Final Technical Report
on the
Surface Electrical Properties
Experiment
report prepared by
The University of Toronto

and submitted to MIT
in fulfillment of the
subcontract on NASA
contract NAS9-11540

Personnel

D.W. Strangway
A.P. Annan
J.D. Redman
J.R. Rossiter
J.A. Rylaarsdam
R.D. Watts

Part II of III Parts
Digital Processing
2) Science Data Processing

R.D. Watts
SCIENCE DATA PROCESSING

The format of science (VCO) data from the acquisition system has been given in Figure 7 of J.D. Redman's report on data digitization. These data are processed in four major stages: 1) frequency and quality determination, 2) merging, 3) demultiplexing, 4) calibration.

In the notation of Figure 7 of Redman's report, one VCO measurement is recorded by 12 digits as follows

\[
N = (N_1 N_2 N_3)_{10} = \text{number of 5.2 KHz cycles counted}
\]

\[
V = (V_1 V_2 V_3 V_4 V_5)_{10} = \text{duration of M VCO cycles (\mu sec)}
\]

\[
R = ([R_0] R_1 R_2 R_3 R_4)_{10} = \text{duration of N 5.2 KHz cycles}
\]

Note that only the four low-order digits of \( R \) are recorded on tape. The high-order digit \( R_0 \) must be inferred from \( V \) and the operational characteristics of the DAS.

The determination of \( R_0 \) is based on the DAS design criterion that \( V \) and \( R \) should not differ by more than \( 1/500 \text{ sec} \approx 200 \mu \text{sec} \). Since \( R_0 \) represents the 10000's unit of \( \mu \text{sec} \), \( R_0 \) can conceivably be only one less, one greater, or equal to \( V_1 \). The algorithm diagrammed in Figure 1 bases the choice of one of these on the criterion of minimizing the difference between \( V \) and \( R \). This function was performed in the routine FREQ.

The foregoing discussion assumed that the DAS output
**Figure 1** Determination of $R_0$
conformed to specifications. Because this was not always true, a syntax analysis was performed by the FREQ routine. Four types of errors were recognized, and an error indicator was set according to the seriousness of the errors which were found. The error indicator was set to the sum of the following individual error levels:

- **level 0**: no errors
- **1**: the measured VCO frequency was outside the range 300 - 3000 Hz., which spans the expected frequencies.
- **2**: the measured reference frequency was more than 100 Hz different from the mean reference frequency of 5213 Hz.
- **4**: the measured periods of VCO and reference cycles differed by more than 210 μsec (1/5200 sec + 9.2%).
- **8**: one or more of the N, V, or R counters read zero.

Error level 8 was a terminal error, for which the frequency could not be computed and was set to zero. Error level 4 indicated a malfunction of the zero-crossing detector for the reference signal or a malfunction in the start/stop circuitry for reference period counting. However, the effects of a level 4 error could be quite small, especially
for low VCO frequencies. Error level 2 indicated either a large random fluctuation in the 5.2 KHz multivibrator in the DSEA, a tape-speed variation, or a counting problem. The tape-speed variation was probably the most frequent cause of this type of error, in which case the VCO frequency would still have been correctly determined. Error level 1 simply chopped off the allowable range of frequencies to a range spanning those observed in instrument calibration. VCO frequencies below 300 were set to 300, and those over 3000 were set to 3000.

For error levels below 8, the VCO frequency was computed by the following formulas:

\[ T_{VCO} = \frac{V}{4} \times 10^{-6} \text{ sec.} \]

\[ T_{REF} = \frac{R}{N} \times 10^{-6} \text{ sec.} \]

\[ f_{VCO} = T_{VCO}^{-1} = \frac{4}{V} \times 10^6 \text{ Hz} \]

\[ f_{REF} = \frac{N}{R} \times 10^6 \text{ Hz} \]

\[ f_{VCO - \text{CORRECTED}} = \frac{f_{VCO}}{f_{REF}} \]

\[ = \frac{5213 \times 4 \times R}{V \times N} \text{ Hz} \]

The source of the correction frequency 5213 has been discussed in Redman's report on data digitization. It represents the mean actual 5.2 KHz reference frequency.
Each of the four final science tapes (SEP400-403) was processed with the program CHANGE, which used FREQ to change all VCO readings to frequency-status pairs. The output data were stored as four files on tape SEPDO4, as shown in Figure 2. Since there were two bad records on tape SEP403, the frequency values from these records were set to zero, and the error status values were set to 16.

The drum-file copies of CHANGE output (T400-403) were merged by the program MERGE4. Output was stored in the first file of tape SEPDO5. Merging was performed according to the following rules.

1. Taking the four corresponding readings from files T400-403, reject those not having the lowest error status.
2. Search the values with lowest error status for the pair of values with the smallest discrepancy.
3. Average the values from the pair with smallest discrepancy. Use this value.

The reason for rule 1 is simply to use the most trustworthy values available, judged according to the seriousness of syntax errors. Rule 2 guarantees that the most repeatable measurements are used. Rule 3, assuming that the errors of measurement are Gaussian (which they are only approximately),
Figure 2  VCO DATA PROCESSING

CONVERT TO FREQ. AND STATUS USING FREQ.

MERGE4

DRUM FILE
T400

DRUM FILE
T401

DRUM FILE
T402

DRUM FILE
T403

SEPDO5
FILE 1

PRPL

DISREPEaney DISTRIBUTIONS

CONT'D
DRUM FILE DB-2

DRUM FILE DB-2-STR

MERGER

MERGES SCIENCE AND NAVIGATION DATA.

DRUM FILE TRAVERSE

BCDTAPES

CONVERTS FROM BINARY TO BCD

DRUM FILE DB-2-BCD

BCDTAPES

TAPECOPY

COPIES TRAVERSE-BCD, WRITES FILE MARK, COPIES DB-2-BCD, WRITES FILE MARK.

SEPD07

SEPD08

SEPD09

SEPD10

MAKES DATA UNIFORM THRU TURN AT EP-4
reduces the measurement error by a factor \( \sqrt{\frac{1}{2}} \).

MERGE4 provided the following ancillary functions:
1) printout of all non-zero status-value data which were used, 2) distributions of frequency discrepancies for the values from rule 3 above, broken down by 100 Hz intervals, with 5 Hz granularity. These are shown in Figure 3. 3) printouts of all points where the discrepancy of accepted values was large (Figure 4).

The discrepancy distributions from MERGE4 were stored in the drum file STATS. Program PRPL graphed the distributions as histograms on the line printer (Figure 3). These graphs show the frequency of occurrence of various discrepancies, where the source of the first accepted value precedes the source of the second accepted value in the sequence SEP400, 401, 402, 403, and the discrepancy is the first accepted value minus the second.

Science data output from MERGE4 were still in multiplexed form (i.e., data of various types were mixed together in a known sequence). The DEMUX program demultiplexed the data, collecting all data of one type (e.g. 4 MHz NS X) into a single array. Because the memory capacity of the computer was insufficient to hold all the data, the demultiplexing was done in two stages. Input data were first demultiplexed in core,
Figure 3

Distributions of discrepancies between the two accepted values (i.e. the values with lowest error level and smallest absolute discrepancy). Data are grouped according to the mean frequency, in 100 Hz intervals, and plotted with 5 Hz granularity.
**Frequency Interval (100Hz)** starts at 300Hz.

<table>
<thead>
<tr>
<th>Frequency (Hz)</th>
<th>Error (Hz)</th>
</tr>
</thead>
<tbody>
<tr>
<td>-100</td>
<td>0.00</td>
</tr>
<tr>
<td>-95</td>
<td>0.00</td>
</tr>
<tr>
<td>-90</td>
<td>0.00</td>
</tr>
<tr>
<td>-85</td>
<td>0.00</td>
</tr>
<tr>
<td>-80</td>
<td>0.00</td>
</tr>
<tr>
<td>-75</td>
<td>0.00</td>
</tr>
<tr>
<td>-70</td>
<td>0.00</td>
</tr>
<tr>
<td>-65</td>
<td>0.00</td>
</tr>
<tr>
<td>-60</td>
<td>0.00</td>
</tr>
<tr>
<td>-55</td>
<td>0.00</td>
</tr>
<tr>
<td>-50</td>
<td>0.00</td>
</tr>
<tr>
<td>-45</td>
<td>0.00</td>
</tr>
<tr>
<td>-40</td>
<td>0.00</td>
</tr>
<tr>
<td>-35</td>
<td>0.00</td>
</tr>
<tr>
<td>-30</td>
<td>0.00</td>
</tr>
<tr>
<td>-25</td>
<td>0.00</td>
</tr>
<tr>
<td>-20</td>
<td>0.00</td>
</tr>
<tr>
<td>-15</td>
<td>0.00</td>
</tr>
<tr>
<td>-10</td>
<td>0.00</td>
</tr>
<tr>
<td>-5</td>
<td>0.00</td>
</tr>
<tr>
<td>-0</td>
<td>0.00</td>
</tr>
<tr>
<td>5</td>
<td>0.00</td>
</tr>
<tr>
<td>10</td>
<td>0.00</td>
</tr>
<tr>
<td>15</td>
<td>0.00</td>
</tr>
<tr>
<td>20</td>
<td>0.00</td>
</tr>
<tr>
<td>25</td>
<td>0.00</td>
</tr>
<tr>
<td>30</td>
<td>0.00</td>
</tr>
<tr>
<td>35</td>
<td>0.00</td>
</tr>
<tr>
<td>40</td>
<td>0.00</td>
</tr>
<tr>
<td>45</td>
<td>0.00</td>
</tr>
<tr>
<td>50</td>
<td>0.00</td>
</tr>
<tr>
<td>55</td>
<td>0.00</td>
</tr>
<tr>
<td>60</td>
<td>0.00</td>
</tr>
<tr>
<td>65</td>
<td>0.00</td>
</tr>
<tr>
<td>70</td>
<td>0.00</td>
</tr>
<tr>
<td>75</td>
<td>0.00</td>
</tr>
<tr>
<td>80</td>
<td>0.00</td>
</tr>
<tr>
<td>85</td>
<td>0.00</td>
</tr>
<tr>
<td>90</td>
<td>0.00</td>
</tr>
<tr>
<td>95</td>
<td>0.00</td>
</tr>
<tr>
<td>100</td>
<td>0.00</td>
</tr>
</tbody>
</table>

**Average Absolute Error = .3Hz.**
**FREQUENCY INTERVAL (100Hz) STARTS AT 400Hz.**

<table>
<thead>
<tr>
<th>Frequency (Hz)</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>-100</td>
<td>0.00</td>
</tr>
<tr>
<td>-95</td>
<td>0.00</td>
</tr>
<tr>
<td>-90</td>
<td>0.00</td>
</tr>
<tr>
<td>-85</td>
<td>0.00</td>
</tr>
<tr>
<td>-80</td>
<td>0.00</td>
</tr>
<tr>
<td>-75</td>
<td>0.00</td>
</tr>
<tr>
<td>-70</td>
<td>0.00</td>
</tr>
<tr>
<td>-65</td>
<td>0.00</td>
</tr>
<tr>
<td>-60</td>
<td>0.00</td>
</tr>
<tr>
<td>-55</td>
<td>0.00</td>
</tr>
<tr>
<td>-50</td>
<td>0.00</td>
</tr>
<tr>
<td>-45</td>
<td>0.00</td>
</tr>
<tr>
<td>-40</td>
<td>0.00</td>
</tr>
<tr>
<td>-35</td>
<td>0.00</td>
</tr>
<tr>
<td>-30</td>
<td>0.00</td>
</tr>
<tr>
<td>-25</td>
<td>0.00</td>
</tr>
<tr>
<td>-20</td>
<td>0.00</td>
</tr>
<tr>
<td>-15</td>
<td>0.00</td>
</tr>
<tr>
<td>-10</td>
<td>0.00</td>
</tr>
<tr>
<td>-5</td>
<td>0.00</td>
</tr>
<tr>
<td>0</td>
<td>0.00</td>
</tr>
<tr>
<td>5</td>
<td>0.00</td>
</tr>
<tr>
<td>10</td>
<td>0.00</td>
</tr>
<tr>
<td>15</td>
<td>0.00</td>
</tr>
<tr>
<td>20</td>
<td>0.00</td>
</tr>
<tr>
<td>25</td>
<td>0.00</td>
</tr>
<tr>
<td>30</td>
<td>0.00</td>
</tr>
<tr>
<td>35</td>
<td>0.00</td>
</tr>
<tr>
<td>40</td>
<td>0.00</td>
</tr>
<tr>
<td>45</td>
<td>0.00</td>
</tr>
<tr>
<td>50</td>
<td>0.00</td>
</tr>
<tr>
<td>55</td>
<td>0.00</td>
</tr>
<tr>
<td>60</td>
<td>0.00</td>
</tr>
<tr>
<td>65</td>
<td>0.00</td>
</tr>
<tr>
<td>70</td>
<td>0.00</td>
</tr>
<tr>
<td>75</td>
<td>0.00</td>
</tr>
<tr>
<td>80</td>
<td>0.00</td>
</tr>
<tr>
<td>85</td>
<td>0.00</td>
</tr>
<tr>
<td>90</td>
<td>0.00</td>
</tr>
<tr>
<td>95</td>
<td>0.00</td>
</tr>
<tr>
<td>100</td>
<td>0.00</td>
</tr>
</tbody>
</table>

**Average Absolute Error = 0.00 Hz.**
REQUENCY INTERVAL (100HZ) STARTS AT 300HZ.

<table>
<thead>
<tr>
<th>Frequency</th>
<th>Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>0.00</td>
</tr>
<tr>
<td>75</td>
<td>0.00</td>
</tr>
<tr>
<td>50</td>
<td>0.00</td>
</tr>
<tr>
<td>25</td>
<td>0.00</td>
</tr>
<tr>
<td>75</td>
<td>0.00</td>
</tr>
<tr>
<td>50</td>
<td>0.00</td>
</tr>
<tr>
<td>25</td>
<td>0.00</td>
</tr>
<tr>
<td>75</td>
<td>0.00</td>
</tr>
<tr>
<td>50</td>
<td>0.00</td>
</tr>
<tr>
<td>25</td>
<td>0.00</td>
</tr>
<tr>
<td>75</td>
<td>0.00</td>
</tr>
<tr>
<td>50</td>
<td>0.00</td>
</tr>
<tr>
<td>25</td>
<td>0.00</td>
</tr>
<tr>
<td>75</td>
<td>0.00</td>
</tr>
<tr>
<td>50</td>
<td>0.00</td>
</tr>
<tr>
<td>25</td>
<td>0.00</td>
</tr>
<tr>
<td>75</td>
<td>0.00</td>
</tr>
<tr>
<td>50</td>
<td>0.00</td>
</tr>
<tr>
<td>25</td>
<td>0.00</td>
</tr>
<tr>
<td>75</td>
<td>0.00</td>
</tr>
<tr>
<td>50</td>
<td>0.00</td>
</tr>
<tr>
<td>25</td>
<td>0.00</td>
</tr>
<tr>
<td>75</td>
<td>0.00</td>
</tr>
<tr>
<td>50</td>
<td>0.00</td>
</tr>
<tr>
<td>25</td>
<td>0.00</td>
</tr>
<tr>
<td>75</td>
<td>0.00</td>
</tr>
<tr>
<td>50</td>
<td>0.00</td>
</tr>
<tr>
<td>25</td>
<td>0.00</td>
</tr>
<tr>
<td>75</td>
<td>0.00</td>
</tr>
<tr>
<td>50</td>
<td>0.00</td>
</tr>
<tr>
<td>25</td>
<td>0.00</td>
</tr>
<tr>
<td>75</td>
<td>0.00</td>
</tr>
<tr>
<td>50</td>
<td>0.00</td>
</tr>
<tr>
<td>25</td>
<td>0.00</td>
</tr>
</tbody>
</table>

AVERAGE ABSOLUTE ERROR = 0.00HZ.
FREQUENCY INTERVAL (100HZ.) STARTS AT 1200HZ.
FREQUENCY INTERVAL (100Hz.) STARTS AT 2200Hz.

---

-1000
-500
500
1000
2000
3000
4000
5000
6000
7000
8000
9000
10000

---

\[ \text{AVERAGE ABSOLUTE ERROR} = 2.4\text{Hz.} \]
FREQUENCY INTERVAL (100HZ.) STARTS AT 25000HZ.

<table>
<thead>
<tr>
<th>Frequency (Hz)</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>0.94</td>
</tr>
<tr>
<td>99</td>
<td>0.92</td>
</tr>
<tr>
<td>98</td>
<td>0.90</td>
</tr>
<tr>
<td>97</td>
<td>0.88</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>100</td>
<td>2.33</td>
</tr>
</tbody>
</table>

Average Absolute Error = 2.00 Hz.
Average Absolute Error = 5.9 Hz.
AVG ABS ERROR = 7.3 HZ.

FREQUENCY INTERVAL (1.1 CHZ) STARTS AT 2 CHZ.
AVERAGE ABSOLUTE ERROR = 3.0 HZ.
NUMBER OF ACCEPTANCES
AT EACH ERROR LEVEL.

NORMAL EXIT: EXECUTION TIME: 3944 MILLISECONDS.
50 records at a time, then output to random-access drum. The next 50 records were then demultiplexed in core, and these data were concatenated to the previous data on the drum. The random access feature was used to skip over space reserved for as yet unprocessed data. Figure 5 shows the general flow of the DEMUX program as well as diagrams of core and drum space allocations.

The DEMUX program contained a coding error which deleted data from the arrays for 4, 8, 16, and 32 MHz. The error occurred in the final transfer of data from drum to tape, in subroutine TAPER (internal subroutine in DEMUX). Figure 6 diagrams the nature of the error. In the tape output routine, the entire 400*N data array (N is the number of samples per record of the data type being considered) should have been processed at once. In Figure 6, the data array in the drum diagram should have been written on tape as 4 contiguous parts of 387 values, truncated to 387. The effect was a compression of the data - 13 values missing after every 387 values on tape. There are (N-1) such error regions. At the end of the tape arrays, there is garbage of length 13* (N-1).

The error is complicated somewhat by the next stage of processing. Because of the design of the DAS, the mode word appearing in record i predicted the receiver mode in
Figure 5 DEMUX - Demultiplexing Flow

Drum space of appropriate size is reserved for each data type (e.g., 4 MHz NS 2).

One core array of appropriate size is reserved for each data type.

Successive records fill up core arrays, which are full after 50 records.

Cont'd
WRITE ARRAYS ONTO DRUM

FOR EACH DATA TYPE

READ FROM DRUM

WRITE TO TAPE

TAPE END-OF-FILE

STOP

DATA IS APPENDED TO PREVIOUSLY WRITTEN DATA OF SAME TYPE ON DRUM.

ALL DATA OF ONE TYPE (387 x N SAMPLES PER RECORD)

N BLOCKS OF 387 WORDS
Figure 6  Demultiplexing Error

Example:  8 MHz data (4 samples / input record)

**DRUM:**

<table>
<thead>
<tr>
<th>RECORDS</th>
<th>1 - 50</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>51 - 100</td>
<td></td>
</tr>
<tr>
<td>101 - 150</td>
<td></td>
</tr>
<tr>
<td>151 - 200</td>
<td></td>
</tr>
<tr>
<td>201 - 250</td>
<td></td>
</tr>
<tr>
<td>251 - 300</td>
<td></td>
</tr>
<tr>
<td>301 - 350</td>
<td></td>
</tr>
<tr>
<td>351 - 387</td>
<td></td>
</tr>
<tr>
<td>GARBAGE</td>
<td></td>
</tr>
</tbody>
</table>

**TAPE:**

<table>
<thead>
<tr>
<th>RECORDS</th>
<th>1 - 50</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>51 - 100</td>
<td></td>
</tr>
<tr>
<td>101 - 150</td>
<td></td>
</tr>
<tr>
<td>151 - 200</td>
<td></td>
</tr>
<tr>
<td>201 - 250</td>
<td></td>
</tr>
<tr>
<td>251 - 300</td>
<td></td>
</tr>
<tr>
<td>301 - 350</td>
<td></td>
</tr>
<tr>
<td>351 - 387</td>
<td></td>
</tr>
<tr>
<td>GARBAGE</td>
<td></td>
</tr>
</tbody>
</table>

* Denotes and end-of-record as written onto tape, with the next 13 values skipped (the next record starts 13 values later). For this example, the final 39 values are garbage.
record i + 1. To make the mode array agree exactly with the data arrays, with no shift, N points were dropped from the start of each data array. This was accomplished by reading N blocks of 387 values (concatenated), deleting the first N values, and writing N blocks of 386 values. This was done by the REBLOCK program. The output was stored in drum file VCO.

Location of errors - data is missing after the following locations:

<table>
<thead>
<tr>
<th>SAMPLE #</th>
<th>APPARENT TIME FROM START</th>
</tr>
</thead>
<tbody>
<tr>
<td>4 MHz</td>
<td></td>
</tr>
<tr>
<td>385</td>
<td>0:20:47</td>
</tr>
<tr>
<td>8 MHz</td>
<td></td>
</tr>
<tr>
<td>383</td>
<td>0:10:20</td>
</tr>
<tr>
<td>770</td>
<td>0:20:47</td>
</tr>
<tr>
<td>1157</td>
<td>0:31:14</td>
</tr>
<tr>
<td>16 MHz</td>
<td></td>
</tr>
<tr>
<td>379</td>
<td>0:05:07</td>
</tr>
<tr>
<td>766</td>
<td>0:10:20</td>
</tr>
<tr>
<td>1153</td>
<td>0:15:34</td>
</tr>
<tr>
<td>1540</td>
<td>0:20:47</td>
</tr>
<tr>
<td>1927</td>
<td>0:26:01</td>
</tr>
<tr>
<td>2314</td>
<td>0:31:14</td>
</tr>
<tr>
<td>2701</td>
<td>0:36:28</td>
</tr>
<tr>
<td>32 MHz</td>
<td></td>
</tr>
<tr>
<td>374</td>
<td>0:03:06</td>
</tr>
<tr>
<td>761</td>
<td>0:06:19</td>
</tr>
<tr>
<td>1148</td>
<td>0:09:32</td>
</tr>
</tbody>
</table>
To correct the timing errors, 13 filler values should be inserted following each of the tabulated error locations. This should be done with care, since at 8, 16, and 32 MHz the insertion of the first 13 fillers moves the location for insertion of subsequent sets of 13 fillers. The last \(13 \times (N-1)\) values should be discarded.

The next step of processing was conversion from VCO frequencies to dB values for scientific data, and from VCO frequency to temperature. The calibration data for the flight unit were typed in and stored in drum files TESTLEVELS and VCOFREQS. These data are printed out in Figure 7. The receiver was operating in the medium (65°) to hot (105°) and above temperature range during the entire traverse, so the cold calibration was ignored.

A two-dimensional linear interpolation was done by
CALIBRATE, first determining the temperature, then using the temperature to interpolate between the medium and high temperature dB vs. frequency curves. Since the calibration data were given as VCO frequency as a function of dB input power, the inverse function first had to be determined. This was done by linear interpolation for VCO frequencies in 1 Hz steps from 300 Hz to 3000 Hz. The dB levels for frequencies below/above the lowest/highest calibrated value were set to the lowest/highest value. Interpolation was not done between these 1 Hz values - they were used purely as a lookup table. The granularity of the data (near the center of the calibration curves) is about 1/27 dB, or .9% on a linear power scale.

Up to this time, there has been no mention of the disposition of the following arrays: MODE, TEMP, TXOFF, CAL. The contents of each will be described here.

The MODE array consists of 386 words of the form \((\text{MAR})_{10}\), where \(M\) (the 100's digit) is 1 or 2 depending on whether the receiver was in sync'ed or sync-search mode during the \(i^{th}\) record, corresponding to the \(i^{th}\) word in the MODE array. The 10's digit, \(A\), is the antenna indicator. Values 1, 2, and 3 correspond to the \(X, Y,\) and \(Z\) antennas during synch search and to the
### Figure 7. Flight Unit Calibration Data

#### 1 MHz

<table>
<thead>
<tr>
<th>DBM</th>
<th>X ANTENNA</th>
<th>Y ANTENNA</th>
<th>Z ANTENNA</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>COLD</td>
<td>MED</td>
<td>HOT</td>
</tr>
<tr>
<td>-134.23</td>
<td>380</td>
<td>366</td>
<td>349</td>
</tr>
<tr>
<td>-129.24</td>
<td>418</td>
<td>400</td>
<td>376</td>
</tr>
<tr>
<td>-119.29</td>
<td>645</td>
<td>631</td>
<td>574</td>
</tr>
<tr>
<td>-109.37</td>
<td>947</td>
<td>940</td>
<td>899</td>
</tr>
<tr>
<td>-99.41</td>
<td>1202</td>
<td>1195</td>
<td>1168</td>
</tr>
<tr>
<td>-89.47</td>
<td>1426</td>
<td>1424</td>
<td>1418</td>
</tr>
<tr>
<td>-79.54</td>
<td>1684</td>
<td>1686</td>
<td>1679</td>
</tr>
<tr>
<td>-59.68</td>
<td>2259</td>
<td>2239</td>
<td>2225</td>
</tr>
<tr>
<td>-49.74</td>
<td>2517</td>
<td>2492</td>
<td>2471</td>
</tr>
<tr>
<td>-39.80</td>
<td>2769</td>
<td>2759</td>
<td>2744</td>
</tr>
<tr>
<td>-34.84</td>
<td>2897</td>
<td>2880</td>
<td>2873</td>
</tr>
<tr>
<td>-29.86</td>
<td>2946</td>
<td>2962</td>
<td>2970</td>
</tr>
</tbody>
</table>

#### 2 MHz

<table>
<thead>
<tr>
<th>DBM</th>
<th>X ANTENNA</th>
<th>Y ANTENNA</th>
<th>Z ANTENNA</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>COLD</td>
<td>MED</td>
<td>HOT</td>
</tr>
<tr>
<td></td>
<td>COLD</td>
<td>MED</td>
<td>HOT</td>
</tr>
<tr>
<td>-134.56</td>
<td>387</td>
<td>376</td>
<td>356</td>
</tr>
<tr>
<td>-129.56</td>
<td>442</td>
<td>422</td>
<td>391</td>
</tr>
<tr>
<td>-119.61</td>
<td>691</td>
<td>680</td>
<td>620</td>
</tr>
<tr>
<td>-109.69</td>
<td>972</td>
<td>964</td>
<td>924</td>
</tr>
<tr>
<td>-99.72</td>
<td>1217</td>
<td>1213</td>
<td>1190</td>
</tr>
<tr>
<td>-89.78</td>
<td>1435</td>
<td>1441</td>
<td>1435</td>
</tr>
<tr>
<td>-79.88</td>
<td>1698</td>
<td>1701</td>
<td>1689</td>
</tr>
<tr>
<td>-60.01</td>
<td>2272</td>
<td>2253</td>
<td>2242</td>
</tr>
<tr>
<td>-50.04</td>
<td>2530</td>
<td>2507</td>
<td>2489</td>
</tr>
<tr>
<td>-40.10</td>
<td>2781</td>
<td>2774</td>
<td>2763</td>
</tr>
<tr>
<td>-35.14</td>
<td>2890</td>
<td>2895</td>
<td>2892</td>
</tr>
<tr>
<td>-30.16</td>
<td>2948</td>
<td>2967</td>
<td>2979</td>
</tr>
</tbody>
</table>

#### 4 MHz

<table>
<thead>
<tr>
<th>DBM</th>
<th>X ANTENNA</th>
<th>Y ANTENNA</th>
<th>Z ANTENNA</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>COLD</td>
<td>MED</td>
<td>HOT</td>
</tr>
<tr>
<td></td>
<td>COLD</td>
<td>MED</td>
<td>HOT</td>
</tr>
<tr>
<td>-135.05</td>
<td>392</td>
<td>380</td>
<td>361</td>
</tr>
<tr>
<td>-130.06</td>
<td>447</td>
<td>430</td>
<td>399</td>
</tr>
<tr>
<td>-120.12</td>
<td>701</td>
<td>690</td>
<td>637</td>
</tr>
<tr>
<td>-110.20</td>
<td>980</td>
<td>974</td>
<td>938</td>
</tr>
<tr>
<td>-100.23</td>
<td>1446</td>
<td>1224</td>
<td>1204</td>
</tr>
<tr>
<td>-90.29</td>
<td>1626</td>
<td>1451</td>
<td>1447</td>
</tr>
<tr>
<td>-80.38</td>
<td>1709</td>
<td>1713</td>
<td>1712</td>
</tr>
<tr>
<td>-60.52</td>
<td>2280</td>
<td>2263</td>
<td>2255</td>
</tr>
<tr>
<td>-50.55</td>
<td>2540</td>
<td>2518</td>
<td>2501</td>
</tr>
<tr>
<td>-40.63</td>
<td>2789</td>
<td>2783</td>
<td>2774</td>
</tr>
<tr>
<td>-35.66</td>
<td>2895</td>
<td>2900</td>
<td>2900</td>
</tr>
<tr>
<td>-30.67</td>
<td>2947</td>
<td>2967</td>
<td>2980</td>
</tr>
</tbody>
</table>
### 8 MHz

<table>
<thead>
<tr>
<th>DBM</th>
<th>X Antenna</th>
<th>Y Antenna</th>
<th>Z Antenna</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>COLD</td>
<td>MED</td>
<td>HOT</td>
</tr>
<tr>
<td>-134.86</td>
<td>437</td>
<td>427</td>
<td>401</td>
</tr>
<tr>
<td>-129.86</td>
<td>515</td>
<td>505</td>
<td>464</td>
</tr>
<tr>
<td>-119.98</td>
<td>789</td>
<td>795</td>
<td>747</td>
</tr>
<tr>
<td>-110.04</td>
<td>1059</td>
<td>1055</td>
<td>1018</td>
</tr>
<tr>
<td>-100.04</td>
<td>1292</td>
<td>1294</td>
<td>1283</td>
</tr>
<tr>
<td>-98.11</td>
<td>1524</td>
<td>1532</td>
<td>1527</td>
</tr>
<tr>
<td>-80.18</td>
<td>1797</td>
<td>1799</td>
<td>1798</td>
</tr>
<tr>
<td>-70.26</td>
<td>2083</td>
<td>2074</td>
<td>2066</td>
</tr>
<tr>
<td>-60.34</td>
<td>2361</td>
<td>2340</td>
<td>2327</td>
</tr>
<tr>
<td>-50.36</td>
<td>2618</td>
<td>2601</td>
<td>2583</td>
</tr>
<tr>
<td>-40.41</td>
<td>2855</td>
<td>2857</td>
<td>2851</td>
</tr>
<tr>
<td>-35.44</td>
<td>2937</td>
<td>2950</td>
<td>2955</td>
</tr>
<tr>
<td>-30.47</td>
<td>2949</td>
<td>2970</td>
<td>2984</td>
</tr>
</tbody>
</table>

### 16 MHz

<table>
<thead>
<tr>
<th>DBM</th>
<th>X Antenna</th>
<th>Y Antenna</th>
<th>Z Antenna</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>COLD</td>
<td>MED</td>
<td>HOT</td>
</tr>
<tr>
<td>-134.58</td>
<td>430</td>
<td>430</td>
<td>408</td>
</tr>
<tr>
<td>-129.60</td>
<td>504</td>
<td>501</td>
<td>468</td>
</tr>
<tr>
<td>-119.80</td>
<td>777</td>
<td>795</td>
<td>756</td>
</tr>
<tr>
<td>-109.91</td>
<td>1051</td>
<td>1059</td>
<td>1033</td>
</tr>
<tr>
<td>-99.92</td>
<td>1285</td>
<td>1299</td>
<td>1297</td>
</tr>
<tr>
<td>-90.00</td>
<td>1517</td>
<td>1538</td>
<td>1543</td>
</tr>
<tr>
<td>-80.11</td>
<td>1799</td>
<td>1813</td>
<td>1817</td>
</tr>
<tr>
<td>-70.31</td>
<td>2076</td>
<td>2081</td>
<td>2082</td>
</tr>
<tr>
<td>-60.42</td>
<td>2354</td>
<td>2344</td>
<td>2338</td>
</tr>
<tr>
<td>-50.43</td>
<td>2612</td>
<td>2605</td>
<td>2596</td>
</tr>
<tr>
<td>-40.51</td>
<td>2853</td>
<td>2865</td>
<td>2868</td>
</tr>
<tr>
<td>-35.54</td>
<td>2940</td>
<td>2956</td>
<td>2967</td>
</tr>
<tr>
<td>-30.57</td>
<td>2955</td>
<td>2971</td>
<td>2987</td>
</tr>
</tbody>
</table>

### 32 MHz

<table>
<thead>
<tr>
<th>DBM</th>
<th>X Antenna</th>
<th>Y Antenna</th>
<th>Z Antenna</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>COLD</td>
<td>AMEIBNNR0T</td>
<td>COLD</td>
</tr>
<tr>
<td>-133.45</td>
<td>415</td>
<td>399</td>
<td>378</td>
</tr>
<tr>
<td>-128.49</td>
<td>499</td>
<td>464</td>
<td>410</td>
</tr>
<tr>
<td>-119.06</td>
<td>762</td>
<td>745</td>
<td>697</td>
</tr>
<tr>
<td>-109.26</td>
<td>1044</td>
<td>1030</td>
<td>985</td>
</tr>
<tr>
<td>-99.34</td>
<td>1282</td>
<td>1276</td>
<td>1238</td>
</tr>
<tr>
<td>-89.42</td>
<td>1513</td>
<td>1513</td>
<td>1499</td>
</tr>
<tr>
<td>-79.62</td>
<td>1806</td>
<td>1800</td>
<td>1788</td>
</tr>
<tr>
<td>-70.21</td>
<td>2066</td>
<td>2048</td>
<td>2034</td>
</tr>
<tr>
<td>-60.41</td>
<td>2337</td>
<td>2310</td>
<td>2274</td>
</tr>
<tr>
<td>-50.49</td>
<td>2593</td>
<td>2566</td>
<td>2488</td>
</tr>
<tr>
<td>-40.57</td>
<td>2834</td>
<td>2826</td>
<td>2756</td>
</tr>
<tr>
<td>-35.62</td>
<td>2928</td>
<td>2931</td>
<td>2927</td>
</tr>
<tr>
<td>-30.65</td>
<td>2950</td>
<td>2969</td>
<td>2981</td>
</tr>
</tbody>
</table>
frequency pairs (32, 16), (8, 4), (2, 1) during calibration (CAL) and transmitter off (TXOFF) frames. The l's digit, R, tells whether the receiver re-synced at the beginning of the record. The MODE array is the first array on the tape.

