IMPLEMENTATION AND EXTENSION OF THE
IMPULSE TRANSFER FUNCTION METHOD FOR FUTURE
APPLICATION TO THE SPACE SHUTTLE PROJECT

Volume II - Program Description and User's Guide

By

G. Patterson

Grumman Data Systems Corporation
Bethpage, New York 11714

April 1973

NAS 9-12333

Prepared for

NASA-MANNED SPACECRAFT CENTER
Houston, Texas 77058
NOTICE

THIS DOCUMENT HAS BEEN REPRODUCED FROM THE BEST COPY FURNISHED US BY THE SPONSORING AGENCY. ALTHOUGH IT IS RECOGNIZED THAT CERTAIN PORTIONS ARE ILLEGIBLE, IT IS BEING RELEASED IN THE INTEREST OF MAKING AVAILABLE AS MUCH INFORMATION AS POSSIBLE.
IMPLEMENTATION AND EXTENSION OF THE
IMPULSE TRANSFER FUNCTION METHOD FOR FUTURE
APPLICATION TO THE SPACE SHUTTLE PROJECT

Volume II - Program Description and User's Guide

By

G. Patterson

Grumman Data Systems Corporation
Bethpage, New York 11714

April 1973

Prepared for

NASA-MANNED SPACECRAFT CENTER
Houston, Texas 77058
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Summary</td>
<td>1</td>
</tr>
<tr>
<td>2. Background and Introduction</td>
<td>3</td>
</tr>
<tr>
<td>3. Program Description</td>
<td></td>
</tr>
<tr>
<td>a. ITF Program; Program I</td>
<td>15</td>
</tr>
<tr>
<td>b. Response Program; Program II</td>
<td>21</td>
</tr>
<tr>
<td>4. User Guide</td>
<td></td>
</tr>
<tr>
<td>a. Program I</td>
<td>24</td>
</tr>
<tr>
<td>b. Program II</td>
<td>29</td>
</tr>
<tr>
<td>5. Conclusions &amp; Recommendations</td>
<td>34</td>
</tr>
<tr>
<td>Appendices</td>
<td></td>
</tr>
<tr>
<td>A. Fortran Listings</td>
<td>36</td>
</tr>
<tr>
<td>B. Deck Setup, Sample Data and Output</td>
<td>59</td>
</tr>
<tr>
<td>C. Flow Diagrams</td>
<td>68</td>
</tr>
</tbody>
</table>
1. **SUMMARY**

This volume of the report describes the data processing procedures and the computer programs developed to predict structural responses using the Impulse Transfer Function (ITF) method. There are three major steps in the process:

- Analog-to-digital (A-D) conversion of the test data to produce Phase I digital tapes
- Processing of the Phase I digital tapes to extract ITF's and storing them in a permanent data bank
- Predicting structural responses to a set of applied loads

The analog to digital conversion is performed by a standard package which will be described later in terms of the contents of the resulting Phase I digital tape.

Two separate computer programs have been developed to perform the digital processing:

**Program I**

- ITF Program - extracts ITF's and stores them in the data bank

**Program II**

- Response Program - predicts structural responses to a set of input forcing functions

All coding was initially done in Fortran IV for the IBM 360/75; this program was used to produce the results presented in Volume I. The data bank used was an IBM 2314 Disk Pack, with random access capability. Both programs have been modified to Fortran V for the Univac 1108, using standard 7-track tape as the data bank.

The programs were originally sized for the present problem of 36 applied impulses and 70 response channels. The tape storage version, however, allows a virtually unlimited number of responses and the number of applied impulses can be increased with only minor program revisions.
IMPULSE TESTING

ANALOG TAPES

ANALOG TO DIGITAL CONVERSION

PHASE I DIGITAL TAPES

PROGRAM I

ITF PROGRAM
- CALIBRATION
- ZERO ADJUST
- IMPULSE AREA
- ITF START TIME
- STORE RAW ITF'S, ZEROS AND SCALE FACTORS

PRINT OF ITF'S

PLOT OF ITF'S

DATA BANK STATUS

O'GRAPHS

PRINT RESP.

PLOT RESP.

PROGRAM II

RESPONSE PROGRAM
- SCALES & FILTERS ITF'S
- CALCULATES RESPONSES

DISK PACK OR TAPE PERMANENT ITF DATA BANK

APPLIED FORCES

FIGURE 1

DATA PROCESSING FLOW CHART
2. BACKGROUND AND INTRODUCTION

Analog Tape

Test data was recorded on 108 KHz wideband FM analog tape at 60 ips. This tape has 14 channels, of which one is used exclusively for an IRIG-B time code, one for the 108 KHz reference, and another for the impulse. The ITF's are multiplexed, 10 to a channel, allowing a maximum of 110 ITF's per tape. For the LTA-ll testing, 70 responses were recorded on channels 2-8, the impulse on channels 1 and 12, IRIG-B on channel 14, and the 108 KHz reference on channel 13. Channels 9 and 10 were not used. Time sequencing of events as recorded on the analog tape is schematically shown on Figure 2 and described in the accompanying table.

<table>
<thead>
<tr>
<th>Time (Seconds)</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 - 9</td>
<td>A.C. calibration signal</td>
</tr>
<tr>
<td>9 - 14</td>
<td>Hard zero, gauges shorted</td>
</tr>
<tr>
<td>14 - 19</td>
<td>D.C. calibration - 100% of full scale</td>
</tr>
<tr>
<td>19 - 24</td>
<td>Hard zero, gauges shorted</td>
</tr>
<tr>
<td>24.00 - 24.01</td>
<td>10 ms event pulse - approximately 100% full scale</td>
</tr>
<tr>
<td>24.01 - 24.200</td>
<td>Recorded zero, gauges in circuit</td>
</tr>
<tr>
<td>24.200 - 30.+</td>
<td>Impulse occurs 200 ms after event pulse and data is recorded for at least 6 seconds</td>
</tr>
</tbody>
</table>

Oscillograph playbacks of each ITF was made to verify the data and for later comparison with plots made of the digitized data. For the present contract, 36 impulses were applied to the vehicle; one analog tape was produced for each applied impulse. Sixty-eight active measurements per impulse were recorded for a total of 2448 ITF's.

Analog-to-Digital Conversion

Conversion of the analog tapes was performed by the LM Data Reduction Station (LDRS) using a general A-D conversion program. Seven Phase I digital tapes are produced from one analog tape, with each containing the impulse, and 10 ITF's. Conversion starts at the beginning of the D.C. calibrations and continues for at least 6 seconds. Most of the ITA-ll conversions were
FIGURE 2
ANALOG TAPE Schematic - Response Track
performed for only 3 seconds due to the storage limitation in the earlier version of the computer system. Subsequent modifications have increased storage to 6 seconds. During conversion, the impulse track is sampled at 20,000 samples/sec, and the ITF's at 5000 samples/sec. Formatting of the Phase I digital tape, which must be compatible with the ITF program (Program I), is discussed next.

Phase I Digital Tapes

Phase I digital tapes produced by the A-D conversion are standard 7 track input tapes for the ITF program. Each tape contains the time history of the impulse track identified by the load point, and ten of the 68 associated ITF's. The histories were directly digitized from the analog records starting with the D.C. calibrations and continued for 3 seconds. In order to satisfy the requirements of the ITF program the following specifications must be adhered to for the Phase I digital tape:

- It must be a 7 track binary tape containing two files of data, separated by a standard End of File mark.
- File 1 contains addressing and record size information which is not used by the ITF program. This is generated by the generalized A-D conversion program and is passed over by specifying on the I/O control cards that the second file is to be used as the input data set.
- File 2 contains the time history data written in records, each 4868 characters long. This may be interpreted as:
  
  29,208 bits
  9,736 octal digits
  4,868 BCD characters
  2,434 12 bit data values (4 octal digits)

- The data is most meaningfully interpreted as 12 bit words ranging from 0 to 7777 octal (4095 decimal).
- Each record represents 32 milliseconds of real time.
Words (12 Bits) | Data
---|---
1 - 34 | Status and time values from IRIG-B
35 - 674 | 640 values of impulse track (32 ms x 20 samples/ms)
675 - 834 | 160 values of response 1 (32 ms x 5 samples/ms)
835 - 994 | 160 values of response 2
995 - 2274 | 160 values each of responses 3-10
2275 - 2434 | Dummy values to fill out record

- Each record contains data for 10 response channels; an eleventh response channel could be accounted for but its inclusion would cause some added complexity in the numbering scheme.
- Data should start (Record 1) within ± 3.2 seconds (100 records) of the start of the DC calibrations (i.e., no more than 100 records of zeroes at start, and at least 50 records of DC calibrations).
- There should be at least 75 records (2.4 seconds) of zeroes between DC calibrations and event pulse. Normally, with direct conversion, there is a period of 5 seconds (~ 156 records) between DC calibrations and event pulse. In the present contract work, some conversions were performed with time jumps in this zero area. This is not considered to be a useful economy, but can be accepted by the ITF program.
- Data should be recorded continuously from the event pulse to the end of data. For the LTA-11 only 3 seconds of response was recorded, but present storage capability for the 1108 program is 6 seconds.
- During A-D conversion, the DC calibrations are used as an indication of the full range of values required. Scaling should be such that the DC calibrations fall at approximately 90% of the full digitized scale.
- No sign bit is used, hence the digital value of 0 is equivalent to -100%.

<table>
<thead>
<tr>
<th>Decimal</th>
<th>Octal</th>
<th>Reading</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>-100%</td>
</tr>
<tr>
<td>2048</td>
<td>4000</td>
<td>0</td>
</tr>
<tr>
<td>4095</td>
<td>7777</td>
<td>+99.95%</td>
</tr>
</tbody>
</table>

- Scale factors of all measurements must be supplied to the ITF program to allow eventual conversion to engineering units.
DC calibrations, the event pulse and the impulse are to be recorded in the positive direction during recording of the analog tape and for A-D conversion. Sign changes to account for direction of the impulse relative to the coordinate system are data input to Program II.

Program I - ITF Processing

This program extracts the ITF's from the Phase I tapes, evaluates the zero adjustment and scale factor, locates time zero, and stores the ITF's in the data bank. Plotted output of the ITF's is available for visual inspection and comparison to the oscillograph records; conventional computer printout is also available.

How the scale factors and zero adjustment are obtained is discussed next with the aid of a specific example. (See Figures B-2, -3, -4)

Consider response 1, which is gauge GA-12. It is a force gauge located on the aft interstage truss. It is calibrated prior to testing, and it is determined that its output, when passed through a Preston Filter, with gain set at 1000 and an RC of 100K, is 2.5 volts D.C. at 5360 #. The nominal range of the analog recorder is ± 2.5 vdc. From a sample run it is observed that the output of GA-12 is low; the RC of the Preston Filter is then reset to 470K, at which value the output of GA-12 is 2.5 vdc at 1143 #. Since this output signal is satisfactory, it is duly recorded that 1143 # = 100% full scale = 2.5 vdc. This data defines the engineering units of the measurement. Following an application of an AC calibration signal to the recorder, GA-12 is in the circuit and recording at a nominal zero level for 5 seconds, then a direct input of 2.5 vdc is applied for 5 seconds, followed by 5 seconds of a "hard" zero level during which the gauge is shorted out of the circuit (see Figure 2). An event pulse of 2.5 vdc is then applied for 10 msec and the impulse and recorded response occurs 200 msec later. During the period between the event pulse and the impulse, gauge GA-12 is in the circuit and its output is being recorded. Examination of the Phase I digital tape in this section of the test shows the following (adjusting counts to the range -2048 to +2047):

<table>
<thead>
<tr>
<th>Test Period</th>
<th>Record</th>
<th>Reading (average)</th>
</tr>
</thead>
<tbody>
<tr>
<td>End of DC cals</td>
<td>250 - 300</td>
<td>1617 counts</td>
</tr>
<tr>
<td>Start of hard zero</td>
<td>310 - 350</td>
<td>8 counts (gauge shorted out)</td>
</tr>
<tr>
<td>Event pulse</td>
<td>463</td>
<td>513 counts</td>
</tr>
<tr>
<td>Event to impulse</td>
<td>464 - 468</td>
<td>10 counts (gauge in circuit)</td>
</tr>
</tbody>
</table>
From this we can surmise that:
\[ 1143 \times \# = 2.5 \text{ vdc} = 1609 \text{ counts} \ (1617-8). \]

It is important to note how the hard zero and applied 2.5 vdc pulse are used to obtain the scaling.

GA-12 is recorded at 10 counts, while in the circuit, prior to the application of the impulse. This reading represents 1.4 \#, and is computed from the tabulated values.

\[ (10 - 8) \times 1143/1609 = 1.4 \]

This indicates that the strain gauge, within the supported structure, prior to impulse application is not in a wholly unloaded state. This level of 10 counts is taken as the nominal zero level of the measurement (zero adjustment). Further complications concerning oscillations of the zero level due to swaying of the test article will be discussed later. To complete these observations, following the impulse, GA-12 records oscillations which range between +183 counts and -209 counts and eventually settle down to a small range around 10 counts. These peak values represent:

\[ (183 - 10) \times 1143/1609 = 123 \# \]
\[ (-209 - 10) \times 1143/1609 = -155 \# \]

This procedure is also followed on the impulse track, to obtain the load time history of the impulse. In a later calculation, this data is used to compute the impulse magnitude (area under the load time history) which is eventually used to scale the impulse response to a unit impulse.

With the above discussion in mind, the processing logic of the ITF program may now be considered. Figure C-2 shows a flow diagram of the sequence of operations in which one Phase I tape, containing an impulse track and ten responses, is processed to obtain the ten ITF's. Records are processed sequentially. Note that each record contains:

* 32 milliseconds of real time
* 640 impulse track values (20 samples/ms x 32 ms)
* 160 values per ITF (5 samples/ms x 32 ms)

The programmed procedure is as follows:

1. Nine records are skipped, the tenth is read and the average values of the impulse and response tracks are calculated and saved. In reading records, counts are automatically adjusted to the range -2048 to +2047 (= 100% full scale).
2. Step 1 is repeated until a record is found where the impulse track average value exceeds 1000 counts. This is the region of DC calibration.

