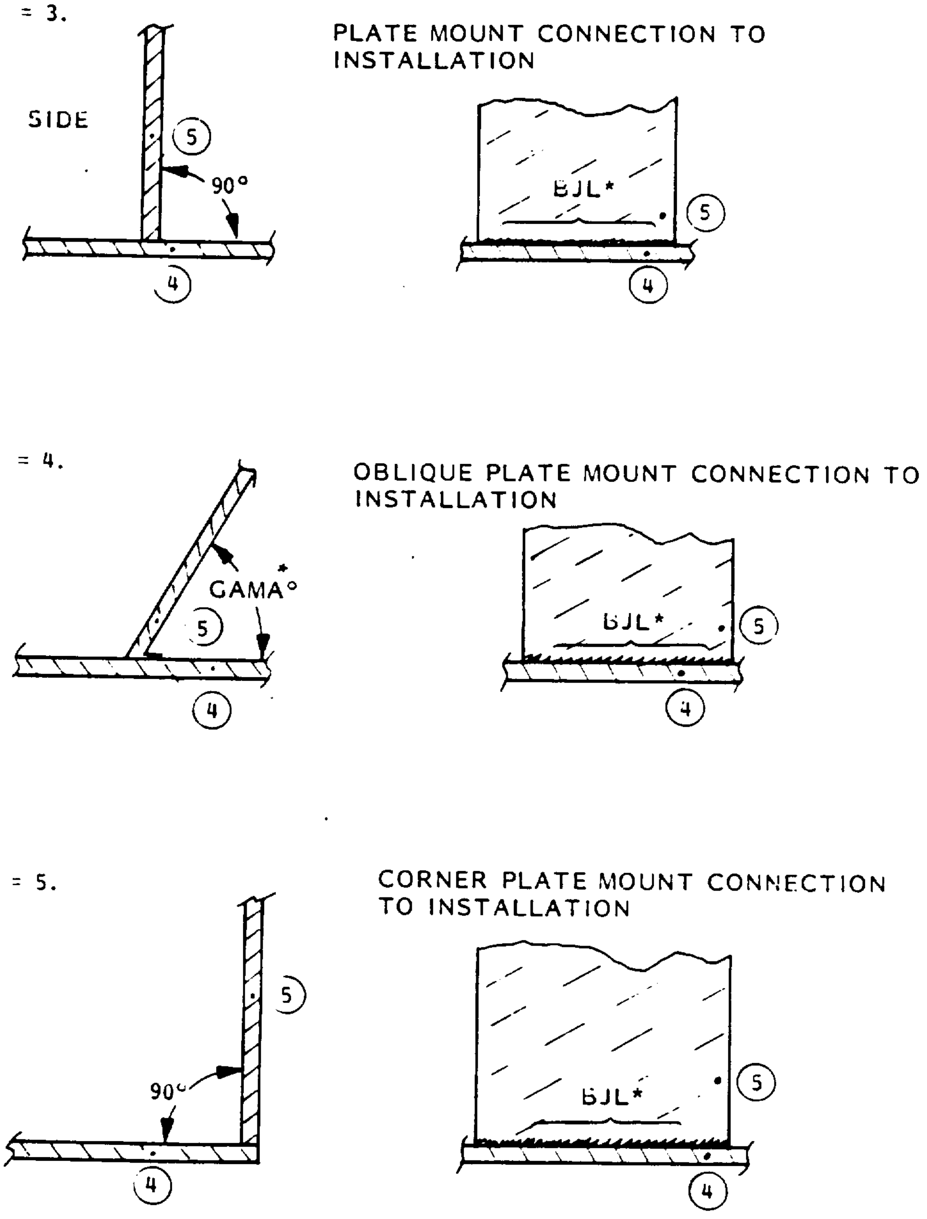


PLATE MOUNT CONNECTION TO INSTALLATION

OBLIQUE PLATE MOUNT CONNECTION TO INSTALLATION

CORNER PLATE MOUNT CONNECTION TO INSTALLATION

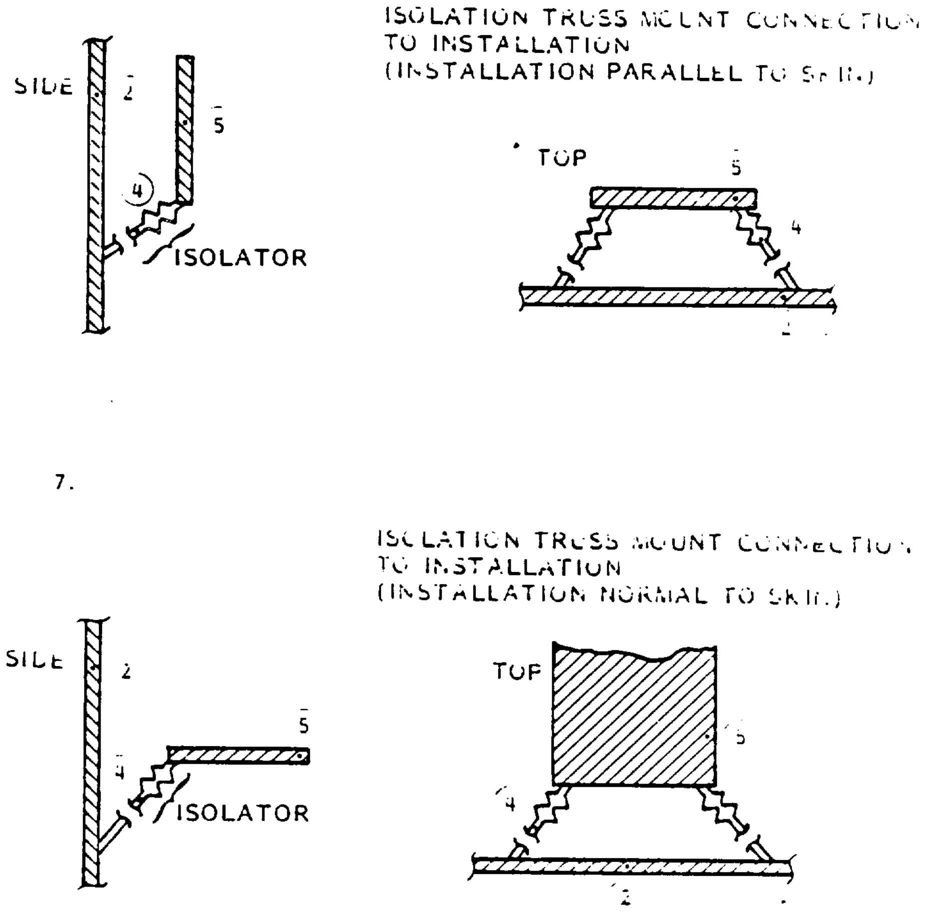
*THIS PARAMETER IS ENTERED INTO THE PARAMETER LIST OF SEA ELEMENT INST

Figure 11-5j

Figure 11.17 MONT SEA element TYPJ = 3., 4. and 5.

MONT DETAIL (CONTINUED)

TYPJ =6.


SIDE
2
5
4
ISOLATOR
ISOLATION TRUSS MOUNT CONNECTION
TO INSTALLATION
(INSTALLATION PARALLEL TO SKIN)
TOP
5
4
2
SIDE
2
5
4
ISOLATOR
ISOLATION TRUSS MOUNT CONNECTION
TO INSTALLATION
(INSTALLATION NORMAL TO SKIN)
TOP
5
4
2

7.

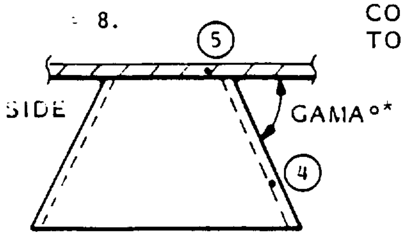
Figure 11.5k

Figure 11.18 MONT SEA element TYPJ = 6. and 7.

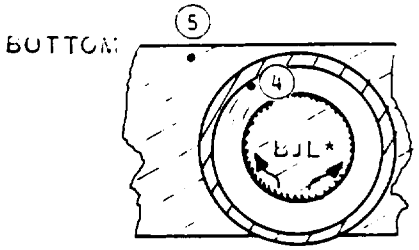
MONT DETAIL (CONTINUED)

TYPJ

8.



CONE MOUNT CONNECTION
TO INSTALLATION



9.

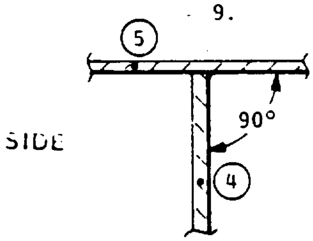
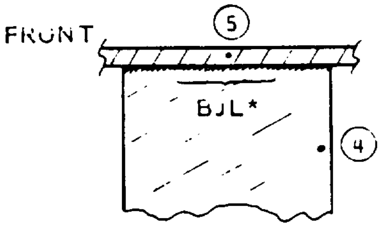
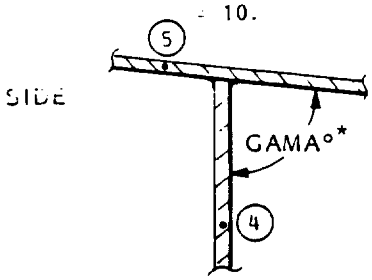


PLATE MOUNT CONNECTION
TO INSTALLATION



10.



*THIS PARAMETER IS ENTERED
INTO THE PARAMETER LIST OF
SEA ELEMENT INST

PLATE MOUNT
OBLIQUE CONNECTION
TO INSTALLATION

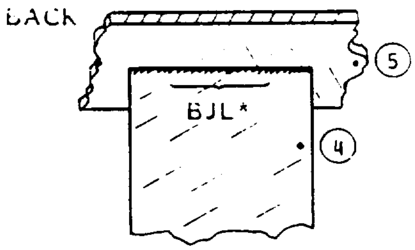


Figure 11-51

Figure 11.19 MONT SEA element TYPJ = 8., 9. and 10.

MONT DETAIL (CONTINUED)

TYPJ

11. BEAM MOUNT CONNECTION
TO INSTALLATION

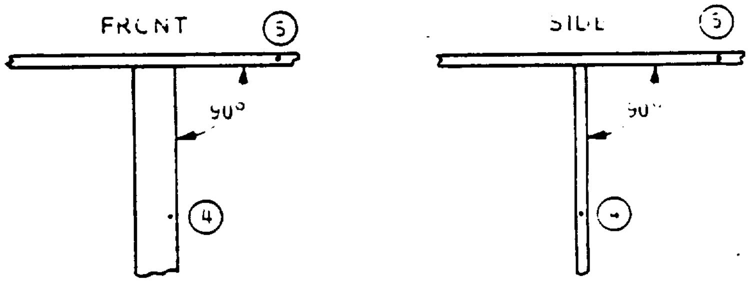


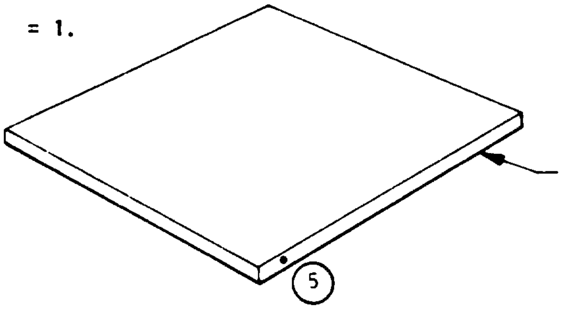
Figure 11-5m

Figure 11.20 MONT SEA element TYPJ = 11.

5 INST
(STRUCTURE)

TYPE

= 1.

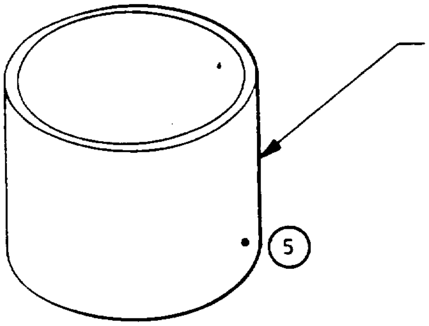


FLAT PLATE

PARAMETERS [UNITS]

TYPE	[ ]
ROW*	[MASS/(LENGTH**3)]
ASMS*	[MASS]
H	[LENGTH]
ALX*	[LENGTH]
ALY*	[LENGTH]
PATA*	[LENGTH]
BJL θ	[LENGTH]
AP	[LENGTH**2]
GAMA θ	[DEGREES]
E	[FORCE/(LENGTH**2)]
DLF	[ ]
CL	[LENGTH/TIME]

= 2.



CYLINDER

PARAMETERS [UNITS]

TYPE	[ ]
ROW*	[MASS/(LENGTH**3)]
ASMS*	[MASS]
H	[LENGTH]
D	[LENGTH]
BL	[LENGTH]
ALX*	[LENGTH]
ALY*	[LENGTH]
PATA*	[LENGTH]
BJLθ	[LENGTH]
AP	[LENGTH**2]
GAMAθ	[DEGREES]
E	[FORCE/(LENGTH**2)]
DLF	[ ]
CL	[LENGTH/TIME]

*SEE SKIN DETAIL (Figures 11-3c and 11 3d)

θSEE MONT DETAIL (Figures 11-5i through 11-5m)

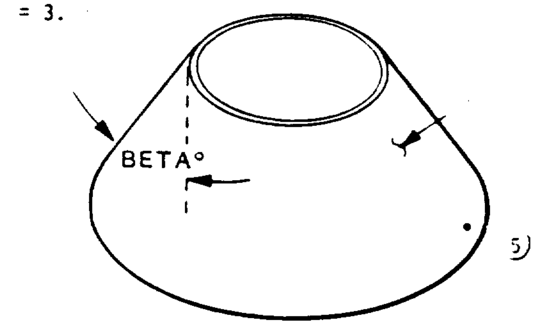
Figure 11-6a

Figure 11.21 INST SEA element, plate and cylinder.

INST (CONTINUED)

TYPE

= 3.

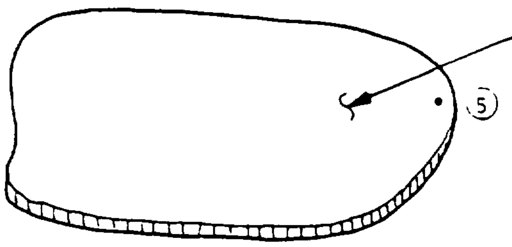


CONE

PARAMETERS [UNITS]

TYPE	[ ]
ROW*	[MASS/(LENGTH**3)]
ASMS*	[MASS]
H	[LENGTH]
D	[LENGTH]
BL	[LENGTH]
ALX*	[LENGTH]
ALY*	[LENGTH]
PATA*	[LENGTH]
BJL"	[LENGTH]
AP	[LENGTH**2]
BETA	[DEGREES]
GAMA"	[DEGREES]
E	[FORCE/(LENGTH**2)]
DLF	[ ]
CL	[LENGTH/TIME]

= 4.



EXTRAP II INSTALLATION

PARAMETERS [UNITS]

TYPE	[ ]
ROWS*	[MASS/(LENGTH**3)]
ASMS*	[MASS]
H	[LENGTH]
AP	[LENGTH**2]
DLF	[ ]

* SEE SKIN DETAIL (Figures 11-3c and 10-3d)

" SEE MONT DETAIL (Figures 11-5i through 11-5m)

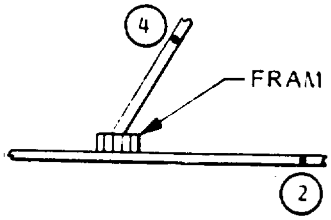
Figure 11-6b

Figure 11.22 INST SEA element, cone and EXTRAP II.

FRAM
(STRUCTURE)

TYPE
= 1.

BEAM FRAME CONNECTION
INTERFACE BETWEEN THE
MOUNT AND THE SKIN



PARAMETERS [UNITS]

TYPE	[ ]
XTYP	[ ]
ROW*	[MASS (LENGTH**3)]
H	[LENGTH]
B	[LENGTH]
BL	[LENGTH]
DI	[LENGTH]
DO	[LENGTH]
RGF	[LENGTH]
A	[LENGTH**2]
CL	[LENGTH/TIME]

* SEE SKIN DETAIL (Figure 11-3c)

Figure 11-7

Figure 11.23 FRAM SEA element.

11.2.2. NON-DIMENSIONAL PARAMETERS

VAPEPS computes values of non-dimensional parameters for each SEA element. These non-dimensional parameters are used for searching out existing models. The SERCH processor, however, searches for integer non-dimensional parameter values. Thus, the values of the non-dimensional parameters are actual computed as integer codes which represent the real value of the non-dimensional parameters. This is best explained by way of example.

Example 1

Suppose that the real value of a non-dimensional parameter was 0.0015983. VAPEPS would convert this real number to the integer code -5159. The sign and thousands digit indicate how many places and which direction to move the decimal point. The hundreds, tens and ones digits of the code are the three most significant digits of the real value. So the integer code 05159 represents the real number 159E-5 or .00159.

Example 2

What is the integer code for the real value 15.933.

-1159 (159E-1)

It may seem that this unnecessarily complicates the situation, but it should be noted that this procedure actually simplifies a number of tasks. Number one, integer searches are more reliable than real searches. Two, the display of the integer codes via the CHECK command in DAMO would not fit on the screen for interactive use unless the maximum length of each number was limited to five characters (four characters plus sign). And third, using integer codes reduces mass storage requirements. So while this may seem like an inconvenience, keep in mind that it is a necessary one. It should also be noted that the error in retaining only the three most significant digits is always less than one percent, which is insignificant when compared with the uncertainties associated with the parameters themselves.

Each SEA element has a maximum of four non-dimensional parameters which may be calculated. The actual number of non-dimensional parameters which will be calculated for each SEA element will be between one and four.