The TEMP array is self-explanatory. It gives the temperature in degrees Fahrenheit. It is the second array on the tape.

The TXOFF data fill 6 tape records (records 3-8). These were recorded during times when the transmitter was turned off, but the receiver was active. Figure 8 shows the contents of the TXOFF records.

Figure 8

TXOFF ARRAY - TRANSMITTER OFF DATA
6 RECORDS - 386 WORDS/RECORD

<table>
<thead>
<tr>
<th>MODE</th>
<th>DIGIT A</th>
<th>FREQ</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>________</td>
<td>1</td>
</tr>
<tr>
<td>Y</td>
<td>________</td>
<td>2</td>
</tr>
<tr>
<td>Z</td>
<td>________</td>
<td>3</td>
</tr>
<tr>
<td>X</td>
<td>________</td>
<td>1</td>
</tr>
<tr>
<td>Y</td>
<td>________</td>
<td>2</td>
</tr>
<tr>
<td>Z</td>
<td>________</td>
<td>3</td>
</tr>
</tbody>
</table>

The TXOFF arrays were fully calibrated (converted to dB)
in the same way as the science VCO data. This procedure required that the TXOFF data be present in core as the interpolation was being done for each frequency - antenna combination (the alternative was to generate the dB vs. VCO tables twice). The TXOFF array was therefore read into core preceeding the science VCO data, calibrated along with the science VCO data back to its position following the TEMP array.

The CAL array consists of three kinds of data:

1) receiver front-end noise measurement, 2) noise-diode source amplified by 20 dB, and 3) noise diode source unamplified. It contains 6 records, as diagrammed in Figure 9. These are the 9th-14th records on tape. Because these data are intended for use in calibrating the VCO - vs. - input power characteristics of the receiver, CAL data were left as VCO frequencies.

Figure 9

CAL ARRAY - CALIBRATION DATA
6 RECORDS
386 WORDS/RECORD

<table>
<thead>
<tr>
<th>MODE</th>
<th>DIGIT</th>
<th>A</th>
<th>FREQ</th>
</tr>
</thead>
<tbody>
<tr>
<td>G</td>
<td>1</td>
<td></td>
<td>32</td>
</tr>
<tr>
<td>NA</td>
<td>2</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>N</td>
<td>3</td>
<td></td>
<td>2</td>
</tr>
<tr>
<td>G</td>
<td>1</td>
<td></td>
<td>16</td>
</tr>
<tr>
<td>NA</td>
<td>2</td>
<td></td>
<td>4</td>
</tr>
<tr>
<td>N</td>
<td>3</td>
<td></td>
<td>1</td>
</tr>
</tbody>
</table>

G = Input grounded (front-end noise)
NA = Noise diode amplified 20 dB
N = Noise diode unamplified
Drum file DB-2 represented the full set of final science data. It was copied verbatim to the second file of the distribution tapes SEPD07 - D10.

For use in scientific interpretation, the turn at EP4, which was "removed" from the navigation data, had to be specially handled for scientific data as well. The treatment which was applied was: the turn was identified by the stops preceding and following the turn. The last values which existed prior to the execution of the turn, during the stop, were repeated through the time when the turn was completed. This gives the appearance for plotting, that the turn was not made. This function was performed by the STRAIGHTEN routine. The output was stored in drum file DB-2-STR.

The output from the navigation data processing, drum file ARROW-RANGES, was merged with DB-2-STR and stored in drum file TRAVERSE by MERGER.

The routine BCDTAPE was used in two versions (which differed only in the number of records they processed) to convert the binary data in TRAVERSE and DB-2 into BCD mode for tape transmission. Title arrays were placed at the beginning of each output file, named TRAVERSE-BCD and DB-2-BCD, respectively. These files were copied to tapes SEPD07-D10 for transmission.

The detailed format of transmission tapes SEPD07-D10 is given in Figure 10.
Figure 10

Distribution Tape Format

Tapes SEPDO7, SEPDO8, SEPDO9, SEPDO10

7-Track
Even Parity
800 BPI
BCD
Unlabeled
Fixed Unblocked Records
Record Size = Block Size = 386 - 6 char words
= 2316 chars.

Two Files

Second File: Unstraightened Science Data only.
<table>
<thead>
<tr>
<th>RECORD</th>
<th>FORMAT</th>
<th>CONTENTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>27(14A6),8A6</td>
<td>TITLE: 27 card images plus padding.</td>
</tr>
<tr>
<td>2</td>
<td>386I6</td>
<td>(MAR) (1).</td>
</tr>
<tr>
<td></td>
<td></td>
<td>M = 1 Data acquisition mode</td>
</tr>
<tr>
<td></td>
<td></td>
<td>M = 2 Sync acquisition mode</td>
</tr>
<tr>
<td></td>
<td></td>
<td>A = 1 X antenna, 32 or 16 MHz</td>
</tr>
<tr>
<td></td>
<td></td>
<td>A = 2 Y antenna, 8 or 4 MHz</td>
</tr>
<tr>
<td></td>
<td></td>
<td>A = 3 Z antenna, 2 or 1 MHz</td>
</tr>
<tr>
<td></td>
<td></td>
<td>R = 0 No receiver resync</td>
</tr>
<tr>
<td></td>
<td></td>
<td>R = 1 Receiver resync</td>
</tr>
<tr>
<td>3</td>
<td>386F6.1</td>
<td>TEMP: Temperature, degrees F.</td>
</tr>
<tr>
<td>4</td>
<td>386F6.1</td>
<td>TXOFF: X antenna, 32, 8, or 2 MHz</td>
</tr>
<tr>
<td>5</td>
<td></td>
<td>Y antenna, &quot; &quot; &quot; &quot; MHz</td>
</tr>
<tr>
<td>6</td>
<td></td>
<td>Z antenna, &quot; &quot; &quot; MHz</td>
</tr>
<tr>
<td>7</td>
<td></td>
<td>X antenna, 16, 4, or 1 MHz</td>
</tr>
<tr>
<td>8</td>
<td></td>
<td>Y antenna, &quot; &quot; &quot; MHz</td>
</tr>
<tr>
<td>9</td>
<td></td>
<td>Z antenna, &quot; &quot; &quot; MHz</td>
</tr>
<tr>
<td>10</td>
<td></td>
<td>CAL: Front-end noise, 32, 8 or 2 MHz</td>
</tr>
<tr>
<td>11</td>
<td></td>
<td>&quot; &quot; Diode + 20 DB, &quot; &quot; &quot;</td>
</tr>
<tr>
<td>12</td>
<td></td>
<td>&quot; &quot; Diode, &quot; &quot; &quot;</td>
</tr>
<tr>
<td>13</td>
<td></td>
<td>&quot; &quot; Front-end noise, 16, 4, or 1 MHz</td>
</tr>
</tbody>
</table>
14 386F6.1 CAL Diode + 20 DB, 16, 4, or 1 MHz
15 " " Diode, " " " "
16 " RANGE1 Range array for 1 MHz data, meters
17 " 1MHz Endfire X antenna power in dBm
18 " " " Y " "
19 " " " Z " "
20 " " Broadside X " "
21 " " " Y " "
22 " " " Z " "
23 " RANGE2 Range array for 2 MHz data
24 " 2 MHz Endfire X
25 " " " Y
26 " " " Z
27 " " Broadside X
28 " " " Y
29 " " " Z
30 2(386F6.1/) RANGE 4
31
32 " 4 MHz Endfire X
33
2(386F6.1/)  4 MHz Endfire Y
4(386F6.1/)  RANGE_8
8(386F6.1/)  RANGE_16
16 MHz Endfire X
128 13(386F6.1/) RANGE32

•

140

141 " 32 MHz Endfire X

•

153

•

206 " 32 MHz Broadside Z

•

218

End-of-file
FILE 2

Structured exactly like File 1 except range arrays are absent.

<table>
<thead>
<tr>
<th>RECORD</th>
<th>CONTENTS</th>
<th>RECORD</th>
<th>CONTENTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>TITLE</td>
<td>112</td>
<td>32 MHz</td>
</tr>
<tr>
<td>2</td>
<td>MODE</td>
<td>.</td>
<td>&quot;</td>
</tr>
<tr>
<td>3</td>
<td>TEMP</td>
<td>189</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>TXOFF</td>
<td>&quot;</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>CAL</td>
<td>&quot;</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>1 MHz</td>
<td>&quot;</td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>2 MHz</td>
<td>&quot;</td>
<td></td>
</tr>
<tr>
<td>28</td>
<td>4 MHz</td>
<td>&quot;</td>
<td></td>
</tr>
<tr>
<td>40</td>
<td>8 MHz</td>
<td>&quot;</td>
<td></td>
</tr>
<tr>
<td>64</td>
<td>16 MHz</td>
<td>&quot;</td>
<td></td>
</tr>
</tbody>
</table>
APPENDIX

Programs for processing science (VCO) data and merging it with Nav Data.
FUNCTION FREQ, Y, X

INTEGER VX, FREQ, SIG, STEP, IPER

FREQ = SIG * (1 - VX)

RETURN
FUNCTION: MODEF(x)
1
2  INTEGER X, XANT, YANT, RESYN, MODE, YNT
3
4  C GET DATA FROM DIF OR MODE
5  MODE=2+LUN(2,1,Y)
6  XANT=1+LUN(24,1,Y)
7  YANT=1+LUN(23,1,Y)
8  RESYN=1+LUN(1,1,Y)
9
10  C CHECK FOR ERROR CONDITIONS
11  IF(YANT.NE.1 .AND. YANT.NE.11) GOTO 12
12
13  C correct the mode indicator if resync has been received
14  IF(RESYN.EQ.1) MODE=1
15
16  C GENERATE ANOTHER DIGIT
17  ANT=3+2*YANT-YNT
18
19  C STACK DIGITS INTO MODE
20  MODE=11+MODE+10*YANT+RESYN
21
22  RETURN
23
24  C ERRORS DETECTED IN DATA
25  IF (XANT.NE.0) MODE, YANT, YANT, RESYN
26  WRITE(1,1) MODE, YANT, YANT, RESYN
27  END FORMAT(* MODE DIGIT ERROR, REAL DATA = *, 6I11)
28
29  RETURN
30  END
CHANGE

URAAH YCTI CHUSI
1  C CHANGE
2  C CONVERT CODE TO FREQUENCY AND STATUS
3  REAL RECORD(11), DATA(109)
4  INTEGER STATUS, P(N)
5  CALL NREAD (DATA(109), P(N))
6  C SKIP BAD RECORDS
7  CALL NREAD (DATA(109), P(N), P(N), P(N), P(N))
8  GO TO 1
9
10
11
12  C PROCESS ALL RECORDS
13  DO 100 P(N) = 1, 10
14
15  C READ A RECORD
16  CALL NREAD (DATA(109), P(N), P(N), P(N), P(N))
17  GO TO 15
18
19  C END OF FILE - WRITE ERRORS
20  WRITE(A, *WRITE, ERRORS)
21  AND FORMAT(*READ RECORD*, 14)
22  STOP
23
24  C FOUND RECORD - CALL FREQUENCY TO P(N), STATUS TO P(N)
25  WRITE(A, *READ RECORD*)
26  DO 12 P(N) = 1, 10
27  DATA(P(N)) = P(N)
28  STATUS(P(N)) = P(N)
29  GO TO 12
30
31  C CONVERT GOOD RECORD
32  DO 12 P(N) = 1, 10
33  DATA(P(N)) = DATA(P(N)) * (P(N) + 1), STATUS(P(N)) = P(N)
34  DATA(P(N)) = DATA(P(N)) * (P(N) + 1)
35  STATUS(P(N)) = 0
36
37  C WRITE DATA
38  WRITE(A, *DATA, STATUS)
39
40  C PROCESSING COMPLETED
41
42  //FUNCTION
43  //WRITE(A, *WRITE
44  //AND RECORDS PROCESSED*
45  //END OF FILE \LETTER ON UNIT 2)
46  STOP
47  END
```
C MERGEY
C MERGE Y, FIND FOUR SCIENCE TAPEs.
C REAL DATA(1,100), OUTPUT(100)
C INTEGER STATUS(4), IROD, COUNTS(17), I70D, MODE(4),
      N7(4), D7(7), RECORD, TAPE, TEMP, UNIT
C EQUIVALENCE (DATA(1,100), MODE(1))
C FOR EACH INPUT RECORD
C READ DATA TO TAPE
C TEMP \n.
C READ (UNIT) DATA(TAPE, "MOD"), "MOD" = 1, 4, (STATUS(4), MODE(4)),
      "MOD" = 1, 4
C CHECK MODES FOR AGREEMENT
C DO 3 TAPE = 2, 4
C IF (MODE(TAPE).NE.MODE(4)) GO TO 1
C IF (STATUS(TAPE).NE.TAPE) GO TO 2
C WRITE (4,100), MODE(4), TAPE,"MOD"(TAPE)
C AND FORMAT(4) MAPS COUNT AGAIN AT RECORD "14/20000000", MODE = 1, 4
C STOP
C CONTINUE
C FOR EACH WORD IN A RECORD, PROCESS THE DATA
C DO 4 MODE = 1, 4
C CALL PROCESS DATA(MODE), STATUS(MODE), OUTPUT(MODE)
C OUTPUT THE RESULTS
C WRITE (4 OUTPUT, MODE(4))
C FINISH PROCESSING
C END FILE 4
C WRITE (4,101)
C END FILE 4
C WRITE (4,102)(1, COUNTS(I7), I = 0, 14)
C AND FORMAT(4) MAPS COUNTS(14) TO OUTPUT(14)
C SAVE ERROR DISTRIBUTIONS FOR PLOTTING
C WRITE (4 OUTPUT)
C END FILE 3
C
```
SUBROUTINE PROCESSES (DATA, STATUS, OUTPUT)  
C SELECTS AND PROCESSES DATA  
REAL DATA(N1), STACK(N1)  
INTEGER COUNT, PSTAT, PIVOT, SOME, COUNT, PIVOT,  
, STATUS(N1), UNIT(N1)  
C COUNT VALUES AT MINIMUM ERROR LEVEL AND PLACE THEM  
C IN STACK.  
COUNT=1  
C MOVE FIRST VALUE INTO STACK, SET PSTAT  
STACK(1)=DATA(1)  
PSTAT=STATUS(1)  
UNIT(1)=1  
C FOR OTHER VALUES  
DO 1IVAL=2,N  
C COMPARE THE STATUS TO PSTAT  
1 IF(STATUS(IVAL)-PSTAT)EQ0, GO TO 1  
C NEW ERROR LEVEL IS LOWER - CLEAR THE LIST  
2 COUNT=0  
C NEW ERROR LEVEL IS SAME  
3 JUDGEMENT COUNTER AND ADD TO LIST  
COUNT=COUNT+1  
STACK(COUNT)=DATA(IVAL)  
PSTAT=STATUS(IVAL)  
3 IF(STATUS(N1+1))GO TO 4  
C YES - THE SINGLE VALUE  
OUTPUT=STACK(1)  
C IF ZERO STATUS, RETURN  
IF(PSTAT.EQ.0)RETURN  
C NON-ZERO STATUS MESSAGE  
WRITE(6,90)RECORD,.word,PSTAT,UNIT(1),OUTPUT  
90 AND FORMAT(1, RECORD,14,word,14, STATUS,13,  
, 2(1, TAPER,17, TINFO,12)))  
100 RETURN
MULTIPLE VALUES OF SAME STATUS
SET MINIMUM DIFFERENCE TO MAX, THEN SEEK ACTUAL MIN

FOR EACH PIVOT VALUE
COUNT = COUNT + 1
DO C PIVOT = 1, COUNT
PIVOT = PIVOT + 1
FOR EACH SECOND VALUE ABOVE PIVOT
DO A SECOND = PIVOT, COUNT
IS DISCREPANCY LARGER FOR THIS PAIR OF VALUES?
DIF = STACK (PIVOT) - STACK (SECOND)
IF DIF > 0.5 * DIFMIN, GO TO A
SMALLER = MAKE SUBSTITUTIONS, USE ANOTHER
DIFMIN = CS (DIF)
DIF = DIF
EFOA = STACK (PIVOT)
TAOFA = N (P (PIVOT))
EFOSE = STACK (SECOND)
TAOFE = N (P (SECOND))
OUTPUT = (EFOA + EFOSE)/2.
CONTINUE FOR MORE PIVOTS AND SECONDS
CONTINUE
CONTINUE

KEEP DIFFERENCE STATISTICS
IF EFO = (OUTPUT - 300) / 100 + 1,
IF EFO = MIN (77, MAX (1, IF EFO))
IF EFO1 = 1/4, +21.5
IF EFO = MIN (41, MAX (1, IF EFO))
DIST (1, IF EFO2) = DIST (1, IF EFO2) + 1

PRINT MESSAGE FOR LARGE DIFFERENCES OR NON-ZERO STATUS
IF EFO = MIN [6, 4] AND P (STAT) .EQ. 0 RETURN
WRITE (6, 401) REC, XORD, P (STAT), I (TAPE), CT (TAPE), L (TAPE), S (TAPE)

ACTION

END
C PROGRAM

1 C PRPL
2 C PLOTS ACCURACY STATISTICS ON PRINTER-PILOTTER
3 C
4 C PARAMETER LWIDE=101
5 C INTEGER DMAX,TOT,BLANK/* /;BAF/* ;STAR/* /*/;
6 C . DIST(41,27),LIVE(LWIDE),FORM1(10),FORM2(10)
7 C
8 C SET UP FORMATS
9 C LWIDE=LWIDE
10 C ENCODE(400,FORM1)LWIDE
11 C 400 FORMAT(*((5X,*'13*(1H-))*)
12 C ENCODE(401,FORM2)LWIDE
13 C 401 FORMAT(*((15X,*'13*'$A1,'F1.2'))
14 C
15 C READ ERROR DISTRIBUTION
16 C READ(4)DIST
17 C
18 C LOOP THROUGH PLOTS
19 C DO 1 J=1,27
20 C JFRTQ=300+100*(J-1)
21 C
22 C SET UP LINE BUFFER
23 C LINE(1)=BAF
24 C DO 2 I=2,LWIDE
25 C 2 LINE(I)=BLANK
26 C
27 C LOCATE MAXIMUM VALUE AND TOTAL WEIGHT
28 C DMAX=DIST(1,J)
29 C TOT=DMAX
30 C DO 3 I=2,41
31 C TOT=TOT+DIST(I,J)
32 C 3 DMAX=MAX(DMAX,DIST(I,J))
33 C DDMAX=DMAX
34 C IF(TOT)1,1*
35 C ATOT=TOT/100.
35* C
37* C PRINT TOP LINE
38* WRITE(6,600)JFREQ
39* 600 FORMAT:*'FREQUENCY INTERVAL (1CDHZ.) STARTS AT*'1E+HZ'.*)
40* C PRINT HEADING
41* WRITE(6,FORM1)
42*
43* C PRINT LINES
44* DO 4 I=1,41
45* IFREQ=E*(I-21)
46* IPO$=1.5+DIST(I,J)\cdot(LWIDE-1)/$2MA$X
47* IPO$=MIN(LWIDE\cdot MAX(1,IPO$))
48* IF(IPO$<EQ.1)GO TO 9
49* DO 3 II=2,IPO$
50* 8 LIN$(II)$=STAR
51* 9 X=DIST(I,J)$/ATC$
52* WRITE(6,FORM2)IFREQ\cdot LINE\cdot X
53* DO 10 II=2,IPO$
54* 10 LINE(II)$=$BLANK
55* LINE(LWIDE)$=BAR
56* 4 CONTINUE
57*
58* C WRITE BOTTOM LINE
59* WRITE(6,FORM1)
60*
61* C COMPUTE MEAN ABSOLUTE ERROR
62* IERR$=0
63* TOT$=0
64* DO 20 I=2,41
65* K=DIST(I,J)
66* TOT$=TOT$+K
67* 20 IERR$=IERR$+K$\cdot IABS(I-21)
68* ERR=(5.*IERR$)/TOT
69* WRITE(6,601)ERR
70* 601 FORMAT:*'AVERAGE ABSOLUTE ERROR ='}1E+HZ'.*)
71*
72* 1 CONTINUE
73* END
SUBROUTINE MOVE (DATA, SOURCE, SIZE)
INTEGER SIZE, SOURCE (SIZE)
REAL DATA (SIZE)
DO 1 I = 1, SIZE
   J = SOURCE (I)
1  DO 1 J = J + 1, SIZE
   DATA (I, CORE, J) = DATA (I, CORE, J) + DATA (I, CORE, J)
RETURN

SUBROUTINE DRUM
INTEGER A, B, D, E
AND FORMAT (MOVING RECORDS TO DRUM, IN (A, B), D)
JUMP =
CALL DRUM (T, 1)
CALL DRUM (D, 1)
CALL DRUM (D, 1)
CALL DRUM (D, 1)
CALL DRUM (D, 1)
CALL DRUM (D, 4)
CALL DRUM (D, 1)
CALL DRUM (D, 1)
RETURN
SUBROUTINE CPPF5 (DATA, SIZE)
INTEGER SIZE, REST
REAL DATA (0: SIZE, 0)
REST = SIZE + (INT (SIZE - 1) + 1) * SIZE
DO 1 ICOMP = 1, SIZE
DO 1 J = 1, SIZE
1 CALL TRANS (DATA (1, J, ICOMP), REST + I)

SUBROUTINE TRANS (DATA, RECORD)
INTEGER RECORD, DATA (0, 0)
WRITE (2, RECORD) DATA
CLOSE (2, INPUT)
RETURN

SUBROUTINE TAPF5
REAL DATA (0: R), OUTPUT (38, 38)
EQUIVALENCE (DATA, OUTPUT)
IF (REC) = 1
WRITE (X, ALL)
WRITE (7, PRINTING) DATA ON TAPF5
DO 1 K = 1, 3
DO 1 J = 1, A
DO 1 I = 1, 8
1 CALL INPUT (DATA (1, I, K), IF (REC))
IF (REC) = REC + 1
WRITE (7, OUTPUT)
RETURN

SUBROUTINE INPUT (DATA, IF)
REAL DATA (0)
READ (38, IF) DATA
CLOSE (3, INPUT)
RETURN

END
```
CALIBRT

DIMENSION VCO(13,1,1,1)
REAL D(13,1)
INTEGER I(13,1,1,1)

C INPUT VCO DATA
DO 1 I=1,13
  READ(1,100)I(1,J,K,L),I=1,11
  FORMAT(1x,I4,1x,4F12.6)

C CHANGE SIGNS
    I=1,13
    I(1,K,L)=I(1,K,L)*(-1)

C OUTPUT BY FREQ
    IFREQ=2*(IFREQ-1)
    WRITE(A,A11)IFREQ
    FORMAT(1x,10x,A,10x,10x,A)
    IFREQ=2*(IFREQ-1)
    WRITE(A,A11)IFREQ
    FORMAT(1x,10x,A,10x,10x,A)

C OUTPUT THE DATA
    FORMAT(10x,10x,10x,10x,10x,10x)
    IFREQ=2*(IFREQ-1)
    FORMAT(10x,10x,10x,10x,10x,10x)

C END FORMAT(1x,10x,A,10x,10x,A)
STOP
END
```
REBLOCK

03AAR0041 TR, REBLOCK

1       C  REBLOCK
2       C  COPIES DATA DELETING FIRST RECORD AND SHIFTS 200F
3       INTEGER MORE(300), SIZE(X) /1, 1, 2, 4, 8, 16/
4       REAL DATA(307,44)
5       C  COPY MORE
6       READ(7) MORE
7       CALL WRITE(MORE)
8
9       C  COPY TEMP
10      CALL COPY(I)
11
12      C  COPY XOFF, CAL, I, AND Y OFF DATA
13      DO 1 T1=1,24
14 1       C  CALL COPY(I)
15      1
16      C  COPY HIGHER FREQUENCY DATA
17      DO 2 T2=1,3,6
18 2       SIZE X=SIZE (T2F4)
19      DO 2 T2=1,6,
20 2       C  CALL COPY(SIZE X)
21      END FILE X
22      WRITE(X,6300)
23      END FORMAT(* END OF FILE WRITTEN ON UNIT 0*)
24      STOP
25
26      SUBROUTINE COPY(I)
27      DO 1 T1=1,N
28 1       C  CALL READ(IN1DATA(I,1))
29      CALL OUT(DATA(N+1,1),N)
30      RETURN
31
32      SUBROUTINE OUT(IN1DATA,M)
33      REAL DATA(307,N)
34      DO 1 T1=1,N
35 1       C  CALL WRITE(DATA(I,N))
36      RETURN
37
38      SUBROUTINE READ(I)
39      REAL N(24)
40      READ(7,7)
41      RETURN
42
43      SUBROUTINE WRITE(I)
44      REAL N(24)
45      WRITE(I)
46      RETURN
47
48      END
CALIBRATE

1 C CALIBRATE
2 C USES MEASUREMENT DATA TO CALIBRATE DENSITY LEVELS
3 INTEGER MODE(35), size(4)/1, 1, 2, 4, 1, 7
4 TANH(A)/1, 2, 1, 1, TANH(C)/1, 2, 1, 2, 1, 2,
5 REAL TEMP(35), TECH(35, 3, 3), table(35)
6 TAN(2, 270/1, 1, DU[2], 1, tCOM[2], 0, 1, 3, 6)

7 C READ CALIBRATION DATA
8 CALL SETUP
9
10 C READ MODE ARRAY
11 READ(J):=MODE
12 C COPY MODE ARRAY
13 WRITE(A)
14
15 C READ TEMP ARRAY
16 READ(J):=TEMP
17 C CONVERT TO TEMPERATURE
18 DO 1 I=1, 35
19 TEMP(I)=5*(1+TEMP(I)-4)
20 C WRITE TEMP ARRAY
21 WRITE(A)
22
23 C READ TYPEF ARRAY
24 DO 2 ITRANS=1, 3
25 DO 2 ICOMP=1, 2
26 2 CALL WEEK(TYPEF(I, ICOMP), ITRANS)
27
28 C COPY CALI ARRAYS
29 DO 3 ICOMP=1, A
30 READ(I):=COMP
31 3 WRITE(A)
32
33 C FOR EACH FREQUENCY
34 DO 4 IFREQ=1, 6
35 IFREQ=2*((IFREQ-1)+1)*MOD(1, IFREQ-1, 2)
36 WRITE(A)
37 AND FORMAT(* START PROCESSING FOR *, E4.1, ... DATA, */)
38 IF(KA=0/1,(I/FREQ))
39
40 C COMPUTE INTERPOLATION TABLE
41 DO 5 ICOMP=1, 3
42 5 CALL INTERP(D1, IFREQ), VCO[1], ICOMP, 2, IFREQ
43 VCO[1, ICOMP, 3, IFREQ], table[1, I, ICOMP])
44
45 C INTERPOLATE THE SCIENCE DATA
46 DO 7 ITRANS=1, 3
47 DO 7 ICOMP=1, 3
48 7 CALL WEEK(TABLE[1, I, ICOMP])
49 DO 7 IFREQ=1, 1, 3
50 READ(J):=COMP
51 WRITE(A)
52 DO 8 I=1, 3, 6
53 8 IWORK(I)=POW(IWORK(I), IFREQ(I))
54 8 WRITE(A)
C INTERPOLATE THE INPUT DATA
DO 14  I = 1, 3
Q
C MEMORE (I )
C MEMORE ( I ) = I
C MA BE THE MORE ANTEA DIGIT
C IF ANTEA DIGIT INDICATES OTHER FREQUENCY, CORD THIS PL
C (INVESTIGATE THE ) GO TO 14
C FREQUENCY ACCRES - CALL CORRESPONDING X FRAME DIGIT
C CALL XNATAG( I )
C INTERPOLATE
C DO 16  ICOM = 1,3
C CALL RESULTABLE ( I , ICOM)

16  TYPEF11, ICON, I = XNATAG(YF11, 1, ICOM, I, I)  I = P(I)
C CONTINUE
C FINISH FOR ALL FREQUENCY
C CONTINUE
C WRITE TYPEI1 DATA
C DO 18  IFRAME = 1, 3
C DO 18  ICOM = 1, 3
C CALL WRTI1( TYPE11, ICOM, IFRAME)
C WRTI1(L, 3)
C END FORMAT END OF THE WRITING ON UNIT O77
C END OF CALULATION PROCESSING
C STOP
C
C SUBROUTINE SETI
C READS CALIBRATION DATA ALAYS
READ ( 5, (H10 ) (DH1 , IFREQ1 , I = 13 ) , I - 1, A )
C FORMAT (7E7/4)
C CHANGE SIGNS
DO 1 I = D2, 4
DO 1 I = D2, 4
1 DRI (1, IFREQ1 ) = - DRI (1, IFREQ1 )
C READ YCO DATA
DO 2 ITEMP = 1, 4
DO 2 ITEMP = 1, 4
DO 2 ITEMP = 1, 4
2 READ ( 4, (H10 ) (IFREQ1 , I = 13 ) , I - 1, A )
C FORMAT (7E7/4)
C RETURN
C
SUBROUTINE interp(DP, VCO, Y)
C INTERFACE FOR SINGLE TIME PERIOD ALG.
REAL DP(3, 2), VCO(1), Y
INTEGER I, J, NEXT

DEFINE N(1) = DPER(1, 1)

C FILL THE ARRAY UP TO THE FIRST VCO VALUE
I = 0
IF (VCO(I) .LT. 360.) GO TO 1
J = VCO(I) - 360.