3. Every tenth record is averaged and the averages saved, until a record is found whose impulse track average is less than 500 counts. This is the hard zero region following DC calibration.

4. Fifty more records are passed, with every tenth being averaged and saved. The last of these is used to establish the zero level for locating the event pulse.

5. One-hundred fifty records are averaged and saved. Each is examined to locate one where the average of the impulse exceeds the zero established in step 4 by 120 counts. This locates the record containing all or most of the event pulse. This record is checked to determine that the sum of three consecutive values exceeds 2000.

6. Seven records are read, averaged and the averages saved. The last two records are saved in their entirety since they contain the impulse and start of the ITF.

7. The two records containing the impulse are examined further. Using the average of the three previous records as a relative zero, three consecutive values over 200 counts are sought to indicate the start of the impulse.

8. Time zero for the ITFs is taken as the nearest millisecond to the first 200 count value of the impulse.

9. The area of the impulse is computed using the preceding 2 milliseconds as the relative zero level.

10. With the time zero established, values from the ITF tracks are stored from the two records previously read.

11. Additional ITF records are read and the values stored until 6 seconds of response is obtained or the end of the data reached.

12. Values are stored as 1 ms averages, with every 5 points being averaged. Values are stored without adjustment to the zero level or the scale factor.

13. Scale factors for the impulse and ITF's are calculated using the average record values saved from the 20th, 30th and 40th records after the start of DC calibrations and the 10th, 20th and 30th
records after the end of DC calibrations. Engineering unit values of the impulse and ITF tracks corresponding to 2.5 vdc (100% scale) are supplied as input to the program. Coupling the engineering unit value, with the area under the pulse, the final scale factor is computed in engineering units per count for a unit pulse.

14. This scale factor is now stored in a status file along with other information such as ITF duration, impulse area, DC calibration step size, engineering units full scale value, and two zero values, one calculated from the average of the records between the event pulse and the impulse, and the other from the record averages of the ITF's.

15. Printed output includes pertinent record averages throughout the process as well as several records in their entirety.

ITF Storage

The ITF processing subroutine assumes that the ITF’s are stored in temporary, direct access data files on Fastran Drum, referenced by Define File statements in Fortran V under the EXEC-8 operating system. The use of disk pack or tape for permanent storage is discussed later.

A full length ITF is 6000 values in 1 ms increments. ITF values are read from the Phase I tapes as integer values in the range of 0 to 4095. They are adjusted to the range -2048 to +2047 for ease of analysis and interpretation. Five point averages are computed; these are also in the range of -2048 to +2047. To store the 6000 values economically, they are reconverted to the range 0 to 4095, which is represented by exactly 12 bits, and packed three to a 36 bit word before storing as a 2000 word string. The file for ITF storage consists of 3600 - 200 word records. One ITF is ten records. The file holds 360 ITF's, representing ten ITF's for each of the 36 applied pulses. Ordering of the ITF's is by measurement number, i.e., ITF 1, Pulses 1-36; ITF 2, Pulses 1-36, etc. The values stored are not adjusted for the nominal zero and are not scaled to engineering units. The zero level and scale factors are stored in a second file of 10 - 216 word records. This is, in effect 360 - 6 word arrays, ordered by ITF number and loads the same as the ITF's. These six words contain floating point values of ITF duration in seconds, two zero values, the scale factor from counts to engineering units, impulse area, and the impulse scale factor. For computing economy, the ITF's are scaled in Program II when predicting responses.
Program II - Response Program

The response to a set of forcing functions is obtained through the application of the Duhamel integral, as described in Volume I. The numerical solution proceeds as follows:

- An applied load time history is read as input.
- A time interval is specified as input; its selection depends on the expected frequency content of the output.
- Total time of the response is specified as input.
- The applied force time history is interpolated to each time interval and saved as an array.
- The ITF resulting from the application of a specific impulse is read in from the data bank along with zero adjust and scaling information.
- The ITF is passed to the filtering routine as specified by the user. (Filtering is discussed in a later paragraph.)
- The filtered ITF is reduced to a set of averaged points corresponding to the time increment over the total time.
- Zero adjustment is made.
- The response is calculated from:

\[ R(t_k) = CAt \sum_{i=1}^{k} F_i h_{n+1} - i \]

where for a typical gauge reading force,

- \( R \) is in lbs
- \( \Delta t \) is in sec
- \( F \) is in lbs (forcing function)
- \( h \) is in counts (the ITF)
- \( C \) is in lbs/\( \text{lb} \)-sec-count, (scale factor)

- This process is repeated for all other load points, with options to plot and print individual and/or summed responses.

Filtering

ITF filtering is not an integral part of the ITF method, but was introduced to eliminate spurious low frequencies in the measured ITF's arising from the LTA-11 supports. These effects are removed from the ITF's during response...
processing by a filtering routine utilizing a Fast Fourier Transform technique, developed at Grumman.

During performance of the present contract work, these low frequency oscillations were observed in the ITF's and several methods were considered to eliminate them. The most useful was the Fourier Filter technique which did not require visual inspection of the ITF plots and was similarly applicable to the different types of responses (forces, strains, accelerations).

Filtering is done in the response program following retrieval of the ITF's from the data bank, and prior to their use in calculating a response.

The ITF is passed to the filtering routine where the following operations are performed.

- Stored data that defines an ITF time history (maximum = 6000) are time averaged to reduce the number of values to a more workable number (typically 300). This averaging eliminates the higher frequencies, but retains the character of the low frequencies of interest here.
- The reduced number of time history points are transformed to the frequency domain.
- All Fourier coefficients above a specified cutoff frequency are zeroed, and the points transformed to the time domain. They now represent the low frequency content of the ITF as a time history.
- This low frequency time history is then interpolated back to the original sample rate and subtracted from the original ITF to obtain a new function which is considered filtered.
- When filtering, it is not necessary to zero adjust the ITF, since the process removes any biasing in the data.

Zero Adjustment

The need for accurate evaluation of the zero level of the ITF's can be recognized if it is recalled that responses are obtained by integrating the ITF's. An error in the zero level will result in a steadily growing error over the duration of the response. Three methods of zero level evaluation are available, under input control.
• When processing the ITF from Phase I tapes, the calibrated zero level, which exists between the event pulse and the time of impulse application, is determined and stored in the status file. This zero level is of short time duration and may not be sufficiently accurate.

• A second estimate of the zero level is obtained by averaging the ITF over all but the early time period. The time skipped should be equal to several periods of the lowest natural frequency of the test article. This zero level will assure that the response to a constant load does not change its gross level significantly after the first few low frequency cycles. This estimate of zero level is done in Program I and is stored in the status file.

• The third method of zero adjusting is performed automatically whenever the filtering routine is used. This routine removes all frequencies below a specified value, including the zero frequency.

Data Bank

The system was originally designed using an IBM 2314 disk pack, capable of permanently filing 2520 - 80,000 bit ITF's. Since this capacity for permanent storage does not at present exist on a direct access data device at the MSC Univac 1108 facility, the system was redesigned to use tape storage. The primary programming problem, in such a system, is to minimize the number of tape operations for efficient use of computer time. This was done by making use of a smaller direct access temporary data set on Fastran Drum. All programs, and the associated data files have been sized to a 10 response system, with provisions in each program to initialize the data files from tape, and, at the end of an operation with any set of 10 responses, to dump the files back to tape. It is necessary to work with 10 responses simultaneously because 10 responses are grouped on each Phase I tape. When executing the ITF processing program, if several Phase I tapes for the same set of 10 responses are processed sequentially, drum to tape I/O does not consume a major portion of operating time.

When executing the response program it is only necessary to load the drum from tape. Final dumping is not required, since the permanent data base is not modified. All subroutines operate on the basis of ten numbered
responses, while the main programs, which perform the loading, relate the responses to the overall numbering scheme. If large scale permanent direct access storage becomes available, either the tapes can be replaced with the permanent data set and the programs operated exactly as with the tapes, or else the temporary data sets may be increased to full size, with minor program modifications.
Program I, the ITF Program, processes Phase I digital tapes to extract the ITF's, stores them in the data bank, and performs other data bank maintenance and utilization functions.

Figure C-1 shows a simplified flow diagram of the main program. It performs CLEAR, LOAD, DUMP, STATUS and PLOT operations on the data bank, and calls subroutine STORED, which contains the bulk of the programming, to process Phase I tapes. To understand these functions it is necessary to review how the program handles the data bank described in Section 2.

Program I references two temporary, direct access data files through define file statements.

```
DEFINE FILE 1 (10, 216, U, IU1)
DEFINE FILE 2 (3600, 200, U, IU2)
```

File 1 contains data describing the status of 360 ITF's. The ten records pertain to 10 responses. The 216 words per record are an array of (6, 36) real variables, six words for each of 36 load points.

- word 1 - conversion factor, units per count, which converts the value of the stored ITF from digital counts to the engineering units associated with the particular response point
- word 2 - the zero level of the ITF, derived by averaging the analog response for the period between the event pulse and in impulse
- word 3 - the duration of the stored ITF in milliseconds; maximum = 6000
- word 4 - the zero level of the ITF derived by averaging the ITF from 320 milliseconds to the end
- word 5 - scale factor for the impulse track in pounds per count
- word 6 - area of the impulse in pound-seconds

File 2 contains the 360 ITF's, ten records per 2000 word ITF. They are ordered by response point (i.e., Response 1, loads 1-36; Response 2, loads 1-36, . . . . Response 10, loads 1-36).
For a specific problem with 10 responses or less, permanent data files could be used and the processes of dumping to, and loading from tape could be dispensed with. The programs can, however, deal with an unlimited number of response points, by the use of binary tapes to store the contents of data files 1 and 2. Through use of the LOAD and DUMP operations, the contents of these files may be dumped to a binary tape or loaded from a binary tape, with no real restrictions on the number of tapes being used for a problem.

The effects of the five data bank operations are as follows.

CLEAR - The contents of files 1 and 2 are set to zero. This operation is usually performed when the first Phase I tapes for a set of responses are being processed and no LOAD operation is performed.

LOAD - Contents of a binary data bank tape are loaded into files 1 and 2. Presumably, this tape was created by a dump following an earlier run.

The second content of the binary data bank tape is:
record 1 - 100 words - an integer variable representing the tape sequence number - repeated 100 times.
record 2 - 2160 words - the entire contents of file 1.
records 3 - 362 - 2000 words each - the entire contents of file 2, but in 2000 word records rather than 200.

DUMP - Contents of files 1 and 2 are dumped to binary tape to create a new or updated data bank tape

STATUS - prints the contents of file 1

PLOT - plots or prints specified ITF's, stored in file 2.

The sequence number on the binary tapes is used to insure that the proper binary tape is mounted, since a number of them will be used for the typical job. While most of the internal logic of the program deals with ten responses, numbered 1 to 10, an overall numbering system should be employed, whereby the responses are numbered consecutively from 1 and are stored with tape number 1 containing responses 1-10, tape 2 containing 11-20, etc. Input data pertaining to response numbers always refer to the overall numbering scheme. This is converted by the program to a tape sequence number and the internal number range 1-10.
An additional storage operation was written into the program in order to transfer data between Grumman's disk pack system and MSC's tape storage system. This operation is:

LOADGQ - loads an existing BCD tape containing a number of ITF's.

The END operation causes a normal termination of the program.

Processing of Phase I Tapes

The RUN operation is used to process Phase I tapes. This function is performed primarily in subroutine STORED which also calls RITREC, READ7, AVG and UNPACK. Figure C-2 shows a flow diagram for subroutine STORED. The basic sequence of operations in processing the Phase I tape is covered in Section 2. As each record of 32 milliseconds is read, it is separated into impulse track values and 10 sets of response track values. The average value of each of these sets is determined for each record examined. Only these averaged values are used until the actual impulse is encountered and the responses start.

While examining records to locate the DC calibrations and event pulse, only the impulse track is examined since events occur simultaneously on each track. Once the record containing all or most of the event pulse is located, the impulse can be readily found. It occurs 200 ms after the event pulse and must be in the 6th or 7th record following the event pulse. The first 5 of these are read and their averages saved to use in determining zero levels. The 6th and 7th are examined to locate the impulse.

A temporary zero level for the impulse track is established using the average value of the three previous records. The impulse is located by finding three consecutive values, each exceeding the average by 200 counts. A new zero level is established to use in calculating the impulse area. It is based on the 40 values preceding the start of the impulse (2 ms). The first and last points on the impulse whose values exceed 50 are located, and a Simpson's Rule integration performed between them. If the number of intervals is even, the first interval is evaluated by the Trapezoidal Rule. One additional point on each end of the impulse is now used to calculate the areas of the end pieces.
Scale factors for the impulse and response tracks are calculated and
the impulse scale factor is used to get the impulse area in lb-seconds. It
should be noted that for this problem, there are 7 phase I tapes for each
impulse, and that the impulse track on each of them is derived from the same
analog track. Any differences between them are due to the precision of the
A-D conversion, and comparison of the impulse areas can serve as a check on
the A-D process.

At this point, 64 ms of data, is present in core. The starting point
of the impulse is known, and this is adjusted to the nearest millisecond,
to define time zero for the responses. The remaining task is to transfer
data from the response tracks of the Phase I tape to the data file in which
the ITF's are stored. To do this, the intermediate array IDISK is established,
dimensioned to (600, 10). It holds 600 ms of ITF data for the 10 responses
on the phase I tape. Records are now read into core through the use of the
JREC array, converted by averaging from 5 points/ms to one point/ms and
transferred to the IDISK array. The process is started by transferring the
appropriate values from the two records containing the impulse. When the
IDISK array is filled, it is packed down to (200, 10) since ITF's are stored
as three values to a word. The 200-word columns are then stored in the
appropriate records of file 2 and the process repeated. Processing continues
until 6000 milliseconds of data is stored, or the data runs out. A record of
the number of milliseconds of valid data is kept for inclusion in the STATUS
file (File 1). When processing of the phase I tape is completed, the status
items are loaded into File 1, summary output is written, the tape is rewound,
and control is returned to the main program.