When the CHECK command is issued in DAMO a table will be displayed, assuming no errors are detected in the module. The table looks like this:

```
SECT TYPE CHAN   A      B      C      D     DESCRIPTIONS

EXTA   1    2  1621     0      0      0     REVERBERANT
```

```
   SKIN    2    6 -2251  -3154  -1101  -1428    CYLINDER
```

The first column indicates the SEA element. The second column is the type (from the TYPE parameter). The third column indicates the number of channels associated with the SEA element. The next four columns are for any non-dimensional parameters which apply to that SEA element. The last column, headed DESCRIPTIONS, contains the description that was assigned to the element in DAMO. Only those SEA elements which are present in the model (TYPE not equal to 0) are listed in the table.

This same table will be displayed by the DUMP command in the MODULES section of the SERCH processor. For more information refer to ELEMENT SEARCHING COMMANDS in the MODULE section of Chapter 7 (SEARCHING THE DATABASE).

The following table presents the formulae used to compute the values of the non-dimensional parameters for each SEA element.

Non-Dimensional Parameters

ELEMENT	TYPE	A	B	C	D
EXTA	Acoustic Space	AP**.5/ (V**(1/3))			
	Fluid Pressure	VEL/CO	FBL*VEL/VISC		
SKIN and INST*	Panel-like	H/(AP**.5)	RHO*AP**.5/RHOS	CO**2*AP**.5/ H*CL**2	
	Cylinder	H/(AP**.5)	RHO*AP**.5/RHOS	CO**2*AP**.5/ H*CL**2	AP**.5/ D
INTA	Acoustic Space	AP**.5/ (V**(1/3))			
MONT and FRAM	(Truss) Rectangular	H/BL	H*(E/(72*RHO))**.5/ (B*CL)		
	Circular	H/BL	(E/(2*RHO))**.5/CL	DO/BL	
	Tubular	H/BL	(E/(2*RHO))**.5/CL	DO/BL	
	Any section	H/BL	(E/(2*RHO))**.5/CL		
MONT	Plate and Cone	H/BL	(E/(2*RHO))**.5/CL		

* INST does not require RHOS to be input so non-dimensional parameter B is not computed.

11.3. DENS

DENS Command

Purpose

The DENS command calculates the modal density (per Hz) of each of the five SEA elements based on the system parameters (in the parameter matrix).

Command

```
DENS NU,VER,NFRQ,SF
```

Where

```
NU   - DAL unit containing the parameter matrix.
VER  - The version name of the parameter matrix.
NFRQ - The number of 1/3 octave bands.
SF   - The center frequency of the lowest 1/3 octave band.
```

Output

The following matrices are output:

```
NU,'DENS',VER = Modal density. (NFRQ X 5)
NU,'FREQ',VER = Frequency vector. (NFRQ X 1)
```

Notes

Use DAMO to input parameters or obtain directly from the VAPEPS data base by using RUN=GMOD.

The output of this command is used by the other prediction commands. The prediction commands that use this output are as follows:

```
TRNF   - Calculates the transfer functions.
PRDICT - Does theoretical and extrap predictions.
```

11.4. PRDICT

PRDICT Command

Purpose

This command predicts the response of a new design based on data from a baseline configuration found in the VAPEPS data base or alternately does a theoretical prediction based on input parameters.

Command

```
PRDICT NU1,VER1,NU2,VER2,IBACK
or
PRDICT NU2,VER2
```

Where

For case 1 - Extrapolated prediction.

NU1,VER1 - DAL unit and version name of baseline model.
NU2,VER2 - DAL unit and version name of new model (model that is to be predicted).
IBACK - A SEA element number indicating the excitation element for which an empirical transfer function is desired.

For case 2 - Theoretical prediction.

NU2,VER2 - DAL unit and version name of model to be predicted.

Input

The following elements are required to make a prediction.

Case 1 - Extrapolated prediction.

NU1,'EXC',VER1 - This element contains the space-averaged excitation data for the baseline system. The data must be mean-square values or dB per 1/3-octave. The dimensions of this element are NFRQ BY #EXC, where #EXC is the number of excitation elements.

NU1,'RSP ',VER1 - This element will contain space-averaged response data for the baseline system. The data must be mean-square values or dB per 1/3-octave. The dimensions of this element are NFRQ by #RESP, where #RESP is the number of response elements.

NU2,'EXC',VER2 - This element contains the space-averaged excitation data for the new system. The data must be mean-square values or dB per 1/3-octave. The dimensions of this element are NFRQ BY #EXC, where #EXC is the number of excitation elements.

NU1,'FREQ',VER1 - Output of DENS command.
NU2,'FREQ',VER2
NU1,'IDES',VER1 - Output of TRNF command.
NU2,'IDES',VER2
NU1,'TRNF',VER1 - Output of TRNF command.
NU2,'TRNF',VER2
NU1,'CONV',VER1 - Output of CNVRT command.
NU2,'CONV',VER2

Case 2 - Theoretical prediction.

NU2,'EXC',VER2 - This element contains the space-averaged excitation data for the system. The data must be mean-square values or dB per 1/3-octave. The dimensions of this element are NFRQ BY #EXC, where #EXC is the number of excitation elements.
NU2,'FREQ',VER2 - Output of DENS command.
NU2,'IDES',VER2 - Output of TRNF command.
NU2,'TRNF',VER2 - Output of TRNF command.
NU2,'CONV',VER2 - Output of CNVRT command.

Output

Both cases:

NU2,'RESP',VER2 - Predicted response for the new system in the same units as designated by CNVRT. (NFRQ X #RESP) #RESP is the the number of response elements.

Notes

In the case of an extrapolated prediction DENS, CNVRT and TRNF must be issued twice, once for the old system and once for the new system, prior to issuing the PRDICT command. For the theoretical prediction DENS, CNVRT and TRNF only need to be issued once for the model.

You can also use power inputs to each SEA element.

11.4.1. SUMMARY

This is a summary of input to the PRDICT command:

COMMAND ARGUMENTS	INPUT ELEMENTS (DIMENSIONS)	OUTPUT ELEMENTS (DIMENSIONS)
DENS NU,VER,NFRQ,SF	NU,'PARA',VER (30,6) From DAMO	NU,'DENS',VER (NFRQ,5) NU,'FREQ',VER (NFRQ,1)
TRNF NU,VER,JPA,JPI ,JEXC,JRES	NU,'FREQ',VER (NFRQ,1) NU,'PARA',VER (30,6) NU,'DENS',VER (NFRQ,5) OPTIONAL INPUT: NU,'COF',VER (#ITEMS,1) NU,'COD',VER (NFRQ,#ITEMS) NU,'ATAF',VER (#ITEMS,1) NU,'ATAD',VER (NFRQ,#ITEMS)	NU,'IDES',VER (10,1) NU,'TRNF',VER (NFRQ,#RES*#EXC) NU,'CO',VER (25,NFRQ) NU,'ATA',VER (25,NFRQ) NU,'FCFR',VER (2,1) NU,'INCO',VER (NEXC,NFRQ)
CNVRT NU,VER,ITYP ,JTYP,KTYP	NU,'FREQ',VER (NFRQ,1) NU,'PARA',VER (30,6)	NU,'CONV',VER (NFRQ,5) NU,'DBFG',VER (2,1) NU,'CPOW',VER (NFRQ,1)
PRDICT N1,VER1 ,N2,VER2,IBACK	Old system: N1,'FREQ',VER1 (NFRQ,1) N1,'IDES',VER1 (10,1) N1,'TRNF',VER1 (NFRQ,#RESP *#EXC) N1,'CONV',VER1 (NFRQ,5) N1,'EXC',VER1 (NFRQ,#EXC) N1,'RSP ',VER1 (NVRQ,#RESP) New system or theoretical: N2,'FREQ',VER2 (NFRQ,1) N2,'IDES',VER2 (10,1) N2,'TRNF',VER2 (NFRQ,#RESP *#EXC) N2,'CONV',VER2 (NFRQ,5) N2,'EXC',VER2 (NFRQ,#EXC) Optional input to both new and old system: NU,'POWF',VER (#ITEMS,1) NU,'POWD',VER (NFRQ,#ITEMS)	N2,'RESP',VER2 (NFRQ,#RESP)

11.4.2. EXAMPLE

This is an example of a theoretical prediction using PRDICT.

```
* INPUT PARAMETERS
DAMO 1,LWP1
EXTA
P* TYPE=1.,ROW=1.12E-7,AP=561868.,V=2198600.,AAC=.02,CO=1.32E4
LIST
SKIN
P* TYPE=1.,ROW=1.66E-4,ROWS=1.882E-5,ASMS=1.5645,H=.385,ALX=40.,ALY=44.
P* PATA=672.,AP=7040.,E=6.42E6,DLF=.1,CL=2.E+5
LIST
INTA
P* TYPE=1.,ROW=1.12E-7,AP=19008.,V=70400.,AAC=.05,CO=1.32E4
LIST
DONE
SET SFRQ=50.,NFRQ=24,HFRQ=10000.
*
* CREATE VECTOR WITH ACOUSTIC EXCITATION
ECOL 1,CBAY,ENVI &NFRQ,1
131.0,132.0,132.5,133.0,133.0,132.5,132.0,136.5,139.5,130.5,130.0,130.0
128.5,128.0,127.0,126.5,126.0,125.0,123.5,122.0,121.0,119.5,118.0,117.0
* CALCULATE MODAL DENSITY
DENS 1,LWP1,&NFRQ,&SFRQ
* CREATE FUNCTIONALIZED DAMPING FOR SKIN BASED ON .10 DAMPING AT
* 250 HZ AND DAMPING=K/FREQUENCY.
ECOL 1,ATAF,LWP1 1,1
22
FUN 1,FREQ,LWP1 1,ATAD,LWP1 1.,25. -1,1
* CREATE TRANSFER FUNCTIONS SINGLE SIDED EXCITATION
TRNF 1,LWP1,12,0,1,2
* DEFINE UNITS
CNVRT 1,LWP1,2,2,4
* EXCITATION VECTOR
CONVERT 1,CBAY,ENVI 1,EXC,LWP1 1,2
* DO THEORETICAL PREDICTION
PRDICT 1,LWP1
* CONVERT RESPONSE TO G**2/HZ THEN PRINT
CONVERT 1,RESP,LWP1 1,SKG2,LWP1 3,6 &SFRQ,&HFRQ
PRINT 1,SKG2
```

11.4.3. POWER INPUTS

To put a power input into any element in a model use the following method:

```
ECOL NU,'POWF',VER IROW,1
(input SEA element numbers that a power input is to used)
ECOL NU,'POWD',VER NFRQ,IROW
(input power inputs by columns)
```

Where

```
NU     = The dal unit number consistent with the model.
'POWF' = The element name.  Must be 'POWF'.
VER    = The version name consistent with the model.
IROW   = The number of power inputs desired.
'POWD' = The element name.  Must be 'POWD'
NFRQ   = The number of frequencies.  Consistent with the model.
```

11.4.3.1. EXAMPLE

Power input example:

```
ECOL 1,POWF,XMPL 2,1
2,3
ECOL 1,POWD,XMPL 16,2
16*3.5
16*-2.5
```

In this example the power input to SEA element number 2 is positive 3.5. The power input to SEA element number 3 is negative 2.5. The negative sign indicates that actually a power output has been specified. The units of the power should be consistent with those in the parameter matrix.

11.5. SCALE

SCALE Command

Purpose

Predicts the acoustically induced vibration response of a panel like structure based on empirical data obtained from a similar (baseline) structure. Corrections are made for differences in structural parameters and differences in acoustic excitation between the baseline and the new system.

Command

```
SCALE N1,IVN1 N2,IVN2 SEANO IUEXC,IURES
```

Where

```
N1,IVN1 - Dal unit and version containing elements for the baseline
          system.
N2,IVN2 - Dal unit and version containing elements for the new system.
SEANO   - Sea element number to predict (elements 2 and 5 are allowed).
IUEXC
IURES   - Codes indicating units of excitation and response
          respectively.  Allowable codes are:

            1 - psi**2
            2 - (n/m)**2
            3 - (m/sec**2)**2
            4 - g**2
            5 - (in/sec**2)**2
            6 - (ft/sec**2)**2
            7 - dB
```

Required input elements

```
N1,PARA,IVN1 - Parameters for the baseline system (generated by DAMO).
N1,EXC,IVN1  - Excitation for the baseline system.
N1,RESP,IVN1 - Response for the baseline system.
N1,FREQ,IVN1 - Center frequencies corresponding to the excitations and the
               responses for both the baseline and new systems.
N2,PARA,IVN2 - Parameters for the new system (generated by DAMO).
N2,EXC,IVN2  - Excitation for the new system.
```

Output element

```
N2,RESP,IVN2 - Response result for the new system.  Units are consistent
```

with those of the baseline response (as indicated by IURES). The response is in standard 1/3 octave bands. If a frequency shift has been made during the prediction, the new response will have zero values in certain frequency bands (see the equations for EXTRAP2 in the VAPEPS USER'S MANUAL).

11.5.1. INPUT

Summary of input needed for the SCALE command.

COMMAND ARGUMENTS	INPUT ELEMENTS (DIMENSIONS)	OUTPUT ELEMENTS (DIMENSIONS)
SCALE N1,VER1 N2,VER2 ,SEANO,IUEXC ,IURES	New system: N2,'PARA',VER2 (30,6) From DAMO N2,'EXC',VER2 (NFRQ,1) Old system: N1,'PARA',VER1 (30,6) From DAMO N1,'FREQ',VER1 (NFRQ,1) N1,'EXC',VER1 (NFRQ,1) N1,'RESP',VER1 (NFRQ,1)	N2,'RESP',VER2 (NFRQ,1)

11.6. TRNF

TRNF Command

Purpose

The TRNF command calculates transfer functions, a SEA model descriptor, a coupling loss factor matrix and a coefficient matrix for the governing energy balance equation.

Command

```
TRNF NU,VER,JPA,JPI,JEXC,JRES
```

Where

NU,VER - Dal unit and version name, respectively. These must be the same as those specified in the DAMO and DENS commands.

JPA - An optional string of numbers which specifies the SEA elements which are present and active in the model. Example: JPA = 345 indicates that elements 3, 4 and 5 (INTA, MONT and INST) are present and active. If not specified, the sign of the TYPE parameter for each SEA element will automatically be used to determine which elements are present and active.

JPI - An optional string of numbers which specify the SEA elements which are present and inactive. A zero in this column indicates that no elements are inactive. If not specified the sign of the TYPE parameter for each SEA element will automatically be used to determine which elements are present and inactive.

JEXC - A string of numbers which specifies which SEA elements supply excitation energy.

JRES - A string of numbers specifying which SEA elements are response elements.

Output

Output from TRNF consists of the following elements:

NU,'IDES',VER - This element contains descriptor flags. A modified form of this element is printed out after 'FLAGS =' when the TRNF command is issued.

NU,'TRNF',VER - This element contains the transfer function(s), one row per frequency and one column per transfer function.

NU,'CO',VER - This element contains the coefficient matrix (see illustration in VAPEPS USER'S MANUAL).

```
NU,'ATA',VER  - This element contains the coupling loss factor matrix
                (the illustration also applies here).
NU,'FCFR',VER - This element contains the critical frequency in row 1
                and the ring frequency in row 2.
NU,'INCO',VER - This element contains the inverse of the coefficient
                matrix.  It only includes the non-excitation elements.
                The user will most likely never be concerned with the
                contents of this element.
```

Notes

DAL elements required for the TRNF command are the parameter element from the DAMO command, the modal density element from the DENS command and the 1/3 octave center frequency element from the DENS command.

The option exists to input your own values for the coupling loss factors or coefficients.

The output of this command is used by the other prediction commands. The prediction commands that use this output are as follows:

PRDICT - Does theoretical and extrap predictions.

11.6.1. ATA_CO

To set coupling loss factors:

```
ECOL NU,'ATAF',VER IROW,1
(input the index values of coupling loss factors to be set)
ECOL NU,'ATAD',VER NFRQ,IROW
(input the coupling loss factors column by column)
```

Where

```
NU     - The dal unit, consistent with previous uses.
'ATAF' - The element name.  Must be 'ATAF'.
VER    - The version name.  This must be consistent with previous
         uses.
IROW   - The number of coupling loss factors to be pre-set.
'ATAD' - Another element name.  This must be 'ATAD'.
NFRQ   - The number of 1/3 octave bands.  This must be the same
         value of NFRQ as used in the DENS command.
```

To preset terms in the coefficient matrix, use the same procedure as above but substitute the element names COD and COF for ATAD and ATAF.

11.6.2. EXAMPLES

Use of the TRNF command.

```
TRNF 1,XMPL,1235,4,12,5
```

This command would access data on DAL unit 1, version name XMPL. SEA elements 1, 2, 3, and 5 are present and active, and element 4 is present and inactive. Elements 1 and 2 are excitation and element 5 is response. Element 3 is also responding to the excitation specified but the magnitude of this response is not being requested.

Use of coupling loss factor input.

```
ECOL 1,ATAF,XMPL,3,1
25,52,13
ECOL 1,ATAD,XMPL,16,3
16*0.1
.1,.2,.3,13*.5
.8,.7,.5,.6,12*.9
```

In this example, terms (2,5), (5,2) and (1,3) have been preset. Each term covers sixteen frequency bands, in this case.

12. TRANSPARENT COMMANDS

This section deals with a special set of commands that are rarely 'needed' by the one-command-at-a-time type DALPRO user. On the other hand they are commands that the advanced user can hardly live without. Their purpose is to provide the user with additional control over a DALPRO execution. They allow the user to make conditional branches in a runstream, invoke 'do-loops', operate on variables, and reset processing modes.