DO 2 I = 1, J
2 P(I) = DP(I)

J = J + 1

C INTERPOLATE UP TO THE LAST VCO VALUE
1 DO 3 NEXT = 2, 13

N(1) = DP(NEXT - 1)
VCO = VCO(NEXT - 1)
VCO = VCO(NEXT)
X = (VCO - VCO) / (VCO - VCO)

DO 4 I = 1, 270
4 IF (I .LE. 360 - VCO) GO TO 2

P(I) = (I + 270 - VCO) * X + NP

RETURN

2 J = 1

C FILL THE ARRAY BEYOND THE LAST VCO VALUE

N(1) = DP(13)

DO 5 I = 1, 270
5 P(I) = DP(13)

RETURN

END
SUBROUTINE READ(7)
REAL Y(dna)
RETURN
END

SUBROUTINE WRITE(7)
REAL Y(dna)
WRITE(7,*)
RETURN
END

C FUNCTION FOR TEMPERATURE INTERPOLATION
C FUNCTION HARDCODED
REAL DERIV(2,2,11)
INTEGER !
IF (ERR0R,ERR) GO TO 1
1-ERROR=ERROR

POW=POW([1,1]+DERIV-A)/40.*(POW([2,1]-POW([1,1]))
RETURN
END

ENTRY INTERPOLATION(VCOUNT,POW)
ENTRY INTERPOLATE ENO ONE FREQUENCY-INTERPOLATION COMBINATION
ENTRY INTERPOLATE FOR 15 AND 10- DEGREE ARCS
ENTRY INTERPOLATE(VCOUNT,POW)
ENTRY WRITE TO ORIGINAL INTERPOLATIONS
ENTRY WRITE (POW)
RETURN
ARRANGE

C MOVE TYPEF DATE BACK INTO POSITION AT END OF FILE
C      REAL F(I,4)
C      INTEGER SIZE(A/1/),1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16
C
C      COPY TEMP, TYPEF
C      CALL COPY(1,TEMP*)
C      CALL COPY(1,TEMP*)
C
C      SKIP CAL:
C      CALL SKIP(6,'CAL*')
C
C      SKIP SCIENCE DATA
C      DO 1 TYPEF=1,8
C         1 CALL SKIP(A,ST/F(I,TYPEF),TITLE(TYPEF))
C
C      COPY TXOFF
C      CALL COPY(6,'TXOFF*')
C
C      START OVER
C      RETURN
C
C      WRITE(A,ALL)
C      ADD FORMAT(* READING UNIT 2*)
C
C      SKIP Typef And Temp
C      CALL SKIP(1,'Typef*')
C      CALL SKIP(1,'TEMP*')
C
C      COPY CAL
C      CALL COPY(6,'CAL*')
C
C      COPY SCIENCE DATA
C      DO 2 TYPEF=1,8
C         2 CALL COPY(A,ST/F(I,TYPEF),TITLE(TYPEF))
C
C      ENDTYPE
C      WRITE(A,ALL)
C      ADD FORMAT(* END OF FILE WRITTEN ON UNIT 2,')
C         '* END PROCESSING')
C
C      SUBROUTINE SKIP(I,TITLE)
C      WRITE(A,ALL,TITLE)
C      ADD FORMAT(* SKIPPING,'11,' RECORDS OF ',' ,DATA.*)
C      DO 1 I=1,11
C      1 READ(7)OKK
C      RETURN
C
C      SUBROUTINE COPY(I,TITLE)
C      WRITE(A,ALL,TITLE)
C      ADD FORMAT(* COPYING,'11,' RECORDS OF ',' ,DATA.*)
C      DO 1 I=1,11
C      1 READ(7)OKK
C      WRITE(OKK)
C      RETURN
C
C      END
MAIN PROGRAM .......

1 C STRAIGHTEN
2 C UMAGAACES DATA THROUGH THE 1ST 9 TIMES
3 REAL DATA(1),12),MOUNES)
4 INTEGER INDEX, DEC/4096, 0 REAL DATA(1), 1, 2, 3, 4, 5
6 C READ AN END IN
7 CALL VAR Ai(12), DEC/4096, 0, 1, 0
8 GO TO 2
10 C STOP
12 C DRC/0 (1184, 0218)
14 FORMAT (1184, 1A)
16 WRITE(A, 1184) (DATA(I), I = 1,16), '
18 C .. TNP: (1216, 1216) 
20 C INDEX = 1, 0/1, 11, 0/4, 0/4, 0/4
22 C DATA[1], 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1
24 C COPY LOC
26 CALL COPY(1)
28 C SKIP NO DATA
30 CALL SKIP(31)
32 C COPY TO A
34 CALL COPY(11)
36 C SKIP TO
38 CALL SKIP(17)
40 C COPY TO END AND C1
42 CALL COPY(1)
44 C FOR EACH FREQUENCY
DO 3 IF F(1,4,9)<0
WRITE(6,REC(1,4,9))
3 CONTINUE

SUBROUTINE SKIP(NREC)
WRITE(A,AMINREC)
L1 FORMAT(* SKIPPING*,12,I RECORD*)
DO 2 J=1,NREC
2 CALL READ(I,DATA,J,5,5,5)
RETURN
1 WRITE(A,AU11)
L0 FORMAT(* SKIPPING AT RECORD*,12)
STOP
17

SUBROUTINE COPY(NREC)
WRITE(A,AMINREC)
L1 FORMAT(* COPYING*,12,1 RECORD*)
DO 2 J=1,NREC
2 CALL READ(I,DATA,J,5,5,5)
RETURN
1 WRITE(A,AU11)
L0 FORMAT(* COPYING AT RECORD*,12)
STOP

STOP
SUBROUTINE CORRDATA
C CORRECTS ONE COMPONENT
REAL DATA(NREC)
WRITE(A,1000)
1 FORMAT(* CORRECTED DATA, I)
C INPUT THE ARRAY
DO 1 I=1,NREC
1 CALL READIT(A,DATA(I),I), 6,7,9,8,2,4,5
C CORRECT II
CALL CORREC(DATA)
C OUTPUT IT
CALL OUTPUT(DATS)
RETURN
STOP
C END
WRITE(A,1000)
100 FORMAT(* AT RECORD N, I)
STOP
WRITE(A,1000)
100 FORMAT(* AT RECORD N, I)
STOP
SUBROUTINE CORREC(DATA)
REAL DATA(NREC,1)
DO 1 I=1,NREC
1 DATA(I,1)=DATA(I,1)**3
RETURN
STOP
STOP
SUBROUTINE OUTPUT(DATS)
REAL DATA(NREC,1)
DO 1 I=1,NREC
1 DATA(I,1)=DATA(I,1)**3
RETURN
STOP
STOP
C MERGER
C MERGE RANGE ARRAYS FROM RANGER WITH DATA FROMズ栽CHER
C INPUT FROM ARROW-RANGES, SKUH81
C OUTPUT TO SPOOL
C REAL RANGES(X,32), DATA(32,9), LABEL(40)
C INTEGER NATK(1,1,1,2,4,4,13), POS(4) /1,7,3,5,9,17/
C READ(2) RANGES
C CALL WINDER(3)
C CALL WINDER(4)

C SKIP TO LABEL
C CALL PRD-AIT(3,1,4,6,1,2,9,5,10)
C GO TO 8
C WRITE(A,ANY)
C END FORMAT(* At LABEL1)
C CONTINUE
C SETUP NEW LABEL
C ENCONE (WIND, LABEL1)
C END FORMAT(* At WIND)
C * ERP SITE TO 4.2 KM, AND DIGITISE, FOR THOM RENOWN
C * UV MAY BEST RANGES, THINS, * 30/1
C C OUTPUT OF LABEL
C CALL WR-AIT(4,1,2,3,4,5,6)
C C COPY MORE, TEMP, IYOF, C9J
C DO 1 I=1,14
C CALL RD-AIT(3,DATA,32A,511,511,511,511)
C 1 CALL WR-AIT(4,DATA,32A,511,511,511)
C C FOR EACH FREQUENCY
C DO 2 IFREQ=1,6
C IFREQ=S/IFREQ)
C IFREQ=S/IFREQ)
C C COPY RANGE
C CALL OUT(RANGES(1,14),14)
C C FOR EACH COMPONENT
C DO 3 TEMP=1,6
C C COPY DATA
C DO 4 I=1,14
C CALL RD-AIT(3,DATA,32A,511,511,511)
C 3 CALL WR-AIT(4,DATA,32A,511,511)
C 4 CONTINUE
C CALL END(4)
C STOP
51     WRITE(6,1011)
52     A00 FORMAT(* AT RECORD*,14)
53          STOP 10
54   10   STOP 19

55     WRITE(6,1011) REND, ICOMP, I
56     A01 FORMAT(* AT REND*,14/* AT COMPONENT*,14/
57          * AT RECORD*,14)
58          STOP 11
59
60     SUBROUTINE OUT(2,1)
61     REAL P(2,8)
62
63     DO 1, I = 1, 1
64       CALL WRITE(4,21,1), P(I,9), 291
65     RETURN
66  1   STOP
67
68     END
BCDTAPES

CONVERTS FILES WITH 386-WORD RECORDS TO BCD

DIMENSION INPUT(386,2), OUTPUT(386,2), TITLE(386)

SET OUTPUT TAPE PARITY
CALL PWPAR(8,0)

READ AND COPY TITLE
READ(2,200) TITLE
200 FORMAT(13A6,A2)
CALL WRWAIT(S,TITLE,386,$9,$9)

READ AND COPY MODE
CALL RDWAIT(7,INPUT,386,$9,$9,$9)
ENCODE(100,OUTPUT) (INPUT(I,1), I=1, 386)
100 FORMAT(236)
CALL WRWAIT(S,OUTPUT,386,$9,$9)

READ, CONVERT, AND COPY EVERYTHING ELSE
IBUF=1
CALL PREAD(7,INPUT,386,$9,$9,$9)
DO 1 IREC=1,216
IBUF=3-IBUF
CALL PREAD(7,INPUT(1,IBUF),386,$10,$10,$9)
CALL CONVRT(INPUT(1,3-IBUF),OUTPUT(1,3-IBUF))
CALL PWRITE(8,OUTPUT(1,3-IBUF),386,$9,$9)
1 CONTINUE
CALL RDWAIT(8,$9,$9)
CALL FILEND(8)
STOP

WRITE(6,600)
600 FORMAT(' UNEXPECTED EOF REACHED ON UNIT 7')
9 STOP

SUBROUTINE FOR REAL CONVERSIONS
SUBROUTINE CONVRT(INPUT, OUTPUT)
REAL INPUT(386), OUTPUT(386)
ENCODE(100, OUTPUT) INPUT
100 FORMAT(236)
RETURN

END
C        TAPECOPY
C        COPIES FINAL DATA TAPES
REAL DATA(386,2)
CALL PREAD(9,9)
CALL TRANS(7,218)
CALL TRANS(8,189)
STOP

SUBROUTINE TRANS(IUNIT,NREC)
CALL PREAD(IUNIT,DATA,386,$9,$9,$9)
IBUF=1
DO 1 IREC=1,NREC
CALL PREAD(IUNIT,DATA(I,3-IBUF),386,$9,$9,$9)
CALL WRWAIT(9(DATA(1,IBUF),386,$9,$9)
1 IBUF=3-IBUF
CALL FILEND(9)
CALL PRWAIT(IUNIT,$2,$2,$2)
2 RETURN
9 STOP
END

$3
23
Apollo 17 SEP

Data Processing

John C. Rylaarsdam

July 1974
# Contents

## Introduction

2

## The Stack

10

### Documentation Routines

LUNALST, LUNALST2, LUNALST3

15

### Editing Routines

LUNACOPY, LUNACPY2, LUNACPY3

17

LUNACPY4

17

LUNACPY5

18

### Plotting Routines

LUNAPLOT, LUNAPLT2

20

LUNAPLT3

20

LUNAPLT4

21

LUNAPLT5

21

ANTENNA0

22

### Statistical Routines

CALSTAT

24

TXOSTAT

25

VLBIRT

26

### Auxiliary Routines

LUNIN, LUNIN2, LUNIN3

28

SEPLOT

29

INTPOL

30

FILTER

30

PLINIT

31

### File Formats

SCI1, SCI2, SCI3

32

SCI1A, SCI2A, SCI3A

36

STAT1

37

EP4

38

NAV1

39
<table>
<thead>
<tr>
<th>Program Listings</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>ANTENNAO</td>
<td>40</td>
</tr>
<tr>
<td>ANTPAT</td>
<td>41</td>
</tr>
<tr>
<td>BUBBLE</td>
<td>43</td>
</tr>
<tr>
<td>CALSTAT</td>
<td>46</td>
</tr>
<tr>
<td>FILTER</td>
<td>47</td>
</tr>
<tr>
<td>GAPLOT</td>
<td>49</td>
</tr>
<tr>
<td>INTPOL</td>
<td>50</td>
</tr>
<tr>
<td>LUNACOPY</td>
<td>52</td>
</tr>
<tr>
<td>LUNACY2</td>
<td>53</td>
</tr>
<tr>
<td>LUNACY3</td>
<td>57</td>
</tr>
<tr>
<td>LUNACY4</td>
<td>60</td>
</tr>
<tr>
<td>LUNACY5</td>
<td>64</td>
</tr>
<tr>
<td>LUNALIST</td>
<td>68</td>
</tr>
<tr>
<td>LUNALST2</td>
<td>72</td>
</tr>
<tr>
<td>LUNALST3</td>
<td>74</td>
</tr>
<tr>
<td>LUNAPLOT</td>
<td>77</td>
</tr>
<tr>
<td>LUNAPLT2</td>
<td>80</td>
</tr>
<tr>
<td>LUNAPLT3</td>
<td>83</td>
</tr>
<tr>
<td>LUNAPLT4</td>
<td>87</td>
</tr>
<tr>
<td>LUNAPLT5</td>
<td>91</td>
</tr>
<tr>
<td>LUNIN</td>
<td>97</td>
</tr>
<tr>
<td>LUNIN2</td>
<td>104</td>
</tr>
<tr>
<td>LUNIN3</td>
<td>106</td>
</tr>
<tr>
<td>ODCINT</td>
<td>109</td>
</tr>
<tr>
<td>PLINIT</td>
<td>111</td>
</tr>
<tr>
<td>RTPLLOT</td>
<td>113</td>
</tr>
<tr>
<td>SEPLOT</td>
<td>113</td>
</tr>
<tr>
<td>STOPT</td>
<td>115</td>
</tr>
<tr>
<td>TXOSTAT</td>
<td>120</td>
</tr>
<tr>
<td>TXPLOT</td>
<td>121</td>
</tr>
<tr>
<td>VLBIRT</td>
<td>125</td>
</tr>
</tbody>
</table>

| SCI2B Plots   | 130 |
Figures

1 Processing Flow 5
2 Examples of Stack Use 11
3 Algorithm for Assembly of Arrays of Range and dB Information 14

Tables

1 Data Set Summary 3
2 Incorrectly Labelled Blocks 16
3 Receiver Timing Sequence 19
4 Calibration Data Obtained From Earth-based Tests 25
5 Format of SCI1 Label Record 32
6 Interpretation of Mode Data 33
7 Record Contents on Files SCI1, SCI2, and SCI3 34
8 dB Data on Files SCI1A, SCI2A, and SCI3A 36
9(a) Contents of File STAT1 37
9(b) Function of the Index K for Transmitter-off and Calibration Arrays on File STAT1 38
10 Record Format for File EP4 39
Introduction

This report is a summary of intermediate stage processing operations performed on the data from the Apollo 17 surface electrical properties experiment. The starting points for these operations are the files designated SCI1, SCI2, and SCI3; the contents of these and all generated files, plots, and listings are summarized in table 1. File SCI1 is a preliminary release of the data; files SCI2 and SCI3 are the final data sets, respectively with and without data for the EP-4 turn, prepared by R. Watts, tape number SEPD09. (N.B. - subsequent references to removal of turn data refer to work described in this report, rather than to Watts' initial processing.)

The diagrams in figure 1 give an indication of the sequences of processing operations. More detailed information is provided by the descriptions of the programs. Annotated listings are also included, as well as precise tabulations of the formats of the various files. The program listings include descriptions of all required card input data.

In addition, two complete sets of plots (SCI2P) are included as a record of the data; in one set dB values are plotted versus the range in metres, and in the second set versus the range in wavelengths.
Table 1 - Data set summary.

**Apollo 17 SEP - 3**

<table>
<thead>
<tr>
<th>LMTTCD</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>A0EXAAB</td>
<td></td>
</tr>
<tr>
<td>BDM-LN</td>
<td></td>
</tr>
<tr>
<td>EPOG</td>
<td></td>
</tr>
<tr>
<td>LFE</td>
<td></td>
</tr>
</tbody>
</table>

**CAL1 Listing**

**EP4**

* dB data for 490 m.

**EP4 Listing**

* <= range <= 535 m.

**EP4 Plot**

**EP4A Listing**

* dB data for 490 m.

**EP4A Plot**

* <= range <= 535 m.

**NAV1**

times and odometer counts relative to beginning of traverse

**RT1 Listing**

* includes VLBI data, converted to ranges

**RT1 Plot**

**SCI1**

* * * * * *

**SCI1 Listing**

* * * * * *

**SCI1A**

* dB data sampled at intervals of 0.1 wavelength

**SCI1A Listing**

**SCI1A Plot**

* *
### Apollo 17 SEP - 4

<table>
<thead>
<tr>
<th>L M T T C R D</th>
<th>B D M - L N</th>
<th>E E P O G</th>
<th>Notes</th>
</tr>
</thead>
</table>

#### SCI2
- ** * * * * * * * * |
- no data for EP-4

#### SCI2 Listing
- ** * * * * * * * |

#### SCI2 Plot
- ** * |

#### SCI2A
- * * |
- no data for EP-4;
- dB data sampled at intervals of 0.1 wavelength

#### SCI2A Listing
- * * |

#### SCI2A Plot
- * |

#### SCI2B Plot
- * * *

#### SCI3
- ** * * * * *

#### SCI3 Listing
- ** * * * * * |
- range data from SCI2

#### SCI3A
- * * |
- dB data sampled at intervals of 0.1 wavelength

#### SCI3A Listing
- * *

#### SCI3A Plot
- *

#### STAT1
- ** * * *
- also contains speed data

#### TX01 Listing
- * * |
- all data except 490 m. <= range <= 535 m.

#### TX01 Plot
- * * |
- and L&V in motion

---

Table 1 - Data set summary (continued).
Figure 1(a) - Processing flow.
Figure 1(b) - Processing flow.
Figure 1(c) - Processing flow.
Figure 1(d) - Processing flow.
Figure 1(e) - Processing flow.
The Stack

A number of routines use a large array (named DATA, hereafter referred to by name, or as "the stack") and a set of indices to the array, as the basis of a system for manipulating range and dB data.

In most cases a particular set of range or dB information is contained in several blocks, which it is generally convenient to combine into one large block before processing. To accomplish this, three indices are associated with the stack as diagrammed in figure 2: IXX and IXY indicate the first words of range and dB data respectively; IORG gives the location of the first word into which data should be read to make an extension to the block currently being assembled. Other parameters relating to the stack may be defined where necessary.

The basic procedure to be used to process a complete file is outlined in figure 3. In this description "read a block" is taken to mean that the contents of one block from the input file are placed in locations IORG through IORG + N - 1 of the stack; the meaning of "last" is that given to it by the LUNIN routines.
Figure 2(a) - Example of stack use: assembling a range array.
Figure 2(b) - Example of stack use: completing a dB array.
Figure 2(c) - Example of stack use: beginning the dB array for a new component.
do for frequency = 1, 2.1, 4, 8.1, 16, 32.1
  iorg = 1;  ixx = 1;  m = 0;

ASSEMBLE THE RANGE ARRAY:
  repeat
    read a block;
    iorg = iorg + n;
    m = m + 1;
  until last;

  ixy = iorg;
  repeat
    1 = 0;  iorg = ixy;

ASSEMBLE THE DB ARRAY FOR ONE COMPONENT:
  repeat
    read a block;
    iorg = iorg + n;
    1 = 1 + 1;
  until 1 = m;

  perform required processing;
  until last;
end:

Figure 3 - Algorithm for assembly of arrays of range and dB information.
Apollo 17 SEP - 15

Documentation Routines

LUNALIST, LUNALST2, LUNALST3

These routines produce complete listings of the data on files SCI1, SCI2, and SCI3 respectively.

The data are read from SCI1 and SCI2 by LUNIN and LUNIN2 respectively, and printed in blocks corresponding to the physical records on the files, fifteen values per line. Each block is preceded by a heading, containing the character information returned by the input routine, identifying the contents of the block.

The procedure for listing SCI3 is more complex, since the file contains no range data. LUNALST3 invokes LUNIN2 to read a record from SCI2, and inspects the returned value of ITYPE(2). If this value is five, indicating a block of range data, the block is listed. Otherwise, LUNIN3 is called to read a record from SCI3, which replaces the data from SCI2, and the new block is listed.

N.B. A bug, in all three routines, results in the identification of transmitter-off, calibration, and one megahertz range and dB data being printed incorrectly, as indicated in table 2. (The numbers in parentheses indicate how many records are affected.)
### Table 2 - Incorrectly labelled blocks.

<table>
<thead>
<tr>
<th>contents of block</th>
<th>label printed</th>
</tr>
</thead>
<tbody>
<tr>
<td>temperature(1)</td>
<td>temperature</td>
</tr>
<tr>
<td>transmitter-off(1)</td>
<td>NONE</td>
</tr>
<tr>
<td>transmitter-off(5)</td>
<td>transmitter-off</td>
</tr>
<tr>
<td>calibration(1)</td>
<td>NONE</td>
</tr>
<tr>
<td>calibration(5)</td>
<td>calibration</td>
</tr>
<tr>
<td>1 MHz. range(1)</td>
<td>NONE</td>
</tr>
<tr>
<td>1 MHz. dB(6)</td>
<td>NONE</td>
</tr>
</tbody>
</table>

Apollo 17 SEP - 16
Editing Routines

LUNACOPY, LUNACPY2, LUNACPY3

These routines read files SCI1, SCI2, and SCI3 respectively, and produce the binary files SCI1A, SCI2A, and SCI3A, containing the label records from the input files, and six blocks of NPTS dB values each, for each frequency, after interpolation at intervals of 0.1 wavelength. NPTS varies with frequency, and is either the maximum number of interpolated values which could be generated, or 1000, whichever is smaller.

The stack mechanism is used to set up the arrays of range and dB values to be given to the interpolation routine, and the array RANGE is initialized with the appropriate equivalent in metres of 0, 0.1, 0.2, ..., 99.9 wavelengths. Subroutine INTPOL is called to do the interpolation, and returns its results in array VCO.

After interpolation the values of NSTART and NPLT (set by INTPOL) indicate the first and last elements of VCO which contain interpolated results. Before writing the array on the output file, the programs set VCO(1) through VCO(NSTART-1) and VCO(NPLT+1) through VCO(1000) to zero.

LUNACPY4

This program produces a binary file (EP4) of range and dB values for the turn at EP-4, using file SCI3 as input. The program is a simple modification of LUNAPLT4, in which the call to subroutine GAPLOT is replaced by a write statement which generates a record on file EP4, and a second statement which writes the contents of the record on the printer.
This program is used to generate file STAT1, containing temperature, calibration, and transmitter-off data, and arrays of the ranges at which the data were obtained. Also, a set of crude LRV speed values, corresponding in time to the transmitter-off data, is computed and written on the output file.

The array of temperature values is simply copied from the input file (SCI2). The associated range data is the set of values for one megahertz, which is also copied directly onto the output file. (The one megahertz data were used since their occurrences are the closest in time to the temperature data - c.f. table 3.)

Transmitter-off data are read from the input file in two groups of three blocks each. For each group, the first, second, and third blocks contain data from the x, y, and z receiving antennae respectively. The frequency for each sample is dependent on the group, and on the tens digit of the corresponding element of the mode array: the first group contains data for frequencies of 32.1, 8.1, and 2.1 megahertz, and the second group for 16.0, 4.0, and 1.0 megahertz, corresponding respectively to tens digits of 1, 2, and 3. The contents of the blocks of calibration data are arranged similarly, with blocks one and four containing values for front-end noise, two and five containing values for the noise diode, and values for the noise diode plus 20 dB amplification in blocks three and six.

Using the number of the block on which it is working, and the appropriate contents of the mode array, the program generates arrays of values which may be indexed by frequency, and either antenna in the case of the transmitter-off data, or noise source in the case of the calibration data.
Approximate range values corresponding to the above data are obtained by selecting the values closest to them in the timing sequence from the 32.1 megahertz range array, and again using the contents of the mode array. Corresponding to each range value, an approximation of the LRV speed is computed by taking the difference of the immediately succeeding and preceding elements of the 32.1 megahertz range array.

Table 3 - Receiver timing sequence.
Each record is 202.5 ms. duration.
### Plotting Routines

LUNAPLOT, LUNAPLT2

These routines produce (CALCOMP) plots of dB versus distance, using interpolated data. LUNAPLOT is used for plotting the data in SCI1A; LUNAPLT2 may be used to plot data from either SCI2A or SCI3A.

One namelist (FREQ) control record is read for each frequency. The parameters which may be specified allow choice of components, maximum and minimum wavelengths, and maximum dB value to be plotted. Filtering of the data before plotting may be requested, a dB level may be specified for plotting a reference mark, and optional plot annotation may be supplied.

Subroutine PLINIT is called to initialize the plotting software. The actual plots are produced using the DATPLT entry point of subroutine SEPLOT.

LUNAPLT3

This program is used to generate (CALCOMP) plots of the data on SCI2, similar to the plots produced by LUNAPLOT and LUNAPLT2. The program differs from the others in that the data are not interpolated, and cannot be filtered before plotting; also, the portion of the data corresponding to the turn at EP-4 is deleted before plotting.

The program uses the stack mechanism to prepare data for the plotting routine. Removal of the data for the turn is accomplished as follows. After an entire array of range values has been assembled, the program locates the values to be deleted; then succeeding values are copied downward to maintain a contiguous set, and IORG is reset to indicate what is
then the first free location. The indices defining the gap are modified to apply to the dB segment of the stack, and each time a complete set of dB data is assembled, an equivalent compacting operation is performed.

Parameters to control the plotting operation are supplied on a set of namelist (CNTL) control records. The plotting is done using the DATPLT entry point of subroutine SEPLDT.

LUNAPLT4

This program produces (CALCOMP or GOULD) plots of dB values for the FP-4 turn versus an implicit time scale, using data from file SCI3. The data are plotted as discrete points (marked by symbols) at equal horizontal intervals. Each component is plotted on a separate graph.

Each set of values to be plotted is assembled in the main program and passed to subroutine GAPLOT, which produces the plot. The method used to define the desired values is basically the complement of the method used in the two preceding programs to remove the same data. However, in this case the required segments of the arrays are merely located; no compacting operation is performed.

LUNAPLT5

This routine is a modification of LUNAPLT3 which allows the distances to be expressed in either metres or wavelengths. In addition, transmitter-off values from file STAT1 are plotted (as points, rather than continuous curves) as a baseline for each dB curve, using entry point BASEL of subroutine SEPLDT.
ANTENNAO

This program is used to generate plots of the patterns of the three receiving antennae, based on the data for the turn at EP-4 contained in file EP4. Two methods are available for computing the angle between the LRV and the SEP transmitter: the plotted points may be at equal angular increments throughout the whole range, or the navigation data may be used to compute an approximate angular displacement for each point.

One namelist (CNTL) control record is required for each frequency. Parameters in each record allow choice of components to be plotted, initial angle and the difference between final and initial angles, and indices defining the data points obtained while the LRV was in motion; a Boolean value (NAVDAT) may be included to indicate whether or not navigation data are to be used in computing angles, and if this value is "true", a time must be supplied corresponding to the first data point in the set for which the LRV was moving.

The main program organizes the control information, and then enters a loop, in each cycle of which it reads one record from file EP4, and if the data in that record are to be plotted, passes the data and required control information to subroutine ANT Patton.

Subroutine ANT Patton begins by drawing a set of x and y axes and plotting a label indicating frequency and component. The total angular range is divided into equal intervals, based on the number of points to be plotted. If navigation data are to be used in computing angular displacements, the number of odometer counts at the beginning and end of the range are obtained by invocations of function ODCINT, and their difference (ODCRAN) is computed. The first dB value and the initial angle are converted to rectangular coordinates and the point is plotted.
loop is then entered, which continues until the last data point has been plotted: the angle is decremented (to give clockwise rotation) either by the constant amount, or by using the result of an invocation of ODCINT to determine a fraction of the total angle; rectangular coordinates are computed, and the point is plotted; the index of the next value is determined, using the supplied parameters.

Function ODCINT is invoked with one argument, a time (T) in seconds. On the first entry a set of times and corresponding left- and right-wheel odometer counts are read from the card reader. For each triple the time and the average of the counts are saved. The value of the function is an odometer count obtained by linear interpolation, using T and the arrays of times and corresponding average counts.
Statistical Routines

CALSTAT

This program is used to compute various statistics for each set of calibration data on file STAT1: the means and standard deviations of the front-end noise; differences between the experimental noise diode values (with and without amplification) and values for the same data obtained from earth-based testing; the means and standard deviations of these differences.

The computations for front-end noise data are straightforward, and should require no explanation. Most of the variable names begin with "EG". The results of the calculations are written on FORTRAN logical eight, which is used as an auxiliary printer.

The calculations for the noise-diode data (with and without amplification - "with" is indicated by "PA" as part of the name of each variable involved) involve computation of an earth-based value, using linear interpolation according to temperature, of the v.c.o. frequencies given in table 4. The difference between the experimental and interpolated values is computed; the remainder of the calculation consists of the accumulation and scaling of the appropriate sums.
transmitter frequency noise diode noise diode + amplifier
66°F 112°F 66°F 112°F
1. 525.6 508.6 1167.7 1157.7
2.1 564.2 546.2 1208.6 1208.6
4. 593.3 566.3 1230.1 1230.1
8.1 694.8 682.8 1298.6 1304.6
16. 756.1 770.1 1293.7 1306.7
32.1 833.9 888.9 1199.6 1198.6

Table 4 - Calibration data obtained from earth-based tests.

TXOSTAT

This routine and its associated subroutines compute mean values and standard deviations for each set of transmitter-off data on file STAT1. Separate statistics are calculated for periods when the rover was stopped and in motion; in the latter case, values for the EP-4 turn are excluded from the computations. Each set of values is displayed twice: once in order of increasing distance from the transmitter, and once in order of increasing LRV speed. The dB values are also plotted versus LRV speed (if plots are not required, subroutine TXPLOT is simply replaced by a dummy routine).

A set of bounds, the same as the 32.1 megahertz bounds input, but adjusted relative to the beginning of the 32.1 megahertz range data rather than the beginning of the turn data, is required as input. These values are used by function STOPT in deciding whether the rover was moving or stopped for each point within the EP-4 turn.

The statistical computations, which are performed in the main program, are relatively straightforward
and should not require explanation. Data for ranges greater than 1667 metres are omitted from all calculations. A possibly confusing action is the assignment of -1 to certain elements of the speed array; the elements are those within the EP-4 turn for which the LRV was in motion. The speed values for these elements are computed (meaninglessly) as zero by LUNACPY5; the value of -1 indicates to the plotting routine that each such datum is to be ignored.

Ordering of the data according to increasing speed is accomplished by subroutine BUBBLE, which performs a bubble sort. Rather than interchanging elements within four parallel arrays (one of speeds and three of dB values) the routine uses an integer array (IX) of equivalent size, supplied by the calling program; IX(I) is initialized to I, and the contents of IX are used as indirect addresses to the actual data arrays, and it is these indices which are interchanged. When the sort is complete, the contents of IX indicate the order in which the other arrays should be indexed to obtain the data in order of increasing speeds.

The dB data are plotted versus speed by subroutine TXPLOT, which is entered once for each frequency. For each entry, three sets of labelled axes are plotted (one for each receiving antenna) within a 8.5 by 11 inch area. The appropriate data points are then simply plotted on each set of axes.

VLBIRT

This program is used to compare results from the VLBI experiment with SEP navigation data; the comparison is done on the basis of distance from the SEP transmitter.

The 16 megahertz range data are read from file SCI2, and an array of corresponding times is generated, using a starting value which is read as a
control parameter. A parameter may also be supplied specifying a value to be added to each range value.

The VLBI times are converted from hours, minutes, and seconds to seconds, and each pair of x and y coordinates is converted to a distance. An interpolated SEP range value is computed for each VLBI datum, using the time arrays; the difference between VLBI and interpolated SEP ranges is calculated, and summations are taken of the differences and their squares, which yield the mean difference and standard deviation.

If PLOT is set to true on the namelist control record, subroutine RTPLOT is called. The subroutine plots a set of time and range axes, using the supplied parameter SCALE. The SEP and VLBI ranges are then plotted versus time in two simple loops.
Auxillary Routines

LUNIN, LUNIN2, LUNIN3

These subroutines are used to read data from files SCI1, SCI2, and SCI3 respectively. Floating-point data are returned to the invoking routine in the array DATA (not to be confused with the stack, although most of the programs which invoke these routines use a portion of the stack for passing data); fixed-point data are returned in array IDATA. The label record in SCI1 contains various fields, the contents of which are passed to the invoking routine through COMMON block LUNDAT. The label records of the other files consist solely of text, which is returned in IDATA.

The values in the fixed-point array TIDX are used to identify each record read from the input file. The values are arranged in seventeen groups of three: the first value in each group indicates the number of records of that type which are on the file; the second and third values are used to select alphanumeric identification from arrays TYPE1 and TYPE2 respectively. The alphanumeric identification is returned in TYPE, and the second and third values from TIDX are returned in ITYPE (both TYPE and ITYPE are in LUNDAT).

LUNIN obtains the value of N (the number of values in each record other than the label record) from the label record; the other two routines expect N to have been set by the invoking routine.