A summary of the subroutine calls follows.

SUBROUTINE STORED (ITAPE, IRESP, IRESPP, ITEST, FULLI, FULL, NG)

ITAPE = fortran unit number on which phase I tape is mounted
IRESP, = range of response numbers to be stored, inclusive. They
IRESPP are internal response numbers in the range 1 to 10
ITEST = a value (from 1 to 5) that controls the number of records
written as printed output
FULLI = scale factor in pounds per full scale for the impulse track
FULL = 10 item array with the scale factors in engineering units per
full scale for the 10 response tracks
NG = a counter set to zero after successful completion

SUBROUTINE RITREC (JB, IB, IC)
JB = record number
IB = 2400 word array containing the impulse and response values
from the record adjusted to the range of -2048 to +2047
IC = (160, 11) array containing the responses.
This routine prints the contents of one phase I tape record.

SUBROUTINE READ7 (JTAPE, IREC1, IREC2, IAVG, KS, NG)
JTAPE = fortran unit number
IREC1 = 34 member array - not used
IREC2 = 2400 member array
IAVG = 11 by 500 array
KS = record number
NG = counter
This routine is used to read one record from the phase I tape. The
actual read is performed by the MSC library routine MREAD, used to read non-
standard tapes. MREAD is called in UNPACK which is called by READ7. The 640
word set of impulse values and 11 to 160 word sets of response values are
loaded into IREC2.
Further processing is under control of the input, NG.
NG = 0 for no further processing
NG = 1 for impulse track only
NG = 2 for responses
NG = 3 for both
2048 is subtracted from each word to modify the range of data from (0 to
4097) to (-2048 to +2047).
Averages are performed on the impulse and responses and stored in the
IAVG array according to record number.
SUBROUTINE AVG (IREC, IAVG, NG)
IREC = 2400 values of tape record
IAVG = 11 averages to be calculated
NG = same as in READ7
Averaging described in READ7 is performed.

SUBROUTINE UNPACK (IN1, IREC2, ISTAT)
IN1 = fortran unit number
IREC2 = 2400 values to be returned
ISTAT = status of read
As described in Section 2, a phase I tape record contains 2436 12 bit words. These are read by MREAD into an array of 812 - 36 bit words. The last 800 of these are unpacked, 12 bits at a time and stored in IREC2.

SUBROUTINE CRUNCH is only used to decode the special tape used to transfer ITF's to the data bank for demonstration purposes. It may be replaced with a dummy routine, or reference to it in the LOADGQ section of the main program may be deleted.

SUBROUTINE PLOT1 (IFILE, INN)
IFILE is a work area of 3000 words supplied by the main program
INN array of 8 control constants
PLOT1 reads ITF's from data file 2 and either plots them, prints them or both.
Plot arrays X and Y are completely set up, but no calls to plotting hardware are supplied.
Provision is made for calling plot initialization and plot termination routines.
Program II, calculates the structural response to a set of forcing functions. It consists of a main program, a filter routine, FILT, a fast fourier transform routine, FAST1, and a plot routine, PLOTB. Flow diagrams for the main program and the filter routine are shown in Figures C-4 and C-5.

The main program uses two data files which are identical to those used in Program I. They contain ITF's for 10 response points and 36 load points. Permanent storage of the ITF's is on the binary data bank tapes created by Program I. The appropriate tape must be read into the data files at the start of execution. In order to run program II it is necessary to have processed, for the responses desired, those load points at which forcing functions are to be applied. Other load point ITF's need not have been processed, since the data bank tape contains dummy values where none have yet been supplied, so as to keep proper sequencing of data. Input data to the program consists of a set of control cards with a function name in columns 1-6 followed by additional data.

For the first pass through the program the order of data is fixed, but additional passes can be made on one run, with some variation allowed. In addition, a set of tables of the forcing functions must be supplied.

The function names on the control cards, in normal input order are LOAD, TIME, FORCE, RESP, PLOT and CYCLE. Forcing function tables follow the force card.

Except for recycling for multiple problems, the flow of the program is straight-through, and is best described in relation to the input data.

The LOAD card specifies an I/O unit number and response numbers and results in a data bank tape being read into Files 1 and 2. All features of this operation are identical to the LOAD operation in program I.

The time card specifies the number of points, or time slices required by the analysis, and their spacing in milliseconds. The number of points should not exceed 1200 and the total time should not exceed 6000 ms.
While the ITF's are stored at 1 ms time intervals, it is not necessary to evaluate the structural responses at 1 ms intervals. Selection of the calculation interval should be based on system frequencies, hence if the highest system frequency is 50 cps (20 ms period), spacing of 4 or 5 ms will adequately represent the results. The forcing functions will be evaluated at the specified times and the ITF's averaged to that time interval before performing the Duhamel integration.

The force card is the principle heading card for the forcing functions. It is followed by a set of cards describing the forcing function time histories at each load point required by the problem. To minimize the effort in preparing data for the forcing functions, all forcing functions are described for the same time points. The time scale is described on the FORCE card as a sequence of points at a fixed time interval, followed by another set of points at a different time interval, up to 6 sets. Hence a 500 ms time scale might be specified as 20 points at 2 ms, 6 points at 10 ms and 4 points at 100 ms.

An array is established giving the time value for each point on the tabulated forcing function. The force card is followed by a number of sets of forces for each load point required. A heading card for each set specifies the load point number and a scale factor to be applied to all forces in the set. The scale factor can be used to reduce the number of digits required to specify the forces, change the sign of the force and to conveniently scale the force without modification of the entire table. The heading card of each set is followed by cards specifying the forces in pounds at the previously established times. Any number of tables may be read, with read-in of tables ending when a blank heading card is encountered. Any forces not specified will be treated as zero by the program. As each forcing function is read in, it is interpolated to a set of values corresponding to the time scale established on the time card and the resulting array is stored in a direct access data file for later use.

Following the tables, a RESP card specifies which responses are to be processed and describes the filtering which should be applied to the ITF's. A PLOT card follows, which specifies whether or not the structural responses should be plotted.
For each response to be evaluated, and for each load point, the required ITF is read from file 2 and unpacked. The ITF status array is read from file 1. If no filtering is required, the zero adjustment value is obtained from the status array. If filtering is required, the ITF is passed to the FILT subroutine which removes specified low frequencies. The filtered or unfiltered ITF is now averaged over the time intervals specified on the time card and zero adjusted. The required forcing function is used from the direct access data file, and the Duhamel integral is evaluated. Finally, scale factors for the ITF and the forcing function are applied. The result is the structural response at one response point to one forcing function. The operation is repeated for each forcing function, with each response being printed and/or plotted, and accumulated in a summed response array which eventually gives the total response at the point to all the supplied forcing functions. The summed response is printed and/or plotted.

After all specified responses have been evaluated, a CYCLE card allows the program to recycle to different places in the input sequence to perform additional operations. The forcing function tables do not have to be repeated if they are unchanged.

The filtering technique, described in Section 2, is applied in subroutine FILT. The sequence of operations closely follows that description. A feature of the filtering which needs further explanation is the apparent elimination of frequencies from the high end of the spectrum as well as the low end. As an example, consider a case of 100 points at spacing of 10 ms, for a 1 second ITF. The fast Fourier transform fits this function with frequencies range from 0 cps to 99 cps in steps of 1 cps. In order to eliminate frequencies at or below 10 cps the first 11 and the last 10 coefficients would be set to zero. The last 10 are zeroed because the apparent fitting of 90-99 cps does not really evaluate this frequency content (there is only 1 data point per cycle). It does in fact produce a fit for the low frequencies 1-10 cps through a process known as aliasing. This is a characteristic of the fast Fourier transform technique, and is taken into account by setting to zero the high order coefficients as well as the low.
The ITF processing program performs seven functions in addition to processing Phase I tapes. Each operation is controlled by a single input card, containing the function name and additional information when required. The functions, which are explained in the Program Description, are CLEAR, LOAD, DUMP, STATUS, LOADGQ, PLOT and END. Phase I processing is evoked by the function RUN and requires an additional data card. Tape unit logical numbers are under control of the input and should conform to the EXEC-8 control cards for the run.

Input Data

Each operation is evoked by a single function card (2nd card required for RUN), with the following format.

Function Card - format (A6, 10I4)

| col 1-6  | function name, left adjusted |
| col 7-10 | INN(1)                      |
| col 43-46| INN(10)                     |

appropriate integer data as described below

CLEAR - no values of INN required.

All data files are cleared.

Usually performed before creating original data tape of a series.

LOAD - INN(1) fortran unit number of data bank tape to be loaded.

INN(2) number of first response on data bank tape (normally 1, 11, 21, 31, etc.) This is used to check tape sequence number.

DUMP - INN(1) fortran unit number of tape to which files will be dumped.

INN(2) number of first response of the 10 to be dumped (1, 11, 21, etc.). This is used to determine the sequence number.
STATUS - INN(1) number of first response in a sequence, whose status is to be printed.

INN(2) number of last response in sequence. It is assumed that appropriate responses are loaded in files.

LOADGQ - INN(1) fortran unit number of tape to be loaded. This is a special tape for transferring data from the Grumman data bank to the MSC program.

PLOT - INN(1), INN(2), INN(3), INN(4)
Routine prints or plots ITF for responses INN(1) through INN(2) for loads (impulses) INN(3) through INN(4)

INN(5) spacing for prints or plots in milliseconds
INN(6) number of points to be printed or plotted

INN(7) = 1 plot only
= 2 print only
= 3 plot and print

INN(8) = 1 open plot routine for this set of plots
= 2 close plot routine after these plots
= 3 open and close plot routine

This is used to initialize or terminate plot tapes as required by system. They are not opened or closed automatically by routine since several PLOT calls may be made on any one run.

END - no values of INN required - terminates run.

RUN - INN(1) fortran unit number of Phase I tape to be processed.

INN(2) load (impulse) number

INN(3) number of first response to be processed
INN(4) number of last response (inclusive) to be processed

INN(5) = 0 minimum number of records
| printed in output
= 5 maximum number printed

RUN card must be followed by a single data card containing the full scale calibration values for the ten responses being processed, and the impulse track format (11E7.0)

col 1-7 impulse calibration (lbs at full scale)
Sample Data Setup

Figure B-1 shows a data sequence in which the primary operations are to process two Phase I tapes for the responses 1-10 for impulses 5 and 6.

The exact operations are:
1 Load responses 1-10 from data bank tape mounted as unit 10. This has been created on a previous run.
2 Print status of responses 1-10.
3 Process Phase I tape on unit 11. It is an impulse 5 data tape - responses 1-10. Calibration data card follows.
4 Process Phase I tape on unit 12 - impulse 6, responses 1-10. Calibration data card follows.
5 Print and plot the stored responses 1-10, impulses 5-6.
6 Print status of responses 1-10.
7 Dump files to unit 20 to create an updated data bank tape with impulses 5-6, responses 1-10 added or corrected.
8 End run with normal termination.

Sample Deck Setup

A deck setup for the preceding data processing operations is shown in Figure B-1.

1 Fastrand assignment for temporary file 1, 2 tracks.
2 Fastrand assignment for temporary file 2, 7 positions.
3 Data bank tape on unit 10 XXX = tape label
4 Phase I tape on unit 11 XXX = tape label
5 Position to file 2
6 Phase I tape on unit 12 XXX = tape label
7 Position to file 2
8 New tape on unit 20, to be saved
9 Fortran program ITF1 and subroutines with appropriate FOR cards
This setup does not include any control cards for plotting.

Output Data

Primary printed output from the program consists of full listings of some records from the Phase I tapes, a summary of the average values of a member of records, listing of a tape status and printed ITF's.

Phase I Tape Records

Figure B-2 shows a 3 page list of a record from a Phase I tape. The record covers 32 milliseconds, indicated on extreme left. The impulse track has 20 samples per millisecond and the responses have 5 samples/ms. Responses are always numbered 1 to 10 regardless of true response numbers associated with the tape. The numbers displayed are in counts which range from -2048 to +2047. Some notable features of the record shown are:

1. The impulse appears during the 26th ms of the impulse track and can be estimated roughly to .2 count-seconds 1.6 lb-seconds based on a calibration of approximately 12,500 lbs = 1600 counts.

2. Response 11 is a dummy.

3. Response 8 shows an oscillation of 16 ms period (60 cps), probably the result of electrical interference. It is removable by filtering but makes the estimated zero unreliable. Status output for this response shows a larger discrepancy between the zero derived from pre-impulse records and the zero obtained by ITF averaging than is normal.

4. 43 numbers at the bottom are 32 - 1 ms averages for the impulse track, one 32 ms impulse average and 10 - 32 ms averages for the responses.

A summary of record averages at the end of processing a tape shows the average value in counts of selected records for the impulse and 10 response tracks (Figure B-3).
Output from the STATUS operation presents the six items in the status file for the impulses and responses requested. The six items are:

1. Conversion factor - engineering units/count.
2. Zero adjustment - derived from region between event pulse and impulse.
3. Duration of stored response in milliseconds.
4. Zero adjustment - derived from record average of response, omitting first 10 seconds (320 ms).
5. Impulse scale factor - lbs/count.
6. Impulse area in lb-seconds.

Printed output from the PLOT operation is in zero adjusted counts as shown in Figure B-4 according to specifications on the PLOT function card.

Plots

Actual calls to plotter software have been omitted since plot routines under EXEC-8 were not wholly operational at time of programming. Furthermore, final layout of plotted output cannot be properly determined until a specific application is undertaken. These plots are considered to be intermediate output, for checking purposes, and are best designed to match oscillograph playbacks of the original analog tapes. The arrays required for plotting are set up in the PLOT1 routine and correspond to the printed output.
The response program computes the structural response to a set of forcing functions, applied at one or more load points. The data required to produce one response consists of the time histories of the forcing functions and the set of ITF's relating the load points to the response point. Output from the program is printed or plotted structural response time histories.