Transparent commands differ from regular commands in that they must be prefaced by a pair of transparent command characters. These characters must be the first two non-blank characters in the command line. While cumbersome to type, these characters are necessary in that they tell DALPRO that the card image is special and must be treated accordingly. The use of the control characters allows the commands to be used virtually anywhere in the runstream. This includes their use in sub-processors such as CALC and BUILD, as well as embedded in data such as that required by CHOP and ECOL. The only time they may not be used is when data is being read according to a defined FORTRAN format (eg. SREAD).

Valid transparent commands are listed in the following table. Note that each command is assigned one or more types. These types indicate the type of control the command affects. A list of other transparent control character pairs is included at the end of the table.These aren't really commands but are used with transparent-type processing. For more information about about a particular command, see the proper section following the table.

Note that several commands contain arguments enclosed in parentheses (). These are optional entries that may or may not be used depending on the occasion. When they are used, do not enclose them in parentheses.

TRANSPARENT COMMANDS	TYPE(S)
##ADD N1,EL1,VR1(,IC1,IC2)(,ICOL)	ORS
##ASK LABL @ PROMPT STRING	ORS,J
##CALC any one-line calc operation	V
##CCHAR '$$','##','++','>>'	RSC
##CLOSE NU	ORS
##COMMENT '*'	RSC
##CONTINUE ';'	RSC
##CYCLE IC1,IC2 (JC1,JC2)	SPM
##GET NAME @ PROMPT STRING	ORS,V
##INC NAME,INCR	ORS,V
##JCAS IVAL LAB1,LAB2,LAB3,....	J
##JEQZ VALUE,LABL	J

##JGEZ	VALUE,LABL	J
##JGTZ	VALUE,LABL	J
##JLTZ	VALUE,LABL	J
##JLEZ	VALUE,LABL	J
##JNEZ	VALUE,LABL	J
##JEQV	VAL1,VAL2 LABL	J
##JGEV	VAL1,VAL2 LABL	J
##JGTV	VAL1,VAL2 LABL	J
##JNEV	VAL1,VAL2 LABL	J
##JLTV	VAL1,VAL2 LABL	J
##JLEV	VAL1,VAL2 LABL	J
##JUMP	LABL	J
##GxxZ	VALUE,LABL	RJ
##GxxV	VAL1,VAL2 LABL	RJ
##OFF	COMMENT,ECHO,FATAL,SPIT	SPM
##ON	COMMENT,ECHO,FATAL,SPIT	SPM
##OPEN	NU,'filename'	ORS
##READ	NU (ELN,VER (IC1,IC2)) (IEOF)	ORS
##READ	THROUGH	ORS
##SET	NAME,VALUE	V
##SET	NAME,NU,ELN,VER (IC1,IC2) IROW,ICOL	V
##SET	NAME,NU,ELN,VER (IC1,IC2) ITEM	V
##VRES	N1,EL1,VR1	V
##VSAV	N1,EL1,VR1	V

J	jump
RJ	rewind and jump
ORS	operate read stack
RSC	reset special character
SPM	set processor mode
V	variables

OTHER TRANSPARENT CHARACTER PAIRS

$$CARD IMAGE	INVOKE PRECODER
++XXXX	SAME AS ##XXXX
>>LABL	LABEL

12.1. ADD

##ADD Command

Purpose

Read the appropriate column from the indicated element as if it were a card image. Note that the element should be a type 4 element (as produced by ECOSYM). This command is useful in reading plot titles.

Command

```
##ADD N1,EL1,VR1 (,IC1,IC2) (,ICOL)
```

Where

```
N1,EL1,VR1 - DAL unit, element and version of element to add.
IC1,IC2    - Cylce numbers of element to add.
ICOL       - Column of element to add. (Default = 1).
```

12.2. ASK GET

##ASK ##GET Commands

Purpose

These commands read one line of data from the previous device on the read stack. If they appear in a run-stream, they will read from the unit that initiated the runstream. If the input device turns out to be the user terminal, the user is prompted. For ASK the user is expected to enter a YES or NO (Y or N). If NO is input, a jump is activated to the label LABL. See jumps below for details on labels. For GET, the user is expected to enter a single data value. The variable (ie NAME) is set to the input value. Note that NAME is the name of a variable and must not be prefaced by an ampersand. For both commands, the @ sign is required and it must be placed as the 12th character in the card image. See ##READ 0 and ##READ THROUGH for details on how to control where the input should come from.

Command

```
##ASK LABL @ Prompt string
##GET NAME @ Prompt string
```

Where

LABL - Label to jump to if the answer is no.
NAME - Variable to assign input value to.

12.3. CALC

##CALC Command

Purpose

Perform any one line CALC operation.

Command

```
##CALC  any one-line calc operation
```

See the regular CALC command for detailed documentation of CALC capabilities. Any one-line operation can be placed on a ##CALC line, eg

```
##CALC  I2 = 2*I
```

12.4. CCHAR

##CCHAR Command

Purpose

Changes transparent character pair values. Four entries should always be input and each entry should contain two characters enclosed in single quotes. The entries are order dependant according to mode number:

Command

```
##CCHAR '$$','##','++','>>'
```

Where

MODE	DEFAULT	USAGE
1	$$	Invoke precoder
2	##	Transparent command or label
3	++	Transparent command or label
4	>>	Label (preferred method for label)

12.5. CLOSE

##CLOSE Command

Purpose

Serves to close/rewind a FORTRAN unit. If NU is negative, the unit is rewound. If NU is positive, the unit is rewound and closed. On most systems the two are equivalent. On VAX, a close operation de-attaches the unit so that subsequent writes will create the next higher version.

Command

```
##CLOSE NU
```

Where

NU - Fortran unit to close.

12.6. COMMENT

##COMMENT Command

Purpose

Resets the comment character. The default is an asterisk (*). Note that all comments must have the appropriate comment character as the first chara`ter in the line. Comment lines will be ignored. If the comment mode is 'on' (see ON/OFF) the line will be printed on unit 6.

Command

```
##COMMENT '*'
```

Where

'*' - Character to designate comment line. Default is '*'.

12.7. CONTINUE

##CONTINUE Command

Purpose

Turns on/off multiple records per line mode and establishes the character to be used for separating logical records.

Command

```
##CONTINUE  ';'
```

Where

';' - Character used to designate new line.

Example

```
##CONTINUE  ';'
```

would allow lines of the form

```
CALC ; I=I+1 ; DONE
```

The default for continue is ##CONTINUE -1 This has the effect of turning off the logical record mode. Note that use of this feature is not really encouraged, but it is there if you want it.

12.8. CYCLE

##CYCLE Command

Purpose

This command is the same as the regular CYCLE command but it has the advantage of being used anywhere a transparent command can be used. It is especially handy in the BUILD command.

Command

```
##CYCLE IC1,IC2 JC1,JC2
```

See the CYCLE command for further description.

12.9. ECHO

##ECHO Command

Purpose

Controls what echoing is performed and when it is turned on.

Command

```
##ECHO  level,type
```

Where

```
level - When you ##READ another symbolic dataset, level controls
        whether that will be echoed.
      < 0 - Never echo added runstream (turn mode off)
      = 0 - Echo only if echo mode is on prior to ##READ
      > 0 - Always echo. Eg turn echo mode on for added streams
type  - Controls what is echo'ed. Normally you want to echo whatever
        was input, which is the default. There is a presym section to
        line input that left adjusts all ## lines, etc. You can use
        type=2 to echo the presym'ed line instead of the raw input
        line.
      = 3 - Give list of how command was decoded.
```

12.10. INC SET

##INC ##SET Commands

Purpose

These are essentially one-line CALC commands that either set a variable or increment a variable by some value. ##SET will work on any type of variable. NO TYPE CONVERSION WILL EVER BE PERFORMED (SET and CALC will do conversions). See the regular DALPRO SET command for details. Note that the SETFL form may be used even though the command is still ##SET. Also note that only one variable may be 'set' per command (as opposed to the SET command where a group of variables may be set). Note that both commands require a NAME for the first argument. Do not precede the name by an ampersand. For ##INC, both integer and real variables may be incremented. The value (second item) determines the type of addition performed. Thus if you have a real variable, make sure you include a decimal point on a constant value, or use a real type variable.

```
##INC  I,1
```

is preferable (cheaper) than

```
CALC
I = I + 1
DONE
```

Command

```
##INC NAME,INCR
##SET NAME,VALUE
##SET NAME,NU,ELN,VER (IC1,IC2) IROW,ICOL
##SET NAME,NU,ELN,VER (IC1,IC2) ITEM
```

Where

```
NAME         - Variable to modify.
INCR         - Amount to increment variable by.
VALUE        - Value to set variable to.
NU,ELN,VER   - DAL unit, element and version name to get data from  to
               set variable to.
IC1,IC2      - Cylce numbers of NU,ELN,VER.
IROW,ICOL    - Row and column to get value from NU,ELN,VER.
ITEM         - Dataset parameter as defined in CALC.
```

12.11. JUMP

##JUMP Commands

Purpose

Branch to a label depending on test.

Command

```
##JCAS  IVAL  LAB1,LAB2,LAB3,....
##JEQZ  VALUE,LABL
##JGEZ  VALUE,LABL
##JGTZ  VALUE,LABL
##JLTZ  VALUE,LABL
##JLEZ  VALUE,LABL
##JNEZ  VALUE,LABL
##JEQV  VAL1,VAL2  LABL
##JGEV  VAL1,VAL2  LABL
##JGTV  VAL1,VAL2  LABL
##JNEV  VAL1,VAL2  LABL
##JLTV  VAL1,VAL2  LABL
##JLEV  VAL1,VAL2  LABL
##JUMP  LABL
##GxxZ  VALUE,LABL
##GxxV  VAL1,VAL2  LABL
```

If a variable is to be tested (anything else is pointless) the variable name must be prefaced with an ampersand (&). LABL may have up to 4 characters and must start with an alphabetic character. When a jump is signaled, all cards are ignored until one of the following cards are encountered:

```
>>LABL    ##LABL    ++LABL
>>MASK    ##MASK    ++MASK
```

where the above assumes that the transparent character pairs have not been changed. Note that the >> form is preferable since the label cannot be confused with a transparent command.

Unconditional jump

```
##JUMP
```

Jumps based on a value compared to zero

```
##JEQZ, ##JGEZ, ##JGTZ, ##JNEZ, ##JLTZ, ##JLEZ
```

Jumps based on the comparison of two values

```
##JEQV, ##JGEV, ##JGTV, ##JNEV, ##JLTV, ##JLEV
```

These jumps are all conditional, and are similar to the above commands except that two values are used. The first value is compared to the second value (rather than with 0, as before). ie. ##JGTV &X,&Y LABL would cause a jump to LABL if X > Y.

Backward jumps

```
##GEQZ, ##GNEZ, ##GGEZ, ##GGTZ, ##GLTZ, ##GLEZ, ##GUMP
##GEQV, ##GNEV, ##GGEV, ##GGTV, ##GLTV, ##GLEV
```

Mimic the operation of their ##J counterparts, but perform a REWIND of the current input device prior to jumping. The net effect is a backwards jump. The only restrictions are that you can only jump within the same dal dataset or FORTRAN unit.

Case jump

```
##JCAS  IVAL  LAB1,LAB2,LAB3,...
```

Invokes a jump to one of the labels, depending on the value of IVAL. ie. IVAL=3, would jump to the third label. If IVAL is non-positive or greater than the number of labels, no jump is performed.

12.12. OFF ON

##OFF ##ON Commands

Purpose

Turn on/off one or more of the following modes. Note that three default values are given. These are for interactive, batch, and RUN command modes respectively.

Command

```
##OFF  MODE1(,MODE2(,MODE3(,MODE4)))
```

Where

MODEi - One or more from the following list.

MODE	DEFAULT	DESCRIPTION
COMMENT	off-on -off	ON - Comments will be printed
		OFF - Comments won't be printed
ECHO	off-on -off	ON - User input is echoed
		OFF - User input is not echoed
FATAL	off-on -on	OFF - Errors result in 'TRY AGAIN' or 'SAME TO YOU'
		ON - Errors cause execution to be terminated.
SPIT	on -on -off	ON - Summary info (spit line) will be printed following most commands.
		OFF - Summary info will not be printed

12.13. OPEN

##OPEN Command

Purpose

Opens a FORTRAN unit by name. This is available on most systems, but not on all systems. On some systems there is a distinction between print files and non-print files. If you use the filename 'PRT$' then the unit will be opened as a print file. The PRT$ option currently works only on the Sperry system (both Fieldata and Ascii versions)

Command

```
##OPEN  NU,'filename'
```

Where

```
NU         - Fortran unit to use for file.
'filename' - Name of file to be associated with NU.
```

12.14. READ

##READ Command

Purpose

Directs DALPRO to read subsequent commands and/or data from either a FORTRAN unit or from a DAL element (types 5 or 6 only) Type 5/6 elements may be created in SYMIN or CSYM. The item IEOF controls what happens when the end of the file or element is reached. If IEOF>1, the unit is rewound and re-read IEOF-1 times. ie:

Command

```
##READ  NU (ELN,VER (IC1,IC2)) (IEOF)
##READ  THROUGH
```

Where

```
NU          - Fortran unit to read from.
NU,ELN,VER  - DAL unit element and version to read from.
IC1,IC2     - Cycle numbers of NU,ELN,VER
IEOF        - Option flag.
```

Example

```
##SET LOOP=3
##READ  1,DAL,SYM  &LOOP
##READ  1  &LOOP
```

would cause the DAL element DAL/SYM from DAL unit 1 to be read 3 times and then FORTRAN unit 1 to be read 3 times.

If IEOF=0, an end of file will result in an EOF status being returned to DALPRO. This is useful in commands like EZOUT and ECOSYM. If the end of file is sensed while in regular DALPRO control mode, DALPRO exits normally (it is equivalent to the DALPRO END command)

##READ ... DATA

When IEOF is set to DATA, the indicated stream is accessed, but not read. The data is read only when a ##READ DATA is encountered later in the runstream. This may be used to maintain position in a dal stream. eg

```
##READ  1,TEST,SYM  DATA
*
ECOSYM  1,TITL,LINE  80
```

```
      ##READ DATA AUTO
      *
      RECEIVE  NR,NC
      ##READ DATA AUTO
      *
      ECOL  1,DATA,MAT  &NR,&NC
      ##READ DATA,AUTO
```

##READ ... AUTO

is shown above in an example. It causes the indicated stream to be read until the current command is completed. Input then returns to the previous stream. This is very useful for processing different input sections using different commands, as shown above.

##READ 0 IEOF

A special case exists when FORTRAN unit 0 is referenced. This tells DALPRO to read from the unit that was stacked just prior to this one. It is used to allow a runstream (eg a RUN command) to read data from the calling runstream. For ##READ 0, IEOF is the number of cards to read from this earlier device. Once the cards have been read, reading resumes right after the ##READ 0. If IEOF is 0, reading will continue until and end-of-file is sensed, or until ##OFF READ is encountered.

##READ THROUGH

This is a special command that controls whether the currently stacked unit is eligible for ##READ 0's that may occur later. The default mode is that all units are eligible for 'dropped' reads. By issuing ##READ THROUGH, you cause any future ##READ 0's to read through this unit and go on to the previous unit. The command ##READ THROUGH -1 may be used to turn on the unit's eligibility after it was turned off. An example of where ##READ THROUGH is needed is when a runstream repeatedly prompts a user to enter information. Ie. a loop element is created and then a ##READ is done to add the element many times. Each time through, the element does an ECOL or GCOL followed by a ##READ 0. Ie the actual data for the ECOL is meant to come from whoever called the original runstream. Normally, ##READ 0 just reads from the previous device on the stack, which in this case is the runstream element. By including a ##READ THROUGH in the runstream prior to invoking the loop, the subsequent #READ 0 will pass down to the correct place, bypassing the runstream element. If you

```
   PRINT  30,RUN,INDE
```

you will see the logic behind RUN=INDEX. It performs the operations described above. Note that ##GET and ##ASK essentially perform ##READ 0's and thus may be controlled by ##READ THROUGH's.

12.15. VRES VSAV

##VRES ##VSAV Commands

Purpose

Restores or saves the variable table. VSAV takes the internal variable table and saves it on the specified element. At the same time it also saves the status of ON/OFF modes (EG. COMMENT,ECHO,FATAL,SPIT). VRES reads the specified element and overwrites the internal variable table with its contents. If the unit number on the VRES command is input as a negative number, the previously mentioned ON/OFF states will be reset to whatever value they had at the time of the original VSAV.

Command

```
##VRES  N1,EL1,VR1
##VSAV  N1,EL1,VR1
```

Where

N1,EL1,VR1 - DAL unit, element and version name of element to save or restore variables from.

12.16. PRECODER

The Precoder

Precoding is an operation that scans a card image and converts all variables to symbolic form. It can best be explained by an example. Consider plot titles. They are hollerith images that are printed on plots. If the card image

```
MODE &I --- FREQUENCY = &F
```

is input as a title, the line will be printed exactly as input. If the precoder is invoked:

```
$$MODE &I(I3) --- FREQUENCY = &F(F6.2)
```

then the line will be converted to

```
MODE  10 --- FREQUENCY =   34.59
```

Thus the $$ is removed and all variables are encoded according to the format provided. Note that all variables must be followed immediately by a format. This format must be enclosed in parentheses!

IMPORTANT NOTE

Precoder formats are limited to 12 characters. Eg 10 characters plus beginning and ending parenthesis.

```
$$TEST &I(I3)    is legal
$$&I('TEST ',I3) is not
```

13. VARIABLES AND SYMBOLS

Variables and symbols are used in VAPEPS to create more flexible runstreams. Variables can be set using the SET command as well as a number of other commands. Symbols are set by using the := operator.