The logical variables FIRST and LAST are set to true if the record read is the first or last of its type respectively (e.g. - the first of the twenty-four eight megahertz dB records).
SF.PLOT (DATPLT, BASEL)

This routine (written by J. J. Proctor, 1973) has been modified in a number of ways:

1. dB values may be plotted versus either wavelengths or metres; the decision is made by examining XSCALE: if it is less than ten it is assumed to be the number of inches per twenty-wavelength segment, while a value of ten or greater is assumed to indicate the number of inches per kilometre.

2. Entry point BASEL has been added in order to allow a set of points to be plotted in conjunction with each curve, using the same (relative) plotter origin. This is for the purpose of indicating a set of background values for each curve.

3. Low dB values are no longer set to zero. Furthermore, the y-origin for each curve is now equivalent to (relative) zero dB, rather than the integral minimum dB value. This was necessary in order to keep the plot of background values on the page.

4. All the labels for individual curves, except the component identification, have been eliminated.

5. A reference mark for each curve is plotted on the y-axis, at a dB level set by the invoking routine.

N.B. The entry point (THEPLT) and associated code for plotting theoretical curves have not been changed. However, since the program was modified, no attempt has been made to verify the integrity of this feature.

Most of the code for the subroutine is concerned with setting up the axes and labels, and with placing
a particular curve on the graph. The range axis markings depend on the value of XSCALE, as described above, and extend from zero to the first multiple of the chosen increment greater than the highest range value in the data. A displacement is computed such that curves on the graph will be equally spaced vertically. All these operations are performed on entry at DATPLT, the entry point for plotting data curves, if a new graph (not just a new curve) is to be plotted.

The curves themselves are plotted in a straightforward manner, and a label is plotted at the right-hand end of each, to provide component identification. After a component has been plotted, the parameters defining the position of the curve are left unchanged until the next entry at DATPLT; therefore an entry at BASEL will result in the baseline points being plotted on the same set of relative axes as the associated data curve. (If BASEL is invoked at any other time, it will not function properly.)

INTPOL

This is a general linear interpolation routine. It accepts "input" arrays XIN and YIN, and an array XOUT, of values on the same scale as XIN, for which interpolated values for YIN are required. The results are placed in array YOUT, and parameters NSTART and NPOPT are set to indicate which values in YOUT are the result of successful interpolation, and which are undefined due to the corresponding elements of XOUT being out of the range of XIN.

FILTER

This is a subroutine which accepts array A, and applies to its contents the filter whose coefficients are contained in array F. Array B is used for
accumulation of sums, and its contents are copied into array A before return to the calling program.

PLINIT

This subroutine is used to systematize the initialization of the University of Toronto CALCOMP software package. Presumably it will be of interest only to users of that installation.
File Formats

SCI1, SCI2, SCI3

Each of these files begins with a label record. The format of this record on file SCI1 is given in table 5. The label records for the other two files are 2316 characters each, consisting of 27 segments of 84 characters each, followed by 48 characters of padding. Each 84-character segment contains one card image and four padding characters.

The next record in each file is the mode array, in format 386I6. Each element is a three (decimal) digit number, MAR; the significance of the values of the digits is indicated in table 6. (The notations "f1" and "f2" in the table refer to transmitter-off and calibration data descriptions given in table 7.)

The format of all remaining records is 386F6.1; the contents of these records are given in table 7. (N.B. - file SCI3 contains no range data.)

<table>
<thead>
<tr>
<th>Format</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>A6</td>
<td>run identification</td>
</tr>
<tr>
<td>A6</td>
<td>site identification</td>
</tr>
<tr>
<td>A6</td>
<td>traverse direction</td>
</tr>
<tr>
<td>A6</td>
<td>forward/reverse traverse</td>
</tr>
<tr>
<td>14A6</td>
<td>title</td>
</tr>
<tr>
<td>T6</td>
<td>number of values in each succeeding block</td>
</tr>
</tbody>
</table>

Table 5 - Format of SCI1 label record.
### Table 6 - Interpretation of mode data.

<table>
<thead>
<tr>
<th>Digit</th>
<th>Value</th>
<th>Significance</th>
</tr>
</thead>
<tbody>
<tr>
<td>M</td>
<td>1</td>
<td>receiver in data acquisition mode</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>receiver in synchronization</td>
</tr>
<tr>
<td></td>
<td></td>
<td>acquisition mode</td>
</tr>
<tr>
<td>A</td>
<td>1</td>
<td>f1 = 32.1 MHz.; f2 = 16 MHz.</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>f1 = 8.1 MHz.; f2 = 4 MHz.</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>f1 = 2.1 MHz.; f2 = 1 MHz.</td>
</tr>
<tr>
<td>R</td>
<td>0</td>
<td>synchronization not received</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>synchronization received</td>
</tr>
</tbody>
</table>

*Note: The table shows the interpretation of mode data for the Apollo 17 mission.*
<table>
<thead>
<tr>
<th>Number</th>
<th>Rec.</th>
<th>Tx.</th>
<th>Freq.</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td>label</td>
</tr>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td>mode</td>
</tr>
<tr>
<td>1</td>
<td>x</td>
<td>f1</td>
<td></td>
<td>temperature</td>
</tr>
<tr>
<td>1</td>
<td>y</td>
<td>f1</td>
<td></td>
<td>transmitter-off</td>
</tr>
<tr>
<td>1</td>
<td>z</td>
<td>f1</td>
<td></td>
<td>transmitter-off</td>
</tr>
<tr>
<td>1</td>
<td>x</td>
<td>f2</td>
<td></td>
<td>transmitter-off</td>
</tr>
<tr>
<td>1</td>
<td>y</td>
<td>f2</td>
<td></td>
<td>transmitter-off</td>
</tr>
<tr>
<td>1</td>
<td>z</td>
<td>f2</td>
<td></td>
<td>transmitter-off</td>
</tr>
<tr>
<td>1</td>
<td>f1</td>
<td></td>
<td></td>
<td>calibration - grounded input</td>
</tr>
<tr>
<td>1</td>
<td>f1</td>
<td></td>
<td></td>
<td>calibration - amplified noise</td>
</tr>
<tr>
<td>1</td>
<td>f2</td>
<td></td>
<td></td>
<td>calibration - noise</td>
</tr>
<tr>
<td>1</td>
<td>f2</td>
<td></td>
<td></td>
<td>calibration - grounded input</td>
</tr>
<tr>
<td>1</td>
<td>f2</td>
<td></td>
<td></td>
<td>calibration - amplified noise</td>
</tr>
<tr>
<td>1</td>
<td>f2</td>
<td></td>
<td></td>
<td>calibration - noise</td>
</tr>
<tr>
<td>1</td>
<td>x</td>
<td>EW</td>
<td>1</td>
<td>dB</td>
</tr>
<tr>
<td>1</td>
<td>y</td>
<td>EW</td>
<td>1</td>
<td>dB</td>
</tr>
<tr>
<td>1</td>
<td>z</td>
<td>EW</td>
<td>1</td>
<td>dB</td>
</tr>
<tr>
<td>1</td>
<td>x</td>
<td>NS</td>
<td>1</td>
<td>dB</td>
</tr>
<tr>
<td>1</td>
<td>y</td>
<td>NS</td>
<td>1</td>
<td>dB</td>
</tr>
<tr>
<td>1</td>
<td>z</td>
<td>NS</td>
<td>1</td>
<td>dB</td>
</tr>
<tr>
<td>1</td>
<td>x</td>
<td></td>
<td>2.1</td>
<td>range *</td>
</tr>
<tr>
<td>1</td>
<td>y</td>
<td></td>
<td>2.1</td>
<td>dB</td>
</tr>
<tr>
<td>1</td>
<td>z</td>
<td></td>
<td>2.1</td>
<td>dB</td>
</tr>
<tr>
<td>1</td>
<td>x</td>
<td></td>
<td>2.1</td>
<td>dB</td>
</tr>
<tr>
<td>1</td>
<td>y</td>
<td></td>
<td>2.1</td>
<td>dB</td>
</tr>
<tr>
<td>1</td>
<td>z</td>
<td></td>
<td>2.1</td>
<td>dB</td>
</tr>
</tbody>
</table>

Table 7 - Record contents on files SCI1, SCI2, and SCI3.  
* not present on SCI3
<table>
<thead>
<tr>
<th>Number</th>
<th>Rec.</th>
<th>Tx.</th>
<th>Freq.</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>4</td>
<td>EW</td>
<td>4</td>
<td>range *</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>EW</td>
<td>4</td>
<td>dB</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>EW</td>
<td>4</td>
<td>dB</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>NS</td>
<td>4</td>
<td>dB</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>NS</td>
<td>4</td>
<td>dB</td>
</tr>
<tr>
<td>4</td>
<td>8.1</td>
<td>EW</td>
<td>8.1</td>
<td>dB</td>
</tr>
<tr>
<td>4</td>
<td>8.1</td>
<td>EW</td>
<td>8.1</td>
<td>dB</td>
</tr>
<tr>
<td>4</td>
<td>8.1</td>
<td>NS</td>
<td>8.1</td>
<td>dB</td>
</tr>
<tr>
<td>4</td>
<td>8.1</td>
<td>NS</td>
<td>8.1</td>
<td>dB</td>
</tr>
<tr>
<td>8</td>
<td>16</td>
<td>EW</td>
<td>16</td>
<td>dB</td>
</tr>
<tr>
<td>8</td>
<td>16</td>
<td>EW</td>
<td>16</td>
<td>dB</td>
</tr>
<tr>
<td>8</td>
<td>16</td>
<td>EW</td>
<td>16</td>
<td>dB</td>
</tr>
<tr>
<td>8</td>
<td>16</td>
<td>NS</td>
<td>16</td>
<td>dB</td>
</tr>
<tr>
<td>8</td>
<td>16</td>
<td>NS</td>
<td>16</td>
<td>dB</td>
</tr>
<tr>
<td>13</td>
<td>32.1</td>
<td>EW</td>
<td>32.1</td>
<td>range *</td>
</tr>
<tr>
<td>13</td>
<td>32.1</td>
<td>EW</td>
<td>32.1</td>
<td>dB</td>
</tr>
<tr>
<td>13</td>
<td>32.1</td>
<td>EW</td>
<td>32.1</td>
<td>dB</td>
</tr>
<tr>
<td>13</td>
<td>32.1</td>
<td>NS</td>
<td>32.1</td>
<td>dB</td>
</tr>
<tr>
<td>13</td>
<td>32.1</td>
<td>NS</td>
<td>32.1</td>
<td>dB</td>
</tr>
<tr>
<td>13</td>
<td>32.1</td>
<td>NS</td>
<td>32.1</td>
<td>dB</td>
</tr>
</tbody>
</table>

Table 7 - Record contents on files SCI1, SCI2, and SCI3 (continued).

* not present on SCI3
SCI1A, SCI2A, SCI3A

These files are written in binary form (i.e. without format control), and contain dB data equivalent to that in files SCI1, SCI2, and SCI3 respectively. Each file begins with the same label information as its parent file.

The remainder of each file consists of thirty-six blocks of dB data; each block is of the form given in table 8. The dB data are interpolated values, at intervals of 0.1 wavelengths, beginning at zero wavelengths; the maximum value of NPTS is one thousand.

<table>
<thead>
<tr>
<th>Position</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>eight characters: frequency</td>
</tr>
<tr>
<td>2</td>
<td>floating-point: frequency</td>
</tr>
<tr>
<td>3</td>
<td>integer: NSTART</td>
</tr>
<tr>
<td>4</td>
<td>integer: NPTS</td>
</tr>
<tr>
<td>5 through</td>
<td>floating-point: 0.0</td>
</tr>
<tr>
<td>NSTART+3</td>
<td></td>
</tr>
<tr>
<td>NSTART+4</td>
<td>floating-point: dB values</td>
</tr>
<tr>
<td>through</td>
<td></td>
</tr>
<tr>
<td>NPTS+4</td>
<td></td>
</tr>
</tbody>
</table>

Table 8 - dB data on files SCI1A, SCI2A, and SCI3A.
STAT1

The six blocks which comprise this binary file are summarized in table 9(a). The index I for the first two blocks selects temperatures and corresponding ranges at successive intervals of 6.48 seconds.

For the remaining blocks, the index I selects the various data at successive times (at varying intervals). Values of one through six for J select data for frequencies 1.0 through 32.1 megahertz respectively. The significance of K is indicated in table 9(b).

<table>
<thead>
<tr>
<th>Block</th>
<th>Contents</th>
<th>Indexing</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>TEMP(I)</td>
<td>I&lt;=386</td>
</tr>
<tr>
<td>2</td>
<td>RANGE(I)</td>
<td>I&lt;=386</td>
</tr>
<tr>
<td>3</td>
<td>CAL(I,J,K), NCAL(J,K)</td>
<td>I&lt;=NCAL(J,K) J&lt;=6 K&lt;=3</td>
</tr>
<tr>
<td>4</td>
<td>TXOFF(I,J,K), NTXOFF(J,K)</td>
<td>I&lt;=NTXOFF(J,K) J&lt;=6 K&lt;=3</td>
</tr>
<tr>
<td>5</td>
<td>RANGE2(I,J), NR(J')</td>
<td>I&lt;=NR(J') J&lt;=6 J'=1+(J-1)/2</td>
</tr>
<tr>
<td>6</td>
<td>SPEED(I,J)</td>
<td>I&lt;=NR(J') J&lt;=6</td>
</tr>
</tbody>
</table>

Table 9(a) - Contents of file STAT1.
Table 9(b) - Function of the index K for transmitter-off and calibration arrays on file STAT1.

<table>
<thead>
<tr>
<th>K</th>
<th>TXOFF</th>
<th>CAL</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>x antenna</td>
<td>grounded input</td>
</tr>
<tr>
<td>2</td>
<td>y antenna</td>
<td>noise diode</td>
</tr>
<tr>
<td></td>
<td></td>
<td>+20dB amplification</td>
</tr>
<tr>
<td>3</td>
<td>z antenna</td>
<td>noise diode</td>
</tr>
</tbody>
</table>

This file contains range and dB data from file SCI3, only for the region of the turn at EP-4 (specifically for the range values on the interval 490 to 535 metres, inclusive. There are 36 records on the file, all of the same form, but of varying length. The form of a record is summarized in table 10.
<table>
<thead>
<tr>
<th>Name</th>
<th>Words</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>F</td>
<td>1</td>
<td>frequency in megahertz</td>
</tr>
<tr>
<td>NCOMP</td>
<td>1</td>
<td>component identification:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1: rho endfire</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2: phi endfire</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3: zed endfire</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4: rho broadside</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5: phi broadside</td>
</tr>
<tr>
<td></td>
<td></td>
<td>6: zed broadside</td>
</tr>
<tr>
<td>YMIN</td>
<td>1</td>
<td>minimum and</td>
</tr>
<tr>
<td>YMAX</td>
<td>1</td>
<td>maximum dB values</td>
</tr>
<tr>
<td>N</td>
<td>1</td>
<td>number of range-dB pairs</td>
</tr>
<tr>
<td>RANGE</td>
<td>N</td>
<td>range data</td>
</tr>
<tr>
<td>DB</td>
<td>N</td>
<td>dB data</td>
</tr>
</tbody>
</table>

Table 10 - Record format for file EP4.

NAV1

This is actually a part of the card input data to ANTENNA0. Any number of cards may be included. Each card contains three values: a time in seconds, and corresponding right-front- and left-rear-wheel odometer counts, in format (3F10.).
Program Listings
C.....ANTENNAO.....PLOT DATA FROM FP-4 TURN
C
DB VALUES ARE PLOTTED ON A POLAR GRID; THE ANGULAR COORDINATES ARE
ESTIMATES OF THE ANGLE BETWEEN THE LEV AXIS AND THE LINE FROM THE
LEV TO THE SEP TRANSMITTER.
C
SEVEN INPUT RECORDS ARE REQUIRED: SIX AS DESCRIBED BELOW, AND ONE
NAMELIST (PLTID) RECORD, READ BY PLINIT. IN ADDITION, ANY
NUMBER OF CARDS CONTAINING NAVIGATION DATA (TIME, RIGHT- AND
LEFT-WHEEL ODOMETER COUNTS IN FORMAT 3F10.0) MAY FOLLOW THE 6TH
RECORD.
C
NAMELIST (CNTL):
C
IPREO - BASE TWO LOG OF FREQUENCY; NO DEFAULT
C
ICOMP - (6) CODES FOR COMPONENTS TO BE PLOTTED, PADDED WITH
ZEROS; DEFAULT SIX ZEROS. THE CODES ARE:
C
ENDIFPP BROADSIDE
C
RHO  212  211
PUH  222  221
ZFD  232  211
C
AO - ANGLE FOR THE FIRST POINT; DEFAULT 3.14159
C
ARANGE - ANGULAR DIFFERENCE BETWEEN THE FIRST AND LAST POINTS
DEFAULT 6.28318
C
BOUNDS - THREE PAIRS OF INDICES DEFINING POINTS TO BE PLOTTED
EACH PAIR DEFINES THE FIRST AND LAST OF A SEQUENCE
OF POINTS TO BE PLOTTED; NO DEFAULT
C
NAVDAT - (LOGICAL) ODOMETER COUNTS TO BE READ (FOLLOWING THE
PLTID RECORD) AND USED IN COMPUTATION OF ANGLES; DEFAULT FALSE
C
TIMEO - (REQUIRED IF NAVDAT IS TRUE) - TIME (ON SCALE OF
NAVIGATION DATA) FOR THE FIRST POINT TO BE PLOTTED; NO DEFAULT
C
REAL*4 Y(600), T(600), TIME(600)
REAL*4 TZERO(6)
REAL*4 ET(6), ET(6), ET(6), ET(6), ET(6), ET(6)
FPAI*P PROGRAM(2) / 'OOGP, ANT', 'ENNA' /
INTEGER*2 BOUND(6, 6), BOUND(6)
LOGICAL*1 DECIDE(6, 6) / 36 * 'TRUE,' / 'NAVDAT' / 'FALSE.' /
INTEGER*2 COMP(6) / 212, 222, 232, 221, 221, 231 /
INTEGER*2 ICOMP(6)

"AMPLIST / CNTL / IFREO, ICOMP, AO, ARANGE, TIME0, BOUND, NAVDAT"

SET UP CONTROL INFORMATION

AO = 3.10152
ARANGE = 6.24318
DO 100 I = 1, 6

DO 10 J = 1, 6
ICOMP(J) = 0
BOUND(J) = 0
10 CONTINUE

GET FREQUENCY INDICATOR AND COMPONENTS TO PLOT

READ(5, CNTL)
TZERO(IFREO + 1) = TIME0
DO 50 J = 1, 6

IF A COMPONENT IS NOT TO BE PLOTTED, RESET ITS MATRIX ENTRY.

IC = COMP(J)
DO 30 K = 1, 6
IF (IC .PO. ICOMP(K))
GO TO 90
.
30 CONTINUE
DECIDE(IFREO + 1, J) = 'FALSE.'
50 CONTINUE

DO 80 J = 1, 6
BOUND(J, IFREO + 1) = BOUND(J)
80 CONTINUE
100 CONTINUE

INITIALIZE THE PLOTTER

CALL PLINIT(PROGRAM)
CALL PLOT (4.25, 5.0, -3)

LOOP THROUGH FREQUENCIES

DO 140 FREO = 1, 6

SET UP THE TIME ARRAY

IF 1 = ROUND(1, FREO)
IF 6 = ROUND(6, FREO)
DO 110 I = 1, 6

TIME(I) = #FREQ (FREQ) + DT (FREQ) * (I - 1) + 1
110 CONTINUE

LOOP THROUGH COMPONENTS

DO 130 JCOMP = 1, 6

READ (I, T) FREO, NCOMP, YMIN, YMAX, N,
.X(T), I = 1, N
.Y(T), I = 1, N

IF (.NOT. DECDSE (FREQ, JCOMP))
GO TO 130

CALL ANTPAT(TIME, Y, N, FREO, JCOMP, ROUND (1, FREO),
AO, AVERAGE, NAVDAT)
130 CONTINUE
140 CONTINUE

CALL PLOT4D

RETURN

END

*******************************************************************************
* ANTPAT
*******************************************************************************

SUBROUTINE ANTPAT (T, H, N, P, JC, B, AO, AVERAGE, NAVDAT)

ROUTINE TO PLOT THE ANTENNA PATTERN
THROUGH THE TURN AT FP-D

PARAMETERS ARE:

T - TIME ARRAY
F - V.C.G. ARRAY
N - NUMBER OF POINTS IN P OR H
F - FREQUENCY
JC - COMPONENT IDENTIFIER:

ENDFIRE BROADSIDE

<table>
<thead>
<tr>
<th>PHO</th>
<th>1</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>PHI</td>
<td>2</td>
<td>5</td>
</tr>
<tr>
<td>ZED</td>
<td>3</td>
<td>6</td>
</tr>
</tbody>
</table>

F - BOUNDS WITHIN H; THE VALUES H(R(1)) - H(R(2)) INCLUSIVE
H(R(3)) - H(R(4)) INCLUSIVE
AND H(R(5)) - H(R(6)) INCLUSIVE ARE PLOTTED.

AO - INITIAL ANGLE - DEFAULTS TO PI

ARANGE - RANGE OF ANGLES - DEFAULTS TO 2*PI

REAL*8 LAB(6) / 'PHO END', 'PHI END', 'ZED END',
                 'PHO HDR', 'PHI HDR', 'ZED HDR' /
REAL*4 H(N), T(N)
INTEGER*4 COMP(F, 2) / 23R, 224, 269, 23P, 224, 269,
               3* 'END', 3* 'HDR' /
               INTEGER*2 R(6)
LOGICAL*1 NAVDAT
WRITE(6,1000) F, LAB(JC), N, P

PLOT X AND Y AXES FOR REFERENCE

CALL PLOT(-4., 0., 3)
CALL PLOT( 4., 0., 2)
CALL PLOT( 0., 4., 3)
CALL PLOT( 0.,-4., 2)

PLOT LABELS: FREQUENCY AND COMPONENT

CALL NUMBER(-1.12, -5., 14, F, 0., 1)
CALL SYMBOL( 999., 999., 14, 7H MHZ, H, 0., 7)
CALL SYMBOL( 999., 999., 14, COMP(JC,1), 0., 1)
CALL SYMBOL( 999., 999., 14, COMP(JC,2), 0., 3)

INITIALIZE THE VALUE OF THE ANGLE
AND PLOT THE FIRST POINT

A = ARANGE / ((B(2) - B(1)) + (B(4) - B(3)) + (B(6) - B(5)) + 2)

20 A=AC

Y = H(B(1)) * COS(A) * 0.1
Y = H(B(1)) * SIN(A) * 0.1
WRITE (6, 2000) B(1), H(B(1)), A, X, Y
CALL SYMBOL (X, Y, .07, 10, 0., -1)
I = B(1) + 1
IF (I .EQ. NAVDAT) GO TO 30
ODCINUE = ODCINT(T(B(1)))
ODCIFAN = ODCINT(T(B(6))) - ODCINUE

DECREMENT THE ANGLE (ROTATION IS CLOCKWISE)

AND PLOT THE NEXT POINT

30 IF (NAVDAT) GO TO 40
A = A - DA
GO TO 45

40 A = AC - ARANGE * (ODCINT(T(I)) - ODCINUE) / ODCIFAN
45 IF (A .LT. -3, 14159) A = A + 6.28318
Y = H(I) * COS(A) * 0.1
Y = H(I) * SIN(A) * 0.1
WRITE (6, 3000) I, H(I), A, X, Y
CALL SYMBOL (X, Y, .07, 10, 0., -2)
I = I + 1
IF (I .EQ. B(2) + 1) I = B(3)
IF (I .EQ. B(4) + 1) I = B(5)
IF (I .EQ. B(6) + 1) GO TO 90

GO TO 30

REDEFINE THE PLOTTED ORIGIN FOR A POSSIBLE NEXT ENTRY;
THEN RETURN

90 CALL PLOT(B, 50, 0.0, -3)
RETURN

1000 FORMAT ('0', F5.1, ' MIN. ', ' A8. ', ' COMPONENT / 1X, I5, ' POINTS' 
. , / '1X, ' POINTS', I5, ' TO ', I5 / 
. , ' 1X, I5, ' TO ', I5 / 
. , ' 4X, 'AND', I5, ' TO ', I5, ' WILL BE PLOTTED')
2000 FORMAT ('0 # BR', 7X, ' ANGLE', 3X, ' Y (PLOT) Y' / 
. , ' 10', I4, 2X, F6.2, F10.5, 5Y, 2PG.3)
3000 FORMAT (1Y, I4, 2X, F6.2, F10.5, 5Y, 2PG.3)
END
SUBROUTINE RUBBLE(X, IX, N)
C
REAL*4 Y(N)
INTEGER*4 IX(N)
C
DO 10 I = 1, N
   IX(I) = I
10 CONTINUE
C
MN = N - 1
C
DO 50 I = 1, MN
   IF(X(IX(I)) .LT. X(IX(I + 1))) GO TO 50
   SWITCH
   IT = IX(I)
   IX(I) = IX(I + 1)
   IX(I + 1) = IT
C
II = I - 1
JJ = I
C
RUBBLE
C
DO 40 J = 1, II
   JJ = JJ + 1
   IF(Y(IX(JJ)) .LT. X(IX(JJ + 1))) GO TO 50
   SWITCH
   IT = IX(JJ)
   IX(JJ) = IX(JJ + 1)
   IX(JJ + 1) = IT
40 CONTINUE
50 CONTINUE
C
RETURN
C
END

*******************************************************************************
CAI STAT
*******************************************************************************

PROGRAM TO COMPARE CALIBRATION DATA OBTAINED ON THE MOON
WITH VALUES FROM TEST RUNS ON EARTH.

REAL*4 TRANGE (396), TEMP (386), CRANGE (140, 6), CAL (140, 6, 3)
REAL*4 AN (6) / -36.06, -39.13, -58.79, -60.9, -104.1, 1.1457 /
REAL*4 GN (6) / 590., 590., 622., 712., 736., 755. /
REAL*4 ANDAMP (6) / -2174., 0., 0., 1304., 2926., -6217 /
REAL*4 ENPAMP (6) / 1182., 1208., 1230., 1290., 1275., 1301. /
REAL*4 EN (140), ENPDA (140), DN (140), DNAPP (140), T (140)
INTEGER*4 NCAL (6, 3), NR (3)

READ THE REQUIRED DATA (THE FOURTH READ STEPING
Separator OVER THE TRANSMITTER-OFER ARRAYS.)

READ (3) TEMP
READ (3) TRANGE
READ (3) CAL, NCAI
READ (3)

LOOP THROUGH FREQUENCIES

DO 100 IFREQ = 1, 6
   IFR = 2 ** (IFREQ - 1)
   WRITE (6, 1000) IFR
   WITHE (6, 6000) IFR
   ENUM = 0.
   ESIG = 0.
   ENAMP = 0.
   ENSIG = 0.
   EGHN = 0.
   EGSIG = 0.
100 CONTINUE
n = 0

LOOP THROUGH THE RANGE ARRAY FOR THIS FREQUENCY.

DO 40 i = 1, 140

FIND TEMPERATURE VALUE FOR CRANGE(I)

IF(CRANGE(I, IFREQ) .GT. 1667.) GO TO 70
IF(CRANGE(I, IFREQ) .GT. TRANGE(I)) GO TO 20
T(I) = TEMP(I)
GO TO 50
20 CONTINUE

DO 40 J = 1, 385

IF(CRANGE(I, IFREQ) .GT. TRANGE(J + 1)) GO TO 40
T(I) = TEMP(J) + (TEMP(J + 1) - TEMP(J)) * (CRANGE(I, IFREQ) - TRANGE(J))
/ (TRANGE(J + 1) - TRANGE(J))

GO TO 50
40 CONTINUE

EBN(I) = AN(IFREQ) * (T(I) - 66.) + BN(IFREQ)
EBNP(A(I) = ANPAMP(IFREQ) * (T(I) - 66.) + PNPAMP(IFREQ)
DN(I) = CAL(I, IFREQ) + BNP(I)
DNPAMP(I) = CAL(I, IFREQ, 2) - EBNPA(I)
N = N + 1
EMN = EMN + DN(I)
EPAMN = EPAMN + DNPAMP(I)
EGMN = EGMN + CAL(I, IFREQ, 1)
60 CONTINUE

70 EBN = EMN / N
EPAMN = EPAMN / N
EGMN = EGMN / N
DO 90 I = 1, N

DDN = ABS(DN(I) - EMN)
DDNP = ABS(DNPAMP(I) - EPAMN)
ESIG = ESIG + DDN * DDN
EPASIG = EPASIG + DDNP * DDNP
WRITE(6, 2000) CRANGE(I, IFREQ), T(I), CAL(I, IFREQ, 3),
EBN(I), DN(I), DDN, CAL(I, IFREQ, 2),
EPAMNP(I), DNPAMP(I), DDNP
CTG = CAL(I, IFREQ, 1) - EGMN
EGSIG = EGSIG + CTG * CTG
WRITE(8, 4000) CRANGE(I, IFREQ), CAL(I, IFREQ, 1), CTG
90 CONTINUE

ESIG = SORT(ESIG / (N - 1))
EPASIG = SORT(EPASIG / (N - 1))
WRITE(6, 3000) EMN, EPAMN, ESIG, EPASIG
EGSIG = SORT(EGSIG / (N - 1))
WRITE(8, 5000) CCW, PFSIG
100 CONTINUE
RETURN
C
1000 FORMAT(*14, ' MHZ', *14, ' CALIBRATION COMPARISON' )
  / *
  . 47X, 'NOISE DIODE', 2PX, 'NOISE DIODE + 20 GH AMP.'
  . / 5X, 'RANGE', 4X, 'TEMPERATURE',
  . 2(10X, 'LUNAR', 5X, 'EARTH', 5X, 'PERIOD',
  . 2X, 'D ERROR') / 2X )
2000 FORMAT(1X, 2(F10.3, 5X), 2(4F10.3, 5X))
3000 FORMAT(*14, 25X, 2(72X, 'MEAN ERROR = ', F10.3) /
  * 26X, 2(14X, 'STANDARD DEVIATION = ', F10.3))
4000 FORMAT(16X, F10.3, 5X, 2F10.3)
5000 FORMAT(*14, 33X, 'MEAN = ', F10.3 )
  . 20X, 'STANDARD DEVIATION = ', F10.3)
6000 FORMAT(*14, 20X, 'RANGE', 10X, 'NOISE D (NOISE)' ) / 2X)
C
END

******************************************************************************
**
** FILTER
**
**
******************************************************************************

SUBROUTINE FILTER(A, N, F, M, B)
C.. N-POINT FILTER FOR ARRAY *A* OF DIMENSION *N*, USING *M* FILTER COEFF
C.. IN ARRAY *F*, ARRAY *B* OF DIMENSION *N* IS USED TO STORE FILTERED B
C.. TEMPORARILY.
C
DIMENSION A(N), B(N), F(M)
C
C.. AVOID ATTEMPTING TO FILTER DATA AT START AND END OF ARRAY.
  K=M/2+1
  L=N-K+1
C
IF(N.LT.K) GO TO 5
C
C.. MAIN LOOP FOR ALL DATA POINTS.
  DO 2 I=K,L
    T=1-K
C
C.. LOOP TO APPLY THE FILTER COEFFICIENTS FOR ONE DATA POINT.
  B(I)=0.
  DO 1 J=1,M
1 E(I) = B(I) + A((IJUB + J) * P(J)
C 2 CONTINUE
C . COPY B BACK INTO A.
DO 3 I=K,L
3 A(I) = B(I)
C
4 WRITE (6,4) M,N,N
C 4 FORMAT ('I4','I4', '4-POINT FILTERING COMPLETED ON', 'I4', ' POINTS. FILTER
C * COEFFICIENTS WERE: '/ (7V,14F9.4/)')
C
C RETURN
C
5 WRITE (6,5) M,N
C 5 FORMAT ('***ERROR*** ATTEMPT TO USE 'I4','I4',-POINT FILTER ON', 'I4,
C * ' POINTS: FILTER REQUEST IGNORED.'/)
C RETURN
C END

*****************************************************************
*****************************************************************
****** SUBROUTINE GAPLOT(FFEO, X, Y, N, YMIN, YMAX, NCOME) ******
*****************************************************************
****** SUBROUTINE GAPLOT(FFEO, X, Y, N, YMIN, YMAX, NCOME) ******
*****************************************************************
****** SUBROUTINE GAPLOT(FFEO, X, Y, N, YMIN, YMAX, NCOME) ******
*****************************************************************

PARAMETERS ARE:

FFEO - FREQUENCY
X - RANGE ARRAY
Y - V.C.O. ARRAY
N - NUMBER OF VALUES IN X OR Y
YMIN - MINIMUM V.C.O. VALUE
YMAX - MAXIMUM V.C.O. VALUE
NCOME - COMPONENT IDENTIFIER:
ENDIRT BROADSIDE

WHO 1
PHI 2
ZFD 3

REAL*4 Y(N), Y(N)
INTEGER*4 TCOMP(6,2) / 230, 724, 769, 235, 724, 769,

WRITE (6,100) PRE0, NCOMP, N, YMIN, YMAX
WRITE(6,200) Y
WRITE(6,300) Y

N POINTS ARE TO BE PLOTTED OVER A RANGE BY (PLOTTER) Y
OF 3,500 INCHES.