Input Data
1. LOAD card FF, IU, IR1, IR2 (A4, 2X, 314)
   col 1-4 FF - 'LOAD' (literal)
   7-10 IU - tape unit on which data bank tape containing ITF's is mounted
   11-14 IR1 - first ITF number on data bank tape
   15-18 IR2 - last ITF number on tape (IR2 = IR1 + 9)
2. TIME card FF, NPTS, NMS (A4, 2X, 214)
   col 1-4 FF - 'TIME' (literal)
   7-10 NPTS - number of points to be calculated in the resulting time histories
   11-14 NMS - spacing, in milliseconds of the response points
   NPTS = 25,0, NMS = 2 results in a time history of 500 ms - 250 points, every 2 ms.
3. FORCE card FF, (MT(I), DT(I), I = 1, 6)
   (A5, 5X, 6(I5, F5.0))
   This card establishes the time scale for the set of forcing functions which will follow. It assumes a set of values will be given at a fixed time increment, a second set at another time increment, and so forth, up to six sets if required (e.g., 50 points at 2 ms, 10 points at 10 ms, 1 point at 100 ms).
   col 1-5 'FORCE' (literal)
   11-15 MT(1) number of points in first set
   16-20 DT(1) increment in ms of first set
   21-25 MT(2) number of points in second set
26-30 DT(2) increment in ms of second set

76-70 DT(6) increment in ms of 6th set

If MT(1) = 0, no forces are read

4. Forcing Function Tables

A set of force time histories for each forcing function must be supplied. The time scale for the forces is supplied by the FORCE card. The number of values required is equal to the sum of MT(I) on the force card. Forcing functions may be supplied for any or all of the load points for the problem. Those not supplied will be treated as zero. The table for each forcing function consists of one heading card followed by the required number of forces, 16 to a card. (If MT(1) from the FORCE card is zero, no tables are read.)

a. Heading Card IL, SCL (I2, F8.0)

Col 1-2 IL load point number

3-10 SCL scale factor to be applied to all forces in table.

This may be used to change the sign of the force, or reduce the number of digits required in the tables (do not omit).

b. Force cards (FEMP(I), I = 1, MTT) (16 F5.0)

As many values of the force table as specified by the FORCE card, 16 to a card in 16 F5.0 format. Force is in lbs.

\[ MTT = \sum_{I=1}^{6} MT(I) \]

c. End of tables card

A blank card is required at the end of the tables to terminate the table read-in.

5. RESP card FF, IRA, IRB, IZFIL, FREQ (A6, 3I4, E12.0)

This card specifies which responses are to be calculated and how they are to be filtered.
col 1-4 FF - 'RESP' (literal)
7-10 IRA - number of first response in sequence to be calculated
11-14 IRB - number of last response in sequence to be calculated
15-18 IZFIL - 1 do not filter; use zero adjustment calculated from
time before the impulse
2 do not filter; use zero adjustment calculated from
ITF record averages
3 use Fourier Filter
19-30 FREQ - when Fourier Filter is used, eliminate all frequencies
below FREQ in cycles/sec.

6. PLOT card FF, NPL (A4, 2X, I4)
   col 1-4 FF - 'PLOT' (literal)
   7-10 NPL - 0 no plots
           1 plot total response
           2 plot individual response to each load as well as
total response

7. CYCLE card FF, NCY (A4, 2X, I4)
After the PLOT card is read, the set of responses specified on the
RESP card are evaluated and the results printed. The program may now
be made to recycle to a point in the input stream to read data for a
new set of cases with a minimum of data repetition.
   col 1-5 FF - 'CYCLE' (literal)
   10 NCY 1 return to beginning of data LOAD card
           2 return to TIME card
           3 return to FORCE card
           4 return to first force header
           5 return to RESP card
           6 terminate program

If the program recycles to the LOAD or TIME card, and the forcing
functions are the same as for the previous run, setting MT(1) = 0 on
the FORCE card will by-pass reading of the forcing functions and use
the values from the previous case.
Sample Deck and Data Setup

Figures B-5, -6 show a sample deck setup with data.

1. Fastrand temporary data file assignments for Files 1, 2 and 3.
2. Binary data bank tape containing ITF's for response points 1 through 10.
3. Fortran decks
4. Map cards
5. Load card specifying fortran unit 11, responses 1 through 10 (see item 2).
6. Time card specifying the responses are to be calculated as 200 points every 2 milliseconds, for a 400 ms duration.
7. Force card specifying that the forcing function values are given at 2 ms intervals for 63 values (126 ms) with a final point at 1126 ms.
8. Forcing functions for 2^4 load points. Each set contains 6^4 values (see item 7). Scale factors for all 2^4 are unity, with sign changes to account for coordinate system definition. Forces are in lbs.
10. RESP card specifying responses 1 through 4, to be run with Fourier Filtering, eliminating frequencies below 5.2 cycles/sec.
11. Plot card specifies omit plots.
12. Cycle card specifies terminate run.

Deck setup does not include control cards for plotting.

Output

Output from the program consists of tables of the forcing functions, interpolated to the time interval specified on the TIME card, and listings of both the responses to each individual forcing function and the summed response to the entire set of forcing functions. The units of the responses correspond to the units originally specified by the engineering units calibration values supplied in Program I when converting the Phase I tapes.
Plotting

As in Program I, a plotting program is provided which does not contain actual calls to the plotting software. Arguments supplied to the skeleton routine are: IR, IL, MPT, NMS, Y.

IR  = response number
IL  = load number
MPT = number of points provided
NMS = spacing of points in milliseconds
Y   = array contain MPT values of the response in appropriate engineering units - corresponds to the printed output.
5. CONCLUSIONS & RECOMMENDATIONS

Both the ITF program and Response program operate with a reasonable degree of efficiency in their present form (i.e., tape data base), provided the operations are sequenced to avoid excessive loading from and dumping to tape. This limitation could be removed by the use of a direct access storage device as the permanent data bank, as is the case for the Grumman system.

While the program handles an unlimited number of response points, the number of load points is presently set to 36. This number can be easily increased by changing the DEFINE FILE statements, appropriate DIMENSION statements and the counter, NI, at the start of each program.
**FORTRAN Listings**

**IMPULSE TRANSFER FUNCTION – ITF PROGRAM 1**

TO PROCESS PHASE 1 DIGITAL TAPES AND STORE ITF'S

**MAIN PROGRAM**

```fortran
COMMON IIN, IOUT, NR, NI, ILOAD
IFILE DIMENSIONED TO 6*NT (MINIMUM OF 3000)
DIMENSION IFILE(3000)
INTEGER FUNC, FUNCX
DIMENSION FULL(10)
DIMENSION STAT(636)
DIMENSION INN(10), FUNC(10)
DIMENSION NTEST(8)
DATA FUNC /'CLEAR', 'LOAD', 'DUMP', 'RUN', 'STATUS'/

1 'LOAD','PLOT', 'DUMMY', 'DUMMY', 'END'/

900 FORMAT(A6,10I4)
901 FORMAT(//' FUNCTION CARD - ['A6,10I4])
902 FORMAT(// 'CLEAR FUNCTION COMPLETE')
903 FORMAT(// 'FILES LOADED FROM UNIT' *** RESPONSES', I3, '-', I3)
904 FORMAT(// 'FUNCTION NOT IDENTIFIED')
905 FORMAT(// 'FILES DUMPED TO UNIT' *** RESPONSES', I3, '-', I3)
906 FORMAT(11E7.0)
907 FORMAT(1H1/' _RESPONSE NO. *I4/)
908 FORMAT(I4*6F12.4)
909 FORMAT(// 'ERROR IN INTEGER FIELDS'*** I4)
910 FORMAT(1H1/ 'EXECUTE PROGRAM I - ITF PROCESSING')
911 FORMAT(// 'ERROR, TAPE MOUNTED IS IS FOR SET', I4)
912 FORMAT(// 'TERMINATE RUN')
913 FORMAT(12J5F10.6)
914 FORMAT(3000I16)
915 FORMAT(// 'PLOTTED IMPULSES', I4, '-', I3, ' RESPONSES', I4, '-', I3)
   ICV(I1)=((I1-1)/NR+1)
   CALL MFC(NTEST)
   SET I/O DEVICES
   IIN=5
   IOUT=6
   SET FILE SIZE
   NUMBER OF RECORDS IN FILES=NT
   FILE 1 NR RECORDS OF 6*NI WORDS
   FILE 2 10*NR*NI RECORDS OF 200 WORDS
   DEFINE FILE 1(10*216+U,IU1)
   DEFINE FILE 2(3600+200+U,IU2)
   NR IS NO. OF RESPONSES PER TAPE
   NI IS NO. OF IMPULSES IN SYSTEM
   NR=10
   NI=36
   NT=NR*NI
   NTI=6*NI
   NTT=6*NT
   NTTT=10*NT
   IZ=0
   WRITE(IOUT,910)
   READ FUNCTION CARD
   1 READ(IIN,900) FUNCX*INN
   WRITE(IOUT,901) FUNCX*INN
   CHECK FUNCTION
   DO 4 IFUNC=1,10
```

**Figure A-1**

Program I – Fortran Listings
IF(FUNCX.EQ.FUNC(IFUNC)) GO TO 6

4 CONTINUE
ILLEGAL FUNCTION
WRITE(IOUT,904)
GO TO 1000
PERFORM FUNCTION
6 GO TO (100,200,300,400,500,600,700,800,950,1000),IFUNC
CLEAR BOTH FILES
100 CONTINUE
DO 7 J=1,10
7 WRITE(IOUT,902)(IZ*I,NTI)
DO 102 J=1,NTTT
102 WRITE(IOUT,902)(IZ*I=1,200)
WRITE(IOUT,902)
GO TO 1
LOAD BOTH FILES FROM TAPE
200 CONTINUE
IN1=INN(1)
IN2=INN(2)
IN3=IN2+9
CHECK TAPE SEQUENCE NUMBER
ICODE=ICV(IN2)
REWIND IN1
READ(IN1) (ICOD,I1*1100)
IF(ICODE.EQ.ICOD) GO TO 201
WRITE(IOUT,911) ICOD
REWIND IN1
GO TO 1000
201 READ(IN1)(IFILE(I),I=1,100)
DO 203 J=1,10
I1=NTI*(J-1)+1
203 WRITE(IOUT,903)(IFILE(I),I=1,10)
DO 202 J=1,NTTT,10
READ(IN1)(IFILE(I),I=1,2000)
DO 202 JJ=1,10
LL=200*JJ
L=LL-199
JJJ=JJ+J-1
202 WRITE(IOUT,903)(IFILE(K),K=L,LL)
REWIND IN1
WRITE(IOUT,903) IN1,IN2,IN3
GO TO 1
DUMP BOTH FILES TO TAPE
300 CONTINUE
IN1=INN(1)
IN2=INN(2)
IN3=IN2+9
REWIND IN1
WRITE TAPE SEQUENCE NUMBER
ICOD=ICV(IN2)
WRITE(IN1)(ICOD,I=1,100)
DO 301 I=1,10
L=1*NTI
K=L-NTI+1
301 READ(IN1)(IFILE(J),J=K,L)

Figure A-1 (Cont'd)

Program I - Fortran Listings
WRITE(IN1) (FILE(I), I=1, NTT)  
DO 302 J=1, NTT; 10  
DO 303 JJ=1, 10  
LL=200*JJ  
L=LL-199  
JJ=JJ+J-1  
303 READ(2. JJ) (FILE(K), K=L, LL)  
302 WRITE(IN1) (FILE(I), I=1, 2000)  
REWRITE IN1  
WRITE(IOUT, 905) IN1 IN2 IN3  
GO TO 1  
PROCESS A PHASE 1 TAPE AND STORE IN FILES  
400 CONTINUE  
ITAPE=IN1  
ILOAD=INN(2)  
IRESP=INN(3)  
IRESPP=INN(4)  
ITEST=INN(5)  
IRX=IRESP-(1/IN1)*NR  
IRXX=IRX*IRESPP-IRESP  
IF(IINN2.LT.1.OR.INN2.GT.1.NI) GO TO 440  
READ(1,IN906) FULL IFULL  
CALL STORED(ITAPE, IRX, IRXX, ITEST, FULL, FULL, NG)  
GO TO 1  
440 WRITE(IOUT, 909) (INN(I), I=1, 4)  
GO TO 1  
PRINT STATUS OF TAPE  
500 CONTINUE  
IN1=INN(1)  
IN2=INN(2)  
DO 510 IR=IN1, IN2  
IRX=IR-(1/IN1)*NR  
WRITE(IOUT, 907) IR  
READ(1, IRX) (STAT(I,J), I=1, 6), J=1, NI  
DO 506 IMP=1, NI  
506 WRITE(IOUT, 908) IMP*STAT(I, IMP), I=1, 6  
510 CONTINUE  
GO TO 1  
LOAD FILES FROM GRUMMAN BCD TAPE  
600 CONTINUE  
IN1=INN(1)  
CALL MRRN(IW, IN1)  
DO 610 I=1, 10  
CALL MRWND(IN1)  
DO 610 I=1, 10  
611 CONTINUE  
IF(NTEST(6).EQ.1) GO TO 611  
CALL CRUNCH(IFILE, 360)  
WRITE(6, 930) (IFILE(J), J=1, 360)  
930 FORMAT(1H1/10(1X, 12))  
DO 613 K=1, 36  
KK=10**K-9  
DECODE(612, IFILE(KK)) (STAT(J, K), J=1, 6)  
612 FORMAT(16F10, 6)  
613 CONTINUE  
DO 710 IFIX=1, 36  
710 STAT(3, IFIX)=3000.  