Example

```
SET I=20
SI50 := SET I=50
LIST I
LISTS
SI50
LIST I
```

In this example the variable I is set to 20. The symbol SI50 is equated to the string 'SET I=50'. The LIST I command will display the value of variable I. At this point it is 20. The LISTS command will list all the symbols. The command SI50 is a symbol so it is translated to be SET I=50 and sets the variable I to 50. The last LIST I will then show that the values of I is now 50. The best way to see this is to type in the above example and observe the results.

Symbols can also be substituted in the middle of a line by surrounding them with " characters.

Example

```
A := 1,A,A
B := 1,B,B
C := 1,C,C
MULT "A" "B" "C"
```

The MULT statement above is equivalent to typing:

```
MULT 1,A,A 1,B,B 1,C,C
```

Symbols can provide a short cut for frequently typed strings which are long or difficult to type. A good example is a list of channel names (e.g. CHANLIST := A1,A2,A12,A32,A128,A20) When ever you wanted to input that list of channels just type "CHANLIST".

13.1. LIST

LIST Command

Purpose

List all variables in run by name and value.

Command

```
LIST  NAM1,NAM2,...
```

Where

NAM1,NAM2,... Are variables to be listed. If none are specified, all variables are listed.

Notes

Variable names should not be prefaced by (&).

13.2. SET

SET Command

Purpose

Define or redefine a named variable. The SET command sets named variables to constants, or to the value of other named variables. The SETFL command sets one named variable equal to a particular term in an existing DAL element or table of contents line.

Command

```
SET NAM1=VAL1(,NAM2=VAL2,NAM3=VAL3,...)
```

Where

```
NAM1 (NAM2,NAM3,etc) are hollerith names of named variables. They are
                     not preceeded by an &.
VAL1 (VAL2,VAL3,etc) are any valid VAPEPS decodeable words. If named
                     variables are used, they must be prefaced by an &.
```

Notes

All values assigned to real variables (1st character A-H,O-Z) are normalized prior to being saved in the variable. No conversions are done in the case of integer variables. For this reason, all variables used to contain hollerith names must be of integer type.

13.3. VAR

VAR Command

Purpose

Remove all or a portion of the variables from the variable table

Command

```
VARR  NAM1,NAM2,.....,NAMN
```

Remove the named variables from the variable table. If no variables are named, no action is taken. If a non- existent variable is specified, the variable name is ignored.

```
VARS  NAM1,NAM2,.....,NAMN
```

Remove all variables, except for those specifically named in the command. If no variables are named, the variable table is completely cleared. If a non-existent variable is named, an error occurs, and no action is taken.

14. LOOPS AND RUNSTREAMS

This section discusses the use of VAPEPS as a high level programming language. The reader is assumed to be familiar with the basic format of VAPEPS commands and be able to use many of the commands without refering to the manual. In particular the reader should be familiar with the use of variables. If you aren't, please look at the CALC command and check the INDEX for commands involving variables.

In many tasks, the user is faced with doing the same operation or group of operations repeatedly. Repeated input of commands is a good way to improve typing but is rather boring and prone to error. As an example we will develop a method to print all real elements in a DAL file.

```
* FIRST GET A TABLE OF CONTENTS AND SAVE IT
*
* ASSUME WE ONLY WANT NEW ELEMENTS IE SINCE 8/1/80
*
FINES  1,MASK,MASK,MASK,MASK,800801
28 TOC,LIST
*
* NOW TRANSPOSE IT, AND SEARCH FOR TYPES 1 AND -1
*
TRAN 28 TOC,LIST  28,TOC,TRAN
*
SEARCH  28,TOC,TRAN  28,ELT,LIST  9 2
1,-1
*
*  NOW CHOP THE MATRIX TO INCLUDE ONLY THOSE
*  ELEMENTS WE HAVE FOUND
*
*  TO BE SAFE, WE SHOULD CHECK MCH$ FROM SEARCH
*  TO MAKE SURE WE REALLY FOUND ANY?
*
##JEQZ  &MCH$  NONE
*
* APARENTLY WE DID, SO CHOP
*
CHOPX  28,TOC,TRAN  28,GOOD,ELTS  &MCH$,4
= 28 ELT,LIST
10,11,12,13
*
* NOW PRINT THEM ALL. THE ONLY WAY IS TO ISSUE
* A PRINT COMMAND FOR EACH ONE. IN ADDITION, WE
*HAVE TO ACCOUNT FOR ALL THE DIFFERENT CYCLE NUMBERS.
*
* THE SOLUTION IS TO CREATE A LOOP ELEMENT. THE
* COMMAND SYMIN WILL BE USED FOR THIS PURPOSE.
* PLEASE NOTE THAT WE ARE ONLY CREATING A LOOP
* ELEMENT, WE ARE NOT EXECUTING IT YET.
*
SYMEND  28,PRIN,LOOP  MASK MASK  SSSS
CALC
* INCREMENT ELEMENT COUNTER
I = I + 1
* RETRIEVE ELEMENT NAMES AND CYCLES
NAME = (28 GOOD ELTS &I,1)
NAMV = (28 GOOD ELTS &I,2)
IC1  = (28 GOOD ELTS &I,3)
IC2  = (28 GOOD ELTS &I,4)
*EXIT CALC
DONE
*
* ISSUE APPROPRIATE CYCLE COMMAND
*
CYCLE  &IC1 &IC2
*
*  WRITE A 1H1 FOR A NEW LINE, THEN PRINT
*
WRITE 6
(1H1)
*
*
PRINT  1,&NAME,&NAMV
*
* END OF LOOP
SSSS
*
* NOTE THAT SYMEND ALLOWS US TO SPECIFY AN
* END SENTINEL. IT WAS 'SSSS' THUS 'SSSS'
* INDICATED THAT THE DECK WAS FINISHED
*
*  THE NEXT STEP IS TO INITIALIZE ANY VARIABLES
*  IE: I
*
SET I=0
*
*  NOW READ THE LOOP &MCH$ TIMES
*
* FIRST GET RID OF THE SPIT LINES ETC
*
##OFF SPIT,COMMENT,ECHO
*
##READ  28,PRIN,LOOP  &MCH$
*
*  THIS STREAM WILL WORK FOR ANY NUMBER OF ELEMENTS
*
```

```
    ##NONE
    * THE ABOVE LABEL WOULD CATCH A JUMP IF NO ELEMENTS WERE
    * FOUND
```

The real secret to looping is the use of SYMIN, and also the use of ##READ. This is the end of the section. I hope you get the general idea. The best way to learn is to do it yourself. If you are lost, please type in the above stream and add any writes or lists that would help you.

Example 2 - A plotting loop

This example will plot each column in a matrix, each on a separate page.

```
*
*  First create some dummy elements to plot.
*
random 28,rand,data 30, 10
ecol 28,rand,nams 10,1
R1:R10
frequency/center 28,rand,freq 10.,10000.
*
* Now create the loop.
*
symend 28,plot,loop wxyz
*
* Increment column pointer.
##inc i,1
*
* Get name of vector.
calc
   name = (28,rand,nams &i,1)
   done
*
* Get the ith column.
chopx 28,rand,data 28,plot,vec &nr, 1
1:&nr
&i
*
* Now plot data.
plot 28,rand,freq 28,plot,vec 13331
FREQUENCY, (HZ)
RANDOM DATA
10.,10000.
0.,1.
PLOT OF RANDOM DATA POINTS
*
* Use precoder to create title containg the name of the vector.
$$VECTOR &NAME(A4)
DEMONSTRATION PLOTTING LOOP
```

```
1,1
*
* End of loop
WXYZ
*
* Get sizes.
calc
  nr = (28,rand,data -6)
  nc = (28,rand,data -7)
  done
*
* Set column pointer.
set i = 0
*
* Set plot type
pset tek
*
* Read loop nc times
##read 28,plot,loop &nc
```

14.1. DEFAULT

DEFAULT Command

Purpose

Conditionally define the value of a variable. If the variable already exists, the default value is ignored.

Command

```
DEFAULT VAR1=VAL1 VAR2=VAL2 ... VARn=VALn
```

Where

```
VARi = The name of a variable (no &).
VALi = The default value for the variable.
```

Example

Look at any run stream on DAL unit 30. (Prinz 30,RUN,...)

Notes

The DEFAULT command is usually used in conjunction with runstreams.

Related Commands

RUN, RECOVER, SET, LIST

14.2. RECOVER

RECOVER Command

Purpose

Retrieves the values of arguments that were used in a call to a command stream via the RUN command.

Command

```
RECOVER  NAME1,NAME2,NAME3,NAME4, .......
or
RECEIVE  NAME1,NAME2,NAME3,NAME4, .......
-DATA-
or
RECIPE   NAME1,NAME2,NAME3,NAME4, .......
-Prompt-
-DATA-
```

Where

```
NAME1,NAME2,NAME3,NAME4, .......
          - List of variable names to place recovered values.
-DATA-    - Input list of values to put in the variable list.
-Prompt-  - Prompt string to use when reading input.
```

RECOVER

Set the variables named in the command to the values that were present on the RUN command. The variable whose name is NAME1 is set to the first value from the RUN command, etc.

RECEIVE

Set the variables in the variable list to the values in the following line.

RECIPE

Set the variables in the variable list to the values in the line following the prompt line. Whatever text is input will be used to prompt for the variables.

Notes

The passing of arguments from the RUN commands is handled as follows:

```
RUN=NAME  ARG1,ARG2,ARG3,ARG4
```

causes the element 28:PARM/LIST-0,0- to be created as a vector 4 words long. The proper run stream is accessed and added as data to the program.

The statement

```
RECOVER  NAME1,NAME2,NAME3,NAME4
```

reads the element 28:PARM/LIST-0,0- and essentially does a

```
SET  NAME1=ARG1  NAME2=ARG2  NAME3=ARG3  NAME4=ARG4
```

It then disables the element 28:PARM/LIST-0,0-

Interactive Runstreams

The second two forms are good for building interactive runstreams. The following example is an illustration of this.

```
* Read some variables and list them.
RECIPE  A,B,C
Input A, B, C >
##READ 0,AUTO
LIST A,B,C
```

If a ##READ were done on the above the display would look something like this:

```
?##READ 28,EXAM,PLE
Input A, B, C> 1.,2.,3.
A    =    1.
B    =    2.
C    =    3.
?
```

Symbol REC$

The symbol REC$ is set to the input from RECEIVE or RECIPE and can be used later for whatever you wish.

Related Commands

RUN, LIST, SET, ##READ, ##READ THROUGH

14.3. RUN

Purpose

Initiate the execution of a previously prepared runstream. The RUN command is functionally equivalent to the FORTRAN CALL statement. The subroutine 'CALLed' is a sequence of VAPEPS commands.

Command

```
RUNx=NAME  ARG1,ARG2,ARG3,......,Argn
```

Where x is an integer number between 1 and 30, and refers to a DAL unit. NAME is a 4 character hollerith name, which indicates a runstream element on DAL unit x.

DAL unit 30 is usually used for the RUN command streams, thus the value of x defaults to 30.

ARG1,ARG2,... Are arguments to the 'subroutine'.

Notes

The use of the RUN command by users is very straightforward, providing that the author of the routine has documented it. Other than the fact that it starts with RUNx=, it is no different than any regular command.

Creating a RUN subroutine is a slightly more complicated matter. The following lines provide an example:

```
SYMIN  5  RUN TEST
*
* The convention for all RUN streams is to start the stream
* with a group of comments that describe the command. In
* this case, the purpose of this stream is to show a
* RUN command stream.
*
*  RUN=TEST   N1,EL1,VR1  N2,EL2,VR2  N3,EL3,VR3
*
*   THIS STREAM PERFORMS THE TRIPLE PRODUCT
*
*          T
*     C = A BA
*
*  FIRST WE WILL RECOVER THE ELEMENT NAMES
*
RECOVER  N1,IL1,IV1,N2,IL2,IV2,N3,IL3,IV3
*
```

```
*  THEN, USE TMUL FOR FIRST PRODUCT
*
TMUL  &N1,&IL1,&IV1  &N2,&IL2,&IV2  28,AT,B
*
*  NOW PERFORM SECOND PRODUCT
*
MULT  28,AT,B  &N1,&IL1,&IV1  &N3,&IL3,&IV3
*
*  AND WE'RE DONE
*
RUN- TEST
SEOF
```

The above run command could be executed as follows:

```
RUN5=TEST  1,PHI,MAT  1,MASS,MAT  1,PHIT,MPHI
```

There are several important items to include when creating run stream elements:

1. Always document the runstream command.
2. The RECOVER command may be used to retrieve arguments.
3. The RUN- NAME command must be used to terminate your run stream element. (RUN+ NAME may be used instead, see below)

The following notes will be usefull:

1. The RUN command automatically
   a) Saves variables prior to starting the runstream.
   b) Turns off the SPIT, ECHO, and COMMENT modes (if they were on).
2. The variable table is left unchanged when the stream is started. Thus any new variables are added to the table. This has advantages, in that variables may be referenced other than through the argument list. It has disadvantages, though in that the variable table might overflow if the sum of existing variables plus new variables gets too large. (Max variables = 50)
3. The RUN- command retrieves the variables that were in effect prior to the runstream. Variables that were created in the stream may not be passed out of the stream. If RUN+ is used to terminate a runstream, the original variable table will not be recovered. This means that the variables defined in the stream will be retained once the stream is finished.
4. The status of SPIT, COMMENT, and ECHO are restored to their original values. Thus a programmer can change the comment mode inside a stream and not have to reset it.

5. An error status may be passed to VAPEPS by supplying an additional argument to the RUN+ or RUN- commands. eg. RUN+ NAME &IERR or RUN- NAME &IERR
   If IERR=0, normal (successful) completion. If IERR=1, VAPEPS will issue a 'TRY AGAIN' message if the session is interactive, or 'FATAL ERROR,....' if not.
6. Note that the command SYMIN was used to create the runstream element. The element must be either a type 5 or a type 6 DAL element. The name is also important, as the command RUNx=NAME will look for the element RUN/NAME on unit x.

Special notes

Users may toggle an automatic runstream selection mode. In english, this translates to: If a command is not found you normally will get a SAME TO YOU error message. If the auto runstream mode is turned on, VAPEPS will search the runstream file for a runstream element by that name. If one is found, execution will continue as if the user had actually typed in RUN=COMMAND. This mode is initially on. It may be turned off by

```
     RUNOFF
```

or back on by   RUNON

The user may also specify an additional unit that should be used for runstreams. In this case, the command is

```
RUNON  3
```

Now an illegal command will be re-submitted as RUN3=COMMAND If that is still no good, the form RUN=COMMAND will be tried. Note that RUN=NAME always refers to runstreams on unit 30.

15. DATABASE WORKSHEETS

The following pages contain worksheets designed to aid in the entry and preparation of data into the VAPEPS database.

Sheet____of____

Event________

PREP CHECKLIST

STATUS* □ COMMAND SECTION

____ □ PREP

____ □ BOOK

____ □ CHANNEL COORDINATES

____ □ CONFIGURATION

____ □ MODULES

____ □ MODULES - DESCRIPTION

____ □ MODUELS - ATTACH CHANNELS

____ □ MODULES - MODULE PARAMETERS

____ □ MODULES - MODULE ATTACHMENT

____ □ DEFINE NEW WORDS

____ □ ADMINISTRATION

STATUS MODULE NAME

4 CHARACTERS

____ □ ________

____ □ ________

____ □ ________

____ □ ________

____ □ ________

____ □ ________

____ □ ________

____ □ ________

____ □ ________

____ □ ________

Sheet ___of___

Event_______

PREP Command

```
? PREP NU,VENT  NSEC
       (___, _____  _____)
```

NU is the DAL unit where the data for the EVENT is stored.
VENT and EVENT are the same four character word.
NSEC is the number of sections ENTERed for the EVENT.

Sheet ____ of ____

Event ________

BOOK Section

>BOOK	AGENCY	PROGRAM	PROJECT	ID*
>PROC	________	________	________	________
>CONT	________	________	________	________
>COGN	________	________	________	________

>DATE

month day year

____ / ____ / ____

>TIME

hours min. sec. fract.

____ : ____ : ____ . ____

* The EVENT and the ID are the same four character word.
All other fields except for TIME and DATE can be up to 12 characters long.
NOTE: All argument fields are SERCHable.

	CLASS	TYPE	ID1	ID2
>EVENT	________	________	________	________
	GLOBAL	LOCAL	ID1	ID2
>LOCATION	________	________	________	________
	CLASS	TYPE	CONFIGURATION	ID
>VEHICLE	________	________	________	________

Channel Coordinates

Sheet ___ of ___

Event ________

Channel	X	Y	Z	System	Channel	X	Y	Z	System

CONFIGURATION Tree

Sheet ____ of ____

Event ________

LAST	THIS	GENERIC	SPECIFIC	MOUNT	ENCLOSURE

Sheet____of____

Event________

MODULES Command

```
>MODULES MXMOD MXCT MXCPS

          _____ _____ _____
```

See the USER'S MANUAL for default values for MXMOD, MXCT, MXCPS.

MXMOD is the maximum number of modules for this event.

MXCT is the maximum number of channels attached to all MODULES entered for the EVENT.

MXCPS is the maximum number of channels per SEA element.

MODULES - Description

Sheet ____ of ____

Event ________

Four character module name

>DAMO ______

Three word module description

>DESC ____________________ ____________________ ____________________

Three word description of EXTA

>EXTA ____________________ ____________________ ____________________

Three word description of SKIN

>SKIN ____________________ ____________________ ____________________

Three word description of INTA

>INTA ____________________ ____________________ ____________________

Three word description of MONT

>MONT ____________________ ____________________ ____________________

Three word description of INST

>INST ____________________ ____________________ ____________________

Sheet ____ of ____

Event ________

MODULES - ATTACHED CHANNELS

Four character module name

>DAMO ______

NOTE: If more than 15 channels are attached to a SEA element, see the MXCPS spcification for the MODULES command.