DY = 8.599 / N
CALL PLOT(0., 8., 3)
CALL PLOT(0., 0., 2)

SET AN INTEGRAL MINIMUM V.C.O. VALUE; IF THE RANGE OF V.C.O.
VALUES IS GREATER THAN 35 DB, ADJUST THE DATA TO FIT WITHIN
A 7 INCH PLOTTER RANGE - OTHERWISE THE SCALE IS FIXED
AT 5 PP. / INCH.

YMIN = INT((YMIN)
DY = AMAX1(5., INT((ATNT(YMAX - YMNN) + 1.) / 7.) + 1.)
WRITE (6, 150) DY
DY = 1. / DY
DO 20 I = 1, 7
YY = 1
YYY = YMIN + YY / DY
CALL SYMBOL(0., YY, 0.07, 13, 90., -1)
CALL NUMBER(-1., YYY, 0.07, 90., 30., -1)
20 CONTINUE

FINISH LABELLING THE AXIS; THEN PLOT A LABEL
GIVING FREQUENCY AND COMPONENT

CALL SYMBOL(-.25, 6.25, .14, 5HV C 0, 90., 5)
CALL NUMBER(4.5, 0., 14, FREQ, 0., 1)
CALL SYMBOL(.999, .990, 14, 7K HMZ, P, 0., 7)
CALL SYMBOL(.999, .990, 14, TCOMP(NCOMP, 1), 0., -1)
CALL SYMBOL(.999, .990, 14, TCOMP(NCOMP, 2), 0., -1)

PLOT A SYMBOL FOR EACH V.C.O. VALUE.
C
DO 40 I=1,N
XX=DX*I
YY=DY*(Y(I)-YMNN)
CALL SYMBOL(XX,YY,.07,10,0.,-1)
40 CONTINUE
C
DEFINE THE ORIGIN FOR THE NEXT PLOT:
C THEN RETURN.
C
CALL PLOT(3.5,0.,-3)
RETURN
100 FORMAT(10F7.2=',15X,'COMPONENT',12/I6,1,' POINTS'/
     ' MIN. V.C.O.=',F6.1/' MAX. V.C.O.=',F6.1)
150 FORMAT('OSCALE=',F6.1,' INP./INCH')
200 FORMAT('ORANGE ARRAY:1/100(1X,10F10.1/)')
300 FORMAT('OV.C.O. ARRAY:1/100(1X,10F10.1/)')
END

******************************************************************************

*                INTPOL
*
******************************************************************************

SUBROUTINE INTPOL(XIN,YIN,N,XOUT,YOUT,NSTART,NPLOT)
C
LINEAR INTERPOLATION OF YIN VS XIN AT POINTS YOUT. FEB 14/73.
C
INPUT:
C XIN = INPUT X ARRAY
C YIN = INPUT Y ARRAY
C N = DIMENSION OF XIN AND YIN
C YOUT = POINTS AT WHICH YIN WILL BE INTERPOLATED
C NPLOT = DIMENSION OF XOUT AND YOUT

OUTPUT:
C YOUT = INTERPOLATED VALUES OF YIN AT POINTS YOUT
C NSTART = NUMBER OF FIRST POINT INTERPOLATED
C NPLOT = NUMBER OF LAST POINT INTERPOLATED
C DIMENSION XOUT(NPLOT),YOUT(NPLOT),XIN(N),YIN(N)
C NSTART=1
C I=1
C
DO LOOP TO INTERPOLATE YOUT AT EACH XOUT POINT.
DO 50 J=1,NPLOT
40 IF(XIN(I)-XOUT(J)) 10,20,30
C 10 IF (I.EQ.1) GO TO 60
   I=I+1
   GO TO 40
C 20 YOUT(J)=YIN(I)
   GO TO 50
C 30 IF (I.EQ.1) GO TO 33
   YOUT(J)=YIN(I-1)+(XOUT(J)-XIN(I-1))*(YIN(I)-YIN(I-1))/YIN(I-1)
   GO TO 50
C 33 NSTART=J+1
50 CONTINUE
C
RETURN
60 NPLOT=J-1
RETURN
END

******************************************************************************
* LUNACOPY
******************************************************************************
REAL*8  TYPE(2), RUN, SITE, DIRECT, FORREV, TITLE(11)
REAL*4   DATA(12000), RANGE(1000), VCO(1000)
REAL*4   FREQ(6)  /1.0, 2.1, 4.0, 8.1, 16.0, 32.1 /
INTEGER*4 IDATA(400)
INTEGER*2 ITYPE(2)
LOGICAL*4 FIRST, LAST
EQUIVALENCE (DATA(1), IDATA(1))
COMMON /LUNDAT/ TITLE, RUN, SITE, DIRECT, FORREV, TYPE,
   ITYPE, N, PIST, LAST
C
READ LUNAR SEP FILE (#1) AND PRODUCE A FILE OF V.C.O. DATA
INTERPOLATED AT INTERVALS OF 0.1 WAVELENGTH
THE RANGE AND V.C.O. DATA ARE ACCUMULATED IN ARRAY "DATA".

READ AND WRITE THE LABEL RECORD.
THIS RECORD CONTAINS N - THE NUMBER OF VALUES IN EACH SUBSEQUENT RECORD.

CALL LUNIN (DATA, IDATA, 6980, 6990)
WRITE (6, 3000) TYPE
WRITE (6, 1000) RUN, SITE, DIRECT, POPREV, TITLE, N
WRITE (3) RUN, SITE, DIRECT, POPREV, TITLE, N

INITIALIZE THE STACK

10 IORG = 1
M = 0
L = 0

CHECK FOR STACK OVERFLOW BEFORE READING THE NEXT RECORD.

20 IF (IORG + N .GT. 12000) GO TO 970
CALL LUNIN (DATA (IORG), IDATA, 6980, 6990)
IF (IYTP (1) .GE. 6) GO TO 40
WRITE (6, 2000) TYPE
GO TO 20

THIS SECTION IS ENTERED ONLY FOR RANGE AND V.C.O. RECORDS.

40 WRITE (6, 3000) TYPE
IF (IYTP (2) .EQ. 6) GO TO 60

FOR RANGE DATA - MOVE IORG TO POINT ONE LOCATION BEYOND THE LAST VALUE, AND INCREMENT THE COUNT OF RANGE BLOCKS (M).

IF (FIRST) IXX = IORG
IORG = IORG + N
M = M + 1
IF (.NOT. LAST) GO TO 20

AFTER READING THE LAST RANGE BLOCK FOR THIS FREQUENCY,
FILL ARRAY "RANGE" WITH DISTANCES IN METERS CORRESPONDING TO 0.1 WAVELENGTH INCREMENTS; THEN COMPUTE THE NUMBER OF VALUES.
DWL=29.9725/FREQ(IYTYPE(1)-5)
DO 50 I=1, 1000
  RANGE(I)=DWL*FLOAT(I-1)
50 CONTINUE
  NPTSTN=M*N
  GO TO 20

TREATMENT OF V.C.O. DATA IS SIMILAR; ONE SET OF V.C.O. VALUES
HAS BEEN ACCUMULATED WHEN THE COUNT OF V.C.O. BLOCKS (I) FORMS N.

60 IF(FIRST) IXY=IORG
    IORG=IOFG+N
    L=L+1
    IF(L .LT. M) GO TO 20
    CALL INTPOL TO OBTAIN V.C.O. VALUES AT EQUAL RANGE INTERVALS;
    THE NEW VALUES ARE RETURNED IN ARRAY "VCO".
    NSTART=1
    NPLOT=1000
    CALL INTPOL(DATA(IYX), DATA(IYX),NPTSTN,RANGE, VCO, NSTART, NPLOT)
    SET TO ZERO V.C.O. VALUES WHICH HAVE NOT BEEN INTERPOLATED.
    NSTM1=NSTART-1
    IF(NSTM1 .LE. 0) GO TO 80
    DO 70 I=1, NSTM1
      VCO(I)=0.0
    70 CONTINUE
    NPLTP1=NPLOT+1
    IF(NPLTP1 .GT. 1000) GO TO 100
    DO 90 I=NPLTP1, 1000
      VCO(I)=0.0
    90 CONTINUE
    NPTS=NPLOT-NSTM1

WRITE HEADER INFORMATION AND ARRAY "VCO".
WRITE(6,4000) TYPE(1),FREQ(IYTYPE(1)-5),NSTART,NPTS,
  (VCO(I),I=1,NPTS)
WRITE(3) TYPE(1),FREQ(IYTYPE(1)-5),NSTART,NPTS,
  (VCO(I),I=1,NPTS)

IF LAST IS TRUE THEN READ A NEW SET OF RANGE VALUES; OTHERWISE
READ V.C.O. DATA FOR THE NEXT COMPONENT (THE CURRENT RANGE DATA
C ARE RETAINED).
C IF(LAST) GO TO 10
10 ORG=IXY
L=0
GO TO 20
C C STACK OVERFLOW MESSAGE
C C
970 WRITE(6,7000)
GO TO 999
C C END OF FILE ON INPUT WHEN MORE DATA WERE EXPECTED
C C
980 WRITE(6, 5000)
GO TO 999
C C PROCESSING COMPLETED NORMALLY
C C
990 WRITE(6, 6000) TYPE
999 END FILE 3
RETURN
C C
1000 FORMAT('RUN', A6/'OSITE', A6/'ODIRECTION', A6/
. '0', A6, 'TRANSMITTER'/'0', 10A8, A4/'0', I4, 'POINTS')
2000 FORMAT('0', 2A8, 'RECORD SKIPPED')
3000 FORMAT('0', 2A8, 'RECORD READ')
4000 FORMAT('1LABEL=', A8, 'FREQ=', F5.1, 'MHz', '/
. '0FIRST POINT=', I4, '0# OF POINTS=', I4/
. '0', 10F10.3/99(1X, 10F10.3/)
5000 FORMAT('0NORMAL END OF JOB')
6000 FORMAT('0END OF FILE OCCURRED WHILE ATTEMPTING TO READ ',
. 'RECORD')
7000 FORMAT('---*** INSUFFICIENT SPACE ON STACK ***')
C C
END
REAL*8 TYPE(2), RUN, SITE, DIRECT, FORREV, TITLE(11)
REAL*4 DATA(12000), RANGE(1000), VCO(1000)
REAL*4 PRFO(6) /1.0, 2.1, 4.0, 8.1, 16.0, 32.1 /
INTEGER*4 IDATA(400)
INTEGER*2 ITYPE(2)
LOGICAL*4 FIRST, LAST
EQUVALENCE (DATA(1), IDATA(1))
COMMON /LUNDAT/ TITLE, RUN, SITE, DIRECT, FORREV, TYPE, 
          ITYPE, N, FIRST, LAST

READ LUNAR SEP FILE (#2) AND PRODUCE A FILE OF V.C.O. DATA
INTERPOLATED AT INTERVALS OF 0.1 WAVELENGTH

THE RANGE AND V.C.O. DATA ARE ACCUMULATED IN ARRAY "DATA".
IORG IS THE INDEX OF THE NEXT FREE LOCATION INTO WHICH DATA
MAY BE STORED. IXN IS THE INDEX OF THE FIRST RANGE VALUE,
AND IXY IS THE INDEX OF THE FIRST V.C.O. VALUE.

READ AND WRITE THE LABEL RECORD.
THIS RECORD CONTAINS N - THE NUMBER OF
VALUES IN EACH SUBSEQUENT RECORD.

N=386
CALL LUNIN2(DATA, IORD, 6980, 6990)
WRITE(6, 3000) TYPE
WRITE(6, 1000) (IDATA(I), I=1, 297)

INITIALIZE THE STACK

10 IORG=1
M=0
L=0

CHECK FOR STACK OVERFLOW BEFORE READING THE NEXT RECORD.

20 IF(IORG+N .GT. 12000) GO TO 970
CALL LUNIN2(DATA(IORG), IORD, 6980, 6990)
IF(ITYPE(1) .GE. 6) GO TO 40
WRITE(6, 2000) TYPE
GO TO 20

THIS SECTION IS ENTERED ONLY FOR RANGEx AND V.C.O. RECORDS.

40 WRITE (6, 1000) TYPE
   IF (ITYPE(2) .EQ. 6) GO TO 60

   FOR RANGE DATA - MOVE IORG TO POINT ONE LOCATION BEYOND THE LAST VALUE, AND INCREMENT THE COUNT OF RANGE BLOCKS (M).

   IF (FIRST) IXY=IORG
   IORG=IORG+N
   M=M+1
   IF (.NOT. LAST) GO TO 20

   AFTER READING THE LAST RANGE BLOCK FOR THIS FREQUENCY, FILL ARRAY "RANGE" WITH DISTANCES IN METERS CORRESPONDING TO 0.1 WAVELENGTH INCREMENTS; THEN COMPUTE THE NUMBER OF VALUES.

   DWL=20.97925/FREQ(ITYPE(1)-5)
   DO 50 I=1, 1000
   RANGE(I)=DWL*FLOAT(I-1)
   CONTINUE
   NPTSIN=M*N
   GO TO 20

   TREATMENT OF V.C.O. DATA IS SIMILAR; ONE SET OF V.C.O. VALUES HAS BEEN ACCUMULATED WHEN THE COUNT OF V.C.O. BLOCKS (L) EQUALS M.

60 IF (FIRST) IXY=IORG
   IORG=IORG+N
   L=L+1
   IF (L .LT. M) GO TO 20

   CALL INTPOL TO OBTAIN V.C.O. VALUES AT EQUAL RANGE INTERVALS; THE NEW VALUES ARE RETURNED IN ARRAY "VCO".

   NSTART=1
   NPLLOT=1000
   CALL INTPOL(DATA(IXY), DATA(IXY), NPTSIN, RANGE, VCO, NSTART, NPLLOT)

   SET TO ZERO V.C.O. VALUES WHICH HAVE NOT BEEN INTERPOLATED.
C
"NSTM1=NSTART-1"
"I(F(NSTM1 .LE.0) GO TO 30"
DO 70 I=1, NSTM1
VCO(I)=0.0
70 CONTINUE
"NPLTP1=NPLLOT+1"
"IF(NPLTP1.GT.1000) GO TO 100"
DO 90 J=NPLTP1, 1000
VCO(I)=0.0
90 CONTINUE
100 NPTS=NPLLOT-NSTM1
C
WRITE HEADER INFORMATION AND ARRAY "VCO".
C
DO 120 I=NSTART, NPLLOT
VCO(I)=VCO(I)+135.0
120 CONTINUE
WRITE(6,4000) TYPE(1),FREQ(I=TYPE(1)-5),NSTART,NPTS,
VCO(I),I=1,NPTS
WRITE(3) TYPE(1),FREQ(I=TYPE(1)-5),NSTART,NPTS,
VCO(I),I=1,NPTS
C
IF LAST IS TRUE THEN READ A NEW SET OF RANGE VALUES; OTHWWISE
READ V.C.O. DATA FOR THE NEXT COMPONENT (THE CURRENT RANGE DATA
ARE RETAINED).
C
IF(LAST) GO TO 10
JORG=JXY
L=0
GO TO 20
C
STACK OVERFLOW MESSAGE
C
970 WRITE(6,7000)
GO TO 969
C
END OF FILE ON INPUT WHEN MORE DATA WERE EXPECTED
C
980 WRITE(6,5000)
GO TO 999
PROCESSING COMPLETED NORMALLY

999 WRITE (6, 6000) TYPE
999 END FILE 3
END

LUNACPY3

REAL*8 TYPE (2), SUN, SITE, DIRECT, FORREV, TITLE (11)
REAL*4 DATA (12000), RANGE (1000), VCO (1000)
REAL*4 FREQ (6) /1.0, 2.1, 4.0, 8.1, 16.0, 32.1 /
INTEGER*4 IDATA (400)
INTEGER*2 ITYPE (?)
LOGICAL*4 FIRST, LAST
EQUIVALENCE (DATA (1), IDATA (1))
COMMON / LUNDAT/ TITLE, "UN, SITE, DIRECT, FORREV, TYPE, 
ITYPE, N, FIRST, LAST

READ LUNAR SEP FILE (#3) AND PRODUCE A FILE OF V.C.O. DATA
INTERPOLATED AT INTERVALS OF 0.1 WAVELENGTH
RANGE DATA ARE TAKEN FROM FILE #2
THE RANGE AND V.C.O. DATA ARE ACCUMULATED IN ARRAY "DATA".
IORD IS THE INDEX OF THE NEXT FREE LOCATION INTO WHICH DATA
EAX YE SCORID. JXY IS THE INDEX OF THE FIRST RANGE VALUE,
AND IXY IS THE INDEX OF THE FIRST V.C.O. VALUE.

READ AND WRITE THE LABEL RECORD.
THIS RECORD CONTAINS N - THE NUMBER OF
VALUES IN EACH SUBSEQUENT RECORD.

N=385
CALL LUNIN2(DATA, IDATA, E980, E990)
CALL LUNIN3(DATA, IDATA, E980, E990)
WRITE(6, 3000) TYPE
WRITE(6, 1900) (IDATA(I), I=1, 297)

INITIALIZE THE STACK

10 IORG=1
   M=0
   L=0

CHECK FOR STACK OVERFLOW BEFORE READING THE NEXT RECORD.

20 IF (IORG+N .GT. 12000) GO TO 970
   CALL LUNIN2(DATA(IORG), IDATA, E980, E990)
   IF (ITYPE(2) .NE. 5) CALL LUNIN3(DATA(IORG), IDATA, E980, E990)
   IF (ITYPE(1) .GT. 6) GO TO 40
   WRITE(6, 2000) TYPE
   GO TO 20

THIS SECTION IS ENTERED ONLY FOR
RANGE AND V.C.O. RECORDS.

40 WRITE(6, 3000) TYPE
   IF (ITYPE(2) .GE. 6) GO TO 60

FOR RANGE DATA - MOVE IORG TO POINT ONE LOCATION BEYOND THE
LAST VALUE, AND INCREMENT THE COUNT OF RANGE BLOCKS (M).

IF (.FIRST) IXN=IORG
   IORG=IORG+N
   M=M+1
   IF (.NOT. LAST) GO TO 20

AFTER READING THE LAST RANGE BLOCK FOR THIS FREQUENCY,
FILL ARRAY "RANGE" WITH DISTANCES IN METERS CORRESPONDING
TO 0.1 WAVELENGTH INCREMENTS; THEN COMPUTE THE NUMBER OF VALUES.

\[ DLW = 0.9735 / PEO (IYPE(1) - 5) \]
\[ DO 50 J = 1, 1000 \]
\[ RANGE (J) = DLW * FLOAT (J - 1) \]
50 CONTINUE

TREATMENT OF V.C.O. DATA IS SIMILAR; ONE SET OF V.C.O. VALUES
HAS BEEN ACCUMULATED WHEN THE COUNT OF V.C.O. BLOCKS (L) EQUALS 1.

IF (FTEST) IXY=TORG
TORG=JORG+N
L=L+1
IF (L .LT. M) GO TO 20

CALL INTUOL TO OBTAIN V.C.O. VALUES AT EQUAL RANGE INTERVALS;
THE NEW VALUES ARE RETURNED IN ARRAY "VCO".

NSTART=1
NPLOT=1000
CALL INTUOL (DATA (IXY), DATA (IXY), NPTSIN, RANGE, VCO, NSTART, NPLOT)

SET TO ZERO V.C.O. VALUES WHICH HAVE NOT BEEN INTERPOLATED.

NSTM1=NSTART-1
IF (NSTM1 .LT. 0) GO TO 80
DO 70 I = 1, NSTM1
VCO (I) = 0.0
70 CONTINUE

80 NPLTB1=NPLOT+1
IF (NPLTB1 .GT. 1000) GO TO 100
DO 90 I = NPLTB1, 1000
VCO (I) = 0.0
90 CONTINUE

100 NPTS=NPLOT-NSTM1

WRITE HEADER INFORMATION AND ARRAY "VCO".

DO 120 I = NSTART, NPLOT
VCO (I) = VCO (I) * 135.0
120 CONTINUE

WRITE (6,4000) TYPE (1), PEO (IYPE(1) - 5), NSTART, NPTS,
                      (VCO (I), I = 1, NPTS)
WRITE (3) TYPE (1), PEO (IYPE (1) - 5), NSTART, NPTS,
IF LAST IS TRUE THEN READ A NEW SET OF RANGE VALUES; OTHERWISE READ V.C.O. DATA FOR THE NEXT COMPONENT (THE CURRENT RANGE DATA ARE RETAINED).

IF (LAST) GO TO 10
IORG=IVY
L=0
GO TO 20

STACK OVERFLOW MESSAGE

970 WRITE(6,7000)
GO TO 999

END OF FILE ON INPUT WHEN MORE DATA WERE EXPECTED

980 WRITE(6,5000)
GO TO 999

PROCESSING COMPLETED NORMALLY

990 WRITE(6,6000) TYPE
999 END FILE 3
RETURN

1000 FORMAT(27(1X,11A4/))
2000 FORMAT('10',2A8,' Record Skipped')
3000 FORMAT('10',2A8,' Record Read')
4000 FORMAT('11LABEL="",AA,""1/ 'OPreq.="",F5.1," MHz.1/
  . '01Just Point="",I4/ '0# CF Points="",I4/
  . '01,10F10.3/99(1X,10F10.3/)
5000 FORMAT('10NORMAL END OF JOB')
6000 FORMAT('10END OF FILE OCCURRED WHILE ATTEMPTING TO READ ','
  . '2A8, ' RECORD')
7000 FORMAT('10*** INSUFFICIENT SPACE ON STACK ***')

 Apollo 17 SEP - 11
END

**************************************************************************
** *
**  LUMACPY4  
** *
****************************************************************************

C ROUTINE TO COPY THE RANGE AND VCO (UNCORRECTED) DATA FOR THE
C FP-4 TUBE, FOR USE BY THE ANTENNA PATTERN PLOT PROGRAM(S)
C
C THE RANGE AND V.C.O. DATA ARE ACCUMULATED IN ARRAY "DATA".
C JORC IS THE INDEX OF THE NEXT FREE LOCATION INTO WHICH DATA
C MAY BE STORED. JRV IS THE INDEX OF THE FIRST RANGE VALUE,
C AND JRV IS THE INDEX OF THE FIRST V.C.O. VALUE.
C
C SIX NAMELIST CARDS ARE REQUIRED AS DESCRIBED BELOW.
C NAMELIST / CNTL /
C
C FREQ - FREQUENCY INDICATOR (BASE 2 LOG OF FREQUENCY)
C NO DEFAULT
C
C ICOMP - ARRAY OF COMPONENTS TO BE COPIED, OR ZEROS TO IAP
C THE ARRAY OUT TO 6 ELEMENTS. DEFAULT & ZEROS
C CODES FOR THE COMPONENTS ARE:
C ENDIRE BROADSJOE
C
C REAL*8 TYPE(2), RUN, SITE, DIRECT, FOPPEV, TITLE(11)
C REAL*8 PROGRAM(2) / 'QQGRJ', 'CAB' / 
C REAL*4 DATA (12000), RANGE (1000), VCO (1000)
C REAL*8 FREQ(6) / 1.0, 2.1, 4.0, 8.1, 16.0, 32.1 / 
C INTEGER*4 IDATA(400)
C INTEGER*2 ITYPE(2)
C LOGICAL*4 FIRST, LAST
C INTEGER*2 ICOMP(6)
C INTEGER*2 COMP(6) / 212, 222, 232, 211, 221, 231 /
LOGICAL*1 DECIDE(6,6) / 36 * .TRUE. /
NAMELIST / CNTL / IFREQ, ICMP
EQUIVALENCE (DATA(1), IDATA(1))
COMMON / LUNDAT/ TITLE, RUN, SITE, DIRECT, FORREV, TYPE,
.setType, N, FIRST, LAST

C BEGIN BY SETTING DEFAULT VALUES FOR COMPONENT SELECTION
C (NO COMPONENTS), READING CONTROL CARDS, AND SETTING
C CONTROL PARAMETERS
C
DO 10 5600 I=1,6
DO 10 10 600 J=1,6
10100 ICMP(J)=0
READ(5,CNTL,END=10600)
IDY=IFREQ+1
DO 10 1200 J=1,6
IC=COMP(J)
DO 10 1100 K=1,6
IF(IC.EQ.ICMP(K)) GO TO 10120
10110 CONTINUE
DECIDE(IDY,J)=.FALSE.
10120 CONTINUE
10500 CONTINUE
10600 CONTINUE
N=386

C SKIP THE LABEL BLOCK
C
CALL LUNIN2(DATA, IDATA, 6980, 6990)
CALL LUNIN3(DATA, IDATA, 6380, 6390)

C INITIALIZE THE STACK
C
10 IORG=1
M=0
L=0
20 IF(IORG+N.GT.12000) GO TO 970
CALL LUNIN2(DATA(IORG), IDATA, 6980, 6990)
IF(IYP(2).NE.5) CALL LUNIN3(DATA(IORG), IDATA, 6980, 6990)
IF(IYP(1).GE.6) GO TO 40
GO TO 20

C CONTINUE
40 CONTINUE
IF(IYP(2).GE.6) GO TO 60

C ACCUMULATE RANGE BLOCKS
C
IF(P=0) IYX=JORG
JORG=JORG+N
M=M+1
IF(.NOT. LAST) GO TO 20
NPTSIN=N*M
IGX=ITYX
IGXEND=ITYX
C
C FIND THE POINTS WHICH LIE BETWEEN 490 AND 535 MESHES:
C THESE WILL BE COPIED
C
DO 50 I=ITYX,NPTSIN
IF(DATA(I).LE.490.) IGX=IGX+1
IF(DATA(I).LE.535.) IGXEND=IGXEND+1
50 CONTINUE
NCOMP=0
GO TO 20
C
C ACCUMULATE VCO BLOCKS
C
60 IF(FIRST) IYV=JORG
JORG=JORG+N
L=L+1
IF(L.LT.M) GO TO 20
IF(NCOMP.GT.0) GO TO 65
C
C ISOLATE THE POINTS TO BE COPIED
C
IGY=IYV+IGX-ITYX
IGXEND=IYV+IGXEND-ITYX
NPTS=IGXEND-IGY
NCOMP=NCOMP+1
IF(.NOT.DECIDE(ITYP(1)-5,NCOMP)) GO TO 150
YMIN=DATA(IGY)+135.
YMAX=YMIN
ICY=ICY
C
C ADJUST THE DATA VALUES TO RELATIVE DB, AND FIND
C MAXIMUM AND MINIMUM VALUES
C
DO 70 I=1,NPTS
DATA(IY)=DATA(IY)+135.
IF(DATA(IY).LT.YMIN) YMIN=DATA(IY)
IF(DATA(IY).GT.YMAX) YMAX=DATA(IY)
ICY=ICY+1
70 CONTINUE
IF(NCOMP .GT. 1) GO TO 75
      IGXFND = IGXFND - 1
      IGYEND = IGYEND - 1
    75  CONTINUE

    C WRITE OUT THE ACCUMULATED DATA
    C
    WRITE (1) FEPO(ITYPE(1)-5), NCOMP, YMIN, YMAX, NPTS,
     , (DATA(I), I=IGY,IGXFND), (DATA(I), I=IGY,IGXFEND)

    IF THIS WAS THE SIXTH COMPONENT FOR THIS FREQUENCY, READ
    RANGE DATA FOR THE NEXT FREQUENCY; OTHERWISE READ
    VCO DATA FOR THE NEXT COMPONENT

    150 IF(LAST) GO TO 10
       IORG=ITYY
       L=0
       GO TO 20

    C STACK ARRAY TOO SMALL
    C
    970 WRITE(6,7000)
       GO TO 999

    C ALL PROCESSING COMPLETED NORMALLY
    C
    980 WRITE(6,5000)
       GO TO 999

    C PREMATURE END OF INPUT FILP
    C
    990 WRITE(6,6000) TYPE
    999 END "FILP 1"
       RETURN

1000 FORMAT(27(1X,11A4/))
2000 FORMAT('0',2A9,' RECORD SKIPPED')
3000 FORMAT('0',2A9,' RECORD READ')
4000 FORMAT('1','LABEL=','A6','/ 'OPRFQ.=',F5.1,' MHZ.'/
     , '0FIRST POINT=',I4,' # OF POINTS=',I4/
     , '0101,10F10.3/99(1X,10F10.3/)')
5000 FORMAT('O(NORMAL END OF FLP')
6000 FORMAT('EAD OF FILE OCCURRED WHILE ATTEMPTING TO READ ','
     , 'A9', ' RECORD')
7000 FORMAT('1-*** INSUFFICIENT SPACE ON STACK ***')
**H**

```
*** LUNACPY5 ***
```

C PROGRAM TO EXTRACT TEMPERATURE, CALIBRATION,
     TRANSMITTER-OFF, AND SELECTED RANGE INFORMATION FROM
     LUNASED FILE # 2.

C THE RANGE DATA ASSOCIATED WITH THE TEMPERATURE DATA ARE
     A DIRECT COPY OF THE 1 MHZ RANGE ARRAY.

C A SECOND ARRAY CONTAINS RANGE VALUES MATCHED WITH THE
     CALIBRATION AND TXOFF DATA BY SELECTING EVERY 11TH LINE
     FROM THE 32 MHZ RANGE ARRAY, BEGINNING WITH THE 7TH.

C THE ARRAY OF TEMPERATURE DATA IS CONSIDERED DIRECTLY OFF THE
     INPUT FILE.  THE CALIBRATION AND TXOFF DATA ARE IN A
     MULTIPLYED FORM ON THE INPUT FILE (C.E. - 1, WATTS NOISE
     - 13.7, 74).  THE PROGRAM DEMULTIPLEXES THIS INFORMATION
     AND STORES IT IN TWO ARRAYS:

     CAL(I, IFR*0, J), AND
     TXOFF(I, IFR*0, K),

     WHERE I INDICATES THE I-TH VALUE IN SEQUENCE AND IFR*0 IS
     THE (INTEGRAL) BASE-2 LOGARITHM OF THE FREQUENCY.

     J = 1, 2, 3 CORRESPOND TO CALIBRATION FOR GROUND,
     NOISE DIODE * 20 DB, AND NOISE DIODE SOURCES RESPECTIVELY.

     K = 1, 2, 3 INDICATE TXOFF INFORMATION FOR THE Y, Y,
     AND Z ANTENNAS RESPECTIVELY.