Figure A-1 (Cont'd)  
Program I - Fortran Listings
Program I - Fortran Listings
FOR IS STORED

SUBROUTINE STORED(ITAPE, IRESP, IRESPP, ITEST, FULL, FULL*NG)
DEFINE FILE 1(10*216U*U1)
DEFINE FILE 2(3600*200U*U1)
COMMON IIN, IOUT, NR, NI, ILOAD, IRESPG
DIMENSION ISAVE(10), ZERO(10), ZEROX(10), IREC2(1280)
DIMENSION IAVG(11500), FULL(10), NREC(500)
DIMENSION IREC1(134), IREC2(2400), IDISK(600*10), IREC(160*10)
EQUIVALENCE (JREC(1,1), IREC2(641)), (IREC2(1), IAVG(1,381))
FOU00010
100 FORMAT(34A212(10OA210OA2))
102 FORMAT(10E12.4)
100 FORMAT(10E12.4)
102 FORMAT(10E12.4)
100 FORMAT(10E12.4)
102 FORMAT(10E12.4)

1 CONTINUE
IRESPG=IRESP
KAVGXX=0
IPH=1
IREC=0
KS=0
KFAIL=0
PASS 9 RECORDS - READ AND AVERAGE THE TENTH
10 DO 12 I=1,9
NG=0
CALL READ7(ITAPE, IREC1, IREC2, IAVG, KS, NG)
IF(NG*EQ.1) GO TO 99
12 CONTINUE
KS=KS+1
IREC=IREC+10
NG=3
CALL READ7(ITAPE, IREC1, IREC2, IAVG, KS, NG)
IF(NG*EQ.1) GO TO 99
NREC(KS)=IREC
IF(ITEST*GT.4) CALL RITREC(IREC, IREC2, IREC2(641))
GO TO(20,30,40,49), IPH
:
LOOKING FOR DC CALS
20 IF(KS*GT.31) GO TO 21
:
300 RECORDS AND NO DC CALS
KFAIL=1
GO TO 99
21 IF(KS*GT.3) KAVGXX=(IAVG(1*KS-2)+IAVG(1*KS-3))/2
IF(IAVG(1*KS)*LT.1000*KAVGXX) GO TO 10
IF(ITEST*GT.3) CALL RITREC(IREC, IREC2, IREC2(641))
IREC=IREC
KSX=KS
IPH=2
GO TO 10
:
LOOKING FOR END OF DC CALS
30 KVAG=KAVGXX+500.
IF(IAVG(1*KS)*LT.KVAGAND IA VG(1*KS-1)*LT.KVAG) GO TO 34

Figure A-1 (Cont'd)
Program I - Fortran Listings
IF(KS.LT.KSX+20) GO TO 10
OVER 6 SECONDS OF DC CALS
KFAIL=2
GO TO 99
34 IREC=IREC
KSY=KS=1
IPH=3
GO TO 10

IN ZERO AREA
SKIP 50 RECORDS
40 IF(KS=KSY*NE*5) GO TO 10

SEARCH 150 RECORDS FOR EVENT PULSE
KAVG=IAVG(1:KS)+120
50 DO 52 I=1,150
KS=KS+1
IREC=IREC+1
NREC(KS)=IREC
NG=3
CALL READ7(ITAPE,IREC1,IREC2,IAVG,KS,NG)
IF(NG*EQ*1) GO TO 99
IF(IAVG(1:KS).GT.KAVG) GO TO 54
52 CONTINUE
FAIL TO FIND EVENT PULSE
KFAIL=3
GO TO 99

FOUND EVENT PULSE RECORD
54 CONTINUE
IRECZ=IREC
KSZ=KS

READ 6 RECORDS AND SAVE LAST IN ENTIRETY IN IDISK ARRAY
DO 60 I=1,6
IREC=IREC+1
NG=3
KS=KS+1
CALL READ7(ITAPE,IREC1,IREC2,IAVG,KS,NG)
IF(NG*EQ*1) GO TO 99
NREC(KS)=IREC
60 CONTINUE
IF(1TEST.GT.0)
1CALL RITREC(IREC,IREC2,IREC2(641))
DO 61 I=1,640
61 IREC22(I)=IREC2(I)
DO 62 J=1,10
DO 62 I=1,160
62 IDISK(I+200,J)=JREC(I,J)

READ AND SAVE ANOTHER RECORD
IREC=IREC+1
KS=KS+1

Figure A-1 (Cont'd)
Program I - Fortran Listings
CALL READ7(ITAPE,IREC1,IREC2,IAVG,KS,NG)

IF(NG*EQ*1) GO TO 99

IF(ITEST.GT.0)

1 CALL RITREC(IREC,IREC2,IREC2(641))

NREC(KS)=IREC

DO 63 I=1,640

63 IREC22(I+640)=IREC2(I)

DO 64 J=1,10

DO 64 I=1,640

64 IDISK(I+360,J)=JREC(I,J)

SEARCH 2 SAVED RECORDS FOR IMPULSE

IAVGX=(IAVG(1,KS-2)+IAVG(1,KS-3)+IAVG(1,KS-4))/3

IF(IREC22(I).GT.IAVGX+200. AND. IREC22(I-1).GT.IAVGX+200. AND.

1 IREC22(I-2).GT.IAVGX+200.) GO TO 66

65 CONTINUE

C

IMPULSE NOT FOUND

KFAIL=4

GO TO 99

C

FOUND IMPULSE I = THIRD ITEM OVER 200

GET LOCAL ZERO FOR IMPULSE AREA CALCULATION

J=I-2

XIASH=0.

IF(K.LT.1) GO TO 99

K=J-6

DO 67 L=K,KK

67 XIASH=XIASH+IREC22(L)

XIASH=XIASH/40.

IA=XIASH+SIGN(*5,XIASH)

FIND FIRST AND LAST POINT 50 OVER AVERAGE

IF(IREC22(J-1).LT.IA+50) GO TO 70

J=J-1

GO TO 68

70 IF(IREC22(I+1).LT.IA+50) GO TO 71

I=I+1

GO TO 70

71 JSRT=J

C

CALCULATE AREA OF END PIECES

AREA=0.000025*((IREC22(J)-XIASH)**2/(IREC22(J)-IREC22(J-1)) +

1 (IREC22(I)-XIASH)**2/(IREC22(I)-IREC22(I+1)) +

IPTS=I-J+1

IF((IPTS/2)*2=IPTS*NEQ.0) GO TO 74

INSURE AN ODD NUMBER OF POINTS FOR SIMPSON RULE
C PERFORM SIMPSON INTEGRATION

74 ISUM=IREC22(J)+IREC2(J-1) -2**XIASH + AREA

75 J1=J+1
    I1=I-1
    IF(I1.LT.J1) GO TO 99

76 DO 77 K=J1,I1+2
    I1=I-1
    IF(I1.LT.J1) GO TO 78

77 I1=J1+1
    DO 76 K=J1,I1+2

78 AREA=AREA+.00005/3*ISUM-.00500*(I-J) *XIASH

C COMPUTE CALIBRATION VALUES

IF(KSY-KSX*.GT.4) GO TO 79

NOT ENOUGH DC CAL RECORDS

KFAIL=5
GO TO 99

79 FULLI=3*FULLI
(IAVG(1*KSX+2)+IAVG(1*KSX+3)+IAVG(1*KSX+4)
1 -IAVG(1*KSX+1) -IAVG(1*KSX+2) -IAVG(1*KSX+3))

C AREA IN POUND-SECONDS

AREAI=AREA*FULLI
DO 42 I=2,11
J=I-1

42 FULL(J)=3*FULL(J)
(IAVG(I*KSX-3)+IAVG(I*KSX-4)+IAVG(I*KSX-5)
1 -IAVG(I*KSX+1) -IAVG(I*KSX+2) -IAVG(I*KSX+3))/AREAI

IT IS NOW NECESSARY TO FILL THE IDISK ARRAY TEN TIMES AND WRITE
IT INTO FILE 2 TO COMPLETE STORAGE OF RESPONSES

2 RECORDS (.64 MS) ARE SAVED IN DISK ARRAY
STORE THE LAST KNUM MILLISECONDS OF THESE VALUES

KNUM=(1290-JSTRT)/20
K=3165*KNUM
DO 228 I=1,KNUM
K=K+5
DO 228 J=1,10

AVERAGE 5 POINTS TO GET 1 MILLISECOND DATA

228 IDISK(I+J)=(IDISK(K+200+J)+IDISK(K+201+J)+IDISK(K+202+J)+
1 IDISK(K+203+J)+IDISK(K+204+J)+2)/5

NSET=1
I=KNUM
K=156
DUR=6000.
GO TO 211

209 DO 210 J=1,10
210 IDISK(I+J)=(JREC(K+J)+JREC(K+1+J)+JREC(K+2+J)+JREC(K+3+J)+
1 JREC(K+4+J)+2)/5
211 IF(K*NE.*156) GO TO 212
   K=I
   KS=KS+1
   IREC=IREC+1
   NG=2
   CALL READ7 ITAPE, IREC1, IREC2, IAVG, KS, NG
   NREC(KS)=IREC
   IF(NG*NE.*1) GO TO 212

EARLY END TO DATA
   DUR=600*(NSET-1)+1
   I=I+1
   IF(I*G0.600) GO TO 220
   KS=KS-1
   DO 213 I=I+1,600
   DO 213 J=1,10
   FILL OUT RECORD WITH AVERAGE OF LAST RECORD
213 IDISK(I*J)=IAVG(J+1,KSM)
   NSET=10
   GO TO 220

212 K=K+5
   IF(I*EQ.600) GO TO 220
   I=I+1
   GO TO 209

C PACK 600 WORDS INTO 200
220 L=1
   M1=4096*4096
   DO 222 LL=LL,1,200
   DO 221 J=1,10
   221 IDISK(LL,J)=(IDISK(L*J)+2048)*M1+(IDISK(L+1,J)+2048)*4096+
   (IDISK(L+2,J)+2048)
   I=1
   L=L+3
   JJ=0
C STORE IN FILE 2
   DO 242 J=IRESPP,RESPP
   JJ=JJ+1
   IFILE=360*(J-1)+10*(LOAD-1)+NSET
   WRITE(2,IFILE)(IDISK(I*JJ),I=1,200)
   I=1
   NSET=NSET+1
   IF(NSET,EQ.11) GO TO 345
   GO TO 209
345 WRITE(IOUT#107) AREA1,DUR
   FORMAT('/// AI MPULSE AREA IN LB-SEC='F10.6/1/' DURATION OF RESPONSE IN SECONDS='F7.0)
C CALCULATE ZERO FROM REGION AFTER EVENT PULSE
   DO 350 I=1,10
   L=L+1
   350 ZERO(I)=IAVG(L*KSZ+2)+IAVG(L*KSZ+3)+IAVG(L*KSZ+4)+IAVG(L*KSZ+5)
   I=I+1
C CALCULATE ALTERNATE ZERO FROM AVERAGE OF ENTIRE ITF EXCEPT

Figure A-1 (Cont'd)

Program I - Fortran Listings
LAST RECORD AND FIRST 320 MILISECONDS
KSFF=KSF+10
KSL=KS-1
DO 354 I=1,10
  ZEROX(I)=0.
DO 352 K=KSFF,KSL
  ZEROK(I)=ZEROX(I)+IAVG(I+1,K)
354  ZEROK(I)=ZEROX(I)/(KSL-KSFF+1)

UPDATE STATUS FILE
I=0
DO 360 J=IRESP,IRESP
  I=I+1
  READ(1,J) STATUS
  STATUS(1,LOAD)=FULL(I)
  STATUS(2,LOAD)=ZERO(I)
  STATUS(3,LOAD)=DUR
  STATUS(4,LOAD)=ZEROX(I)
  STATUS(5,LOAD)=FULL
  STATUS(6,LOAD)=AREAI
360  WRITE(1,J) STATUS
  CONTINUE
  CALL MRWND(ITAPE)
  WRITE(IOU0T,101)(J9J1,I9J1)
101    FORMAT(///' SUMMARY OF RECORD AVERAGES'// ' REC IMP ')
  J1=KSX-4
  J2=KSY+4
  WRITE(IOU0T,108)(NREC(J)*IAVG(I,J),I=1,11),J=J1,J2
  J1=KSZ-2
  J2=KSZ+6
  WRITE(IOU0T,108)(NREC(J)*IAVG(I,J),I=1,11),J=J1,J2
  J1=KS-4
  J2=KS
  WRITE(IOU0T,108)(NREC(J)*IAVG(I,J),I=1,11),J=J1,J2
108    FORMAT(12(16))
  RETURN
END

Figure A-1 (Cont'd)
Program I - Fortran Listings
SUBROUTINE RITREC(JB, IB, IC)
COMMON IIN, IOUT, NRNI, LOAD, IRESP
DIMENSION ICSUM(11), IBSUM(32)
DIMENSION IC(160, 11), IB(2400)
100 FORMAT('1RECORD NO.', I4, 'LOAD NO.', I3, 'RESP. NO.', I3)
1  //40X, 'IMPULSE TRACK', 60X, 'RESPONSE NO.', 1/
101 FORMAT(I3, 2X, 2015, 2X, 515)
102 FORMAT(IHO, 1X, 5(5X, 'RESPONSE NO.', I3, 6X))
103 FORMAT(I3, 515, 4(1X, 515))
104 FORMAT(I1H1, 1X, 5(5X, 'RESPONSE NO.', I3, 6X))
105 FORMAT(I1X, 1615, 1X, 1615, 1X, 1615, 16, I10, 105)
DO 9 I=1, 11
ICSUM(I) = 0
DO 8 J=1, 160
ICSUM(I) = ICSUM(I) + IC(J, I)
9 ICSUM(I) = ICSUM(I) / 160 + 5
WRITE(IOUT, 100) JB, LOAD, IRESP
DO 4 I=1, 32
IBSUM(I) = 0
J1 = 20*I-19
J2 = J1 + 19
DO 12 J=J1, J2
IBSUM(I) = IBSUM(I) + IB(J)
12 IBSUM(I) = IBSUM(I) / 20 + 5
J3 = 5*I-4
J4 = J3 + 4
4 WRITE(IOUT, 101) I, (IB(J), J=J1, J2) , (IC(J, I), J=J3, J4)
WRITE(IOUT, 102) (K, K=2, 6)
DO 5 I=1, 16
J1 = 5*I-4
J2 = J1 + 4
5 WRITE(IOUT, 103) I, ((IC(J, K), J=J1, J2) , K=2, 6)
WRITE(IOUT, 104) (K, K=2, 6)
DO 7 I=17, 32
J1 = 5*I-4
J2 = J1 + 4
7 WRITE(IOUT, 103) I, ((IC(J, K), J=J1, J2) , K=2, 6)
WRITE(IOUT, 102) (K, K=7, 11)
DO 6 I=1, 32
J1 = 5*I-4
J2 = J1 + 4
6 WRITE(IOUT, 103) I, ((IC(J, K), J=J1, J2) , K=7, 11)
ICSUM(11) = 0
DO 11 I=1, 32
ICSUM(11) = ICSUM(11) + IBSUM(I)
11 ICSUM(11) = ICSUM(11) / 32 + 5
WRITE(IOUT, 105) IBSUM(1), ICSUM(11), (ICSUM(J), J=1, 10)
RETURN
END
SUBROUTINE READ7(JTAPE,IREC1,IREC2,IAVG,KS,NG)