> ______ : SEA ELEMENT, i.e., EXTA, SKIN, INTA,MONT,INST

Channel names to be attached to this element

C* ______ ______ ______ ______ ______ ______ ______ ______ ______ ______

C* ______ ______ ______ ______ ______ ______ ______ ______ ______ ______

> ______ : SEA ELEMENT, i.e., EXTA, SKIN, INTA,MONT,INST

Channel names to be attached to this element

C* ______ ______ ______ ______ ______ ______ ______ ______ ______ ______

C* ______ ______ ______ ______ ______ ______ ______ ______ ______ ______

> ______ : SEA ELEMENT, i.e., EXTA, SKIN, INTA,MONT,INST

Channel names to be attached to this element

C* ______ ______ ______ ______ ______ ______ ______ ______ ______ ______

C* ______ ______ ______ ______ ______ ______ ______ ______ ______ ______

Sheet____of____

Event________

MODULES - PARAMETERS

>DAMO ______

> ______ : SEA ELEMENT, i.e., EXTA, SKIN, INTA,MONT,INST

Parameters needed for this element

P* TYPE = ______ ______ = ______ ______ = ______ ______ = ______ ______ = ______

P* ______ = ______ ______ = ______ ______ = ______ ______ = ______ ______ = ______

P* ______ = ______ ______ = ______ ______ = ______ ______ = ______ ______ = ______

P* ______ = ______ ______ = ______ ______ = ______ ______ = ______ ______ = ______

P* ______ = ______ ______ = ______ ______ = ______ ______ = ______ ______ = ______

> ______ : SEA ELEMENT, i.e., EXTA, SKIN, INTA,MONT,INST

Parameters needed for this element

P* TYPE = ______ ______ = ______ ______ = ______ ______ = ______ ______ = ______

P* ______ = ______ ______ = ______ ______ = ______ ______ = ______ ______ = ______

P* ______ = ______ ______ = ______ ______ = ______ ______ = ______ ______ = ______

P* ______ = ______ ______ = ______ ______ = ______ ______ = ______ ______ = ______

P* ______ = ______ ______ = ______ ______ = ______ ______ = ______ ______ = ______

MODULES - MODULE ATTACHMENT

Sheet ____ of ____

Event ________

	MODULE	CONF	ZONE	SYSTEM	X	Y	Z
>ATTACH							

MODULE - CONF - ZONE must be unique within a given EVENT

MODULE is the four character name used to identify a MODULE within a EVENT

CONF is the branch, "This", on the EVENT's configuration tree where the MODULE attaches.

ZONE is a four character name which designates a general area in the CONFIGURATION where the MODULE attaches.

Data Dictionary worksheet. -- Previously defined words.

Sheet ___ of ___

Event ______

Word	Definition number.	Word	Definition number.	Word	Definition number.
	□		□		□
	□		□		□
	□		□		□
	□		□		□
	□		□		□
	□		□		□
	□		□		□
	□		□		□
	□		□		□
	□		□		□
	□		□		□
	□		□		□
	□		□		□
	□		□		□

Data Dictionary worksheet. -- Synonyms.

Sheet ____ of ____

Event ________

Word ____________________

Synonyms ____________________ ____________________ ____________________ ____________________ ____________________

Word ____________________

Synonyms ____________________ ____________________ ____________________ ____________________ ____________________

Word ____________________

Synonyms ____________________ ____________________ ____________________ ____________________ ____________________

Word ____________________

Synonyms ____________________ ____________________ ____________________ ____________________ ____________________

Word ____________________

Synonyms ____________________ ____________________ ____________________ ____________________ ____________________

Word ____________________

Synonyms ____________________ ____________________ ____________________ ____________________ ____________________

Word ____________________

Synonyms ____________________ ____________________ ____________________ ____________________ ____________________

Word ____________________

Synonyms ____________________ ____________________ ____________________ ____________________ ____________________

Word ____________________

Synonyms ____________________ ____________________ ____________________ ____________________ ____________________

Data Dictionary worksheet. -- Undefined words.

Sheet ___ of ___

Event ______

Word ____________

Definition ____________

Synonyms ______ ______ ______ ______ ______

Word ____________

Definition ____________

Synonyms ______ ______ ______ ______ ______

16. SEMOD WORKSHEETS

The following pages contain worksheets designed to aid in the creation of SEMOD models.

Element Name: ________________ Type = **1 - REVB** (Reverberant Acoustic Space)

DESCRIPTION = '____'

RHO = ____________ Mass density of acoustic medium. **(Note: This is MASS density not WEIGHT density.)**

CO = ____________ Speed of sound in acoustic medium.

VOLUME = ____________ Volume of acoustic space.

AP = ____________ Surface area of acoustic space.

AAC = ____________ Acoustic absorbtion coefficient.

Element Name: ______________ Type = **2 - NREV** (Non-Reverberant Acoustic Space)

DESCRIPTION = ' **NOT SUPPORTED FOR PREDICTIONS** '

RHO = ____________ Mass density of acoustic medium. **(Note: This is MASS density not WEIGHT density.)**

CO = ____________ Speed of sound in acoustic medium.

VOLUME = ____________ Volume of acoustic space.

AP = ____________ Surface area of acoustic space.

AAC = ____________ Acoustic absorbtion coefficient.

VISC = ____________ Kinematic viscosity of fluid.

VEL = ____________ Velocity of flying body.

Element Name: ________________ Type = **3 - PLAT** (Flat Plate)

DESCRIPTION = '____'

RHO = ______________ Mass density of plate. **(Note: This is MASS density not WEIGHT density.)**

CL = ______________ Longitudinal wave speed.

H = ______________ Equivalent thickness.

AP = ______________ Surface area.

ALX = ______________ Sub-panel dimension - X direction.

ALY = ______________ Sub-panel dimension - Y direction.

DLF = ______________ Damping loss factor.

E = ______________ Young's modulus.

PATA = ______________ Total length of discontinuity.

RHOS = ______________ Surface mass density. **(Note: This is MASS density not WEIGHT density.)**

ASMS = ______________ Non-structural mass. **(Note: This is MASS not WEIGHT.)**

PIVOTFRQ = ______________ Frequency for pivoting functionalized damping. (optional)

Element Name: ____________________ Type = **4 - CYLN** (Cylinder)

DESCRIPTION = '____'

RHO = ________________ Mass density of cylinder**(Note: This is MASS density not WEIGHT density.)**

CL = ________________ Longitudinal wave speed.

H = ________________ Equivalent thickness.

AP = ________________ Surface area.

BL = ________________ Length of cylinder.

ALX = ________________ Sub-panel dimension - X direction.

ALY = ________________ Sub-panel dimension - Y direction.

D = ________________ Diameter of cylinder.

DLF = ________________ Damping loss factor.

E = ________________ Young's modulus.

PATA = ________________ Total length of discontinuity.

RHOS = ________________ Surface mass density. **(Note: This is MASS density not WEIGHT density.)**

ASMS = ________________ Non-structural mass. **(Note: This is MASS not WEIGHT.)**

CO = ________________ Speed of sound in medium surrounding the cylinder.

PIVOTFRQ = ________________ Frequency for pivoting functionalized damping. (optional)

Element Name: ____________ Type = **5 - CONE** (Cone)

DESCRIPTION = '______________________________________'

RHO = ____________ Mass density of cone. **(Note: This is MASS density not WEIGHT density.)**

CL = ____________ Longitudinal wave speed.

H = ____________ Equivalent thickness.

AP = ____________ Surface area.

BL = ____________ Length (height) of cone.

ALX = ____________ Sub-panel dimension - X direction.

ALY = ____________ Sub-panel dimension - Y direction.

D = ____________ Diameter of base of cone.

DLF = ____________ Damping loss factor.

E = ____________ Young's modulus.

PATA = ____________ Total length of discontinuity.

BETA = ____________ Half apex angle of cone.

RHOS = ____________ Surface mass density. **(Note: This is MASS density not WEIGHT density.)**

ASMS = ____________ Non-structural mass. **(Note: This is MASS not WEIGHT.)**

CO = ____________ Speed of sound in medium surrounding the cone.

PIVOTFRQ = ____________ Frequency for pivoting functionalized damping. (optional)

Element Name: ________________ Type = **6 - BEMR** (Rectangular beam or truss.)

DESCRIPTION = '____'

RHO = ____________ Mass density of truss. **(Note: This is MASS density not WEIGHT density.)**

ASMS = ____________ Non-structural mass. **(Note: This is MASS not WEIGHT.)**

H = ____________ Equivalent thickness of truss.

B = ____________ Width of truss.

BL = ____________ Length of truss member.

E = ____________ Young's modulus.

G = ____________ Shear modulus.

DLF = ____________ Damping loss factor.

CL = ____________ Longitudinal wave speed.

CNT = ____________ Number of truss members.

Element Name: ________________ Type = **7 - BEMC** (Circular beam or truss.)

DESCRIPTION = '____'

RHO = ______________ Mass density of truss. **(Note: This is MASS density not WEIGHT density.)**

ASMS = ______________ Non-structural mass. **(Note: This is MASS not WEIGHT.)**

BL = ______________ Length of truss member.

DO = ______________ Outside diameter.

E = ______________ Young's modulus.

G = ______________ Shear modulus.

DLF = ______________ Damping loss factor.

CL = ______________ Longitudinal wave speed.

CNT = ______________ Number of truss members.

Element Name: ______________ Type = **8 - BEMT** (Tubular beam or truss.)

DESCRIPTION = '____'

RHO = ____________ Mass density of truss. **(Note: This is MASS density not WEIGHT density.)**

ASMS = ____________ Non-structural mass. **(Note: This is MASS not WEIGHT.)**

BL = ____________ Length of truss member.

DI = ____________ Inside diameter.

DO = ____________ Outside diameter.

E = ____________ Young's modulus.

G = ____________ Shear modulus.

DLF = ____________ Damping loss factor.

CL = ____________ Longitudinal wave speed.

CNT = ____________ Number of truss members.

Element Name: ______________ Type = **9 - BEMO** (Other beam or truss.)

DESCRIPTION = '____'

RHO = ____________ Mass density of truss. **(Note: This is MASS density not WEIGHT density.)**

ASMS = ____________ Non-structural mass. **(Note: This is MASS not WEIGHT.)**

BL = ____________ Length of truss member.

RGF = ____________ Flexural radius of gyration.

A = ____________ Cross section area.

PJ = ____________ Polar moment of inertia.

E = ____________ Young's modulus.

G = ____________ Shear modulus.

T = ____________ Torsional stiffness.

DLF = ____________ Damping loss factor.

CL = ____________ Longitudinal wave speed.

CNT = ____________ Number of truss members.

17. FREQUENCY TABLES

This section contains table for standard 1/3 octave and 1/6 octave center frequencies, band widths, and lower and uppper band edge freqeuncies.

The following runstream was used to generate the numbers in these tables:

```
frequency/center 28,cen,frq 10.,10000.
frequency/width  28,wid,frq 10.,10000.
frequency/edge   28,edg,frq 10.,10000.
calc
  nf = (28,cen,frq -6)
  nf2 = nf+1
  done
build 28,frq,tabl &nf,4
   mix 28,cen,frq
   mix 28,wid,frq 1,&nf  1,1 1,2
   mix 28,edg,frq 1,&nf  1,1 1,3
   mix 28,edg,frq 2,&nf2 1,1 1,4
   done
prinf 28,frq,tabl -7
   (4f8.1)
frequency/set 2
frequency/center 28,cen,frq 10.,10000.
frequency/width  28,wid,frq 10.,10000.
frequency/edge   28,edg,frq 10.,10000.
calc
  nf = (28,cen,frq -6)
  nf2 = nf+1
  done
build 28,frq,tabl &nf,4
   mix 28,cen,frq
   mix 28,wid,frq 1,&nf  1,1 1,2
   mix 28,edg,frq 1,&nf  1,1 1,3
   mix 28,edg,frq 2,&nf2 1,1 1,4
   done
prinf 28,frq,tabl -7
   (4f8.1)
end
```

Standard 1/3 Octave Center frequencies

Center	Band	Band	Edge
Frequency	Width	Lower	Upper
10.0	2.2	9.0	11.2
12.5	2.8	11.2	14.0
16.0	4.0	14.0	18.0
20.0	4.4	18.0	22.4
25.0	5.6	22.4	28.0
31.5	7.5	28.0	35.5
40.0	9.5	35.5	45.0
50.0	11.0	45.0	56.0
63.0	15.0	56.0	71.0
80.0	19.0	71.0	90.0
100.0	22.0	90.0	112.0
125.0	28.0	112.0	140.0
160.0	40.0	140.0	180.0
200.0	44.0	180.0	224.0
250.0	56.0	224.0	280.0
315.0	75.0	280.0	355.0
400.0	95.0	355.0	450.0
500.0	110.0	450.0	560.0
630.0	150.0	560.0	710.0
800.0	190.0	710.0	900.0
1000.0	220.0	900.0	1120.0
1250.0	280.0	1120.0	1400.0
1600.0	400.0	1400.0	1800.0
2000.0	440.0	1800.0	2240.0
2500.0	560.0	2240.0	2800.0
3150.0	750.0	2800.0	3550.0
4000.0	950.0	3550.0	4500.0
5000.0	1100.0	4500.0	5600.0
6300.0	1500.0	5600.0	7100.0
8000.0	1900.0	7100.0	9000.0
10000.0	2200.0	9000.0	11200.0

Standard 1/6 Octave Center frequencies

Center	Band	Band	Edge
Frequency	Width	Lower	Upper
10.0	1.2	9.4	10.6
11.2	1.3	10.6	11.9
12.5	1.4	11.9	13.3
14.0	1.7	13.3	15.0
16.0	1.8	15.0	16.8
18.0	2.0	16.8	18.8
20.0	2.3	18.8	21.1
22.4	2.6	21.1	23.7
25.0	2.9	23.7	26.6
28.0	3.3	26.6	29.9
31.5	3.6	29.9	33.5
35.5	4.1	33.5	37.6
40.0	4.6	37.6	42.2
45.0	5.1	42.2	47.3
50.0	5.8	47.3	53.1
56.0	6.5	53.1	59.6
63.0	7.2	59.6	66.8
71.0	8.2	66.8	75.0
80.0	9.1	75.0	84.1
90.0	10.3	84.1	94.4
100.0	11.6	94.4	106.0
112.0	13.0	106.0	119.0
125.0	14.0	119.0	133.0
140.0	17.0	133.0	150.0
160.0	18.0	150.0	168.0
180.0	20.0	168.0	188.0
200.0	23.0	188.0	211.0
224.0	26.0	211.0	237.0
250.0	29.0	237.0	266.0
280.0	33.0	266.0	299.0
315.0	36.0	299.0	335.0
355.0	41.0	335.0	376.0
400.0	46.0	376.0	422.0
450.0	51.0	422.0	473.0
500.0	58.0	473.0	531.0
560.0	65.0	531.0	596.0
630.0	72.0	596.0	668.0
710.0	82.0	668.0	750.0
800.0	91.0	750.0	841.0
900.0	103.0	841.0	944.0
1000.0	116.0	944.0	1060.0
1120.0	130.0	1060.0	1190.0
1250.0	140.0	1190.0	1330.0
1400.0	170.0	1330.0	1500.0
1600.0	180.0	1500.0	1680.0
1800.0	200.0	1680.0	1880.0

Standard 1/6 Octave Center frequencies
(continued)

Center	Band	Band	Edge
Frequency	Width	Lower	Upper
2000.0	230.0	1880.0	2110.0
2240.0	260.0	2110.0	2370.0
2500.0	290.0	2370.0	2660.0
2800.0	330.0	2660.0	2990.0
3150.0	360.0	2990.0	3350.0
3550.0	410.0	3350.0	3760.0
4000.0	460.0	3760.0	4220.0
4500.0	510.0	4220.0	4730.0
5000.0	580.0	4730.0	5310.0
5600.0	650.0	5310.0	5960.0
6300.0	720.0	5960.0	6680.0
7100.0	820.0	6680.0	7500.0
8000.0	910.0	7500.0	8410.0
9000.0	1030.0	8410.0	9440.0
10000.0	1160.0	9440.0	10600.0

18. CONVERSION TABLES

This section contains some commonly used conversions. You can use the CONVERT command to perform many of these conversions. The general units give the values in terms of Mass (M), Length (L), and Time (T).

To convert from	to	multply by	general units
ft/s**2	m/s**2	0.3048	L*(T**-2)
in/s**2	m/s**2	0.0254	L*(T**-2)
ft**2	m**2	0.0929	L**2
in**2	mm**2	645.2	L**2
kip	kN	4.448	M*L*(T**-2)
lbf	N	4.448	M*L*(T**-2)
ozf	N	0.2780	M*L*(T**-2)
lbf-s	N-s	4.448	M*L*(T**-1)
ft	m	0.3048	L
in	mm	25.40	L
mile	km	1.609	L
oz mass	g	28.35	M
lb mass	kg	0.4536	M
slug	kg	14.59	M
ton	kg	907.2	M
lbf-ft	N-m	1.356	M*(L**2)*(T**-2)
lbf-in	N-m	0.1130	M*(L**2)*(T**-2)
in**4	mm**4	0.4162E6	L**4
lbf-ft-s**2	kg-m**2	1.356	M*(L**2)
lbf-s	kg-m/s	4.448	M*L*(T**-1)
ft-lbf/s	W	1.356	M*(L**2)*(T**-3)
hp	W	745.7	M*(L**2)*(T**-3)
lbf/ft**2	Pa	47.88	M*(L**-2)
lbf/in**2 (psi)	kPa	6.895	M*(L**-2)
ft/s	m/s	0.3048	L*(T**-1)
in/s	m/s	0.0254	L*(T**-1)
mi/h (mph)	m/s	0.4470	L*(T**-1)
mi/h (mph)	km/h	1.609	L*(T**-1)
ft**3	m**3	0.02832	L**3
in**3	cm**3	16.39	L**3
gal	l	3.785	L**3
qt	l	0.9464	L**3

19. INDEX

VAPEPS has an interactive index that is available by accessing the RUN stream called INDEX (RUN=INDEX). The following pages list the keys that are present in this index and the commands which relate to each of the keys.