REAL*4 RANGE(336), DATA(5019), RANGE2(140, 6)
REAL*4 CAL(140, 6, 3), TXOFF(140, 6, 3)
REAL*4 SPEED(140, 6)
INTEGER*4 MODE(336)
INTEGER*4 NCAL(6, 3) / 18 * 0 /
INTEGER*4 NTXOFF(6, 3) / 18 * 0 /
INTEGER*4 NR(3) / 3 * 0 /
INTEGER*2 ITYPE(2)
```
LOGICAL*4 FIRST, LAST

COMMON / LUNDAF / JUNE(34), ITYPE, N, FIRST, LAST

N = 386
IORG = 1
DO 15 K = 1, 6
   DO 10 J = 1, 140
      RANGE(J, K) = 0.
   DO 5 L = 1, 3
      CALL(J, K, L) = 0.
      TDIFF(J, K, L) = 0.
  5 CONTINUE
10 CONTINUE
15 CONTINUE

20 CALL LUMIN2(DATA(IORG), MODE, 8900, 8900)
IT = ITYPE(1)
GO TO (20, 40, 100, 200, 300, 400, 20, 20, 20, 20, 20, 20, 20), IT

DELETE THE "?" DIGIT FROM EACH ELEMENT OF THE MODE ARRAY.

40 CONTINUE
DO 60 I = 1, N
   MODE(I) = MODE(I) / 10
60 CONTINUE
GO TO 20

THE TEMPERATURE ARRAY IS WRITTEN OUT IMMEDIATELY.

100 WRITE(3) (DATA(I), I = 1, N)
WRITE(6, 1000) (DATA(I), I = 1, N)
GO TO 20

DEMULTIPLY TXOFF DATA INTO ARRAY TXOFF;
TXOFF(IREFD, M) CONTAINS THE MAXIMUM K FOR
TXOFF(K, IREFD, M).

200 IF(FIRST) MM = 0
   M = MOD(MM, 3) + 1
   MM = MM + 1
   L = 0
IF (MM .GT. 3) L = 1
DO 250 I = 1, N
IFREO = 2 * (M - MOD(MODE(I), 10) - 1
NTXOFF(IFREO, M) = NTXOFF(IFREO, M) + 1
TXOFF(NTXOFF(IFREO, M), IFREO, M) = DATA(I)
250 CONTINUE
GO TO 20

FOLLOW THE SAME PROCEDURE TO DEMULTIPLEX THE
CALIBRATION DATA.

300 IF (FIRST) MM = 0
MM = MOD(MM, 1) + 1
MN = MM + 1
L = 0
IF (MM .GT. 3) L = 1
DO 350 I = 1, N
IFREO = 2 * (M - MOD(MODE(I), 10) - 1
NCAL (IFREO, M) = NCAL (IFREO, M) + 1
CAL (NCAL(IFREO, M), IFREO, M) = DATA(I)
350 CONTINUE
GO TO 20

THE RANGE ARRAY FOR 1 MUZ. IS PAIRED WITH THE
TEMPERATURE ARRAY.

400 IF (ITYPE(I) .EQ. 6) GO TO 20
WRITE (3) (DATA(I), I = 1, N)
WRITE (6, 1020) (DATA(I), I = 1, N)
GO TO 20

ACCUMULATE 32 MUZ. RANGE BLOCKS

500 IF (FIRST) NN = 0
MN = NN + N
IONG = IO2G + N
IF (.NOT. LAST) GO TO 20

DEMULTIPLEX THE RANGE DATA TO MATCH THE CALIBRATION
AND TXOFF ARRAYS.

DO 600 I = 7, NN, 13
IT = (I - 7) / 13 + 1
IFREO = M - MOD(MODE(IT), 10)
NR(IFRFO) = NR(IFRFO) + 1
RANGE2(NR(IFRFO), 2 * IPREO) = DATA(I)
RANGE2(NR(IFRFO), 2 * IPREO - 1) = DATA(I)
SPEED(NR(IFRFO), 2 * IPREO) =
  1.234569 * (DATA(I + 1) - DATA(I - 1))
SPEED(NR(IFRFO), 2 * IPREO - 1) =
SPEED(NR(IFRFO), 2 * IPREO)

600 CONTINUE

WRITE AND LIST THE CALIBRATION, TXOFF, AND
ASSOCIATED RANGE INFORMATION.

WRITE(3) CAL, NCAL
WRITE(3) TXOFF, NTXOFF
WRITE(3) RANGE2, NR
WRITE(3) SPEED

DO 700 IPREO = 1, 6
  IFR = 2 ** (IPREO - 1)
  NK = NR((IPREO - 1) / 2 + 1)
  WRITE(6, 1050) IFR, (RANGE2(L, IPREO),
  (CAL(L, IPREO, K), K = 1, 3),
  (TXOFF(L, IPREO, J), J = 1, 3), L = 1, NK)

700 CONTINUE

900 RETURN

1000 FORMAT(' TEMPERATURE' // 26(1X, 15FR.1 / ))
1020 FORMAT(' ORANGES FOR TEMPERATURE ARRAY' // 26(1X, 15FR.1 / ))
1050 FORMAT(' 1', 13, ' W7', 29X, 'CALIBRATION', 37X,
  'TRANSMITTER-OFF' / 11X, 'RANGE', 14Y, 'GROUND', 6Y,
  // 10(1X, E15.1, 2(5X, 3F15.1) / ))

END
***************

LUNALIST

***************

C
PROGRAM TO LIST LUNAR DATA
C
REAL*8 TITLE(11), RUN, SITE, DIRECT, FORREV, TYPE(2)
REAL*4 DATA(825)
INTEGER*4 IDATA(825)
INTEGER*2 ITYPE(2)
INTEGER*2 ITYSAV / 0 /, LINCNT / 0 /
C
COMMON /LUNDAT/ TITLE, RUN, SITE, DIRECT, FORREV,
TYPE, ITYPE, N
C
FLAGS FOR TEMPORARY TRAP
C
LOGICAL*1 TRAP / .FALSE. /, SKIP / .FALSE. /
C
BEGIN EXECUTABLE CODE
C
GET (NEXT) INPUT RECORD
C
20 CALL LUNIN (DATA, IDATA, 400, 500)
21 CONTINUE
IF (.NOT. SKIP)
  GO TO 30
C
ELSE
  IF (ITYPE(1) .NE. 1)
    GO TO 20
C
  ELSE
    IF (LINCNT .LE. 44)
      GO TO 22
C
  ELSE
    WRITE(6, 35) TYPE
    LINCNT = 0
    GO TO 29
C
22 WRITE(6, 25) TYPE
25 FORMAT ('0<<', 2A8, ' >>> / 2X)
28 LINCNT = LINCNT + 16
GO TO 30
30 IF (ITYPE(1) .LE. 3 .OR. ITYPE(2) .NE. ITYSAV)
  WRITE(6, 35) TYPE
35 FORMAT ('1<<', 2A8, ' >>> / 2X)
LINCNT = 3
ITYSAV = ITYPE(1)

CHOOSE APPROPRIATE OUTPUT FORMAT

IF (ITYPE(1) = 2) 100, 200, 300

HEADER

100 WRITE (6, 195) RUN, SITE, DIRECT, FORREV, TITLE, N
105 FORMAT ('RUN ', A6, ' SITE ', A6, ' DIRECT ', A6, /
0', A6, ' TRANSMITTER ', 10', 10A6, M4, /
0', 16 ' POINTS ')

CHECK FOR ARRAY OVERFLOW

IF (N .GT. 825)
   N = 825

COMPUTE NUMBER OF LINES REQUIRED FOR LISTING

NL = MAX0 (N / 15, N / 15 + 1) + 2

TEMPORARY TRAP TO RESTRICT PRINTOUT

IF (TRAP)
   SKIP = .TRUE.
   TRAP = .TRUE.
   GO TO 20

MODE

200 CONTINUE

TRAP

IF (SKIP)
   GO TO 20

WRITE (6, 1090)
WRITE (6, 225) (I DATA (T), T=1,N)
225 FORMAT (44 (1X, 5F17.7), 1X, 15I7)
GO TO 20

ALL OTHER DATA

TRAP
300 CONTINUE
   IF (SKIP)
      GO TO 20
C
   IF (LINCNT + NL .LE. 60)
      GO TO 320
C
   ELSE
      WRITE (6, 75) TYPE
      LINCNT = 3
      WRITE (6, 1000)
      WRITE (6, 350) (DATA (I), I = 1, N)
      FORMAT (54 (1X, 15F7.1 / ), 1X, 15F7.1)
      LINCNT = LINCNT + NL
      GO TO 20
C
RETURN POINTS FOR END OF FILE CONDITIONS
C
400 WRITE (6, 410)
410 FORMAT ('-NORMAL END OF FILE DETECTED')
GO TO 900
C
500 WRITE (6, 510)
510 FORMAT ('-ABNORMAL END OF FILE DETECTED')
C
300 RETURN
C
1000 FORMAT ('N')
C
END

********************************************************************

C

PROGRAM TO LIST LUNAR DATA
C

REAL*4 TITLE (11), RUN, SITE, DIRECT, FORREV, TYPE (2)
REAL*4 DATA (925)
INTEGER*4 I, DATA (825)
INTEGER*2 I, TYPE (2)
INTEGER*2 ITYSAV / 0 /, LINCNT / 0 /
C
COMMON /LUNDAT/ TITLE, RUN, SITE, DIRECT, FORREV,
TYPE, ITYPE, N

FLAGS FOR TEMPORARY TRAP

LOGICAL#1 TRAP / .FALSE. /, SKIP / .FALSE. /

BEGIN EXECUTABLE CODE

GET (NEXT) INPUT RECORD

M=336

20 CALL LUNTIN2 (DATA, IRATA, 5400, 6500)
21 CONTINUE

IF (.NOT. SKIP)
   .GO TO 30

ELSE
   IF (ITYPE(1) .NE. 1)
      .GO TO 20

   ELSE
      IF (LINCT. LE. 44)
         .GO TO 22

   ELSE
      WRITE (6, 35) TYPE
      LINCT = 0
      GO TO 28

22 WRITE (6, 25) TYPE
25 FORMAT ('0'<<' ', 2AH, ' ' >>' / 2X)
28 LINCT = LINCT + 1
      GO TO 30

30 IF (ITYPE(1) .LE. 3 .OR. ITPP(2) .NE. ITPSAVE)
   .WRITE (6, 35) TYPE

35 FORMAT ('1'<<' ', 2AH, ' ' >>' / 2X)
      LINCT = 3
      ITYSAV = ITPP(1)

CHOOSE APPROPRIATE OUTPUT FORMAT

80 IF (ITYPE(1) .EQ. 2) 100, 200, 300

100 WRITE (6, 105) (IDATA(I), I=1, 207)
105 FORMAT (27(1X, 11A4 /), 2X)

CHECK FOR ARRAY OVERFLOW

IF (N .GT. 925)
   .N = 925
C
C  COMPUTE NUMBER OF LINES REQUIRED FOR LISTING
C  NL = MAX( N / 15 , N / 15 + 1 ) + 2
C
C  TEMPORARY TRAP TO RESTRICT PRINTOUT
C
C  IF ( TRAP )  
C     SKIP = .TRUE. 
C     TRAP = .TRUE. 
C     GO TO 20 
C
C  MODE
C
200 CONTINUE
C
C  TRAP
C
C  IF ( SKIP )
C     GO TO 20 
C
C  WRITE(6, 1000)
C  WRITE(6, 225) ( IDATA(I), I=1,N )
225  FORMAT(54 ( 1X, 15I7 / ) , 1X, 15I7)
  GO TO 20 
C
C  ALL OTHER DATA
C
C  TRAP
C
390 CONTINUE
C  IF ( SKIP )
C     GO TO 20 
C
C  IF ( LINCNT + NL . LE. 60 )
C     GO TO 320 
C  ELSE 
C     WRITE(6, 35) TYPE
C     LINCNT = 3
C 320  WRITE(6, 1000)
C  WRITE(6, 350) ( IDATA(I), I=1,N )
350  FORMAT(54 ( 1X, 15F7.1 / ) , 1X, 15F7.1)
   LINCNT = LINCNT + NL
  GO TO 20 
C
C  RETURN POINTS FOR END OF FILE CONDITIONS
C
400 WRITE(6, 410)
410 FORMAT ('/NORMAL END OF FILE DETECTED')
GO TO 900

C
500 WRITE (6, 510)
510 FORMAT ('/ABNORMAL END OF FILE DETECTED')
C
900 RETURN
C
1000 FORMAT ('/')
C
END

******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
*
******************************************************************************
*
*
IF (IY=1) 
GO TO 20

C
ELSE IF (LINCT.E. 44)
    GO TO 22
C
ELSE
    WRITE (6, 35) TYPE
    LINCT = 0
    GO TO 28
22
    WRITE (6, 25) TYPE
25
    FORMAT (' 0<<< ', 2A8, ' >>> / 2Y)
28
    LINCT = LINCT + 1
    ITYSAV = IY
30
IF (IY=1 .OR. IY=2) 
    100, 200, 300
C
    HEADER
C
100
    WRITE (6, 105) (idata(i), i=1, 297)
105
    FORMAT (27(1X, 1A4 /), 2X)
C
    CHECK FOR ARRAY OVERFLOW
C
IF (N.GT. 925)
    N = 925
C
    COMPUTE NUMBER OF LINES REQUIRED FOR LISTING
NL = MAXO (N / 15, N / 15 + 1) + 2
C
    TEMPORARY TRAP TO RESTRICT PRINTOUT
C
IF (IY=1)
    SKIP = .TRUE.
    TRAP = .TRUE.
    GO TO 27
C
    NODE
C
200
    CONTINUE
C TRAP
C IF(SKIP)
      GO TO 20
C
WRITE(6, 1000)
WRITE(6, 225) (*DATA(I), I=1,N)
225 FORMAT(54 (1Y, 15F7.1 /), 1Y, 15F7.1)
GO TO 20
C
C ALL OTHER DATA
C
C TRAP
C
300 CONTINUE:
C IF(SKIP)
      GO TO 20
C
IF(LINCNT + NL .LE. 60)
      GO TO 320
C
310 WRITE(6, 35) TYPE
      LINCNT = 3
320 WRITE(6, 1000)
WRITE(6, 350) (*DATA(I), I=1,N)
350 FORMAT(54 (1Y, 15F7.1 /), 1Y, 15F7.1)
      LINCNT = LINCNT + NL
GO TO 20
C
C RETURN POINTS FOR END OF FILE CONDITIONS
C
400 WRITE(6, 410)
410 FORMAT(' -NORMAL END OF FILE DETECTED')
      GO TO 900
C
500 WRITE(6, 510)
510 FORMAT(' -ABNORMAL END OF FILE DETECTED')
C
900 RETURN
C
1000 FORMAT(' - ')
ROUTINE TO PLOT LUNAR DATA
FROM FILE #1
AFTER INTERPOLATION BY THE COPY PROGRAM
SEVEN NAMELIST CARDS ARE REQUIRED AS INPUT, SIX AS DESCRIBED
BELOW, AND ONE TITLE CARD AS DESCRIBED IN PLOT1.

NAMELIST / PLOT /

IDFH - FREQUENCY INDICATOR (BASE 2 LOG OF FREQUENCY)
NO DEFAULT
XMIN - MINIMUM WAVELENGTH TO BE PLOTTED, DEFAULT 0.6
XMAX - MAXIMUM WAVELENGTH TO BE PLOTTED, DEFAULT 100.6
YMAY - MAXIMUM (RELATIVE DB VALUE TO BE PLOTTED, DEFAULT 7.5
ICOMP - ARRAY OF COMPONENTS TO BE PLOTTED, OR ZEROS TO PAD
THE ARRAY OUT TO 6 ELEMENTS, DEFAULT 6 ZEROS
CODES FOR THE COMPONENTS ARE:

RFDF, BROADSIDE

RHO 212 211
PHI 222 221
ZP 232 231

FILF - (LOGICAL) FILTERING REQUIRED, DEFAULT .FALSE.
COEFS - FILTER COEFFICIENTS, OR ZEROS TO PAD THE ARRAY
TO 100 ELEMENTS (COEFFICIENTS SHOULD BE LEFT-JUSTIFIED
IN THE ARRAY, FOR DEFAULTS SEE DECLARATION OF COEF
NCOEFS - NUMBER OF FILTER COEFFICIENTS, DEFAULT 11
PPF - RELATIVE DB VALUE AT WHICH A REFERENCE MARK IS TO BE
PLOTTED ON THE Y AXIS, DEFAULT 45.0
APOLLO 17 SNP - 91

ABSERR - ABSOLUTE DB VALUE CORRESPONDING TO ERR, DEFAULT 45.

NOTES - UP TO 32 CHARACTERS OF ANNOTATION, NO DEFAULT.

REAL*4 TITLE(2) / 2* /
 REAL*4 VCO(1000), RANGE(1000), COPY(1000)
 REAL*4 NOTES(8) / 8* /
 REAL*4 XLM(6,?) / X*0.0, 16.0, 32.0, 60.0, 84.0, 100.0 /
 REAL*4 YMAXS(6) / 6*100.0 /
 REAL*4 PFF

REAL*4 YMIN, YMXX, YMXX
 REAL*4 COFF(100) /-0.023, -0.041, -0.046, -0.123, -0.207, -0.240, -0.276, -0.350 /
 INTEGER*4 COMP(-) / 212, 232, 232, 211, 221, 231 /
 INTEGER*4 ICMP, ICMP(6), COFFP
 INTEGER*4 NCOFF, NCOFF/
 LOGICAL*1 DECIDE(6,6) /36*TRUE/, FILTE(+)/6*.FALSE./
 LOGICAL*1 POTH, POTH
 PLOTLIST /PLOT/ TPEIO, YMIN, YMXX, YMXX, ICMP, POTH, COFF,
 NCOFF, PTH, NOTES, ABSERR

INITIALIZE RANGE ARRAY

DO 5 I=1,1000
 RANGE(I) = 0.1*FLOAT(I-1)
 CONTINUE

SKIP THE LABEL RECORD

READ (3)
 DO 500 I=1,6

INITIALIZE PLOTTING PARAMETERS TO DEFAULT VALUES, IF ANY

XMIN=0.
XMAX=100.
PITH=.FALSE.
YMAX=67.5
TPE=45.0
ABSERR=45.0
 DO 100 J=1,6

100 ICMP(J) = 0

READ PLOTTING PARAMETERS
READ(F,FBFO,END=520)
IDX=IFBO+1
DO 120 J=1,6
TC=COMP(J)
DO 110 K=1,6
IF(IC .EQ. ICOMP(K)) GO TO 120
110 CONTINUE
DECIDE(IDX,J)=.FALSE.
120 CONTINUE
IF(XMIN.GT.XLIM(IDX,1)) XLIM(IDX,1)=XMIN
IF(XMAX.LT.XLIM(IDX,2)) XLIM(IDX,2)=XMAX
IF(YMAX.LT.YMAXS(IDX)) YMAXS(IDX)=YMAX
FILT(IDX)=FILT
500 CONTINUE
520 DX=0.

C
C INITIALIZE PLOTTTER
C
CALL PLINIT('GOOG,JCR,LUNAR ')
C
C LOOP THROUGH FREQUENCIES
C
DO 900 I=1,6
C
Determine number of curves per graph
C
NA=0
NB=0
XMIN=XLIM(I,1)
XMAX=XLIM(I,2)
DO 550 J=1,3
IF(DECIDE(I,J)) NA=NA+1
IF(DECIDE(I,J+3)) NB=NB+1
550 CONTINUE
BOTH=.FALSE.
CPBRG=NA
IF(NA+NB.GT.3) GO TO 560
BOTH=.TRUE.
CPBRG=CPBRG+NB
560 CONTINUE
MST=IFIX(XMIN*10.0+1.5)
MPT=IFIX(YMAX*10.0+0.5)
C
C Loop through components
C
DO 580 J=1,6
IF(DECIDE(I,J)) GO TO 563
READ(3,END=999)
GO TO 580
580 READ(3,END=239) TITLE(I),F,NST,NPT,(VCO(K),K=1,NPT)
585 K=1,NPT
590 IF(VCO(K),GT.,YMAXS(I)) NST=K+1
595 CONTINUE
C
COMPUTE FIRST POINT AND NUMBER OF POINTS TO BE PLOTTED

NST=MAX0(NST,NST)
NPT=MIN0(NPT,NPT)-NST+1
C
FILTER IF REQUESTED

IF(FILTER(I)) CALL FILTER(VCO(NST),NPT,COEFF,NCoeff,NCST)
C
PLOT THE CURVE

CALL DATPLT(TITLE,NOTES,CPECG,6.18,15.0,CORP(J),VCO(LST),
RANGE(NST),NPT,1,17.0,1.1,F,PEF,ARSEP)
580 CONTINUE
900 CONTINUE
C
TERMINATE THE PLOT WHEN EOF IS DETECTED ON TAPE
C
379 CALL PLOTND
RETURN
END

*******************************************************************************/
*                              LUNAPLT2                                   */
*                                                                            */
*******************************************************************************/
C
ROUTINE TO PLOT LUNAR DATA
 FROM FILE #2
AFTER INTERPOLATION BY THE COPY PROGRAM

SEVEN NAMELIST CARDS ARE REQUIRED AS INPUT, SIX AS DESCRIBED
BELOW, AND ONE PLTID CARD AS DESCRIBED IN PLINIT.
C
NAMELIST / PRED /
Apollo 17 SEP - 94

IFREO - FREQUENCY INDICATOR (BASE 2 LOG OF FREQUENCY)
NO DEFAULT

XMIN - MINIMUM WAVELENGTH TO BE PLOTTED, DEFAULT 0.0

XMAX - MAXIMUM WAVELENGTH TO BE PLOTTED, DEFAULT 100.0

YMAX - MAXIMUM (RELATIVE) DB VALUE TO BE PLOTTED, DEFAULT 67.6

ICOMP - ARRAY OF COMPONENTS TO BE PLOTTED, OR ZEROS TO PAD
THE ARRAY OUT TO 6 ELEMENTS. DEFAULT 6 ZEROS
CODES FOR THE COMPONENTS ARE:

ENDFIRE BROADSIDE

| 000 | 212 | 211 |
| PHI | 222 | 221 |
| ZED | 232 | 231 |

FILT - (LOGICAL) FILTERING REQUIRED, DEFAULT .FALSE.

COEFF - FILTER COEFFICIENTS, OR ZEROS TO PAD THE ARRAY
TO 100 ELEMENTS (COEFFICIENTS SHOULD BE LEFT-JUSTIFIED
IN THE ARRAY, FOR DEFAULTS SEE DECLARATION OF COEFF

NCOEFF - NUMBER OF FILTER COEFFICIENTS, DEFAULT 11

REF - RELATIVE DB VALUE AT WHICH A REFERENCE MARK IS TO BE
PLOTTED ON THE Y AXIS, DEFAULT 45.0

AFSREF - ABSOLUTE DB VALUE CORRESPONDING TO REF, DEFAULT 45.0

NOTES - UP TO 32 CHARACTERS OF ANNOTATION, NO DEFAULT

REAL*4 TITLE(2) / 2*1, / 
REAL*4 VCO(1000), RANG3(1000), WORK(1000)
REAL*4 NOTES (8) / 8*1, / 
REAL*4 YLIN (6,2) / 6*0.0, 16.0, 32.0, 60.0, 3*100.0 / 
REAL*4 YMAXS (6) / 6*100.0 / 
REAL*4 REF 
REAL*4 XMIN, XMAY, YMAX 
REAL*4 COEFF (100) / -.0023, .0041, .0445, .1237, .2078, 
- .2440, .2078, .1239, .0445, .0041, -.0023, 
- 39*0.0 
INTEGER*4 C0MP (6) / 212, 222, 232, 211, 221, 231 /
```fortran
INTEGER*4 IFREQ, ICOMP(6), CDEFG
INTEGER*4 NCOEFF/11/
LOGICAL= DECISION(6,6) /36*.TRUE./, FILTER(6)/6*.FALSE./
LOGICAL= BOTH, FILT
NAMPLIST /IFREQ/ IFREQ, XMIN, XMAX, YMAX, ICOMP, FILT, CDEFG,
NCOEFF, REF, NOTES, ABSDEF

C INITIALIZE RANGE ARRAY

DO 5 I=1,1000
RANGE(I)=0.1*FLOAT(I-1)
5 CONTINUE

C INITIALIZE PLOTTING PARAMETERS TO DEFAULT VALUES, IF ANY

XMIN=0.
XMAX=100.
FILT=.FALSE.
YMAX=67.5
REF=45.0
ABSCOL=45.0
DO 100 J=1,6
100 ICOMP(J)=0

C READ PLOTTING PARAMETERS

READ(5,IFREQ,END=520)
IDX=IFREQ+1
DO 120 J=1,6
IC=COMP(J)
DO 110 K=1,6
IF (IC .EQ. ICOMP(K)) GO TO 120
110 CONTINUE
DECISION(IDX, J)=.FALSE.
120 CONTINUE
IF (XMIN .GT. XLIM(IDX,1)) XLIM(IDX,1)=XMIN
IF (XMAX .LT. XLIM(IDX,2)) XLIM(IDX,2)=XMAX
IF (YMAX .LT. YMAXS(IDX)) YMAXS(IDX)=YMAX
FILTER(IDX)=FILT
500 CONTINUE
520 DY=0.