COMMON IIN*,IOUT
DIMENSION IAVG(11,1)
DIMENSION IREC1(34),IREC2(2400)
100 FORMAT(34A2.12(10OA210OA2))
101 FORMAT('READ ERROR ON UNIT','KS', 'PREVIOUS RECORD USED')
102 FORMAT('END OF TAPE, KS')
90 CALL UNPACK(JTAPE,IREC2,ISTAT)
IF(ISTAT.EQ.2) GO TO 110
IF(ISTAT.EQ.-1) GO TO 120
93 IF(NG.EQ.0) RETURN
C CONVERT IMPULSE NG=1 RESPONSES NG=2 BOTH NG=3
IF(NG.EQ.2) GO TO 11
DO 10 I=1,640
10 IREC2(I)=IREC2(I)-2048
IF(NG.EQ.1) GO TO 13
11 DO 12 I=641,2400
12 IREC2(I)=IREC2(I)-2048
13 CONTINUE
105 CALL AVG(IREC2,IAVG(1,KS),NG)
NG=0
RETURN
110 NG=1
WRITE(IOUT,102) KS
RETURN
120 WRITE(IOUT,101) JTAPE,K
RETURN
IF(NG.EQ.0) RETURN
IF(KS.LT.2) GO TO 122
DO 121 I=1,11
121 IAVG(I,KS)=IAVG(I,KS-1)
GO TO 124
122 DO 123 I=1,11
123 IAVG(I,KS)=0.
124 NG=0
DO 125 I=1,2400
125 IREC2(I)=0
RETURN
END

Figure A-1 (Cont'd)
Program I - Fortran Listings
*FOR* IS AVG

SUBROUTINE AVG(IREC,IAVG,NG)
DIMENSION IAVG(11)
DIMENSION IREC(2400)
ISUM=0
DO 1 I=1,11
1 IAVG(I)=0
C AVERAGE IMPULSE NG=1 RESPONSES NG=2 BOTH NG=3
IF(NG.EQ.2) GO TO 3
2 ISUM=ISUM+IREC(I)
ISUM=ISUM/640+.5
IAVG(I)=ISUM
IF(NG.EQ.1) RETURN
3 DO 6 J=2,11
ISUM=0
K=641+(J-2)*160
L=K+159
DO 4 I=K,L
ISUM=ISUM+IREC(I)
4 IAVG(J)=ISUM/160+.5
RETURN
END

*FOR* IS UNPACK

SUBROUTINE UNPACK(IN1,IREC,ISTAT)
DIMENSION IREC(2400),N(8),DUM(812)
CALL MFC(N)
CALL MREAD(IN1,DUM,N,M)
3 CONTINUE
IF(N(6).EQ.1) GO TO 3
ISTAT = N(6)
IREC(1)=FLD(12,12,DUM(12))
IREC(2)=FLD(24,12,DUM(12))
IREC(2400)=FLD(0,12,DUM(812))
DO 2 K=13,811
J=3*K-36
IREC(J)=FLD(0,12,DUM(K))
IREC(J+1)=FLD(12,12,DUM(K))
2 IREC(J+2)=FLD(24,12,DUM(K))
RETURN
END

Figure A-1 (Cont'd)
Program I - Fortran Listings
*FOR*IS PLOT1.PLOT1
SUBROUTINE PLOT1(IFILE,INN)

COMMON INN, IOUT, NR+NI
DIMENSION STAT(6,36)
DIMENSION X(301), Y(301), IFILE(1), INN(8)
IR1=INN(1)
IR2=INN(2)
IL1=INN(3)
IL2=INN(4)
MS=INN(5)
NPT=INN(6)
IPR=INN(7)
IOPEN=INN(8)

C
IF(IOPEN.EQ.1 .OR. IOPEN.EQ.3) CALL OPEN PLOT ROUTINE
XL=10.
YL=2.
X(1)=Q.
Y(1)=O.
WRITE(IOUT,100)
100 FORMAT(/ // ' PLOT PACKAGE EXECUTING')

1 NPTS=MS/NPT
   IF(NPTS.LE.300) GO TO 2
   NPT=NPT+1
   GO TO 1
2 DX=XL/NPTS
   DO 3 I=1,NPTS
3 X(I+1)=DX*I
   MS=NPT*NPTS
   DO 200 IL=IL1+IL2
   DO 200 IR=IR1,IR2
      IRX=IR-((IR-1)/NR)*NR
      READ(1,I(1)(STAT(I J),I=1,6)+J=1,NI)
      CON=STAT(1 IL)
      ZERO=STAT(2 IL)
      IF=(IRX-1)*10*NI+10*(IL-1)
   DO 6 J=1,10
      LL=200*J
      L=LL-199
      IF((3*L-3).GT.MS) GO TO 7
}

Figure A-1 (Cont'd)
Program I - Fortran Listings
IF$=IF+J$

6 READ(2*IFF)(FILE(K),K=L*LL)
7 NPTSS=NPTS+1
DO 40 I=1*NPTS
   IP=IP+1
   Y(IP)=0.
   DO 10 J=1*NPT
      K=(I-1)*NPT+J
      KW=(K-1)/3+1
      KP=1ABS(K-(KW-1)*3-1)*12
10 Y(IP)=Y(IP)+FLD(KP,12*FILE(KW))
40 Y(IP)=Y(IP)/NPT-2048.0-ZERO
IF(IPR*EQ.2) GO TO 60
C PLOT X VS Y NPTSS VALUES
C X IS IN INCHES RANGE 0 TO 10 INCHES IS TIME FROM 0 TO MS MILLISECONDS
C Y IS IN COUNTS FROM -2048 TO +2048
C CON IS MEASUREMENT UNITS PER POUND SECONDS PER COUNT
IF(IPR*EQ.1) GO TO 200
60 WRITE(IOUT,902) IR,IL,NPTS,NPT,NPT
902 FORMAT(1H1/"RESPONSE","13.1" LOAD","13.10X","14.6 POINTS STARTING","1 AT","13.1" MS, SPACED EVERY","13.1" MS IN COUNTS, ZERO ADJUSTED")
903 WRITE(IOUT,903)(Y(J),J=1,NPTSS)
903 FORMAT(//(1H0,10F10.1))
200 CONTINUE
C IF(IOOPEN*GT.*1) CALL CLOSE PLOT ROUTINE
RETURN
END

Figure A-1 (Cont'd)

Program I - Fortran Listings
*FOR.IS_RESP,RESP*  
DEFINE FILE 1(10,216,U*IU1)  
DEFINE FILE 2(3600,200,U*IU2)  
DEFINE FILE 3(36,1200,U*IU3)  
COMMON IN*,IOUT*,NR*,NI*,LOAD  
INTEGER FF  
DIMENSION IFILE(6000),ITF(1200)*YIF(1200)  
DIMENSION F(1200),SC(36),DT(6),TEN(1200),F1(1200),F2(1200)  
DIMENSION STAT(6,36),IFILL(200,10)  
EQUIVALENCE (IFILL(1,1),ITF(1201))  
EQUIVALENCE (ITF(1,1),F1(1201))  
EQUIVALENCE (IFILL(200,10),ITF(1201))  
DATA F/'LOAD *TIME *FORCE *RESP *PLOT *CYCLE '/  
901 FORMAT(A6,314)  
902 FORMAT(/'NO LOAD CARD - QUIT'/)  
903 FORMAT(/'ERROR - RESPONSES ON TAPE* , I4')  
904 FORMAT(/'ERROR - MOUNTED TAPE * NEED TAPE* , I4')  
905 FORMAT(/'RESPONSES*, I4, TO*, I4, LOADED FROM UNIT* , I4')  
906 FORMAT(/'USING FILES WITHOUT LOADING FROM TAPE* ')  
907 FORMAT(/'ERROR* , UNIT* , I4')  
908 FORMAT(/'NO TIME CARD - QUIT'/)  
909 FORMAT(/'SET TO RUN* , I5, POINTS AT*, I3, MS* /
 1  TOTAL TIME* , I5, MS* )  
910 FORMAT(/'SOMETHING WRONG THERE - QUIT'/)  
911 FORMAT(A6,4X,6(15F5.0))  
912 FORMAT(/'NO FORCING CARD - QUIT'/)  
913 FORMAT(/'USE PREVIOUS FORCES'/)  
914 FORMAT(/'TOO MANY VALUES IN FORCING TABLE - QUIT'/)  
915 FORMAT(A6,314)  
916 FORMAT(/'ERROR 1L GT. NI - QUIT'/)  
917 FORMAT(A6,314)  
918 FORMAT(A6,314)  
919 FORMAT(/'ERROR IN RUN CARD* , IRA* IRB* , I214* , QUIT'/)  
920 FORMAT(/'SET TO PROCESS RESPONSES*, I4, TO*, I4')  
921 FORMAT(/'INCORRECT FILTER INFO - QUIT'/)  
922 FORMAT(/'USE CALIBRATED ZERO'/)  
923 FORMAT(/'USE AVERAGED ZERO'/)  
924 FORMAT(/'USE FILTER TO KILL FREQUENCIES BELOW* , F6.1, CPS'/)  
925 FORMAT(1H1, 'RESPONSE* , I3, LOAD* , I3, EVERY* , I4, MS* /
 1 (10E13, 4/))  
926 FORMAT(1H1, 'SUM OF ABOVE RESPONSES* /(10E13, 4/) )  
927 FORMAT(A6,314)  
928 FORMAT(/'RECYCLE TO POINT* , I3')  
929 FORMAT(1H1, 'FORCING FUNCTION* , I3, EVERY* , I3, MS, STARTING AT',
 1 I3, ' MS*/(10E13, 4/) )  
930 FORMAT(/'ERROR IN PLOT CARD')  
IIN=5  
IOUT=6  

Figure A-2  
Program II - Fortran Listings
NR=10
NI=36
NT=NR*NI
NTT=6*NT
NTT=10*NT
NTI=6*NI
CALL PLOTA

C READ LOAD CARD
1 CONTINUE
READ(IIN,901) FF, IU, IR1, IR2
IF(FF.EQ.*11) GO TO 10
WRITE(IOUT,902)
GO TO 99
10 IF(IU.EQ.0) GO TO 34
IF(IU.GT.6.AND.IU.LT.100) GO TO 12
WRITE(IOUT,907) IU
GO TO 99
12 REWIND IU
ICD1=(IR1-1)/NR+1
ICD2=(IR2-1)/NR+1
IF(ICD1.EQ.ICD2) GO TO 20
WRITE(IOUT,903) ICD1, ICD2
GO TO 99
20 READ(IU) (ICD(I)=1,100)
IF(ICD.EQ.ICD1) GO TO 30
REWIND IU
WRITE(IOUT,904) ICD, ICD1
GO TO 99
30 READ(IU)(IFILE(I)=1,100)
DO 31 I=1,NTT
J2=I*NTI
J1=J2-NTI+1
31 WRITE(IOUT,905) ICD, IU
READ(IU) ((IFILL(K,L)=1,100),L=1,NRT)
DO 32 L=1,10
JJL=J+L-1
32 WRITE(IOUT,906) IFILL(K,L), JJL
REWIND IU
IRX=ICD*NR-9
IRY=IRX+9
WRITE(IOUT,905) IRX, IRY, IU
GO TO 36
34 WRITE(IOUT,906)
C READ TIME CARD
36 READ(IIN,901) FF, NPTS, NMS
IF(FF.EQ,*2) GO TO 40
WRITE(IOUT,908)
GO TO 99
40 MTIME=NPTS*NMS

Figure A-2 (Cont'd)

Program II - Fortran Listings
WRITE(IOUT,909) NPTS,NMS,MTIME
IF(NPTS.GT.1.AND.NPTS.LT.1201.AND.MTIME.LT.6001) GO TO 44
WRITE(IOUT,910)
GO TO 99

C SET UP FORCING FUNCTIONS
C READ FORCE CARD
44 READ(IIN,911) FF(MT(I),DT(I),I=1,6)
IF(FF.EQ.0) GO TO 50
WRITE(IOUT,912)
GO TO 99
30 IF(MT(I).GT.0) GO TO 52
WRITE(IOUT,913)
GO TO 70
50 NSET=MT(I)
MTT=NSET
DO 2 I=1,NI
2 SC(I)=0.
DO 54 I=1,MTT
54 TEMP(I)=I*DT(I)
DO 56 J=2,6
IF(MT(J).LT.1) GO TO 60
MTJ=MT(J)
MTT=MTT+MTJ
IF(MTT.GT.1200) GO TO 58
DO 56 J=1,MTJ
NSET=NSET+1
56 TEMP(NSET)=TEMP(NSET-1)+DT(J)
GO TO 60
58 WRITE(IOUT,914)
GO TO 99
C INTERPOLATE TIMES
60 DO 604 I=1,NPTS
604 IF(TG.GT.TEMP(I)) GO TO 606
JTEMP(I)=1
606 F(I)=TG/TEMP(I)
GO TO 612
607 J=J+1
GO TO 607
608 JTEMP(I)=J
F(I)=(TG-TEMP(J-1))/(TEMP(J)-TEMP(J-1))
610 CONTINUE
612 CONTINUE
61 READ(IIN,915) IL,SCL
IF(IL.LT.1) GO TO 70