The purpose of the index is to help users find the appropriate commands. The section containing the command can be found by looking in the Table of Contents.

20. SUMMARY

```
***** COMMAND SUMMARY
*
*

  NAME                        DESCRIPTION

HELP........VAPEPS ONLINE DOCUMENTATION.
MENU         VAPEPS MENU UTILITY
AND..........REDUCE TWO LISTS TO ONE LIST OF LIKE ITEMS
ASMD.........TERM BY TERM ADD, SUBTRACT, MULTIPLY AND DIVIDE
BUILD(X).....BUILD MATRICES FROM OTHER MATRICES
CALCULATOR...CALCULATOR MODE -- OPERATES ON VARIABLES
CBIT........SEARCH BIT MAP MATRIX FOR SEQUENCE OF BITS
CEAL........CONVERT EAL ELEMENTS TO DAL AND VISA VERSA
CHANGE......CHANGE NAME, TYPE, OR NJ FOR ELEMENTS
CHL.........CHOLESKI DECOMPOSITION -- INVERSE OF SYMMETRIC MATRICES.
CHOP(X).....SELECT ROWS AND COLUMNS OF A MATRIX.
CMOD........COMPLEX TO MODULUS/PHASE OR REAL/IMAGINARY, POLAR TO RECTANGULAR
CNFG........STAND ALONE CONFIGURATION TREE EDITOR (VAPEPS)
CONVERT.....VIBROACOUSTIC UNIT CONVERSION.
COPY........COPY INDIVIDUAL OR MULTIPLE ELEMENTS
CORE........SET MAXIMUM CORE LIMIT.
CPU.........PRINT ELAPSED CPU (SUP) TIME
CRYPT.......CHARACTER TRANSLATION, CRYPTOLOGY.
CSYM........TRANSFER SYMBOLIC ELEMENTS TO/FROM SYSTEM FILES.
CYCLE.......CHANGE DEFAULT CYCLE NUMBERS.
DISABLE......DISABLES DAL ELEMENTS
DOC.........DOCUMENT FORMATER.
DPTOR.......DOUBLE PRECISION TO REAL, AND VISA VERSA.
ECOL........FREE-FIELD DATA ENTRY
ENABLE.......ENABLES DISABLED DAL ELEMENTS
EULER.......CREATE EULER ROTATION MATRIX.
EXPAND......CHANGE BLOCK SIZE OF AN ARRAY.
EZOUT.......WRITE EASY FORMATTED SYMBOLIC ELEMENTS
FFT.........FAST FOURIER TRANSFORM.
FILES.......LIST FILES ASSIGNED TO CURRENT EXECUTION.
FIND........LIST TOC LINES FOR GROUP OF ELEMENTS.
FIT..........DO A LEAST SQUARES CURVE FIT TO AN NTH ORDER POLYNOMIAL
FNAME.......RESET LOGICAL DAL UNIT NAME.
FREQUENCY...FREQUENCY CALCULATOR 1/3 OCTAVE - 1/6 OCTAVE ETC..
FUN.........COMPUTE Y AS A FUNCTION OF X (SIN,COS,LOG,ETC)
GCOL........READ 12 CHARACTER WORDS, BLOCKED SETS, ETC.
GLOSS.......GLOSSARY CREATION, INSERTION, CHECKING
GREAD.......READ DATA IN FREE-FIELD FORMAT.
GROUP.......COLLECT SEVERAL ELEMENTS INTO ONE (SIMPLIFIED BUILD)
```

```
INDEX.......REPLACE DATA WITH POSITION IN AN INDEX LIST.
INTERPOLATE.INTERPOLATE AN ORDERED SET TO A NEW ABSCISSA.
INVERT.....MATRIX INVERSION.
JPI.........OBTAIN SYSTEM PARAMETERS (SYSTEM DEPENDENT)
LAST........PRINT TOC LINE OF LAST ELEMENT IN A DAL FILE.
LOCATE......LOCATE A SPECIFIC ELEMENT.
LOOK........QUICK LOOK PROCESSOR.
MAXMIN......FIND MAXIMUM AND MINIMUM VALUES OF AN ARRAY.
MDIFF.......GET THE DERIVATIVE OF A VECTOR.
MGRATE......INTEGRATE A VECTOR OR GROUP OF VECTORS.
MODIFY......CHANGE INDIVIDUAL TERMS IN A MATRIX.
MSAP........MULTIPLY NEPSAP STIFFNESS MATRIX BY REGULAR MATRIX.
MSPAR.......SPARSE MATRIX MULTIPLICATION.
MTJ.........TRANSPOSE MULTIPLICATION USING 'NJ'
MTRAN......MATRIX MULTIPLICATION A*BT = C
MULTIPLY....MATRIX MULTIPLICATION  A*B=C
NORM........NORMALIZE A MATRIX TO 1.
NOT.........GET NUMBERS NOT PRESENT IN A LIST.
NTOW........NARROW TO WIDE BAND CONVERSION.
OR..........COMBINE TWO LISTS INTO ONE LIST CONTAINING ALL ITEMS.
ORDER.......SORT A MATRIX IN ASCENDING ORDER.
PACK........REMOVE ALL DISABLED ELEMENTS FROM A DAL FILE.
POW.........RAISE A VECTOR TO SUCCESSIVE POWERS.
RANDOM......CREATE A MATRIX OF UNIFORMLY DISTRIBUTED RANDOM NUMBERS
RLJUST......RIGHT OR LEFT JUSTIFIES HOLLERITH WORDS.
RTOI........REAL TO INTEGER CONVERSION AND VISA VERSA.
SEARCH......FIND VALUES OF MATRIX WITHIN A GIVEN RANGE.
SEQUENCE....MAKE A LINEAR OR LOGARITHMIC SEQUENCE OF REAL NUMBERS.
SHOCK.......COMPUTE SHOCK SPECTRA.
SREAD.......READ DATA IMAGES USING FORTRAN FORMATTED READ
SSTRNG......SEARCH COLUMNS FOR A STRING OF INTEGERS.
STAT........GET MEAN, STANDARD DEVIATIONS, MEAN +/- X SIGMA.
STORE.......STORE A VALUE IN A PARTICULAR LOCATION OF A MATRIX.
STRESS......CALCULATE MEAN SQUARE STRESS FROM MEAN SQUARE RESPONSE.
SYMIN.......INPUT IMAGES TO A DAL SYMBOLIC (TYPE 5/6) ELEMENT.
TDUMP.......READ ANY TAPE INTO A DAL FILE.
TMUL........MATRIX MULTIPLICATION  AT*B = C.
TOC.........PRINT THE TABLE OF CONTENTS OF A DAL UNIT.
TRANSPOSE...MATRIX TRANSPOSITION.
TRDAL.......TRANSLATE DAL FILE FROM ONE SYSTEM TO ANOTHER.
TTYP........CHANGE THE TYPE OF TOC LISTING.
WTAPE.......WRITE A MATRIX TO TAPE.
PLOT........PLOT X VERSUS Y.
PRCRN.......PRINT CORNERS OF A MATRIX.
PRINT.......PRINT THE CONTENTS OF A DAL ELEMENT.
PSET........PLOT SET - PLOT INITIALIZATION SUBPROCESSOR.
RITE........PRINT AN ELEMENT AS A FUNCTION OF SOME VECTOR.
TABLE.......PRINT MATRICES IN REPORT TYPE FORMAT.
WRITE.......WRITE VARIABLES/CONSTANTS USING FORTRAN FORMATS.
ENTER        DATABASE SPECTRAL DATA INPUT COMMAND.
```

```
PREP........DATABASE EVENT DEFINITION AND STANDARDIZATION COMMAND.
SERCH.......SEARCH VAPEPS DATABASE.
DICTIONARY..DIRECT DICTIONARY INTERFACE PROCESSOR.
SEMOD.......STATISTICAL ENERGY MODELER.
PREDICT.....THEORETICAL, EXTRAP I AND EXTRAP II PREDICTIONS.
CNVRT.......SETS UP CONVERSION FACTORS FOR SEA PREDICTIONS
DAMO........STAND ALONE DATA MODULE EDITOR.
DENS........CALCULATES MODAL DENSITY.  NON-SEMOD MODELS.
PRDICT......DOES THEORETICAL AND EXTRAPOLATED SEA PREDICTIONS.
SCALE.......EXTRAP II PREDICTION COMMAND.
TRNF........CALCULATES SEA TRANSFER FUNCTIONS
LIST........LIST VALUE OF ALL OR SELECTED VARIABLES
SET.........DEFINE THE VALUE OF A VARIABLE
VAR.........REMOVE VARIABLES FROM THE VARIABLE TABLE
DEFAULT.....SPECIFY DEFAULT VALUES FOR VARIABLES.
RECOVER.....RECOVER ARGUMENTS FROM A RUN COMMAND.
RUN           INITIATE A 'VAPEPS' SUBROUTINE CALL
AC           (NOT A COMMAND) COMMAND AND RUNSTREAM SUMMARY.
AS             ARITHMETICS AND STATISTICS COMMAND SUMMARY
AU             ADVANCED USAGE COMMAND SUMMARY
EC             ELEMENT CREATION MANIPULATION COMMAND SUMMARY
EF             ELEMENT FILE SYSTEM COMMAND SUMMARY
GR           GRAPHICS COMMAND SUMMARY
HH           HELP WITH HELP
IN         INTRODUCTION TO VAPEPS.
IO         INPUT/OUTPUT COMMAND SUMMARY
LS             LOGIC SORTING OPERATONS COMMAND SUMMARY
UG         USER GUIDE TO GENERAL INFORMATION.
UH             USING THE VAPEPS HELP PROCESSOR.
VC             VARIABLE CONTROL COMMAND SUMMARY
VD             VIBROACOUSTIC DATABASE COMMAND SUMMARY
VP             VIBROACOUSTIC PREDICTION COMMAND SUMMARY
VT             (NOT A COMMAND) VIBROACOUSTIC TOOLS COMMAND SUMMARY.
```