C INITIALIZE PLOTTER

CALL PLINIT('COEP,JCR.LUNAR '.)
```
LOOP THROUGH FREQUENCIES

DO 900 J=1,6

DETERMINE NUMBER OF CURVES PER GRAPH

NA=0
NB=0
XMIN=XLIM(I,1)
XMAX=XLIM(J,2)
DO 550 J=1,3
IF (DECIDE(I,J)) NA=NA+1
IF (DECIDE(I,J+3)) NB=NB+1
550 CONTINUE
ROT=.FALSE.
CPERG=NA
IF (NA+NB.GT.3) GO TO 560
ROT=.TRUE.
CPERG=CPERG+NB
560 CONTINUE
MST=IFIY (XMIN*10.0+1.5)
MPT=IFIY (XMAX*10.0+0.5)

LOOP THROUGH COMPONENTS

DO 580 J=1,6
IF (DECIDE(I,J)) GO TO 563
READ(3,END=393)
GO TO 580
563 READ(3,END=393) TITLE(1),F,MST,NPT,(VCO(K),K=1,NPT)
DO 566 K=1,NPT
IF (VCO(K).GT.YMAXS(I)) MST=K+1
565 CONTINUE

COMPUTE FIRST POINT AND NUMBER OF POINTS TO BE PLOTTED

MST=MAX0 (NST,MST)
NPT=MIN0 (NPT,MPT)-NST+1

FILTER IF REQUESTED

IF (FITPER(I)) CALL FILTER(VCO(NST),NPT,COEFF,MCOEFF,STRT)

PLOT THE CURVE

CALL DATPLOT(TITLE,NOTES,CPERG,6,18,15,0,COMP(J),VCO(NST),
          RANGE(NST),NPT,1,17,0,1,1,F,DEF,ARSPY)
IF (ROT=.OR. J.NE.4) GO TO 580
CPEGENR
940 CONTINUE
990 CONTINUE

C TERMINATE THE PLOT WHEN TP0F IS DETECTED ON TAPE
C
990 CALL PLOTED
RETURN
END

*******************************************************************************

LUAPLIT

*******************************************************************************

C ROUTINE TO PLOT LUNAR SED DATA FROM FILE #2;
C NO INTERPOLATION;
C VALUES WITH 510. M. < RANGE < 520. M.
C ARE DELETED BEFORE PLOTTING.
C
C THE RANGE AND V.C.O. DATA ARE ACCUMULATED IN ARRAY "DATA".
C LORG IS THE INDEX OF THE NEXT FREE LOCATION INTO WHICH DATA
C MAY BE STORED, LXX IS THE INDEX OF THE FIRST RANGE VALUE,
C AND LYY IS THE INDEX OF THE FIRST V.C.O. VALUE.
C
C SEVEN NAMELIST CARDS ARE REQUIRED AS INPUT, SIX AS DESCRIBED
C BELOW, AND ONE PLOT Card AS DESCRIBED IN PLINIT.
C
NAMELIST / CNTL /
C
IPRF - FREQUENCY INDICATOR (BASE 2 LOG OF FREQUENCY)
C NO DEFAULT
C
XMIN - MINIMUM WAVELENGTH TO BE PLOTTED, DEFAULT 0.0
C
XMAX - MAXIMUM WAVELENGTH TO BE PLOTTED, DEFAULT 100.0
C
YMAX - MAXIMUM (RELATIVE) DB VALUE TO BE PLOTTED, DEFAULT 47.5
C
ICOMP - ARRAY OF COMPONENTS TO BE PLOTTED, OR ZEROS TO PUT
THE ARRAY OUT TO 6 ELEMENTS, DEFAULT 6 ZEROS
C
CODES FOR THE COMPONENTS ARE:
ENDFIRE  BROMESIDE

RHO  212  211
PHI  222  221
THETA  232  231

REF  - RELATIVE DB VALUE AT WHICH A REFEREE MARK IS TO BE PLOTTED ON THE Y AXIS, DEFAULT 45.0
ABSREF - ABSOLUTE DB VALUE CORRESPONDING TO REF, DEFAULT 45.0
NOTES  - UP TO 32 CHARACTERS OF ANNOTATION, NO DEFAULT

REAL*8  TYPE(2), RUN, SITE, DIRECT, FORPEV, TITLE(11)
REAL*4  DATA(12000), RANGE(1000), VCO(1000)
REAL*4  FREQ(6) /1.0, 2.1, 4.0, 8.1, 16.0, 32.1 /
INTEGER*4  IDATA(400)
INTEGER*2  ITYPE(2)
LOGICAL*1  FIRST, LAST
'EQUIVALENCE  (DATA(1), IDATA(1))
REAL*4  NOTES(4) /***/
REAL*4  XLM(6,2) /6*0.0,16.0,32.0,60.0,3*100.0/
REAL*4  YMAXS(6) /6*100.0/
INTEGER*4  ICOMP(6), NA(6), NR(6), CPERSG(6,6)
INTEGER*4  COMP(6) /217,222,232,211,221,231/
LOGICAL*1  DECIDE(6,6) /6*TRUE,7*FALSE/
COMMON /LUNDAT/ TITLE, RUN, SITE, DIRECT, FORPEV, TYPE,
          ITYPE, N, FIRST, LAST

DO 10500  I=1,6
INITIALIZE PLOTTING PARAMETERS TO DEFAULT VALUES , IF ANY
XMIN=0.0
XMAX=100.0
YMAX=67.5
REF=45.0
ABSREF=45.0
DO 10100  J=1,6
10100  ICOMP(J)=0
10500  CONTINUE
READ PLOTTING PARAMETERS
READ (5,CNTL,END=10600)
IDY = ITPF + 1
DO 1020 J=1, N
IC = COMP(J)
DO 10110 K = 1, G
IF (IC.EQ. [COMP(K)]) GO TO 1020
10110 CONTINUE
DECIDE(IDY, J) = .FALSE.
10120 CONTINUE
C
SET MIN AND MAX RANGE, AND MAX VCO FOR THIS COMPONENT
C
IF (XMIN.GT.XLIM(IDY, 1)) XLIM(IDY, 1) = XMIN
IF (YMAX.LT.XLIM(IDY, 2)) XLIM(IDY, 2) = YMAX
IF (YMAX.LT.YMAXS(IDY)) YMAXS(IDY) = YMAX
NA(IDY) = 0
NB(IDY) = 0
C
DECIDE ON THE NUMBER OF CURVES PER GRAPH
C
DO 10200 J=1, 3
IF (DECIDE(IDY, J)) NA(IDY) = NA(IDY) + 1
IF (DECIDE(IDY, J+1)) NB(IDY) = NB(IDY) + 1
10200 CONTINUE
BOTH(IDY) = .FALSE.
IF (NA(IDY) + NB(IDY).GT. 3) BOTH(IDY) = .FALSE.
DO 10300 J=1, 3
IF (BOTH(IDY)) GO TO 10250
CPEEG(IDY, J ) = NA(IDY)
CPEEG(IDY, J+ 3) = NB(IDY)
GO TO 10300
10250 CPEEG(IDY, J ) = NA(IDY) + NB(IDY)
CPEEG(IDY, J+ 3) = CPEEG(IDY, J)
10300 CONTINUE
10500 CONTINUE
10600 CONTINUE
C
INITIALIZE THE PLOTTER
C
CALL PLINIT('DOGP, JCR, LUNAR ')
N = 306
C
READ THE LABEL RECORD
C
CALL LUNIN2(IDATA, IDATA, 6980, 6990)
WRITE(6, 1000) TYPE
WRITE(6, 1000) (IDATA(I), I=1, 297)
C
INITIALIZE THE STACK BEFORE READING RANGE DATA
10  IORG=1
   M=0
   L=0
20  IF (IORG+N .GT. 12000) GO TO 970
   CALL TUMIN2 (DATA (IORG), IDATA, 5900, L990)
   IF (I TYPE (1) .IE. 5) GO TO 20

   IF (I TYPE (2) .IE. 6) GO TO 60

   ACCUMULATE RANGE BLOCKS

   IF (FIRST) IXX=IORG
   IORG=IORG+N
   M=M+1
   IF (.NOT. LAST) GO TO 20
   XMTOWL=FREQ (I TYPE (1) -5)/299.7925
   NPTSIN=M*N
   NXR GAP=0
   NXAGAP=0

   FIND RANGE VALUES TO BE OMITTED

   DO 45 I=1,NPTSIN
   IF (DATA (I),GE,510.0) GO TO 50
   NXR GAP=NXR GAP+1
   DATA (I)=XMTOWL* (DATA (I)+3.0)
45  CONTINUE

   INEXT=NXR GAP+1
   ISTART=INEXT

   DELETE VALUES BY COMPRESSING THE ARRAY

   DO 55 I=ISTART,NPTSIN
   IF (DATA (I),LE,520.0) GO TO 55
   DATA (INEXT)=XMTOWL* (DATA (I)+3.0)
   NXAGAP=NXAGAP+1
   INEXT=INEXT+1
55  CONTINUE

   RESET THE ORIGIN FOR VCO DATA, AND ZERO THE COMPONENT COUNT.

   IORG=NXR GAP+NXAGAP+1
   NCOMP=0
   GO TO 20
ACCUMULATE VCO BLOCKS

60 IF(FIRST) TV=I0RG
    T0RG=10RG+N
    L=L+1
    IF(1.LT.M) GO TO 20
    IDX=ITYP(1)-6
    NCOMP=NCOMP+1
    IF(.NOT.RECIRC(TX,NCOMP)) GO TO 150

    FIND THE LAST VALUE REPORT THE GAP, AND ADJUST TO VALUES
    TO A RELATIVE SCALE

    IY=IYX+NXBGAP-1
    DO 70 J=IYX,IY
    70 CONTINUE

    IY=IYX+NXBGAP
    ISTART=T0RG-NXAGAP
    IEND=T0RG-1

    COMPRESS THE VCO DATA, ADJUSTING
    TO RELATIVE SCALE IN THE PROCESS

    DO 80 I=ISTART,IEND
    DATA(INEXT)=DATA(I)*135.0
    INEXT=INEXT+1
    80 CONTINUE

    NSTY=IYX
    Nx=IYX
    NPL0T=IYX+NXBGAP+NXAGAP-1
    IY=IYX

    OMIT VALUES OUTSIDE THE OUTER BOUNDS

    DO 90 IX=NX,NPL0T
    IF(DATA(IX).LT.XLIM(IDX,1).OR.DATA(IX).GT.YMAXS(IDX))
    NSTX=IX+1
    IY=IY+1
     90 CONTINUE

    NSTY=IYX+NSTX-IYX
    IY=NPL0T
    NX=NPL0T
    DO 100 I=NSTX,NX
    IF(DATA(IX).LT.XLIM(IDX,2)) NPL0T=NPL0T-1
    IX=IX-1
     100 CONTINUE

    NPTS=NPL0T-NSFX+1
PLOT THE CURVE

CALL DATPLT(TYPE, NOTES, CPARSG(IDY, NCOMP), 6.10, 15.0, COMP(NCOMP),
  DATA(NSTY), DATA(NSTX), NPTS, 1.17, 1.1, FREQ(IDY),
  REF, ABSREF)

IF THIS WAS THE SIXTH COMPONENT, GET A NEW RANGE ARRAY, OTHERWISE
GET A VCO ARRAY FOR THE NEXT COMPONENT

150 IF (LAST) GO TO 10

100 IF (K=STY) L=0

GO TO 20

STACK ARRAY TOO SMALL

270 WRITE(6, 7000)
GO TO 399

NORMAL COMPLETION

390 WRITE(6, 5000)
GO TO 999

DATA (NSTY), DATA (NSTX), NPTS, 1.17, 1.1, FREQ (IDY),

END
**LUAPL14**

ROUTINE TO PLOT SEP DATA THROUGH THE TUBE AT EP-4 VS. RECORD NUMBER

THE RANGE AND V.C.O. DATA ARE ACCUMULATED IN ARRAY "DATA".
IOPG IS THE INDEX OF THE NEXT FREE LOCATION INTO WHICH DATA
MAY BE STORED, IIX IS THE INDEX OF THE FIRST RANGE VALUE,
AND IIVV IS THE INDEX OF THE FIRST V.C.O. VALUE.

SEVEN NAMFILIST CARDS ARE REQUIRED AS INPUT, SIX AS DESCRIBED
BELLOW, AND ONE PLOTID CARD AS DESCRIBED IN PLOTIT.

NAMFILST / CNTL /

FREQO - FREQUENCY INDICATOR (BASE 2 LOG OF FREQUENCY)
NO DEFAULT

ICOMPO - ARRAY OF COMPONENTS TO BE PLOTTED, OR ZEROS TO PAD
THE ARRAY OUT TO 6 ELEMENTS, DEFAULT IF ZEROS
CODES FOR THE COMPONENTS ARE:

<table>
<thead>
<tr>
<th>CODE</th>
<th>FREQUENCY CATEGORY</th>
</tr>
</thead>
<tbody>
<tr>
<td>830</td>
<td>BROADSIDE</td>
</tr>
<tr>
<td>212</td>
<td></td>
</tr>
<tr>
<td>211</td>
<td></td>
</tr>
<tr>
<td>PHI</td>
<td></td>
</tr>
<tr>
<td>222</td>
<td></td>
</tr>
<tr>
<td>221</td>
<td></td>
</tr>
<tr>
<td>ZED</td>
<td></td>
</tr>
<tr>
<td>232</td>
<td></td>
</tr>
<tr>
<td>231</td>
<td></td>
</tr>
</tbody>
</table>

REAL*8  TYPE(2), RUN, SITE, DIRECT, POLREV, TITLE(11)
REAL*8  PROGM(2) / 'OOGP.JCR', 'GAP' / 
REAL*4  DATA(12000), RANGE(1000), VCO(1000)
REAL*4  PRS(6) / 1.0, 2.1, 4.0, 8.1, 16.0, 32.1 / 
INTEGER*4 IDATA(400)
INTEGER*2 ITYPE(2)
LOGICAL*4 FIRST, LAST
INTEGER*2 ICOMP(6)
INTEGER*2 COMP(6) / 212, 222, 232, 211, 221, 231 / 
LOGICAL*1 DECIDE(6,6) / 36 * .TRUE. / 
NAMFILST / CNTL / FREQO, ICOMP
EQUIVALENCE (DATA(1), IDATA(1))
COMMON /UNITAT/, TITLE, RUN, SITE, DIRECT, FORREV, TYPE,
       IYPE, N, FIRST, LAST

C
C  DO 10500 I=1,6
C
C  NO COMPONENTS PLOITED UNLESS REQUESTED
C
C  DO 10100 J=1,6
10100 ICOMMP(J) = 0
C
C  READ FREQUENCY INDICATOR AND COMPONENTS TO BE PLOITED
C
C  READ (5, CPTL, END = 10600)
IDX = IFRPO + 1
C
C  DO 10120 J = 1, 6
C  IC = COMP(J)
C  DO 10110 K = 1, 6
C  IF(IC.EQ.JCOMP(K)) GO TO 10120
10110 CONTINUE
C  DECIDE (IDX, J) = .FALSE.
10120 CONTINUE
10500 CONTINUE
10600 CONTINUE
C
C  INITIALIZE THE PLOTTER
C
C  CALL FLINIT (TOGNM)
N = 396
C
C  SKIP THE LABEL BLOCK
C
C  CALL LUNIT2 (DATA, IDATA, 5280, 5280)
C  CALL LUNIT3 (DATA, IDATA, 5280, 5280)
C
C  INITIALIZE THE STACK
C
10 IORG = 1
M = 0
I = 0
20 IF(IORG + N .GT. 12000) GO TO 970
C  CALL LUNIT2 (DATA(IORG), IDATA, 5280, 5280)
IF(ITYPE(2) .NE. 5) THEN
   CALL LUNIT3 (DATA(IORG), IDATA, 5280, 5280)
   IF(ITYPE(1) .GE. 6) GO TO 40
GO TO 20
ACCUMULATE RANGE BLOCKS

40 CONTINUE
IF (ITYPE(2) .EQ. 6) GO TO 60
IF (FIRST) IXX=IORG
IOPG=IOPG+N
M=M+1
IF (.NOT. LAST) GO TO 20
NPTSIN=N*M
IGX=IXX
IGXEND=IXX

FIND THE GAP

DO 50 I=IXX,NPTSIN
IF (DATA(I) .LE. 0.90) IGX=IGX+1
IF (DATA(I) .LE. 535.) IGXEND = IGXEND + 1
50 CONTINUE
NCOMP=0
GO TO 20

ACCUMULATE VCO BLOCKS

60 IF (FIRST) IXX=IORG
IOPG=IOPG+N
L=L+1
IF (L .LT. M) GO TO 20
IGY=IXX+IGY-IXX
IGXEND=IXX+IGXEND-IXX
NPTS=IGXEND-IGX
NCOMP=NCOMP+1
IF (.NOT. DECIDE (ITYPE(1)-5,NCOMP)) GO TO 150
YMIN=DATA(IGY)+135.
YMAX=YMIN
IY=IGY

ADJUST DB VALUES TO RELATIVE SCALE, AND FIND MINIMUM AND
MAXIMUM VCO THROUGH THE TUN

DO 70 I=1,NPTS
DATA(IY)=DATA(IY)+135.
IF (DATA(IY) .LT. YMIN) YMIN=DATA(IY)
IF (DATA(IY) .GT. YMAX) YMAX=DATA(IY)
IY=IY+1
70 CONTINUE

PLOT THE POINTS
CALL G4PLOT(FPRO,(ITYPE(1)-5),DATA(IGX),DATA(I0Y),
        NTST, YMIN, YMAX, NCOMD)
C IF THIS WAS THE SIXTH COMPONENT, GET NEW RANGE DATA; OTHERWISE
C GET NEW VCO DATA
C 150 IF (LAST) GO TO 10
TORG=IYY
L=0
GO TO 20
C STACK ARRAY TOO SMALL
C 170 WRITE(6,5000)
        GO TO 900
C NORMAL COMPLETION
C 900 WRITE(6, 5000)
        GO TO 979
C PREMATURE END OF DATA FILE
C 970 WRITE(6, 6000) TYPE
      999 CALL PLOTND
      RETURN
C
C 1000 FORMAT(27(1X,114/))
2000 FORMAT(' 10',2A8,' RECORD SKIPPED')
3000 FORMAT(' 10',2A8,' RECORD READ')
4000 FORMAT(' 10',2A8,' ')
      4000 FORMAT('REPORT. ',A5,' SUM')/
      4000 FORMAT('FIRST POINT=',1G,' 0# CP POINTS=',1G/
      4000 FORMAT('0',10=10.339(1Y,10=10.3/))
5000 FORMAT('NORMAL END OF JOB')
6000 FORMAT('END OF FILE OCCURRED WHILE ATTEMPTING TO READ ',
      6000 FORMAT('RECORD')
7000 FORMAT('*** INSUFFICIENT SPACE ON STACK ***')
C
C END
ROUTINE TO PLOT LUNAR SRF DATA FROM FILE #2;
NO INTERPOLATION;
VALUES WITH 510. M. <= RANGE <= 520. M.
ARE DELETED BEFORE PLOTTING.
TRANSMITTER-OFF DATA ARE PLOTTED AS A BASELINE FOR EACH COMPONENT.
THE RANGE AND V.C.O. DATA ARE ACCUMULATED IN ARRAY "DATA",
IOPG IS THE INDEX OF THE NEXT FREE LOCATION INTO WHICH DATA
MAY BE STORED. IXX IS THE INDEX OF THE FIRST RANGE VALUE,
AND YXY IS THE INDEX OF THE FIRST V.C.O. VALUE.
SEVEN NAMELIST CARDS ARE REQUIRED AS INPUT, SIX AS DESCRIBED
BELOW, AND ONE PLOT D卡D AS DESCRIBED IN PLINIT.

NAMELIST / CNTL /

IFREQ  - FREQUENCY INDICATOR (BASE 2 LOG OF FREQUENCY)
NO DEFAULT

XMIN,  - MINIMUM AND MAXIMUM RANGE VALUES TO BE PLOTTED, IN
WAVELENGTHS IF VSWL = .TRUE., OTHERWISE IN METERS;
DEFAULTS: 0.0, 100.0

YMAX  - MAXIMUM (RELATIVE) DB VALUE TO BE PLOTTED, DEFAULT 67.5

ICOMP  - ARRAY OF COMPONENTS TO BE PLOTTED, OR ZEROS TO PAD
THE ARRAY OUT TO 6 ELEMENTS, DEFAULT 6 ZEROS;
CODES FOR THE COMPONENTS ARE:

ENDFIRE  BROADSIDE

PHI   212  211
PHI   222  221
ZED   232  231

VSWL  - (LOGICAL) IF TRUE (DEFAULT), DB VALUES ARE PLOTTED
VS. RANGE IN WAVELENGTHS, OTHERWISE VS. RANGE
DO 1050 T=1,6

INITIALIZE PLOTTING PARAMETERS TO DEFAULT VALUES, IF ANY

XMIN=0.0  
XMAX=100.0  
YMAX=67.5  
YFF=45.0  
ARDEP=45.0  
DO 10100 J=1,6

10100 ICOMP(J)=0

READ PLOTTING PARAMETERS

READ (5,CNTL,END=10600) 
IDX=IFREQ+1
DO 10120 J=1,6
TC= COMP(J)
DO 10110 K=1,6
IF(TC.EQ.TCOMP(K)) GO TO 10120
10110 CONTINUE
DECIDE(IDX,J)=.FALSE.
10120 CONTINUE
C
C SET MIN AND MAX RANGE, AND MAX VCO FOR THIS COMPONENT
C
VLIM(IDX,1) = XMIN
VLIM(IDX,2) = XMAX
VMAXS(IDX) = YMAX
NA(IDX) = 0
NB(IDX) = 0
C
C DECIDE ON THE NUMBER OF CURVES PER GRAPH
C
DO 10200 J=1,3
IF(DCIDE(IDX,J )) NA(IDX)=NA(IDX)+1
IF(DCIDE(IDX,J+3)) NB(IDX)=NB(IDX)+1
10200 CONTINUE
RORTH(IDX)=.TRUE.
IF(NA(IDX)+NB(IDX).GT.3) RORTH(IDX)=.FALSE.
DO 10300 J=1,3
IF(RORTH(IDX)) GO TO 10250
CPERG(IDX,J )=NA(IDX)
CPERG(IDX,J+3)=NB(IDX)
GO TO 10300
10250 CPERG(IDX,J )=NA(IDX)+NB(IDX)
CPERG(IDX,J+3)=CPERG(IDX,J)
10300 CONTINUE
10500 CONTINUE
10600 CONTINUE
XSCHANGE = 10.
IF(VSWL) XSCHANGE = 6.18
C
C READ TRANSMITTER-OFF DATA AND THE ASSOCIATED RANGE VALUES.
C
READ(3)
READ(3)
READ(3)
READ(3) TXOFF, NTONF
READ(3) RANGE?, NR
C
C REMOVE DATA FOR 510 M. <= RANGE <= 520 M.
C
DO 20500 IF7 = 1, 6
T W Y T = 1
P = N F ((I F P - 1) / 2 + 1)
X M T O W L = P P E C (I F P) / 237.7925
I F (. N O T . VS W) X M T O W L = 1.
D O 2 0 4 0 0  I = 1, N
I F (R A N G E 2 (J, I F P) . L T . X L I M (I F P, 1) / X M T O W L - 2.) G O T O 2 0 4 0 0
I F (R A N G E 2 (I, I F P) . G T . X L I M (I F P, 2) / X M T O W L - 2.) G O T O 2 0 4 0 0
I F (R A N G E 2 (I, I F P) . G E . 5 1 0.) G O T O 2 0 2 0 0
R A N G E 2 (I N E X T , I F P) = (R A N G E 2 (I, I F P) + 3.) * X M T O W L.
D O 2 0 1 0 0  J = 1, 3
2 0 1 3 0 C O N T I N U E
G O T O 2 0 1 5 0
2 0 2 0 0 I F (R A N G E 2 (I, I F P) . L E . 5 2 0.) G O T O 2 0 4 0 0
R A N G E 2 (I N E X T , I F P) = (R A N G E 2 (I, I F P) + 3.) * X M T O W L
D O 2 0 3 0 0  J = 1, 3
2 0 3 5 0 C O N T I N U E
2 0 3 5 0 N N (I F P) = I N E X T
I N E X T = I N E X T + 1
2 0 4 0 0 C O N T I N U E
2 0 5 0 0 C O N T I N U E
C
C I N I T I A L I Z E T H E P L O T T E R
C
N = 3 8 5
C
C R E A D T H E L A B E L R E C O R D
C
C C A L L L U N I M 2 (D A T A , T D A T A , 5 9 8 0 , 5 9 9 0 )
W R I T E ( 6 , 3 0 0 0 ) T Y P E
W R I T E ( 6 , 1 0 0 0 ) ( T D A T A ( I ) , I = 1 , 2 9 7 )
C
C I N I T I A L I Z E T H E S T A C K B E F O R E R E A D I N G R A N G E D A T A
C
1 0 I O R G = 1
M = 0
L = 0
2 0 I F ( I O R G * N . G T . 1 2 0 0 0 ) G O T O 9 7 0
C A L L L U N I M 2 (D A T A ( I O R G ) , T D A T A , 5 9 8 0 , 5 9 9 0 )
I F ( T T Y P E ( 1 ) . L T . 5 ) G O T O 2 0
C
C I F ( T T Y P E ( 2 ) . E Q . 6 ) G O T O 6 0
C
C A C C U M U L A T E R A N G E B L O C K S
IF (FIRST) IXY=Iorg
Iorg=Iorg+N
M=M+1
IF (.NOT. LAST) GO TO 20
XMTOWL=FPE0 (ITYPE (1)-5)/299.7925
IF (.NOT. WSWL) XMTOWL = 1.
NPTSIN=M*N
NXAGAP=0
NYAGAP=0

C FIND RANGE VALUES TO BE OMITTED
C
DO 45 I=1,NPTSIN
IF (DATA (I).GE.510.0) GO TO 50
NXAGAP=NXAGAP+1
DATA (I)=XMTOWL*(DATA (I)+3.0)
45 CONTINUE
50 INEXT=NXAGAP+1
Istart=INext

C DELETE VALUES BY COMPRESSING THE ARRAY
C
DO 55 J=Istart,NPTSIN
IF (DATA (J).LE.520.0) GO TO 55
DATA (INEXT)=XMTOWL*(DATA (J)+3.0)
NXAGAP=NXAGAP+1
INEXT=INEXT+1
55 CONTINUE

C RESET THE ORIGIN FOR VCO DATA, AND ZERO THE COMPONENT COUNTS
C
Iorg=NXAGAP+NYAGAP+1
NCOMP=0
GO TO 20

C ACCUMULATE VCO BLOCKS
C
60 IF (FIRST) IXY=Iorg
Iorg=Iorg+N
L=L+1
IF (L.LT. M) GO TO 20
IDX=ITYPE (1)-5
NCOMP=NCOMP+1
IF (.NOT. DECIDE (IDX, NCOMP)) GO TO 150

C C FIND THE LAST VALUE BEFORE THE GAP, AND ADJUST DR VALUES:
C TO A RELATIVE SCALE

LY = IXY + NYGAP - 1
DO 70 I = IXY, LY
DATA (I) = DATA (I) + 135.0
70 CONTINUE
INEXT = IXY + NYGAP
ISTART = TOPG - NYGAP
TEND = TOPG - 1

C COMPRESS THE VCO DATA, ADJUSTING
C TO RELATIVE SCALE IN THE PROCESS

DO 80 I = ISTART, TEND
DATA (INEXT) = DATA (I) + 135.0
INEXT = INEXT + 1
80 CONTINUE
NSTY = IXY
NX = IXY
NPLT = IXY + NYGAP + NYGAP - 1
TY = IXY

C OMIT VALUES OUTSIDE THE OUTER BOUNDS

DO 90 IX = NX, NPLT
IF (DATA (IX) .LT. YLIM (IDX, 1) .OR. DATA (IX) .GT. YMAXS (IDX))
   \ NSTX = IX + 1
   \ IX = IX + 1
90 CONTINUE
NSTY = IXY + NSTX - IXX
IX = NPLT
NX = NPLT
DO 100 I = NSTX, NX
TY = DATA (IX) .GT. XLM (IDX, 2) NPLT = NPLT - 1
IX = IX + 1
100 CONTINUE
NPTS = NPLT - NSTX + 1

C PLOT THE CURVE

CALL DATPLOT (TYPE, NOTES, CPERSG (IDX, NCOMP), XSCALE, 15, COM (NCOMP),
   \ DATA (NSTY), DATA (NSTX), NPTS, 1, 1, 1, 1, TOPG (IDX),
   \ REP, ABSEP)

C PLOT THE BASELINE.

CALL PASEL (TXOFF (1, IDX, MOD (NCOMP - 1, 3) + 1),
   \ RANGE2 (1, IDX), NN (IDX))
IF THIS WAS THE SIXTH COMPONENT, GET A NEW RANGE ARRAY, OTHERWISE
GET A VCO ARRAY FOR THE NEXT COMPONENT

150 IF(LAST) GO TO 10
IORG=IXY
L=0
GO TO 20

STACK ARRAY TOO SMALL

970 WRITE(6,7000)
GO TO 999

NORMAL COMPLETION

980 WRITE(6,5000)
GO TO 999

PREMATURE END OF FILE ("TAPE" INPUT)

990 WRITE(6,6000) TYPE
999 CALL FLOATND
RETURN

1000 FORMAT(27(1X,11A4/))
2000 FORMAT('0',2A8,' RECORD SKIPPED')
3000 FORMAT('0',2A4,' RECORD READ')
4000 FORMAT('1LABEL="',A8,'"/' OFREQ. = ',F5.1,' MHZ.'/
 . '0FIRST POINT=',I4/ '0# OF POINTS=',I4/
 . '0',10F10.3/99(1X,10F10.3/))
5000 FORMAT('1NORMAL END OF JOB')
6000 FORMAT('2END OF FILE OCCURRED WHILE ATTEMPTING TO READ ',
 . '2AR, ' RECORD')
7000 FORMAT('3*** INSUFFICIENT SPACE ON STACK ***')

END
SUBROUTINE LUNIN (DATA, IDATA, *, *)

READ LUNAR DATA TAPE

TAPES DATA ARRAYS

REAL*4 DATA (1)
INTEGER*4 IDATA (1)

CHARACTER DATA

REAL*8 TYPE 1 (11) / 'LABEL', 'MODE', 'TEMPERAT',
     'TRANSMIT', 'CALIBRAT', '1 MHZ.', '2 MHZ.',
     '4 MHZ.', 'F MHZ.', '16 MHZ.', '32 MHZ.' /

REAL*8 TYPE 2 (6) / '
     'TON', 'RANGE', 'V. C. C.' /

REAL*8 RUN, SITE, DIRECT, FORREV, TITLE (11), TYPE (2)

INDICES TO TYPE ARRAYS

INTEGER*2 TDIM (3, 17) / 1, 1, 1, 1, 2, 1, 1, 3, 2,
     6, 4, 3, 6, 5, 4, 1, 5, 6,
     6, 6, 6, 1, 7, 5, 6, 7, 6,
     2, 8, 5, 12, 9, 6, 4, 0, 5,
     24, 0, 6, 8, 10, 5, 40, 10, 6,
     13, 11, 5, 78, 11, 6 /

LOGICAL*4 FIRST, LAST

TYPE CODE RETURNED

INTEGER*2 ITYPE (2)

COMMON BLOCK FOR RETURNED DATA

COMMON /LUNDAT/ TITLE, RUN, SITE, DIRECT, FORREV, TYPE,
     ITYPE, N, FIRST, LAST
RECORD COUNTERS

INTEGER IBLK /17/, IRRC /78/

BEGIN EXECUTABLE CODE

RESET RECORD COUNTER

FIRST = .FALSE.
IRRC = IRRC + 1
IF (IRRC .LE. TIDX(1, IBLK))
   .
   IF (TIDX(1, IBLK) .LT. 1)
      IRRC = 1
   ELSE
      FIRST = .TRUE.
      IBLK = IBLK + 1
      IF (IBLK .GT. 17)
         TDLK = 1
      TTYPE(1) = TIDX(2, IBLK)
      TTYPE(2) = TIDX(3, IBLK)
      TTYPE(1) = TYPE1(TTYPE(1))
      TTYPE(2) = TYPE2(TTYPE(2))

SELECT APPROPRIATE RECORD TYPE

10 IF (IBLK - 2) 100, 200, 300

HEADER RECORD

100 READ (4, 1000, END=390) RUN, SITE, DIRECT, FORREV, TITLE, N
GO TO 999

MODE RECORD

200 READ (4, 2000, END=395) (IData(I), I=1, N)
GO TO 999

ALL OTHER TYPES

300 READ (4, 3000, END=395) DATA(I), I = 1, N
GO TO 999

END OF FILE CONDITIONS

PREDICTABLE

( LABEL RECORD EXPECTED )
( => BEGINNING OF A NEW RUN )
SUBROUTINE LUNIN2(DATA, IDATA, *, *)

READ LUNAR DATA TAPE
TAPE DATA ARRAYS

REAL*8 DATA(1)
INTEGER*4 IDATA(1)

CHARACTER DATA

REAL*8 TYPE1(11) / 'LABEL', 'MODE', 'TEMPERATURE', 'TRANSMIT', 'CALIBRAT', '1 MHZ', '2 MHZ', '4 MHZ', '8 MHZ'.

END
REAL*8 TRUE, PERIOD OFF, RANGE, V.C.O.

REAL*8 RUN, SITE, DIRECT, FORREV, TITLE(11), TYPE(2)

INDICES TO TYPE ARRAYS

INTEGER*2 TIDX(3,17) / 1, 1, 1, 1, 2, 1, 1, 3, 2, 2,
6, 4, 3, 6, 5, 4, 1, 6, 5,
6, 6, 6, 1, 7, 5, 6, 7, 6,
2, 8, 5, 12, 8, 6, 4, 9, 5,
24, 9, 6, 8, 10, 5, 48, 10, 4,
13, 11, 5, 78, 11, 6 /

LOGICAL*4 FIRST, LAST

TYPE CODE RETURNED

INTEGER*2 ITYPE(2)

COMMON BLOCK POP RETURNED DATA

COMMON /LUNDAT/ TITLE, RUN, SITE, DIRECT, FORREV, TYPE,
ITYPE, 4, FIRST, LAST

RECORD COUNTERS

INTEGER*4 IBLK /17/, TREC /78/

BEGIN EXECUTABLE CODE

RESET RECORD COUNTER

FIRST=.FALSE.
IFREC = IRBC + 1
IF(IFREC .LE. TIDX(1, IBLK))
  GO TO 10
ELSE
  IFREC = 1
  FIRST=.TRUE.
  IBLK = IBLK + 1
  IF(IBLK .GT. 17)
    IBLK = 1
  ITYPE(1) = TIDX(2, IBLK)
  ITYPE(2) = TIDX(3, IBLK)
TYPE(1) = TYPE1(ITYPE(1))
TYPE(2) = TYPE2(ITYPE(2))

SELECT APPROPRIATE RECORD TYPE

10 IF( IBLK = 2 ) 100, 200, 300

HEADER RECORD

110 READ(4, 1000, END=990) (TDATA(I), I=1, 207)
GO TO 990

MODE RECORD

200 READ(4, 2000, END=995) ( TDATA(I), I=1, )
GO TO 990

ALL OTHER TYPES

300 READ(4, 3000, END=995) ( TDATA(I), I = 1, )
GO TO 990

END OF FILE CONDITIONS

PREDICTABLE
(LABEL RECORD EXPECTED)
(=> BEGINNING OF A NEW RUN)

990 RETURN 1

UNEXPECTED
(NON-LABEL RECORD EXPECTED)
(=> MIDDLE OF A RUN)

995 WRITE(6, 4000) TYPE, IREC
RETURN 2

RETURN DATA

999 LAST=.FALSE.
IF( IREC .GT. TIDX(1, IBLK)) LAST = .TRUE.
RETURN

1000 FORMAT(27(11A4))
2000 FORMAT(5 (20016) )
3000 FORMAT(5 (20016,1) )
4000 FORMAT('*** END OF FILE FOUND WHILE ATTEMPTING TO READ ')
SUBROUTINE LUNIT3 (DATA, IDATA, *, *)
READ LUNAR DATA TAPE
Tape data arrays
REAL*4 DATA(1)
INTEGER*4 IDATA(1)
Character data
REAL*8 TYPE1(11) / 'LABEL', 'MODE', 'TEMPERAT',
TRANSMIT', 'CALIBRAT', '1 MHz.',
'2 MHz.', '4 MHz.', '8 MHz.',
'16 MHz.', '32 MHz.' /
REAL*8 TYPE2(6) / 'ION', 'RANGE', 'V. C. O.' /
REAL*8 RUN, SITE, DIRECT, FORREV, TITLE(11), TYPE(2)
Indices to type arrays
INTEGER*2 TIDY(3,17) / 1, 1, 1, 1, 2, 1, 1, 2, 2, 3, 2,
6, 4, 3, 6, 5, 4, 1, 6, 5,
6, 6, 6, 1, 7, 6, 6, 7, 6,
2, 8, 5, 12, 4, 6, 4, 9, 5,
24, 9, 6, 8, 10, 5, 48, 10, 6,
13, 11, 5, 78, 11, 6 /
LOGICAL*4 FIRST, LAST
Type code returned
INTEGER*2 ITYPE(2)

COMMON BLOCK FOR RETURNED DATA

COMMON /LUNDAT/ TITLE, RUN, SITE, DIRECT, FORREV, TYPE, ITYPE, N, FIRST, LAST

RECORD COUNTERS

INTEGER*4 IBLK /17/, IREC /78/

BEGIN EXECUTABLE CODE

RESET RECORD COUNTER

FIRST=.FALSE.
ITYPE = IREC + 1
IF(ITYPE .LE. TIDX(1, IBLK))
   GO TO 10
ELSE
ITYPE = 1
FIRST=.TRUE.
ITYPE = IBLK + 1
IF(TIDX(3, IBLK) .EQ. 5)
   IBLK = IBLK + 1
IF(ITYPE .GT. 17)
   IBLK = 1
ITYPE(1) = TIDX(2, IBLK)
ITYPE(2) = TIDX(3, IBLK)
TYPE(1) = TYPF1(ITYPE(1))
TYPE(2) = TYPF2(ITYPE(2))

SELECT APPROPRIATE RECORD TYPE

10 IF(ITYPE - 2) 100, 200, 300

HEADER RECORD

100 READ(2,1000,END=990) (IDATA(I), I=1, 297)
GO TO 999

MCDF RECORD

200 READ(2,2000,END=995) (IDATA(I), I=1, N)
GO TO 999

ALL OTHER TYPES
300 READ(2,1000,END=395) (DATA(I), I=1, N)
GO TO 999
C
END OF FILE CONDITIONS