Figure A-2  (Cont'd)
Program II - Fortran Listings
IF(IL*LE.NI) GO TO 62
WRITE(IOUT,916)
GO TO 99
62 READ(IIN*917)(FEMP(I),I=1,NPTS)
DO 66 I=1,NPTS
J=TEMP(I)
FOR(I)=FEMP(J)*F1(I)
IF(J*EQ.1) GO TO 66
FOR(I)=FOR(I)+FEMP(J-1)*(1.-F1(I))
66 CONTINUE
WRITE(3*IL)(FOR(I),I=1,NPTS)
WRITE(IOUT*929) IL*NMS,NMS*(FOR(I),I=1,NPTS)
GO TO 61
70 CONTINUE
C FINISHED READING, INTERPOLATING AND STORING FORCES
C READ RESP CARD
71 CONTINUE
READ(IIN*918) FF,IRA,IRB,IZFIL,FREQ
IF(FF*NE.F(4)) GO TO 99
IF(IRA.GE.IRX.AND.IRB*LE.IRY.AND .IRA.LE.IRB) GO TO 72
WRITE(IOUT*919) IRA,IRB
GO TO 99
72 IF(IZFIL.GT.0.GA.IZFIL*LT.4) GO TO 74
WRITE(IOUT*920)
GO TO 99
74 GO TO (76,78,80,IZFIL
76 WRITE(IOUT*921)
GO TO 82
78 WRITE(IOUT*922)
GO TO 82
80 WRITE(IOUT*923) FREQ
82 CONTINUE
C READ PLOT PRINT CARD
83 READ(IIN*927) FF,NPL
IF(FF*EQ.F(5)) GO TO 84
WRITE(IOUT*930)
GO TO 99
84 CONTINUE
DO 800 IR=IRA,IRB
DO 110 I=1,NPTS
110 SUM(I)=0.
SUM(NPTS+1)=0.
IRZ=IR-((IR-1)/NR)*NR
READ(1*IRZ) STAT
DO 700 IL=1,NI
IF(SC(IL)*EQ.0) GO TO 700
IF=((IRZ-1)*NI+(IL-1))*10
DO 120 I=1,10
120 WRITE(IOUT,924) IRZ,IR
CONTINUE

Figure A-2 (Cont'd)
Program II - Fortran Listings
**Figure A-2 (Cont'd)**

Program II - Fortran Listings

```fortran
      IFF=IFF+1
  120  READ(2,IFF)(IFILL(J),J=1,200)
      MPAC=STAT(3*IL)+1
      IF(IFILL*EQ.3) GO TO 122
      IF (MTIME*LT*MPAC) MPAC=MTIME
  122  J=0
      DO 124 I=1,MPAC
            K=4001+(I-1)/3
            IFILE(I)=FLD(J,12,IFILE(K))-2048
            J=J+12
            IF (J*EQ.*36) J=0
      CONTINUE
      GO TO (130,131,150)*IZFIL
  130  ZER=STAT(2*IL)*NMS
      GO TO 132
  131  ZER=STAT(4*IL)*NMS
  132  MPTS=NPTS
      IF (MPAC*LT*MTIME) MPTS=MPAC/NMS
      GO TO 133
  150  CONTINUE
      MPTS=NPTS
      IF (MPAC*LT*MTIME) MPTS=MPAC/NMS
      NEED=MPTS*NMS
      CALL FILT(IFILE,MPAC,NEED,FREQ)
      ZER=0.
  133  CONTINUE
      DO 140 I=1,MPTS
            ITF(I)=0.
      DO 138 J=1,NMS
            K=(I-1)*NMS+J
      138  ITF(I)=ITF(I)+IFILE(K)
      140  YITF(I)=ITF(I)-ZER
  400  CONTINUE
      READ(3*IL)(FOR(I),I=1,MPTS)
      MPTSS=MPTS+1
      DO 402 I=1,MPTSS
  402  RESP(I)=0.
      DO 404 I=1,MPTS
            JM=MPTSS-I
      404  RESP(JP)=RESP(JP)+FOR(I)*YITF(J)
      SCALE=STAT(1*IL)*.001*SC(IL)
      DO 406 I=1,MPTSS
  406  SUM(I)=SUM(I)+RESP(I)
      WRITE(IOUT,925) IR,IL,NMS*(RESp(I),I=2,MPTSS)
      IF (NPL*EQ.*2) CALL PLOTB(IR,IL,MPTSS,NMS,RESP)
  700  CONTINUE
      WRITE(IOUT,926)(SUM(I),I=2,MPTSS)
```

---

**Explanation:**

This Fortran program appears to be a continuation of a larger program, possibly for signal processing or data analysis. The program reads data files (possibly from a DDS system or similar), processes the data, and outputs processed values. The program includes loops for iterating over data files, calculations for computing statistical values, and outputting results. The structured code suggests it is part of a larger software system, likely related to signal processing or scientific computing.
IF(NPLAGE.1) CALL PLOTB(IR,IL,MPT,NMS,SUM)

800 CONTINUE
C READ RECYCLE CARD
READ(IN,927) FF,NCY
IF(FF.NE.6) GO TO 99
WRITE(IOUT,928)
GO TO (1,36,64,61,71,99),NCY
99 CONTINUE
CALL PLOTC
STOP
END

*FORIS PLOTB PLOTB
SUBROUTINE PLOTB(IR,IL,MPT,NMS,Y)
DIMENSION Y(1)
C PLOT ROUTINE HERE
RETURN
ENTRY PLOTA
C PLOT INITIALIZATION HERE
RETURN
ENTRY PLOTB
C PLOT TERMINATION HERE
RETURN
END

Figure A-2 (Cont'd)
*FOR* IS FILT*FILT*

SUBROUTINE FILT(IFILE,NPTS,NEED,FREK)
DIMENSION IFILE(1),X(2,300),Y(2,300),W(2,300),Z(2,600)
FREQ=ABS(FREK)

NMS=1
1 MPT=NPTS/NMS
   IF(MPT.LT.300) GO TO 2
   NMS=NMS+1
   GO TO 1
2 NPT=MPT*NMS
   DF=1000.*/NPT
   NKILL=2.01+FREQ/DF
   MKILL=MPT+2-NKILL
   IF(FREK.GT.*1.) GO TO 4
   NMS=20
   MPT=125
   KILL=FREQ/4.*.01
   NKILL=KILL+2
   MKILL=125-KILL
4 CONTINUE
   DO 3 I=1,300
3   X(2,I)=0.
   K=1
   DO 6 J=1,MPT
      X(1,I)=0.
      DO 6 J=1,NMS
         X(1,I)=X(1,I)+IFILE(K)
      K=K+1
6   DO 16 I=1,MPT
16   X(1,I)=X(1,I)/NMS
   CALL FASTI(X,Y,MPT,Z,W,1)
   DO 7 I=NKILL,MKILL
      Y(1,I)=0.
7   Y(2,I)=0.
   CALL FASTI(Y,X,MPT,Z,W,-1)
   DO 8 I=1,MPT
8   X(1,I)=X(1,I)/MPT
   DO 12 I=1,NEED
      P=I-1
      P=P/NMS+1.
      II=P
      FF=P-1
      F=1.-FF
12   IFILE(I)=IFILE(I)-(F*X(1,II)+FF*X(1,II+1))
RETURN
END

Figure A-2 (Cont'd)

Program II - Fortran Listings
SUBROUTINE FAST1(X,Y,N,Z,W,S)
      C FAST FOURIER TRANSFORM OF COMPLEX DATA BY DAVID IVES,
      C GRUMMAN
      C X.....N INPUT VALUES (COMPLEX)
      C Y.....N OUTPUT VALUES (COMPLEX)
      C N.....NUMBER OF VALUES
      C Z.....DUMMY STORAGE OF LENGTH 2N (COMPLEX)
      C W.....DUMMY STORAGE OF LENGTH N (COMPLEX)
      C S.....SIGN CONTROLLING DIRECTION OF TRANSFORM
      THIS PRODUCES 'OUTPUT Y' FROM 'INPUT X', WHERE
      K=N
      Y(J)=SUM X(K)*EXP(SIGN(1,S)*1*2*PI*(J-1)*(K-1)/N)
      K=1
      WITH I=SQRT(-1) S=+1, OR S=-1 AND PI=3.14159.
      COMPLEX NUMBERS ARE HANDLED IN FORTRAN 4 CONVENTION.
      REAL AND IMAGINARY PARTS ARE STORED IN ALTERNATE CELLS,
      STARTING WITH THE REAL PART OF X(1) IN THE FIRST LOCATION.
      DIMENSION X(1)*Z(1)*W(1)*Y(1)
      MOD(J,K)=J-(J/K)*K
      DO 1 I=1,N
      W(2*I-1)=COS((6.28318530717959/N)*(I-1))
      W(2*I)=SIGN(1,S)*SIN(6.28318530717959/N)*(I-1))
      Z(2*I-1)=X(2*I-1)
      1 Z(2*I)=X(2*I)
      ID=N
      DO 4 J=1,N
         IF(ID=1) 5,5
         IF(MOD(ID,IDX)) 3,3
         2 CONTINUE
         3 ID=ID/IX
         IS=IDX
         DO 4 L=1,IS
            1 Z(L+I)=Z(I)
            4 CONTINUE
            5 DO 6 I=1,N
               K=2*MOD(J+1,2)*N+I
               Y(2*I-1)=Z(K)
               K=K+1
               6 Y(2*I)=Z(K)
      RETURN
      END
* (RUN CARD)

*ASG,T 1,F2/2/7/4
*ASG,T 2,F2/7/POS/7
*ASG,TB 10,8C,006515
*ASG,TB 11,8C,TAPE5
*MOVE 11,1
*ASG,TB 12,8C,TAPE6
*MOVE 12,1
*ASG,TB 20,8C,SCRATCH

--------- FORTRAN DECKS ---------

**MAP, I .MAP*, PROG
    SEG TOP
    IN ITF1
    LIB MSC*LOCALIB
*XQT, PROG

LOAD 10 1 10
STATUS 1 10
RUN 11 5 1 10 5
    12500. 143. 639. 1243. 524. 1092. 1110. 653. 640. 179. 179.
RUN 12 6 1 10 5
PLOT 1 10 5 6 500 4 2 0
STATUS 1 10
DUMP 20 1 10
END
*PMD,E
*FIN

* = 7/8 PUNCH

Figure B-1
Program I - Deck Setup and Data
<table>
<thead>
<tr>
<th>RECORD NO.</th>
<th>LOAD NO.</th>
<th>IMPULSE TRACK</th>
<th>RESPONSE NO.</th>
<th>RESPONSE NO.</th>
<th>RESPONSE NO.</th>
<th>RESPONSE NO.</th>
<th>RESPONSE NO.</th>
<th>RESPONSE NO.</th>
<th>RESPONSE NO.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
<td>11</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
<td>11</td>
<td>12</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
<td>11</td>
<td>12</td>
<td>13</td>
</tr>
<tr>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
<td>11</td>
<td>12</td>
<td>13</td>
<td>14</td>
</tr>
<tr>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
<td>11</td>
<td>12</td>
<td>13</td>
<td>14</td>
<td>15</td>
</tr>
</tbody>
</table>

Figure B-2
Program 1 - Record Output
<table>
<thead>
<tr>
<th>RESPONSE NO. 2</th>
<th>RESPONSE NO. 3</th>
<th>RESPONSE NO. 4</th>
<th>RESPONSE NO. 5</th>
<th>RESPONSE NO. 6</th>
</tr>
</thead>
<tbody>
<tr>
<td>17 15 12 17 16 -1 -1 2 8 13</td>
<td>7 9 8 7 5 17 18 16 10</td>
<td>14 19 16 10 9</td>
<td>5 7 2 0 5</td>
<td></td>
</tr>
<tr>
<td>19 15 13 12 4 11 15 12 6 -1 -4</td>
<td>2 -1 -1 2 6 9 -4 16 13</td>
<td>-2 9 -7</td>
<td>19</td>
<td>17</td>
</tr>
<tr>
<td>20 12 10 8 9 3 6 9 10 10</td>
<td>5 5 3 1 0 11 11 11 11</td>
<td>19 11 3 -3 -7</td>
<td>11</td>
<td>7</td>
</tr>
<tr>
<td>21 1 -1 12 12 12 21 21 -2 -3</td>
<td>8 2 3 -2 21 21 21 21 21</td>
<td>12 12 12 12</td>
<td>12</td>
<td>7</td>
</tr>
<tr>
<td>23 13 10 9 7 7 13 12</td>
<td>-1 3 7</td>
<td>15 17 17 16</td>
<td>14</td>
<td>4</td>
</tr>
<tr>
<td>24 11 12 9 8 2 3 5 6 5</td>
<td>1 0 2 4 5 12 10 11 13</td>
<td>13</td>
<td>1 5 6 6</td>
<td></td>
</tr>
<tr>
<td>25 7 8 7 12 -4 1 -4 2 14 19</td>
<td>6 6 8 5 9 7 10 14 16</td>
<td>-6 5 0 -3</td>
<td>13</td>
<td>7</td>
</tr>
<tr>
<td>26 14 11 9 8 7 13 13 1 1 9</td>
<td>-1 -5 -3 1 7 10 6 8 12 14</td>
<td>4</td>
<td>6 3 0</td>
<td></td>
</tr>
<tr>
<td>27 1 1 10 10 10 1 1 1 1 1 1 1</td>
<td>10 10 11 11 11 11 11 11 11 11 11 11</td>
<td>11</td>
<td>11 11 11</td>
<td></td>
</tr>
<tr>
<td>29 9 12 13 11</td>
<td>4 1 2 6</td>
<td>4</td>
<td>11 11 13 11</td>
<td>-3 2 3 4</td>
</tr>
<tr>
<td>30 13 12 6 3 -4 15 18 15 11</td>
<td>6 1 -1 1 5 14 13 5 5</td>
<td>7 7 7 10 11</td>
<td>11</td>
<td>7</td>
</tr>
<tr>
<td>31 -18 -26 39 -40 -38 10 10 -8 -8 -11</td>
<td>-9 -17 18 -24</td>
<td>27 3 -1 -2 2 10</td>
<td>-4 -2 -4 -1 0</td>
<td></td>
</tr>
<tr>
<td>32 -33 -29 -30 -39 -52 14 17 15 16</td>
<td>3</td>
<td>31 29 32 17 20 21 22 21</td>
<td>4 5 3 4 8</td>
<td></td>
</tr>
</tbody>
</table>