20.1. AC

COMMAND AND RUNSTREAM SUMMARY - ALPHABETICAL.

```
 NAME                         DESCRIPTION
-----------------------------------------------------------------------------
 ABLD-ABLE...ENABLE AND DISABLES DAL ELEMENTS.
 RUN=ADDC....ADD NEW COMMANDS TO SETCOM ROUTINE.
 ADMIN.......DATABASE ADMINISTRATION COMMAND.
 RUN=AJNV....ADJUST THE INVERSE OF A MATRIX.
 RUN=ALL.....PRINTS OUT ENTER HEADER AND CHANNEL INFO IN A DAL FILE.
 AND.........REDUCE TWO LISTS TO ONE LIST OF LIKE ITEMS.
 APE.........CURVE FIT TO NTH ORDER POLYNOMIAL.
 ASMD........TERM BY TERM ADD, SUB, MULT, DIV.
 RUN=AVG.....CALCULATES THE AVERAGE BY ROW OR COLUMN.
 RUN=BCL.....BUILD COM/LIST ELEMENT FOR SAVING THE COMMAND LIST.
 BIGSEA......LINKS SEPERATE SEA MODELS TOGETHER.
 RUN=BLIM....CREATES A VECTOR OF 1/3 OCTAVE BAND LIMITS.
 RUN=BOOT....CREATES A VAPEPS BOOT TAPE FOR THE NAMEDE SYSTEM.
 RUN=BSC.....CREATES A SETCOM ROUTINE FOR DALPRO/VAPEPS TYPE SYSTEM.
 BUILD(X)....BUILD MATRICIES FROM OTHER MATRICIES.
 RUN=BWTH....CREATES A VECTOR OF 1/3 OCTAVE BANDWIDTHS.
 CALC........CALCULATOR MODE -- OPERATE ON VARIABLES.
 CBIT........SEARCH BIT MAP MATRIX FOR SEQUENCE OF BITS.
 CHANGE......CHANGE NAME, TYPE, OR NJ FOR ELEMENTS.
 CHL.........CHOLESKI DECOMPOSTION --INVERSE OF SYMMETRIC MATS.
 CHOP........SELECT ROWS AND COLUMNS OF A MATRIX.
 RUN=CIMV....CALCULATES THE CONFIDENCE INTERAL FOR A TRUE MEAN OF X.
 RUN=CLFGET..Gets a set of ATA'S from a SEMOD path.
 RUN=CLFPUT..Puts a set of ATA'S into a SEMOD path.
 CMOD........COMPLEX TO MODULUS/PHASE OR REAL/IMAGINARY.
 CNFG........STAND ALONE CONFIGURATION TREE EDITOR (VAPEPS).
 CNVRT.......SETS UP CONVERSION FACTORS FOR PREDICTIONS.
 COMMAND.....(NOT A COMMAND) COMMAND FORMAT DESCRIPTION.
 CONVERT.....Vibroacoustic Unit Conversion.
 CONTROL.....(NOT A COMMAND) INFO ON CONTROL COMMANDS (##) AND PRECODER ($$).
 COPY........COPY INDIVIDUAL OR MULTIPLE ELEMENTS.
 RUN=COPS....WRITES ALL SYMBOLIC ELEMENTS TO A FORTRAN UNIT.
 CORE........SET MAXIMUM CORE LIMIT.
 CPU.........PRINT ELAPSED CPU (SUP) TIME.
 RUN=CPYS....CREATES FILE OF CARD IMAGES FROM MATCHING SYMBOLIC ELEMENTS.
 CRYPT.......CHARACTER TRANSLATION, CRPTOLOGY.
 CSYM........TRANSFER SYMBOLIC ELEMENTS TO/FROM SYSTEM FILES.
 CYCLE.......CHANGE DEFAULT CYCLE NUMBERS.
 DALDOC......(NOT A COMMAND) DAL FILE DESCRIPTION.
```

```
DAMO........STAND ALONE DATA MODULE EDITOR (VAPEPS).
RUN=DATA....EXTRACTS AND AVERAGES CHANNEL DATA FROM DATABASE.
DEFAULT.....SPECIFY DEFAULT VALUES FOR VARIABLES.
DENS........CALCULATES MODAL DENSITY.
DPTOR.......DOUBLE PRECISION TO REAL, AND VICE VERSA.
ECOL........FREE-FIELD DATA ENTRY.
ENTER.......DATABASE VIBROACOUSTIC DATA INPUT COMMAND.
RUN=ESAV....WRITES ENTER DECK OUT TO A FORTRAN UNIT.
EXPAND......CHANGE BLOCK SIZE OF AN ARRAY.
EZOUT.......WRITE EASY FORMATTED SYMBOLIC ELEMENTS.
FFT.........FAST FOURIER TRANSFORM.
FILES.......LIST FILES ASSIGNED TO CURRENT EXECUTION.
FIND........LIST TOC LINES FOR GROUP OF ELEMENTS.
FNAME.......RESET LOGICAL DAL UNIT NAME.
FORMAT......(NOT A COMMAND) FREE-FIELD FORMAT DESCRIPTION.
FRQ.........Generates Frequencies.
FUN.........COMPUTES Y AS A FUNCTION OF X (SIN,COS,LOG,ECT).
RUN=GCLS....ORDERS COMMAND ABBREVIATIONS AND ENTRY POINTS.
GCOL........READ 12 CHARACTER WORDS, BLOCKED SETS, ECT.
RUN=GDOC....PREPARES A DOC FILE FROM GENERAL AND HELP FILE.
RUN=GENP....INTERACTIVE PLOTING.
RUN=GETRESP.Stores SEMOD responsed on a specified DAL element.
GLOSS.......GLOSSARY CREATION, INSERTION, CHECKING.
RUN=GLST....LIST RUN=GMOD OUTPUT.
RUN=GMOD....RETRIEVES DATABASE MODULE DATA.
RUN=GPSD....CALCULATES PSD FROM A TIME HISTORY.
GREAD.......READ DATA IN FREE-FIELD FORMAT.
RUN=GRLT....UPDATES THE 'READ ELTS' ELEMENT IN THE 'EASY' FILE.
RUN=GRMS....VIBRATION SPEC MANIPULATOR AND RMS CALCULATOR.
GROUP.......COLLECT SEVERAL ELEMENTS INTO ONE (SIMPLIFIED BUILD).
RUN=GVAL....RETRIEVES DISTRIBUTION VALUE GIVEN SAMPLE SIZE AND CONFIDENCE.
GXI.........GET X INTERCEPT - FIND ALL ZERO CROSSINGS.
RUN=HEAD....PRINTS OUT ENTER HEADER AND CHANNEL INFO.
HELP........LIST COMMANDS, HELP.
INDEX.......REPLACE DATA WITH POSITION IN AN INDEX LIST.
RUN=INDE....INTERACTIVE COMMAND INDEX.
RUN=INTE....INTERACTIVE PROMPTING ROUTINE.
INVERT......MATRIX INVERSION.
JPI.........OBTAIN SYSTEM PARAMETERS (SYSTEM DEPENDENT).
LAST........PRINT TOC LINE OF LAST ELEMENT IN FILE.
RUN=LFU.....CREATES LOWWER OR UPPER TRIANGULAR, AND FULL MATRICIES.
LIST........LIST VALUE OF ALL OR SELECTED VARIABLES.
LOCATE......LOCATES A SPECIFIC ELEMENT.
LOOPS.......(NOT A COMMAND) INFORMATION ON LOOPING TECHNIQUES.
RUN=LRMS....CALCULATES RMS OF VIBRATION SPEC.
RUN=LTRP....INTERPOLATES VIBRATION SPEC TO 1/3 OCTAVES.
LUMP........DISPERSE A LIST OF VALUES INTO PRESCRIBED SLOTS.
RUN=MANU....WRITES A VAPEPS MANUAL.
MAXIN.......FINDS THE MAX AND MIN VALUES OF AN ARRAY.
MDIFF.......GET THE DERIVATIVE OF A VECTOR.
```

```
RUN=MEZ.....READS EZOUT FILES AND RELATES THEM TO C RUN=BLOCK REFS.
MGRATE......INTEGRATE A VECTOR OR GROUP OF VECTORS.
MODIFY......CHANGE INDIVIDUAL TERMS IN A MATRIX.
RUN=MPNT....PRINTS A MASTER FILE SUMMARY.
MTRAN.......MATRIX MULTIPLICATION A*BT  RUN= C.
MULTIPLY....MATRIX MULTILPICATION A*B RUN=C.
RUN=MVC.....CALCULATE MEAN VARIANCE AN CO-VARIANCE.
RUN=NEWS....PRINTS OUT NEWS ELEMENT ON RUNFILE.
RUN=NOML....Normalize baseline response to a new excitation.
NORM........NORMALIZE A MATRIX.
RUN=NORM....NORMALIZES DATABASE VIBROACOUSTIC DATA.
NOT.........GET ITEMS NOT PRESENT IN A LIST.
RUN=OCT3....CONVERT NARROW BAND PSD TO 1/3 OCTAVE PSD.
OR..........COMBINE TWO LISTS INTO ONE LIST CONTAINING ALL ITEMS.
ORDER.......SORT A MATRIX IN ASCENDING ORDER.
RUN=ORTH....CHECKS THE ORTHOGONALITY OF MATRIX.
RUN=OVER....CALCULATES OVERALL LEVEL OF 1/3 OCTAVE VIBROACOUSTIC DATA.
PACK........REMOVES ALL DISABLED ELEMENTS FROM A DAL FILE.
RUN=PGLS....ORDERS A GLOSSARY ALPHABETICALY, THEN PRINTS IT.
RUN=PGRM....PLOTS THE SPEC CREATED BY RUN=GRMS.
RUN=PICK....RETRIEVES DATABASE ACOUSTIC AND VIBRATION DATA.
PLOT.......PLOT X VERSUS Y
PLU.........CONVERT PACKED  UPPER-LOWER TRIANGULAR TO REGULAR AND VICE VERSA.
POW.........RAISE A VECTOR TO SUCCESSIVE POWERS.
PRCRN.......PRINT CORNERS OF A MATRIX.
PRDICT......DOES THEORETICAL AND EXTRAPOLATED SEA PREDICTIONS.
PREP........DATABASE EVENT DEFINITION AND STANDARDIZATION COMMAND.
PRINT.......PRINT THE CONTENTS OF A ELEMENT.
RUN=PRST....CALCULATES STATISTICAL INFORMATION FOR PREDICTED RESPONSE.
PSET........PLOT SET --PLOT SUBPROCESSOR.
RUN=PTMP....CALCULATES AT*B*A TRIPLE PRODUCT.
RUN=PUTC....PUTS A C RUN=DECK IN ALL (C1,C2)1,0 ELEMENTS.
PVP.........PEAK AND VALLEY PICKER -- DIGITIZATION.
RANDOM......CREATAES A MATRIX OF UNIFORMLY DISTRIBUTED RANDOM NUMBERS.
RECOVER.....RECOVER ARGUMENTS FROM A RUN COMMAND.
RITE........PRINT AN ELEMENT AS A FUNCTION OF A VECTOR.
RLJUST......RIGHT OR LEFT JUSTIFIES HOLLERITH WORDS.
RUN=ROUN....CONVERTS FLOATING POINT TO INTEGER WITH ROUNDING OFF.
RUN=RRAN....CREATES A MATRIX OF RANDOM BETWEEN LIMITS.
RTABS.......CONVERTS TAB CHARACTERS TO PROPER SPACING.
RTOI........REAL TO INTEGER CONVERSION AND VICE-VERSA.
RUN=RTOC....PULLS OUT FILES FROM CURRENT VAX DIRECTORY.
RUN.........INITIATE A VAPEPS SUBROUTINE CALL.
RUN=SAVG....Averages the responses from a SEMOD model.
RUN=SAVE....SAVES ALL PREPED EVENTS FROM A DAL UNIT TO A MASTER FILE.
SCALE.......EXTRAP2 PREDICTION COMMAND.
SEARCH......FIND VALUES OF A MATRIX WITHIN A GIVEN RANGE.
SEMOD.......Statistical Energy Modeler.
RUN=SEND....FINDS AND PRINTS OUT MATCHING ELEMENTS.
SEQUENCE....MAKE A LINEAR OR LOGRITHMIC SEQUENCE OF REAL NUMBERS.
```

```
SET.........DEFINE THE VALUE OF A VARIABLE.
RUN=SHIFT...Shifts SEMOD'S FPAR parameters up or down.
SHOCK.......COMPUTE SHOCK SPECTRA FROM TIME HISTORY.
SHUFFLE.....CREATE A MATRIX BY PLACING SQUARE MATS AT SPEC.LOCATIONS.
RUN=SNDP....FINDS DAL ELEMENTS THAT HAVE BEEN CHANGED SINCE A DATE.
RUN=SPIL....SPILLS ALL EVENTS FROM A DAL UNIT TO A ELEMENT.
SPLIT.......FORM MATRIX BY SELECTING SPECIFIC ROWS AND COLUMNS FROM ANOTHER.
SREAD.......READ IMAGES ACCORDING TO INPUT FORTRAN FORMAT.
SSTRNG......SEARCH COLUMNS FOR A STRING OF INTEGERS.
RUN=STAS....Calculates normal and log-normal statistical quantities.
RUN=STAT....CALCULATES STATISTICAL INFO ON VIBROACOUSTIC DATA.
STAT........GET MEAN, STANDARD DEVIATIONS MEAN +/- X SIGMA.
RUN=STAC....PRINTS OUT ELEMENTS TO BE TRANSFERED TO ANOTHER SYSTEM.
STORE.......STORE A VALUE IN A PARTICULAR LOCATION OF A MATRIX.
RUN=STTS....SAME AS RUN=STAT PLUS TABULATES RESULTS.
SYMIN.......INPUT IMAGES TO A DAL SYMBOLIC (TYPE 5/6).
TABLE.......PRINT MATRICIES IN REPORT TYPE FORMAT.
RUN=TCPY....WRITE DAL UNIT TO TAPE UNIT.
TDUMP.......READ ANY TAPE INTO A DAL FILE.
TMUL........MATRIX MULTILPICATION AT*B=C.
RUN=TOCO....ORDER A TOC/TABLE OF CONTENTS LISTING.
TOC.........PRINTS THE TABLE OF CONTENTS OF A DAL FILE.
TRANSPOSE...MATRIX TRANSPOSTION.
RUN=TRFU....CALCULATES VIBROACOUSTIC TRANSFER FUNCTIONS.
TRNF........CALCULATES TRANSFER FUNCTIONS FOR PRDICT.
RUN=TSTA....STATISTIC COMMAND WITH TRANSPOSED INPUT.
TURP........INTERPOLATE AN ORDERED SET TO A NEW ABSCISSA.
RUN=UCON....VIBROACOUSTIC UNIT CONVERSION OF DATABASE DATA.
RUN=UPDS....COPIES SYMBOLIC ELEMENTS FROM DAL UNIT TO FORTRAN UNIT.
VAPUP.......(Not a command) Procedure for updating VAPEPS code.
VAR.........REMOVE VARIABLES FROM THE VARIABLE TABLE.
RUN=VFIX....DEVELOP A VIBROACOUSTIC TEST SPEC FROM FIXTURE SURVEY OF DATA.
RUN=VTSE....SETS UP VT TERMINAL FOR PLOTTING.
RUN=WLSQ....WEIGHTED LEAST SQUARES.
WRITE.......WRITE VARIABLES/CONSTANTS USING FORTRAN FORMATS.
WTAPE.......WRITE A MATRIX TO TAPE.
XQT.........EXECUTE USER PROGRAM.
```

20.2. AS

COMMAND AND RUNSTREAM SUMMARY - ARITHMETICS AND STATISTICS

NAME	DESCRIPTION
RUN=AJNV	ADJUST THE INVERSE OF A MATRIX.
APE	CURVE FIT TO NTH ORDER POLYNOMIAL.
ASMD	TERM BY TERM ADD, SUB, MULT, DIV.
RUN=AVG	CALCULATES THE AVERAGE BY ROW OR COLUMN.
BUILD(X)	BUILD MATRICIES FROM OTHER MATRICIES.
CALC	CALCULATOR MODE -- OPERATE ON VARIABLES.
CHL	CHOLESKI DECOMPOSTION --INVERSE OF SYMMETRIC MATS.
RUN=CIMV	CALCULATES THE CONFIDENCE INTERAL FOR A TRUE MEAN OF X.
CMOD	COMPLEX TO MODULUS/PHASE OR REAL/IMAGINARY.
FFT	FAST FOURIER TRANSFORM.
FUN	COMPUTES Y AS A FUNCTION OF X (SIN,COS,LOG,ECT).
RUN=GVAL	RETRIEVES DISTRIBUTION VALUE GIVEN SAMPLE SIZE AND CONFIDENCE.
GXI	GET X INTERCEPT - FIND ALL ZERO CROSSINGS.
HELP	LIST COMMANDS, HELP.
INVERT	MATRIX INVERSION.
MAXIN	FINDS THE MAX AND MIN VALUES OF AN ARRAY.
MDIFF	GET THE DERIVATIVE OF A VECTOR.
MGRATE	INTEGRATE A VECTOR OR GROUP OF VECTORS.
MTRAN	MATRIX MULTIPLICATION A*BT = C.
MULTIPLY	MATRIX MULTILPICATION A*B=C.
RUN=MVC	CALCULATE MEAN VARIANCE AN CO-VARIANCE.
NORM	NORMALIZE A MATRIX.
RUN=ORTH	CHECKS THE ORTHOGONALITY OF MATRIX.
PLU	CONVERT PACKED UPPER-LOWER TRIANGULAR TO REGULAR AND VICE VERSA.
POW	RAISE A VECTOR TO SUCCESSIVE POWERS.
RUN=PTMP	CALCULATES AT*B*A TRIPLE PRODUCT.
PVP	PEAK AND VALLEY PICKER -- DIGITIZATION.
RANDOM	CREATAES A MATRIX OF UNIFORMLY DISTRIBUTED RANDOM NUMBERS.
RUN=ROUN	CONVERTS FLOATING POINT TO INTEGER WITH ROUNDING OFF.
RUN=RRAN	CREATES A MATRIX OF RANDOM BETWEEN LIMITS.
RTOI	REAL TO INTEGER CONVERSION AND VICE-VERSA.
SEQUENCE	MAKE A LINEAR OR LOGRITHMIC SEQUENCE OF REAL NUMBERS.
RUN=STAS	Calculates normal and log-normal statistical quantities.
STAT	GET MEAN, STANDARD DEVIATIONS MEAN +/- X SIGMA.
TMUL	MATRIX MULTILPICATION AT*B=C.
RUN=TPVL	Calculates a theoretical percentile value given a mean value.
TRANSPOSE	MATRIX TRANSPOSTION.
RUN=TSTA	STATISTIC COMMAND WITH TRANSPOSED INPUT.
TURP	INTERPOLATE AN ORDERED SET TO A NEW ABSCISSA.

```
 RUN=WLSQ....WEIGHTED LEAST SQUARES.
----------- ---------------------------------------------------------------
```

20.3. AU

COMMAND AND RUNSTREAM SUMMARY - ADVANCED USAGE

```
   NAME                 DESCRIPTION
----------- ---------------------------------------------------------------
 RUN=ADDC....ADD NEW COMMANDS TO SETCOM ROUTINE.
 RUN=BCL.....BUILD COM/LIST ELEMENT FOR SAVING THE COMMAND LIST.
 RUN=BOOT....CREATES A VAPEPS BOOT TAPE FOR THE NAMEDE SYSTEM.
 RUN=BSC.....CREATES A SETCOM ROUTINE FOR DALPRO/VAPEPS TYPE SYSTEM
 CBIT........SEARCH BIT MAP MATRIX FOR SEQUENCE OF BITS.
 CONTROL.....(NOT A COMMAND) INFO ON CONTROL COMMANDS (##) AND PRECODER ($$).
 CNFG........STAND ALONE CONFIGURATION TREE EDITOR (VAPEPS).
 RUN=COPS....WRITES ALL SYMBOLIC ELEMENTS TO A FORTRAN UNIT.
 CORE........SET MAXIMUM CORE LIMIT.
 CPU.........PRINT ELAPSED CPU (SUP) TIME.
 RUN=CPYS....CREATES FILE OF CARD IMAGES FROM MATCHING SYMBOLIC ELEMENTS.
 CRYPT.......CHARACTER TRANSLATION, CRPTOLOGY.
 CSYM........TRANSFER SYMBOLIC ELEMENTS TO/FROM SYSTEM FILES.
 DPTOR.......DOUBLE PRECISION TO REAL, AND VICE VERSA.
 EZOUT.......WRITE EASY FORMATTED SYMBOLIC ELEMENTS.
 RUN=GCLS....ORDERS COMMAND ABBREVIATIONS AND ENTRY POINTS.
 RUN=GDOC....PREPARES A DOC FILE FROM GENERAL AND HELP FILE.
 GLOSS.......GLOSSARY CREATION, INSERTION, CHECKING.
 RUN=GRLT....UPDATES THE 'READ ELTS' ELEMENT IN THE 'EASY' FILE.
 HELP........LIST COMMANDS, HELP.
 RUN=INDE....INTERACTIVE COMMAND INDEX.
 RUN=INTE....INTERACTIVE PROMPTING ROUTINE.
 JPI.........OBTAIN SYSTEM PARAMETERS (SYSTEM DEPENDENT).
 LOOPS.......(NOT A COMMAND) INFORMATION ON LOOPING TECHNIQUES.
 RUN=MEZ.....READS EZOUT FILES AND RELATES THEM TO C=BLOCK REFS.
 RUN=PGLS....ORDERS A GLOSSARY ALPHABETICALY, THEN PRINTS IT.
 RUN=PUTC....PUTS A C=DECK IN ALL (C1,C2)1,0 ELEMENTS.
 RECOVER.....RECOVER ARGUMENTS FROM A RUN COMMAND.
 RLJUST......RIGHT OR LEFT JUSTIFIES HOLLERITH WORDS.
 RTABS.......CONVERTS TAB CHARACTERS TO PROPER SPACING.
 RUN.........INITIATE A VAPEPS SUBROUTINE CALL.
 RUN=SEND....FINDS AND PRINTS OUT MATCHING ELEMENTS.
 RUN=STAC....PRINTS OUT ELEMENTS TO BE TRANSFERED TO ANOTHER SYSTEM.
 SYMIN.......INPUT IMAGES TO A DAL SYMBOLIC (TYPE 5/6) ELEMENT.
 RUN=TCPY....WRITE DAL UNIT TO TAPE UNIT.
 TDUMP.......READ ANY TAPE INTO A DAL FILE.
 RUN=TOCO....ORDER A TOC/TABLE OF CONTENTS LISTING.
 RUN=UPDS....COPIES SYMBOLIC ELEMENTS FROM DAL UNIT TO FORTRAN UNIT.
 VAPUP.......(Not a command) Procedure for updating VAPEPS code.
```

```
WTAPE.......WRITE A MATRIX TO TAPE.
XQT.........EXECUTE USER PROGRAM.
----------- -------------------------------------------------------------
```

20.4. EC

COMMAND AND RUNSTREAM SUMMARY - ELEMENT CREATION/MANIPULATION

NAME	DESCRIPTION
BUILD(X)	BUILD MATRICIES FROM OTHER MATRICIES.
CHOP	SELECT ROWS AND COLUMNS OF A MATRIX.
ECOL	FREE-FIELD DATA ENTRY.
FORMAT	(NOT A COMMAND) FREE-FIELD INPUT DESCRIPTION.
GCOL	READ 12 CHARACTER WORDS, BLOCKED SETS, ECT.
GREAD	READ DATA IN FREE-FIELD FORMAT.
GROUP	COLLECT SEVERAL ELEMENTS INTO ONE (SIMPLIFIED BUILD).
HELP	LIST COMMANDS, HELP.
RUN=LFU	CREATES LOWWER OR UPPER TRIANGULAR, AND FULL MATRICIES.
MODIFY	CHANGE INDIVIDUAL TERMS IN A MATRIX.
RANDOM	CREATES A MATRIX OF UNIFORMLY DISTRIBUTED RANDOM NUMBERS.
RUN=RRAN	CREATES A MATRIX OF RANDOM BETWEEN LIMITS.
SEQUENCE	MAKE A LINEAR OR LOGRITHMIC SEQUENCE OF REAL NUMBERS.
SHUFFLE	CREATE A MATRIX BY PLACING SQUARE MATS AT SPEC. LOCATIONS.
SPLIT	FORM MATRIX BY SELECTING SPECIFIC ROWS AND COLUMNS FROM ANOTHER.
SREAD	READ IMAGES ACCORDING TO INPUT FORTRAN FORMAT.
STORE	STORE A VALUE IN A PARTICULAR LOCATION OF A MATRIX.