**Figure B-2 (Cont'd)**

*Program 1 - Record Output*
<table>
<thead>
<tr>
<th>RECORD NO.</th>
<th>LOAD NO.</th>
<th>RESP. NO.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>5</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>6</td>
<td>4</td>
</tr>
<tr>
<td>5</td>
<td>7</td>
<td>5</td>
</tr>
<tr>
<td>6</td>
<td>8</td>
<td>6</td>
</tr>
<tr>
<td>7</td>
<td>9</td>
<td>7</td>
</tr>
<tr>
<td>8</td>
<td>10</td>
<td>8</td>
</tr>
<tr>
<td>9</td>
<td>11</td>
<td>9</td>
</tr>
<tr>
<td>10</td>
<td>12</td>
<td>10</td>
</tr>
</tbody>
</table>

**RESPONSE NO. 1**

<table>
<thead>
<tr>
<th>RESPONSE NO. 1</th>
<th>RESPONSE NO. 2</th>
<th>RESPONSE NO. 3</th>
<th>RESPONSE NO. 4</th>
<th>RESPONSE NO. 5</th>
<th>RESPONSE NO. 6</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
</tr>
<tr>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
</tr>
</tbody>
</table>

**Program I - Record Output**
### Summary of Record Averages

<table>
<thead>
<tr>
<th>REC</th>
<th>IMP</th>
<th>R1</th>
<th>R2</th>
<th>R3</th>
<th>R4</th>
<th>R5</th>
<th>R6</th>
<th>R7</th>
<th>R8</th>
<th>R9</th>
<th>R10</th>
</tr>
</thead>
<tbody>
<tr>
<td>120</td>
<td>10</td>
<td>9</td>
<td>14</td>
<td>7</td>
<td>2</td>
<td>10</td>
<td>2</td>
<td>10</td>
<td>15</td>
<td>22</td>
<td>27</td>
</tr>
<tr>
<td>130</td>
<td>9</td>
<td>10</td>
<td>13</td>
<td>7</td>
<td>1</td>
<td>13</td>
<td>5</td>
<td>9</td>
<td>16</td>
<td>23</td>
<td>25</td>
</tr>
<tr>
<td>140</td>
<td>7</td>
<td>11</td>
<td>13</td>
<td>5</td>
<td>1</td>
<td>9</td>
<td>4</td>
<td>11</td>
<td>15</td>
<td>23</td>
<td>26</td>
</tr>
<tr>
<td>150</td>
<td>7</td>
<td>11</td>
<td>13</td>
<td>6</td>
<td>1</td>
<td>11</td>
<td>2</td>
<td>9</td>
<td>14</td>
<td>22</td>
<td>27</td>
</tr>
<tr>
<td>160</td>
<td>1596</td>
<td>1616</td>
<td>1583</td>
<td>1614</td>
<td>1617</td>
<td>1596</td>
<td>1764</td>
<td>1601</td>
<td>1615</td>
<td>1793</td>
<td>1802</td>
</tr>
<tr>
<td>170</td>
<td>1596</td>
<td>1617</td>
<td>1583</td>
<td>1614</td>
<td>1617</td>
<td>1596</td>
<td>1764</td>
<td>1601</td>
<td>1615</td>
<td>1793</td>
<td>1802</td>
</tr>
<tr>
<td>180</td>
<td>1595</td>
<td>1617</td>
<td>1583</td>
<td>1614</td>
<td>1617</td>
<td>1595</td>
<td>1766</td>
<td>1601</td>
<td>1615</td>
<td>1793</td>
<td>1802</td>
</tr>
<tr>
<td>190</td>
<td>1595</td>
<td>1617</td>
<td>1583</td>
<td>1614</td>
<td>1617</td>
<td>1595</td>
<td>1767</td>
<td>1401</td>
<td>1615</td>
<td>1793</td>
<td>1802</td>
</tr>
<tr>
<td>200</td>
<td>1594</td>
<td>1617</td>
<td>1583</td>
<td>1614</td>
<td>1617</td>
<td>1594</td>
<td>1767</td>
<td>1401</td>
<td>1615</td>
<td>1793</td>
<td>1802</td>
</tr>
<tr>
<td>210</td>
<td>1594</td>
<td>1617</td>
<td>1583</td>
<td>1614</td>
<td>1617</td>
<td>1594</td>
<td>1767</td>
<td>1401</td>
<td>1615</td>
<td>1793</td>
<td>1802</td>
</tr>
<tr>
<td>220</td>
<td>1594</td>
<td>1617</td>
<td>1583</td>
<td>1614</td>
<td>1617</td>
<td>1594</td>
<td>1768</td>
<td>1601</td>
<td>1615</td>
<td>1793</td>
<td>1802</td>
</tr>
<tr>
<td>230</td>
<td>1594</td>
<td>1617</td>
<td>1583</td>
<td>1614</td>
<td>1617</td>
<td>1594</td>
<td>1768</td>
<td>1601</td>
<td>1615</td>
<td>1793</td>
<td>1802</td>
</tr>
<tr>
<td>240</td>
<td>1594</td>
<td>1617</td>
<td>1583</td>
<td>1614</td>
<td>1617</td>
<td>1594</td>
<td>1768</td>
<td>1601</td>
<td>1615</td>
<td>1793</td>
<td>1802</td>
</tr>
<tr>
<td>250</td>
<td>1594</td>
<td>1617</td>
<td>1583</td>
<td>1614</td>
<td>1617</td>
<td>1595</td>
<td>1766</td>
<td>1602</td>
<td>1615</td>
<td>1793</td>
<td>1803</td>
</tr>
<tr>
<td>260</td>
<td>1594</td>
<td>1617</td>
<td>1583</td>
<td>1614</td>
<td>1617</td>
<td>1595</td>
<td>1766</td>
<td>1602</td>
<td>1615</td>
<td>1793</td>
<td>1802</td>
</tr>
<tr>
<td>270</td>
<td>1594</td>
<td>1617</td>
<td>1583</td>
<td>1614</td>
<td>1617</td>
<td>1594</td>
<td>1767</td>
<td>1601</td>
<td>1615</td>
<td>1794</td>
<td>1802</td>
</tr>
<tr>
<td>280</td>
<td>1594</td>
<td>1617</td>
<td>1583</td>
<td>1614</td>
<td>1617</td>
<td>1594</td>
<td>1767</td>
<td>1601</td>
<td>1616</td>
<td>1794</td>
<td>1802</td>
</tr>
<tr>
<td>290</td>
<td>1594</td>
<td>1617</td>
<td>1583</td>
<td>1614</td>
<td>1617</td>
<td>1594</td>
<td>1768</td>
<td>1602</td>
<td>1615</td>
<td>1794</td>
<td>1802</td>
</tr>
<tr>
<td>300</td>
<td>1594</td>
<td>1617</td>
<td>1583</td>
<td>1614</td>
<td>1617</td>
<td>1595</td>
<td>1771</td>
<td>1602</td>
<td>1616</td>
<td>1794</td>
<td>1803</td>
</tr>
<tr>
<td>310</td>
<td>13</td>
<td>8</td>
<td>9</td>
<td>8</td>
<td>6</td>
<td>14</td>
<td>17</td>
<td>11</td>
<td>9</td>
<td>18</td>
<td>12</td>
</tr>
<tr>
<td>320</td>
<td>13</td>
<td>8</td>
<td>9</td>
<td>8</td>
<td>6</td>
<td>14</td>
<td>16</td>
<td>10</td>
<td>9</td>
<td>18</td>
<td>11</td>
</tr>
<tr>
<td>330</td>
<td>9</td>
<td>8</td>
<td>9</td>
<td>8</td>
<td>6</td>
<td>14</td>
<td>14</td>
<td>9</td>
<td>18</td>
<td>11</td>
<td></td>
</tr>
<tr>
<td>340</td>
<td>14</td>
<td>8</td>
<td>9</td>
<td>8</td>
<td>6</td>
<td>14</td>
<td>18</td>
<td>10</td>
<td>9</td>
<td>18</td>
<td>11</td>
</tr>
<tr>
<td>350</td>
<td>8</td>
<td>10</td>
<td>8</td>
<td>6</td>
<td>13</td>
<td>16</td>
<td>10</td>
<td>9</td>
<td>18</td>
<td>11</td>
<td></td>
</tr>
<tr>
<td>360</td>
<td>14</td>
<td>7</td>
<td>9</td>
<td>8</td>
<td>6</td>
<td>14</td>
<td>20</td>
<td>10</td>
<td>9</td>
<td>17</td>
<td>10</td>
</tr>
<tr>
<td>370</td>
<td>12</td>
<td>7</td>
<td>9</td>
<td>7</td>
<td>6</td>
<td>13</td>
<td>20</td>
<td>10</td>
<td>8</td>
<td>18</td>
<td>10</td>
</tr>
<tr>
<td>380</td>
<td>487</td>
<td>513</td>
<td>490</td>
<td>503</td>
<td>461</td>
<td>499</td>
<td>491</td>
<td>467</td>
<td>439</td>
<td>495</td>
<td>519</td>
</tr>
<tr>
<td>390</td>
<td>446</td>
<td>6</td>
<td>18</td>
<td>6</td>
<td>0</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>-15</td>
<td>21</td>
<td>22</td>
</tr>
<tr>
<td>400</td>
<td>10</td>
<td>9</td>
<td>1</td>
<td>6</td>
<td>1</td>
<td>10</td>
<td>4</td>
<td>11</td>
<td>-19</td>
<td>21</td>
<td>26</td>
</tr>
<tr>
<td>410</td>
<td>8</td>
<td>11</td>
<td>11</td>
<td>7</td>
<td>2</td>
<td>10</td>
<td>7</td>
<td>-9</td>
<td>-24</td>
<td>23</td>
<td>27</td>
</tr>
<tr>
<td>420</td>
<td>14</td>
<td>9</td>
<td>16</td>
<td>6</td>
<td>0</td>
<td>11</td>
<td>7</td>
<td>9</td>
<td>-26</td>
<td>22</td>
<td>25</td>
</tr>
<tr>
<td>430</td>
<td>9</td>
<td>10</td>
<td>16</td>
<td>7</td>
<td>-1</td>
<td>12</td>
<td>6</td>
<td>10</td>
<td>-23</td>
<td>22</td>
<td>25</td>
</tr>
<tr>
<td>440</td>
<td>15</td>
<td>9</td>
<td>9</td>
<td>6</td>
<td>3</td>
<td>11</td>
<td>4</td>
<td>12</td>
<td>-19</td>
<td>22</td>
<td>25</td>
</tr>
<tr>
<td>450</td>
<td>0</td>
<td>9</td>
<td>16</td>
<td>7</td>
<td>-1</td>
<td>12</td>
<td>5</td>
<td>9</td>
<td>-19</td>
<td>22</td>
<td>25</td>
</tr>
<tr>
<td>460</td>
<td>0</td>
<td>12</td>
<td>7</td>
<td>5</td>
<td>8</td>
<td>13</td>
<td>5</td>
<td>7</td>
<td>15</td>
<td>22</td>
<td>24</td>
</tr>
<tr>
<td>470</td>
<td>0</td>
<td>10</td>
<td>16</td>
<td>7</td>
<td>0</td>
<td>11</td>
<td>3</td>
<td>7</td>
<td>16</td>
<td>21</td>
<td>24</td>
</tr>
<tr>
<td>480</td>
<td>0</td>
<td>9</td>
<td>18</td>
<td>7</td>
<td>-2</td>
<td>12</td>
<td>5</td>
<td>7</td>
<td>16</td>
<td>21</td>
<td>27</td>
</tr>
<tr>
<td>490</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

**Figure B-3**

Program I - Record Summary
RUN CARD

*ASG*T 1*F2/2//4
*ASG*T 2*F2/7/POS/7
*ASG*T 3*F2/30/30
*ASG*TB 11*BC*A06515

FORTRAN DECKS

*MAP*I *MAP**PROG
    SEG TOP*
    IN RESP
    LIB MSC*LOCALIB
    *XQT**PROG

DATA DECK

*PMD
*FIN

* 7/8 PUNCH

Figure B-5
Program II - Deck Setup
FIGURE C-1
PROGRAM I - MAIN PROGRAM
FIGURE C-2
PROGRAM I - SUBROUTINE STORED
FIGURE C-2 (CONT'D)

PROGRAM I - SUBROUTINE STORED
FIGURE 6-3
PROGRAM I - SUBROUTINE READ7
FIGURE C-4 (CONT'D)

PROGRAM II - MAIN PROGRAM
START

Select Time interval

Calc. freq. Cutoff

Average down to 300 pts or less

Fourier Transform

eliminate middle frequencies

inverse transform

interpolate to original time scale

Subtract from IIF

return

FIGURE C-5

PROGRAM II - SUBROUTINE FIIF