20.5. EF

```
     COMMAND AND RUNSTREAM SUMMARY - ELEMENT ,FILE ,SYSTEM CONTROL


   NAME                 DESCRIPTION
----------- ---------------------------------------------------------------------

 ABLD-ABLE...ENABLE AND DISABLES DAL ELEMENTS.
 CHANGE......CHANGE NAME, TYPE, OR NJ FOR ELEMENTS.
 COPY........COPY INDIVIDUAL OR MULTIPLE ELEMENTS.
 COMMAND.....(NOT A COMMAND) COMMAND FORMAT DESCRIPTION.
 CYCLE.......CHANGE DEFAULT CYCLE NUMBERS.
 DALDOC......(NOT A COMMAND) DAL FILE DESCRIPTION.
 EXPAND......CHANGE BLOCK SIZE OF AN ARRAY.
 FILES.......LIST FILES ASSIGNED TO CURRENT EXECUTION
 FIND........LIST TOC LINES FOR GROUP OF ELEMENTS.
 FNAME.......RESET LOGICAL DAL UNIT NAME.
 HELP........LIST COMMANDS, HELP.
 LAST........PRINT TOC LINE OF LAST ELEMENT IN FILE.
 LOCATE......LOCATES A SPECIFIC ELEMENT.
 PACK........REMOVES ALL DISABLED ELEMENTS FROM A DAL FILE.
 RUN=RTOC....PULLS OUT FILES FROM CURRENT VAX DIRECTORY.
 RUN=SNDP....FINDS DAL ELEMENTS THAT HAVE BEEN CHANGED SINCE A DATE.
 TOC.........PRINTS THE TABLE OF CONTENTS OF A DAL FILE.
----------- ---------------------------------------------------------------------
```

20.6. GR

COMMAND AND RUNSTREAM SUMMARY - GRAPHICS

NAME	DESCRIPTION
RUN=GENP	INTERACTIVE PLOTING.
HELP	LIST COMMANDS, HELP.
RUN=PGRM	PLOTS THE SPEC CREATED BY RUN=GRMS.
PLOT	PLOT X VERSUS Y
PSET	PLOT SET --PLOT SUBPROCESSOR.
RUN=VTSE	SETS UP VT TERMINAL FOR PLOTTING.

20.7. HH

```
General Information - Help on HELP

       Summary of commands                  For further information..Type
  -----------------------------------     ------------------------------------
   ?.....Reprints first page of current     * ? and ??..............H?
         help section.
                                            * SKIP..................SKIP
   ??....Prints subtopics of current
         help section.                      * QUIT..................QUIT

   SKIP..Skips to subtopics of the          * Carriage return........CARRIAGE
         current help section.
                                            * RUN= commands..........RCOMMANDS
   QUIT..Takes you out of help.           ------------------------------------

   Carriage return
    .....Go up one help level or go to
         next page of current section.

   RUN=..Gives help on RUN= command
         specified.
  -----------------------------------
```

20.7.1. H?

```
Use of ? and ??

  ?  If you are within the print out of a section (i.e. when the prompt
     ends in a question mark), "?" reprints the current section starting
     at the beginning.  If you have a prompt ending with ">", "?" prints
     the help section indicated by the prompt (see example 1).

  ?? Used to find out what the subtopics are of the current help level
     when you have a prompt ending with ">" (see example 2).  You can
     also use "?" or "??" as the last argument of a string of help subtopics
     to find the subtopics of the specified level (see example 3).

     Examples
              1) HELP HH > ?            - Reprints the information for
                                          the HH section.
              2) HELP HH > ??           - Prints out subtopics of the HH
```

```
                                  section
3) HELP > PREP MODULES ?? - prints out subtopics the MODULES
                                  section of PREP.
```

20.7.2. SKIP

Use of SKIP

SKIP is used only within the print out of a help section to skip to the end of that particular section and print out the subtoptics, if any. If no subtopics exist then you are moved up one level in help. See "Use of ? and ??" for information on how to find out what the subtopics are in other cases.

20.7.3. QUIT

Use of QUIT

QUIT will take you completly out of the help processor except when a subtopic call QUIT is available. If a QUIT subtopic is available then it is treated the same as any other subtopic.

20.7.4. CARRIAGE

Use of carriage return

A carriage return will move you up one level in help when you have a prompt ending with ">" and will take you out of help when you are at the top (i.e when the prompt is "HELP >"). When a carriage return is used when you have a prompt ending with "?" the next page of the current help section is printed out.

20.7.5. RCOMMANDS

Use of RUN= commands

To obtain help at any time on RUN= commands type RUN= followed by the command name. Once one RUN= command has been printed out you will receive a prompt like this:

```
HELP RUN= >
```

When this prompt is displayed the HELP processor assumes that subsequent commands are RUN= commands. A carriage return will put you back to the top level of help even if you were not there when you first typed RUN=.

20.8. IN

General Information - Introduction to VAPEPS

Purpose

VAPEPS (Vibroacoustic Payload Environment Prediction System) software:

- Manages a database composed of acoustic/vibration ground test or flight data obtained from payload components of an Expendable Launch Vehicle or Space Shuttle; the database also allows for storage of payload component physical parameters.

- Has prediction routines (using Statistical Energy Analysis or scaling techniques) to establish the vibroacoustic environment of new payload components.

- Has general data processing routines which perform functions such as: arithmetics, statistics, results display, etc.

- Uses DAL (Direct Access Library) files to store and retrieve data.

20.9. IO

COMMAND AND RUNSTREAM SUMMARY - INPUT/OUTPUT

```
   NAME               DESCRIPTION
----------- ------------------------------------------------------------
 CSYM........TRANSFER SYMBOLIC ELEMENTS TO/FROM SYSTEM FILES.
 ECOL........FREE-FIELD DATA ENTRY.
 FORMAT......(NOT A COMMAND) FREE-FIELD FORMAT DESCRIPTION.
 GCOL........READ 12 CHARACTER WORDS, BLOCKED SETS, ECT.
 GREAD.......READ DATA IN FREE-FIELD FORMAT.
 HELP........LIST COMMANDS, HELP.
 RUN=MANU....WRITES A VAPEPS MANUAL.
 RUN=NEWS....PRINTS OUT NEWS ELEMENT ON RUNFILE.
 PRCRN.......PRINT CORNERS OF A MATRIX.
 PRINT.......PRINT THE CONTENTS OF A ELEMENT.
 RITE........PRINT AN ELEMENT AS A FUNCTION OF A VECTOR.
 SREAD.......READ IMAGES ACCORDING TO INPUT FORTRAN FORMAT.
 SYMIN.......INPUT IMAGES TO A DAL SYMBOLIC (TYPE 5/6).
 TABLE.......PRINT MATRICIES IN REPORT TYPE FORMAT.
 TDUMP.......READ ANY TAPE INTO A DAL FILE.
 WRITE.......WRITE VARIABLES/CONSTANTS USING FORTRAN FORMATS.
 WTAPE.......WRITE A MATRIX TO TAPE.
----------- ------------------------------------------------------------
```

20.10. LS

COMMAND AND RUNSTREAM SUMMARY - LOGIC, SORTING OPERATIONS

NAME	DESCRIPTION
AND	REDUCE TWO LISTS TO ONE LIST OF LIKE ITEMS.
INDEX	REPLACE DATA WITH POSITION IN AN INDEX LIST.
LUMP	DISPERSE A LIST OF VALUES INTO PRESCRIBED SLOTS.
NOT	GET ITEMS NOT PRESENT IN A LIST.
OR	COMBINE TWO LISTS INTO ONE LIST CONTAINING ALL ITEMS.
ORDER	SORT A MATRIX IN ASCENDING ORDER.
SEARCH	FIND VALUES OF A MATRIX WITHIN A GIVEN RANGE.
SSTRNG	SEARCH COLUMNS FOR A STRING OF INTEGERS.

20.11. UG

```
General Information - User Guide

   New User Information...........Type   Information (all users)........Type
 -------------------------------------- --------------------------------------
   * Vapeps operation........OPERATION    * Getting a manual.......RUN=MANUAL
   * Vapeps structure...........VAPEPS
   * Executing Vapeps........EXECUTION
   * Command format............COMMAND
   * Read formats...............FORMAT
   * All about Dal files .......DALDOC

   Advanced User Information
 --------------------------------------
   * Looping techniques..........LOOPS
   * User created routines.........RUN
   * Control and Precoder......CONTROL
```

20.12. UH

General Information - Using HELP

Purpose

Use HELP to display information on VAPEPS functions. While in HELP, use the commands shown below to move through the information.

```
                              HELP commands
Command..Description        -----------------
----------------------------------------------------------------------------
   ?.....Display the beginning of      QUIT..Exit from the HELP processor.
         current help section.
                                       Carriage return (<CR>)..
   ??....Display subtopics of the          ..If you are at the end of
         current help section.               a section, go up one level.
                                             If you are in the middle
   SKIP..Skip to the end of the              of a section, display the next
         current section and display         page.
         subtopics - if any.
----------------------------------------------------------------------------
```

20.12.1. EXAMPLE

General Information - Using HELP - example

Example

Enter the input in the order shown below to get a feel for how HELP works.

```
   prompt > you enter      - what happens
-----------------------------------------------------------
1) HELP UH > ?             - Redisplays the information for
                             the UH (Using HELP) section.
2) HELP UH > ??            - Displays subtopics of the UH
                             section.
3) HELP UH > <CR>          - Bumps you back to the HELP >
                             level.
4) HELP > ?                - Redisplays the main HELP
                             directory.
5) HELP > VD               - Displays a summary of
```

```
                                          Vibroacoustic Database commands.
                 6) HELP > ENTER ??     - Displays the subtopics of the
                                          ENTER command.
                 7) HELP ENTER > EXAMPLE - Displays the information in
                                          EXAMPLE subtopic of ENTER.
                 8) HELP ENTER EXAMPLE> QUIT
                                        - takes you out of HELP back
                                          to the VAPEPS command level.
-short form-     9) ?> HELP PREP BOOK   - Displays the information for
                                          the BOOK section of PREP.
```

20.13. VC

COMMAND AND RUNSTREAM SUMMARY - VARIABLE CONTROL

```
   NAME                  DESCRIPTION
----------- ----------------------------------------------------------------
 CALC........CALCULATOR MODE -- OPERATE ON VARIABLES.
 CONTROL.....(NOT A COMMAND) INFO ON CONTROL COMMANDS (##) AND PRECODER ($$).
 DEFAULT.....SPECIFY DEFAULT VALUES FOR VARIABLES.
 LIST........LIST VALUE OF ALL OR SELECTED VARIABLES.
 SET.........DEFINE THE VALUE OF A VARIABLE.
 VAR.........REMOVE VARIABLES FROM THE VARIABLE TABLE.
----------- ----------------------------------------------------------------
```

20.14. VD

COMMAND AND RUNSTREAM SUMMARY - VIBROACOUSTIC DATABASE

```
  NAME                  DESCRIPTION
----------- ----------------------------------------------------------------
ADMIN.......DATABASE ADMINISTRATION COMMAND.
RUN=ALL.....PRINTS OUT ENTER HEADER AND CHANNEL INFO IN A DAL FILE.
RUN=DATA....RETRIEVES AND AVERAGES DATABASE ACOUSTIC AND VIBRATION DATA.
ENTER.......DATABASE VIBROACOUSTIC DATA INPUT COMMAND.
RUN=ESAV....WRITES ENTER DECK OUT TO A FORTRAN UNIT.
RUN=GLST....LIST RUN=GMOD OUTPUT.
RUN=GMOD....RETRIEVES DATABASE MODULE DATA.
RUN=HEAD....PRINTS OUT ENTER HEADER AND CHANNEL INFO.
HELP........LIST COMMANDS, HELP.
RUN=MPNT....PRINTS A MASTER FILE SUMMARY.
RUN=PICK....RETRIEVES DATABASE ACOUSTIC AND VIBRATION DATA.
PREP........DATABASE EVENT DEFINITION AND STANDARDIZATION COMMAND.
RUN=SAVE....SAVES ALL PREPED EVENTS FROM A DAL UNIT TO A MASTER FILE.
RUN=SPIL....SPILLS ALL EVENTS FROM A DAL UNIT TO A ELEMENT.
RUN=STAT....CALCULATES STATISTICAL INFO ON VIBROACOUSTIC DATA.
RUN=STTS....SAME AS RUN=STAT PLUS TABULATES RESULTS.
----------- ----------------------------------------------------------------
```

20.15. VP

COMMAND AND RUNSTREAM SUMMARY - VIBROACOUTIC PREDICTION

NAME	DESCRIPTION
BIGSEA	Links seperate SEA models together.
RUN=CLFGET	Gets a set of ATA'S from a SEMOD path.
RUN=CLFPUT	Puts a set of ATA'S into a SEMOD path.
CNVRT	SETS UP CONVERSION FACTORS FOR PREDICTIONS.
DAMO	STAND ALONE DATA MODULE EDITOR (VAPEPS).
DENS	CALCULATES MODAL DENSITY.
RUN=EQPL	Calculates SEA parameters for complex panels.
RUN=GETRESP	Stores SEMOD responses on a specified DAL element.
HELP	LIST COMMANDS, HELP.
PRDICT	DOES THEORETICAL AND EXTRAPOLATED SEA PREDICTIONS.
RUN=PRST	CALCULATES STATISTICAL INFORMATION FOR PREDICTED RESPONSE.
RUN=SAVG	Averages the responses from a SEMOD model.
SCALE	EXTRAP2 PREDICTION COMMAND.
SEMOD	Statistical Energy Modeler.
RUN=SHIFT	Shifts SEMOD'S FPAR parameters up or down.
TRNF	CALCULATES TRANSFER FUNCTIONS FOR PRDICT.

20.16. VT

```
     COMMAND AND RUNSTREAM SUMMARY - VIBROACOUSTIC TOOLS


   NAME                 DESCRIPTION
----------- ------------------------------------------------------------
 RUN=BWTH....CREATES A VECTOR OF 1/3 OCTAVE BANDWIDTHS.
 RUN=BLIM....CREATES A VECTOR OF 1/3 OCTAVE BAND LIMITS.
 CONVERT.....Vibroacoustic Unit Conversion.
 RUN=CIMV....CALCULATES THE CONFIDENCE INTERAL FOR A TRUE MEAN OF X.
 RUN=DATA....EXTRACTS AND AVERAGES CHANNEL DATA FROM DATABASE.
 FRQ.........Generates Frequencies.
 RUN=GPSD....CALCULATES PSD FROM A TIME HISTORY.
 RUN=GRMS....VIBRATION SPEC MANIPULATOR AND RMS CALCULATOR.
 HELP........LIST COMMANDS, HELP.
 RUN=LRMS....CALCULATES RMS OF VIBRATION SPEC.
 RUN=LTRP....INTERPOLATES VIBRATION SPEC TO 1/3 OCTAVES.
 RUN=NOML....Normalize baseline response to a new excitation.
 RUN=NORM....NORMALIZES DATABASE VIBROACOUSTIC DATA.
 RUN=OCT3....CONVERT NARROW BAND PSD TO 1/3 OCTAVE PSD.
 RUN=OVER....CALCULATES OVERALL LEVEL OF 1/3 OCTAVE VIBROACOUSTIC DATA.
 SHOCK.......COMPUTE SHOCK SPECTRA FROM TIME HISTORY.
 RUN=STAT....CALCULATES STATISTICAL INFO ON VIBROACOUSTIC DATA.
 RUN=STAS....Calculates normal and log-normal statistical quantities.
 RUN=STTS....SAME AS RUN=STAT PLUS TABULATES RESULTS.
 RUN=TPVL....Calculates a theoretical percentile value given a mean value.
 RUN=TRFU....CALCULATES VIBROACOUSTIC TRANSFER FUNCTIONS.
 RUN=UCON....VIBROACOUSTIC UNIT CONVERSION OF DATABASE DATA.
 RUN=VFIX....DEVELOP A VIBROACOUSTIC TEST SPEC FROM FIXTURE SURVEY OF DATA.
----------- ------------------------------------------------------------
```

BIBLIOGRAPHIC DATA SHEET

1. Report No. NASA CR-180781	2. Government Accession No.	3. Recipient's Catalog No.

4. Title and Subtitle VAPEPS USER'S REFERENCE MANUAL VERSION 5.0	5. Report Date July 1988
	6. Performing Organization Code
7. Author(s) D. M. Park	8. Performing Organization Report No. LMSC/F186312/SSV-2167
9. Performing Organization Name and Address Lockheed Missiles and Space Company 1111 Lockheed Way Sunnyvale, California 94086	10. Work Unit No.
	11. Contract or Grant No. NAS5-29452
12. Sponsoring Agency Name and Address Goddard Space Flight Center Greenbelt, Maryland 20771 Technical Monitor: F. On	13. Type of Report and Period Covered Final
	14. Sponsoring Agency Code

15. Supplementary Notes

16. Abstract

This is the reference manual for the VibroAcoustic Payload Environment Prediction System (VAPEPS). The system consists of a computer program and a vibroacoustic database. The purpose of the system is to collect measurements of vibroacoustic data taken from flight events and ground tests, and to retrieve this data and provide a means of using the data to predict future payload environments. This manual describes the operating language of the program. Topics covered include database commands, Statistical Energy Analysis (SEA) prediction commands, stress prediction command, and general computational commands.

17. Key Words (Selected by Author(s)) Space Shuttle, Vibroacoustics, database, VAPEPS, environment, Vibroacoustic predictions, payloads, SEA, Statistical Energy Analysis	18. Distribution Statement

19. Security Classif. (of this report) Unclassified	20. Security Classif. (of this page) Unclassified	21. No. of Pages 800	22. Price*

*For sale by the National Technical Information Service, Springfield, Virginia 22151.