C
PREDICTABLE
( LABEL RECORD EXPECTED )
( => BEGINNING OF A NEW RUN )
C
990 RETURN 1
C
UNEXPECTED
( NON-LABEL RECORD EXPECTED )
( => MIDDLE OF A RUN )
C
995 WRITE(6,4000) TYPE, TPCC
RETURN 2
C
RETURN DATA
C
397 LAST=.FALSE.
IF(_TPCC .EQ. Tidy(1, TALK)) LAST = .TRUE.
RETURN
C
1000 FORMAT(27(11A4))
2000 FORMAT(5(206ED.1))
3000 FORMAT(5(206F6.1))
4000 FORMAT('0999 END OF FILE FOUND WHILE ATTEMPTING TO READ '1',
* 288 '' RECORD ', ]3)
C
END

*********************************************************************~********
* * *
ODCINT
* *
* ***FUNCTION ODCINT(I)
C
LOGICAL*1 FIRST / .TRUE. /
REAL*4 TIME(500), ODC(500)
C
C
IF(.NOT. FIRST) GO TO 100
FIRST = 'FALSE.
N = 0
20 READ(6, 1000, END = 30) TT, OFF, OLR
N = N + 1
TIM(N) = TT
ODC(N) = 0.5 * (OFF + OLR)
GO TO 20
C
30 NN = N / 5
IF(MOD(N, 5) .NE. 0) NN = NN + 1
DO 50 I = 1, 44
IF(MOD(I - 1, 50) .EQ. 0) WRITE(6, 2000)
JJ = 4 * NN + I
IF(JJ .GT. N) JJ = JJ - NN
WRITE(6, 3000) (TIME(II), ODC(II), II = I, JJ, NN)
50 CONTINUE
WRITE(6, 4000)
C
100 IF(T .LE. TIME(1)) GO TO 300
IF(T .GT. TIME(N)) GO TO 400
DO 200 I = 2, N
IF(T .GE. TIME(I)) GO TO 200
ODCINT = ODC(I - 1) + (ODC(I) - ODC(I - 1))
* (T - TIME(I - 1)) / (TIME(I) - TIME(I - 1))
GO TO 500
200 CONTINUE
C
300 ODCINT = ODC(1)
GO TO 500
C
400 ODCINT = ODC(N)
C
500 RETURN
C
1000 FORMAT(3F10.0)
2000 FORMAT('NAVIGATION DATA ', ' ', 'TIME OD. CNT.',
4(F14X, 'TIME OD. CNT.'), ') ')
3000 FORMAT(1X, 2F10.1, 4(8X, 2F10.1))
4000 FORMAT('')
C
END
SUBROUTINE PLINIT (NAME)
C
PLOTTING INITIALIZATION AND SETUP
C
REAL*4 NAME(4), INIT*/.TRUE./
LOGICAL ZIP/* .TRUE. /
REAL*8 CODE/*'SGS1410'*/, SETUP (5) /*'BLANK'*/
DATA LIMIT, PLTLEN, PAGWID/30, 20, 11. /
NAMLST /PLTID/ INIT, CODE, SETUP, LIMIT, PLTLEN, PAGWID, ZIP
NAMLST /PLECHO/ LIMIT, PLTLEN, PAGWID, ZIP
COMMON /PLTCON/ IL, IT, I2
IT=12
LSET=0
READ (5, PLTID)
NAME (4) = INIT
IF (SETUP (1) .NE. 'BLANK') LSET=40
IZIP=-1822
IF (ZIP) ZIP=-IZIP
CALL PLTSET (LIMIT, SETUP, LSET)
CALL PLOTST (NAME, 16, CODE, IZIP)
CHKE : (/PLECHO)
CALL PLTPAG (PAGWID)
CALL PLTXMM (PLTLEN)
RETURN
END
SUBROUTINE SYMBOL (X, Y, Z, IBCP, ANGLE, N)
C
****** NOTE ****** USE THIS SUBROUTINE ONLY IN PRODUCTION; REMOVE FOR
CALL SYMBOL (X, Y, Z, IBCP, ANGLE, N)
RETURN
END

SUBROUTINE RTPLOT
C
REAL*4 SR (3008), ST (3008), VR (2565), VT (2565)
COMMON / BLOCK / SR, ST, VR, VT, SCALE
INITIALIZE THE PLOTTTER SOFTWARE AND LOCAL VARIABLES.
(THIS IS REQUIRED LATER, FOR DRAWING THE TIME AXIS.)

CALL PLINIT('QQGP.RANGES', 'Q')
SC = 1. / SCALE
TORG = AINT(MAX1(ST(3088), VT(2565)) * SC) + 1.
T = AINT(MIN1(ST(1), VT(1)) * SC)
TS = T

DRAW THE RANGE AXIS.

CALL PLOT(0., TORG - T, 3)
P = AINT(MAX1(ST(3088), VR(2565)) * SC) + 1.
CALL SYMBOL(R, TORG - T, 0.07, 6, -90., -2)
N = IFIV(N) - 1
Y = R

AND LABEL IT.

DO 20 T = 1, N
   X = X - 1.
   CALL SYMBOL(X, TORG - T, 0.07, 13, 0., -1)
   CALL NUMBER(X - .14, TORG - T + .07, 0.07, X * SCALE, 0., -1)
20 CONTINUE

CALL SYMBOL(R * .6, TORG - T + .2, 14, 'RANGE (METERS)', 0., 14)

IDENTIFY THE TWO PLOTS: SEP IS A SOLID LINE; VLRI IS SOLID
AND MARKED BY A SYMBOL AT EVERY 100TH POINT

CALL PLOT(R - 2., TORG - T - 2.4, 3)
CALL PLOT(R - 2., TORG - T - 3.4, 2)
CALL SYMBOL(R - 2.07, TORG - T - 3.6, .14, 'SEP', -90., 3)
CALL SYMBOL(R - 2.2, TORG - T - 2.4, .07, 0.0, 0., -1)
CALL SYMBOL(R - 2.2, TORG - T - 2.9, .07, 0.0, 0., -2)
CALL SYMBOL(R - 2.2, TORG - T - 3.4, .03, 0.0, 0., -2)
CALL SYMBOL(R - 2.27, TORG - T - 3.6, .14, 'VLRI', -90., 4)

LABEL THE TIME AXIS.

CALL SYMBOL(-.34, (TORG - T) * .5, .14, 'TIME (SECONDS)', -90., 14)
N = IFIV(TORG - T) - 1
DO 40 T = 1, N
   T = T + 1.
   CALL SYMBOL(0., TORG - T, .07, 13, 90., -1)
   CALL NUMBER(-.14, TORG - T + .14, .07, T * SCALE, -90., -1)
40 CONTINUE
AND THEN DRAW IT.

CALL SYMBOL(0., TORG - T - 1., .07, 0, 180., -1)
CALL PLOT(0., TORG - TS, 2)

MOVE TO THE FIRST SEP POINT, AND THEN DRAW THE LINE.

CALL PLOT(SR(1) * SC, TORG - ST(1) * SC, 3)
DO 60 I = 2, 3088
   CALL PLOT(SR(I) * SC, TORG - ST(I) * SC, 2)
60 CONTINUE

PLOT THE VLBI DATA WITH A SYMBOL AT EVERY 100'th POINT.

CALL SYMBOL(VR(1) * SC, TORG - VT(1) * SC, .03, 0, 9., -1)
DO 80 I = 2, 2564
   IF(MOD(I, 100) .EQ. 1) GO TO 70
   CALL PLOT(VR(I) * SC, TORG - VT(I) * SC, 2)
   GO TO 80
70 CALL SYMBOL(VR(I) * SC, TORG - VT(I) * SC, .03, 0, 9., -2)
80 CONTINUE

CALL SYMBOL(VR(2565) * SC, TORG - VT(2565) * SC, .03, 0, 9., -2)
CALL PLOTND
RETURN
END

*****************************************************************************
***
**
*****************************************************************************

SUBROUTINE SEPLOT(TITLE, NOTES, CPERSG, YSCALE, YSCALE, COMP, II, IF, MIN,
* INDEX, D, K1, K2, LT1, K2, LT2, SITE, PUN, FPRO, REP, ANSREP)

PLOT OF EITHER THEORETICAL OR EXPERIMENTAL SEP DATA.
WRITTEN BY J. J. PROCTOR, SPRING 1973. UNIVERSITY OF TORONTO.

INPUT:
TITLE = PLOT TITLE (16 DIGITS)
NOTES = ADDITIONAL NOTES (32 DIGITS)
CPERSG = CURVES PER GRAPH (<= 6)
YSCALE = NUMBER OF INCHES PEP 20.0 WL (6.1' IS STANDARD)
YSCALE = NUMBER OF INCHES PER CURVE FOR LINEAR PLOTS (< 5.),
   = DB PER INCH FOR DB PLOTS (=> 5.)
COMP = COMPONENT LABEL - A 3-DIGIT INTEGER:
C  FIRST DIGIT  1=F, 2=H;
C  SECOND DIGIT  1=RHO, 2=PHI, 3=ZED;
C  THIRD DIGIT  1=BROADCAST, 2=ENDFILE
C
C  H = FIELD-STRENGTH ARRAY
C  R = RANGE ARRAY (IN WL)
C  NIN = DIMENSION OF H AND R
C  INDEX = INDEXING THROUGH H AND R ARRAYS (USUALLY =1)
C
C  REAL*4 K1,K2,LT1,LT2
C  INTEGER*4 TITLE(4),NOTES(3)
C  INTEGER*4 COMTAB(7) /'E', 'H', '736', '224', '269', 'PHI', 'ZED'/
C  INTEGER*4 CTR/0/,GCTR/0/,CPPPG,COMP
C  LOGICAL*4 DATOLD,DATNEW
C  DIMENSION H(NIN),R(NIN)
C  INTEGER*4 LABELS(3)
C  INTEGER*4 FXA(3) /'RHO', 'PHI', 'ZED' /
C  RETURN
C
C
C..ENTRY POINT FOR THEORY CURVES
C  ENTRY THEPLOT(TITLE,NOTES,CPPPG,XXSCALE,YSCALF,COMP,H,F,NIN,INDEX,
C    *    D,K1,LT1,K2,LT2)
C  DATNEW=.FALSE.
C  GO TO 2
C
C
C..ENTRY POINT FOR DATA TYPE CURVES
C  ENTRY DATPLOT(TITLE,NOTES,CPPPG,XXSCALE,YSCALF,COMP,H,F,NIN,INDEX,
C    *    SITE,RUN,FREQ,REF,ADSREF)
C  DATNEW=.TRUE.
C
C  2  CTR=CTR+1
C
C..IF THIS IS THE FIRST CURVE ON THE GRAPH, PLOT GRAPH OUTLINE
C  IF(CTR.EQ.1) GO TO 10
C
C..TEST FOR A FULL GRAPH
C  IF(CTR.EQ.1.E,CPPPG) GO TO 70
C
C..FULL-GRAPH LOGIC
C  CTR=1
C  CALL PLOT(XXX+4,20,0,-3)
C
C..GRAPH OUTLINE
C  10  GCTR=GCTR+1
C
C..SET RANGE AND FIELD STRENGTH ARRAY DIMENSION ON INDEX BOUNDARY
\[ XAX = XAX - XSPACE \]

35 CALL SYMBOL(XAX, 0., 05, 13, 0., -2)

C

C..NUMBER FIRST HALF OF X-AXIS
DNUM = IRIST-3.9
IF(XSCALE.GT.10.) DNUM = -199.9
XNUM = -XSPACE-.05
DO 36 I=1,1HALF
TF(XSCALE.LT.10.) DNUM = DNUM + 4.
TF(XSCALE.GE.10.) DNUM = DNUM + 200.
XNUM = XNUM + XSPACE
36 CALL NUMBER(XNUM, -.15, .07, DNUM, 0., -1)

C

C..LABEL X-AXIS
CALL SYMBOL(XLABEL, -.3, 1, 6, RANGE, 0., 6)
TF(XSCALE.GE.10.) CALL SYMBOL(999., 999., 1, 200., 0., 2)
IF(XSCALE.LT.10.) CALL SYMBOL(999., 999., 14, 41, 0., -1)

C

C..NUMBER SECOND HALF OF X-AXIS
DO 37 I=1HALF,NUMP
IF(XSCALE.LT.10.) DNUM = DNUM + 4.
IF(XSCALE.GE.10.) DNUM = DNUM + 200.
XNUM = XNUM + XSPACE
37 CALL NUMBER(XNUM, -.15, .07, DNUM, 0., -1)

C

C..LABEL Y-AXIS
IF(YSCALE.GT.5.) GO TO 38
CALL SYMBOL(-.15, 4.5, 1, 6, HLINEPAB, 90., 6)
GO TO 39
38 CALL NUMBER(-.15, 4.5, 1, YSCALE, 90., -1)
CALL SYMBOL(999., 999., 1, 3H DB, 90., 3)
CALL SYMBOL(-.1, 4.25, .06, 13, 90., -1)
CALL SYMBOL(-.1, 4.27, .06, 180., -1)
CALL SYMBOL(-.1, 5.23, .04, 6, 0., -2)
CALL SYMBOL(-.1, 5.25, .06, 13, 90., -1)
CALL SYMBOL(-.1, 6.0, 1, 7, HREF. AT, 90., 7)
CALL NUMBER(-.15, 6.8, 1, ABSREF, 90., 1)
CALL SYMBOL(999., 999., 1, 4H DBM, 90., 4)

C

C..END OF GRAPH VARIABLE SET-UPS
39 DATOLD=.NOT.DATNEW
CPFPG=MINO(6,CPFPG)
SHIFT=6./(CPFPG-1.)
ORIGIN=6.+SHIFT
GO TO 71

C

C

C..ENTRY POINT FOR PLOTTING A CURVE
70: N=((MIN-1)/INDEX)*INDEX+1
    NFIRST=N
C
C..SPF THE ORIGIN FOR THIS CURVE
71: ORIGIN=ORIGIN-SHIFT
    IC1=5+MOD(COMP,10)
    IC2=2+MOD(COMP/10,10)
    IC1=COMP/100
    LABELS(1)=COMTAP(IC1)
    LABELS(2)=RAXA(IC2-2)
    LABELS(3)=COMTAP(IC3)
    WRITE(6,90050)FREQ.LABELS
90050 FORMAT(*PLOTTING',5E1,3Y,2A2,2A4)
C
C..FIND MAXIMUM AND MINIMUM FIELD STRENGTH VALUES
    YMAX=H(I)
    YMNT=YMNX
    DO 97 I=1,N,INDEX
        YMNT=AMTN1(H(I)),YMNX)
    97 YMAX=AAMAX1(H(I)),YMAY)
C
C..TEST FOR LINEAR PLOTS
    IF(VSCALE.LT.5.)GO TO 700
C
C..CONVERT DP VALUES TO INCHES AND ZERO LOW VALUES
    DO 93 I=1,N,INDEX
    93 H(I)=H(I)/VSCALE
    GO TO 96
C
C..CONVERT LINEAR VALUES TO INCHES
700: HDELTA=VSCALE/YMAX
    DO 701 I=1,N,INDEX
    701 H(I)=H(I)*HDELTA
C
C..PLOT THE CURVE
99 CALL PLOT(-.05,REF/VSCALE+ORIGIN,3)
    CALL PLOT(.05,REF/VSCALE+ORIGIN,2)
    CALL PLOT((R(1)-TRIST)*XSCALD,H(1)+ORIGIN,3)
    NST=INDEX
    DO 900 I=NST,N,INDEX
    900 CALL PLOT((R(I)-TRIST)*XSCALD,H(I)+ORIGIN,2)
C
C..IF FIRST CURVE ON GRAPH, PLOT *COMP* AND *MAX* HEADINGS
    YYNO=H(N)+ORIGIN
    YYHD=YYNO+.15
    IF(CTR.FQ.1)CALL SYMBOL(XXX+.035,YYHD,.07C,4HCMP,0.,4)
C
C..PLOT COMPONENT AND MAXIMUM
CALL SYMBOL(XXX+1.015,YYN0,070,COMTAB(IC1),0.,3)
CALL SYMBOL(XXX+1.505,YYN0,070,K1,0.,2)
CALL SYMBOL(XXX+1.925,YYN0,070,LT1,0.,4)
CALL SYMBOL(XXX+2.555,YYN0,070,K2,0.,2)
CALL SYMBOL(XXX+2.975,YYN0,070,LT2,0.,3)

C C
C .CURVE LABELLING TESTS FOLLOW
IF (DATNEW) GO TO 40
C
C .THEORY CURVE
IF (.NOT.DATOLD) GO TO 41
C
C .THEORY HEADINGS
CALL SYMBOL(XXX+1.015,YYN0,070,32HDEPTH X1 LT1 L2 172
* 0.,32)
C
C .THEORY VARIABLES
41 CALL NUMBER(XXX+1.015,YYN0,070,D 0.,3)
CALL NUMBER(XXX+1.505,YYN0,070,K1 0.,2)
CALL NUMBER(XXX+1.925,YYN0,070,LT1 0.,4)
CALL NUMBER(XXX+2.555,YYN0,070,K2 0.,2)
CALL NUMBER(XXX+2.975,YYN0,070,LT2 0.,3)

C 40 CONTINUE
999 DATOLD=DATNEW
RETURN
ENTRY BASER(H, R, NTY)
DO 90100 I = 1, NIN
CALL SYMBOL((I-1)*ST) XSCALD, U(I)/YSCALD+ORIGIN, C7, 11, C.,-1)
90100 CONTINUE
RETURN
END

************************************************************************
************************************************************************
************************************************************************
************************************************************************
LOGICAL FUNCTION STOPT*1 (I, IPP)
C
C RETURNS .TRUE. IF THE LRV WAS STOPPED DURING THE LP-4 TURN, .FALSE. OTHERWISE. (THIS DECISION IS BASED ON THE VALUES PLACED IN THE ARRAY "H" BY THE CALLING ROUTINE.)
C
C

INTEGER*2 B(6)
INTEGER*2 C(6) / 33, 33, 20, 20, 7, 7 /
C
COMMON /BOUNDS/ B
C
C
J = 13 * T + C(LFP)
STOP = ( J .GT. B(2)
* .AND. J .LT. B(3) )
* .OR. ( J .GT. B(4)
* .AND. J .LT. B(5) )
RETURN
END

*****************************************************************************

TXOSTAT
*****************************************************************************

C

PROGRAM TO COMPARE TRANSMITTER-OFF DATA
WITH APPROXIMATE LRV SPEED.

C ONE INPUT CARD IS REQUIRED, CONTAINING SIX INTEGER VALUES IN
FORMAT 615; THESE VALUES SHOULD BE THE SAME AS THE "BOUNDS" FOR
THE 32.1 MHZ. INPUT TO ANTEMIAD.
C
C REAL*4 SST(3), SMO(3), SSTSO(3), SMOSO(3)
REAL*4 RANGE(140, 6), SPEED(140, 6), TXOFF(140, 6, 3)
REAL*4 TXOPTS(140, 6, 3), SMOFS(3), SSTFS(3)
REAL*4 SSTEOS(3), SMOSO(3)
INTEGER*4 NTXOFF(6, 3), NR(3), INDEX(140)
INTEGER*2 ROUNDS(5)
LOGICAL*1 STOPT
LOGICAL*1 SWITCH
C
COMMON /BOUNDS/ BOUND
C
CALL PLOTST('00GP.JCR.TXOFF ', 16, 'PGS1410 ')
C
READ(3)
READ(3)
READ(3)
READ(3) TXOFF, NTXOFF
READ(3) RANGE, NR
READ(3) SPEED
READ(5, 3000) BOUND

*LOOP THROUGH FREQUENCIES*

DO 100 IFR = 1, 6
   IFPFO = 2 ** (IFR - 1)
   I = 0
   N = 0
   SWITCH = .FALSE.
   NST = 0
   NMO = 0
   DO 5 J = 1, 3
      SST(J) = 0.
      SMO(J) = 0.
      SMOFS(J) = 0.
      SSTFS(J) = 0.
      SSO(J) = 0.
      SSOS(J) = 0.
      SSTFSO(J) = 0.
      SMOFSO(J) = 0.
   5 CONTINUE
   WRITE(6, 1000) IFPFO
   LIN = 0
   20 I = I + 1
   IF (RANGE(I, IFR) .GT. 1667.) GO TO 50
   N = N + 1
   IF (RANGE(I, IFR) .EQ. 513.) AND .NOT. STOPT(I, IFR) GO TO 20
   DO 23 J = 1, 3
      TXOFF(I, IFR, J) = 10. ** (.05 * TXOFF(I, IFR, J))
   23 CONTINUE
   CONTINUE
   WRITE(6, 2000) RANGE(I, IFR), SPED(I, IFR),
                  (TXOFF(I, IFR, J), J = 1, 3),
                  (TXOFFS(I, IFR, J), J = 1, 3)
   IF (SPEED(I, IFR) .EQ. 0.) GO TO 30
   NMO = NMO + 1
   DO 25 J = 1, 3
      SMO(J) = SMO(J) + TXOFF(I, IFR, J)
      SMOFS(J) = SMOFS(J) + TXOFFS(I, IFR, J)
   25 CONTINUE
   GO TO 40
   30 NST = NST + 1
   DO 35 J = 1, 3
SST (J) = SST (J) + TXOFF (I, IFR, J)
SSTFS (J) = SSTFS (J) + TXOPS (I, IFR, J)

35 CONTINUE

40 IF (LIN .LT. 55) GO TO 20
GO TO 10

C DO A SORT ON THE APPROPRIATE SEGMENT OF THE SPEED ARRAY.
C SUBROUTINE BUBBLE RETURNS THE VECTOR INDEX CONTAINING
C INDICES TO THE DATA ARRAYS SUCH THAT IF I < J, THEN
C SPEED (INDEX (I), IFR) < SPEED (INDEX (J), IFR)
C

50 CALL BUBBLE (SPEED (1, IFR), INDEX, N)
C LIST THE SPEED VALUES IN ASCENDING ORDER WITH THE CORRESPONDING
C TXOFF VALUES FOR EACH ANTENNA.
C
LIN = 0
DO 60 J = 1, 3
SMO (J) = SMO (J) / NSM
SST (J) = SST (J) / NST
SMOPS (J) = SMOPS (J) / NSM
SSTFS (J) = SSTFS (J) / NST
60 CONTINUE

DO 80 I = 1, N
IF (LIN .EQ. 0) WRITE (6, 1010) IFR
IX = INDEX (I)
IF (RANGE (IX, IFR) .NE. 5139 OR. STOPT (IX, IFR)) GO TO 70
SPEED (IX, IFR) = -1.
GO TO 90
70 IF (SWITCH .OR. SPEED (IX, IFR) .EQ. 0) GO TO 75
DO 72 J = 1, 3
SSTSQ (J) = SQRT (SSTSQ (J) / (NST - 1))
SSTFSQ (J) = SQRT (SSTFSQ (J) / (NST - 1))
72 CONTINUE
WRITE (6, 4000) NST, SST, SSTFS, SSTSQ, SSTFSQ
LIN = LIN + 5
SWITCH = .TRUE.
75 CONTINUE

DO 79 J = 1, 3
IF (SWITCH) GO TO 77
A = TXOFF (IX, IFR, J) - SST (J)
B = TXOPS (IX, IFR, J) - SSTFS (J)
SSTSQ (J) = SSTSQ (J) + A * A
SSTFSQ (J) = SSTFSQ (J) + B * B
GO TO 79
77 A = TXOFF (IX, IFR, J) - SMO (J)
B = TXOPS (IX, IFR, J) - SMOPS (J)
SMOSQ (J) = SMOSQ (J) + A * A
SMOFSQ(J) = SMOFSQ(J) * B * B

CONTINUE
WHITE(6, 10) SPEED(I, IF),
      (TXOFF(I, IF), J), J = 1, 3),
      (TXOFFS(I, IF), J), J = 1, 3)
LIN = MOD(LIN + 1, 50)

CONTINUE
DO 85 J = 1, 3
SMOFSQ(J) = SORT(SMOFSQ(J) / (NMO - 1))
SMOFSQ(J) = SORT(SMOFSQ(J) / (NMO - 1))

CONTINUE
WHITE(6, 4010) NMO, SMO, SMOFS, SMOFSQ, SMOFSQ

PLOT TXOFF VS. SPEED

CALL TXPLOT(SPEED(1, IF), TXOFF(1, IF), 1), TXOFF(1, IF), 2),
       TXOFF(1, IF), 3), N, IFP3)

CONTINUE

CALL PLOTND

RETURN

FORMAT('1', I4, ' MHZ. -- LRV SPEED AND TXOFF DATA ORDERED BY IANCEF
      / 45X, 'TXOFF DR', 25X, 'TXOFF FIELD STRENGTH' /
      6X, 'RANGE', 10X, 'SPEED', 12X, 2( 'X', 11X, 'Y', 11X,
           'Z', 14X) / 2X )

FORMAT('1', I4, ' MHZ. -- TXOFF DATA ORDERED BY LRV SPEED' /
      47X, 'TXOFF DR', 25X, 'TXOFF FIELD STRENGTH' /

FORMAT(1X, F10.1, 5X, F10.4, 3X, 3(2X, F10.1), 3Y, 3(2X, 1PE10.3))

FORMAT(19X, F10.4, 3X, 3(2X, F10.1), 3X, 3(2X, 1PE10.3))

FORMAT(615)

FORMAT('0', I4, ' RECORDS WITH LRV STOPPED' /
      6Y, 'MEAN VALUES', 15X, 3(2X, F10.2), 3X, 3(2X, 1PE10.3) /
      6X, 'STANDARD DEVIATIONS', 7X, 3(2X, 1PE10.5),
      3X, 3(2X, 1PE10.3) / 2X )

FORMAT('0', I4, ' RECORDS WITH LRV MOVING' /
      6X, 'MEAN VALUES', 15X, 3(2X, F10.2), 3X, 3(2X, 1PE10.3) /
      6X, 'STANDARD DEVIATIONS', 7X, 3(2X, 1PE10.5),
      3X, 3(2X, 1PE10.3) )

END
SUBROUTINE TXPLOT(S, TXX, TXY, TXZ, N, IFH)

REAL*4 S(N), TXX(N), TXY(N), TXZ(N)
REAL*4 TXORG(I) / 0., 3., 6. /
REAL*4 SCALE
SCALE(ARGV) = .1 * (ARGV + 135.)

DRAW THE SPEED AXES

X = 6.
DO 5 I = 1, 3
   CALL PLOT(0., TXORG(I), 3)
   CALL SYMBOL(X, TXORG(I), .07, 6., -90., -2)
5 CONTINUE

AND LABEL THEM

DO 10 J = 1, 5
   X = X - 1.
   DO 8 J = 1, 3
      CALL SYMBOL(X, TXORG(J), .07, 13, 0., -1)
8 CONTINUE
   CALL NUMBER(X - .105, -.2, .07, X, 0., 1)
10 CONTINUE
   CALL SYMBOL(1.5, -.5, .14, 22HdRe SPEED (M. / Sec.), 0., 22)

DRAW THE DR AXES

DO 30 J = 1, 3
   CALL PLOT(0., TXORG(J), 3)
   CALL SYMBOL(0., TXORG(J) + 2.5, .07, 6, 0., -2)
30 CONTINUE

AND LABEL THEM

DO 20 I = 1, 3
   CALL SYMBOL(0., TXORG(J) + 3 - I, .07, 13, 90., -1)
   CALL NUMBER(-.13, TXORG(J) + 2.75 - I, .07, -105., -10. * I, 90., 1)
20 CONTINUE
30 CONTINUE
CALL SYMBOL (-.36, .3, .14, .22) TRANSMITTED-OFF (DEM.), 20, 22

LABEL THE GRAPH

CALL SYMBOL (3, 9.16, .14, FLOT (IFS), 0, -1)
CALL SYMBOL (399, 399, .14, 188 MHZ, APOLLO 17, 0, 18)

PLOT THE DATA POINTS

DO 40 I = 1, N
   IF (S(I) .LT. 0.) GO TO 40
   CALL SYMBOL (S(I), SCALE (TXY(I)) + 6, .07, 9, 0, -1)
   CALL SYMBOL (S(I), SCALE (TXY(I)) + 3, .07, 9, 0, -1)
   CALL SYMBOL (S(I), SCALE (TXZ(I)) + .07, 9, 0, -1)
40 CONTINUE

MOVE ON TO BEGIN A POSSIBLE NEW PLOT

CALL PLOT (8.5, 0, -3)

RETURN

END

******************************************************************************

* VIPIRT

******************************************************************************

C PROGRAM TO COMPARE VLBI DATA WITH SEP NAVIGATION DATA. VLBI DATA
C MAY BE EITHER HIGH- OR LOW-SPEED; SEP DATA ARE OBTAINED FROM THE
C 16 MHZ, RANGE ARRAY ON FILE SCI?, AND CORRESPONDING TIMES ARE
C GENERATED INTERNALLY.
C
C ONE NAMLIST (CNTL) CONTROL CARD IS REQUIRED:
C
C TO - TIME (GM) OF FIRST 16 MHZ, RANGE POINT.
C RO - DISTANCE OF FIRST 16 MHZ, RANGE POINT FROM
C SEP TRANSMITTER.
C STAT - (BOOLEAN) OUTPUT COMPARISON STATISTICS.
C PLOT - (BOOLEAN) PLOT RANGES FOR VLBI AND SEP VS. TIME.
C SCALE - INDICATES NUMBER OF METRES AND 100-WAVELENGTH INTERVALS
C PER INCH ON THE PLOT; NOT REQUIRED IF PLOT = FALSE.
ALSO A PLTID NAMFLIST CARD IS REQUIRED BY PLINIT (IF PLOT = TRUE).

C
INTEGER N, M, S
REAL X(5), Y(5)
REAL SR(3088), ST(3088), VP(2565), VT(2565)
REAL TO / 1452. /, RO / 0. /
LOGICAL STAT / .TRUE. /, PLOT / .FALSE. /
COMMON / BLOCK / SR, ST, VP, VT, SCALE
NAMFLIST / CNTL / TO, RO, STAT, PLOT, SCALE

C
SET AND READ CONTROL PARAMETERS, AND SKIP OVER UNWANTED SEP DATA.

C
SCALE = 500.
READ (5, CNTL)
DO 10 I = 1, 71
READ (1, 1000)
10 CONTINUE

C
READ 16 MHz. RANGE DATA.

C
K = 1
L = 386
DO 20 I = 1, 9
READ (3, 2000) (SR(J), J = K, L)
K = K + 386
L = L + 386
20 CONTINUE

C
ADJUST SEP RANGE AND D3 DATA USING SUPPLIED PARAMETERS

C
DO 30 I = 1, 3088
ST(I) = .81 * (I - 1) + TO
SR(I) = ST(I) + RO
30 CONTINUE

C
READ VERT DATA IN GROUPS OF ONE TIME AND FIVE X-Y PAIRS, CONVERT
C
TIME TO SECONDS AND X-Y PAIRS TO RANGES, AND STORE.

C
DO 50 I = 1, 2565, 5
READ (4, 100) N, M, S, (X(J), Y(J), J = 1, 5)
T = S + 60 * (M + 60 * H)
DO 40 J = 1, 5
II = J - 1
JJ = I + II
VT(JJ) = T + II
VR(JJ) = SQRT(X(J) * X(J) + Y(J) * Y(J))
40 CONTINUE

C
50 CONTINUE
C
IF(PLOT) CALL RTPLOT
IF(.NOT. STAT) GO TO 99
JST = 1
C
SKIP ALL VLBI TIMES WHICH ARE LESS THAN THE FIRST SEP TIME.
C
DO 60 J = 1, 2565
   IF(VT(J) .GE. ST(J)) GO TO 65
   JST = JST + 1
60 CONTINUE
C
65 LIN = 0
   E = 0.
   S = 0.
   I = 1
C
FOR EACH VLBI TIME-RANGE PAIR
C
DO 90 J = 1, 2565
C
FIND THE PAIR OF SEP TIMES WHICH BRACKET THE VLBI TIME.
C
DO 70 K = I, 3088
   IF(VT(J) .LT. ST(K)) GO TO 80
70 CONTINUE
GO TO 95
80 I = K
II = I - 1
C
COMPUTE AN INTERPOLATED SEP RANGE, AND THE DIFFERENCE BETWEEN
C IT AND THE VLBI RANGE, AND INCREMENT THE SUM OF DIFFERENCES AND
C THE SUM OF SQUARES OF DIFFERENCES.
C
   S = SR(IT) + (SR(I) - SR(IT)) * (VT(J) - ST(I))
   / (ST(I) - ST(IT))
   D = VP(J) - R
   F = E + D
   S = S + D * D
   IF(MOD(LIN, 50) .EQ. 0) WRITE(6, 200)
      WRITE(6, 200) VT(J), VR(J), R, D
   LIN = LIN + 1
90 CONTINUE
C
COMPUTE THE MEAN AND STANDARD DEVIATION.
C
95 F = F / LIN
   S = SORT(S / FLOAT(LIN - 1))
WRITE(6, 400) E, S

RETURN

100 FORMAT(3(I2,1X), 1X, 10F5.0)
200 FORMAT(''1''/''-1'', 33X, 'INTERPOLATED', 5X, 'DIFFERENCE' /
  7X, 'VIBI TIME', 5X, 'VIBI RANGE',
  6X, 'SEP RANGE', 7X, 'IN RANGE' / 1X )
300 FORMAT(10X, F6.0, 9X, F6.0, 8X, F7.2, 8Y, F7.2)
400 FORMAT('SUM(DIFF.) / N = ', F7.2 /
  'SQR(SUM(DIFF. ** 2) / (N - 1)) = ', F7.3)
1000 FORMAT(200A6, 186A6)
2000 FORMAT(200F6.1, 186F6.1)

END
SCI2B Plots
MEMORANDUM

July 29, 1974

TO: Distribution

PFCM: J. C. Pylaarsdam

SUBJECT: Modifications to data on tape SEP009

As described in Watts' memorandum of July 2, 1974, a processing error during generation of data tapes SEP007 through SEP010 resulted in the loss of small amounts of dR data for 4, 8.1, 16, and 32.1 megahertz. These losses lead to erroneous correlation of the dR data with the range information, which was processed correctly. This memorandum describes a procedure for producing a set of data which is correctly matched, by removing range data corresponding to the the dR data which were lost.

In the context of my report (Apollo 17 SEP Data Processing - July 1974) the processing is performed by program LUNACPY6, using file SCI2 as input. The modified data are designated as file SCI2M; this file is of exactly the same format as file SCI2, and contains all the same data, except for the changes described. File SCI2M is intended as a replacement for file SCI2 in any of the processing functions described in the report. (However, since some of the missing data occurred during the turn at FP-4, no processing by LUNAPLT4, LUNACPY4, or ANTENNAO was attempted, and none is recommended using the modified data.)

The following is a recapitulation of Watts' description of the error, including one item which was not explained in his memorandum.

For each frequency, \( M \times 400 \) data words were assembled in memory for each component, where \( M = 1, 1, 2, 4, 8, \) and 13 for frequencies of 1, 2.1, 4, 8.1, 16, and 32.1 megahertz respectively. Then \( M \) blocks of length 387 words were defined, with origins of 1,
401, ..., (M - 1) * 400 + 1; the origins should have been 1, 388, ..., (M - 1) * 387 + 1. What is not made clear by Watts' memorandum is that the first M words of the first block did not contain data, and were discarded; hence M blocks, each of length 386 words, were written on the output tape. Then output block i would contain

(a) the last 387 - (M - i + 1) words of block i, followed by

(b) the first M - i words of block i + 1.

N. B.

(1) part (b) above does not apply to output block M (M - i = 0, and block i + 1 does not exist).

(2) if the above-mentioned origins had been defined correctly, then the above definition of output blocks would yield the desired result.

(3) In the cases where M = 1, the data on the output file are correct.

(4) The last (M - 1) * 13 words of the last output block do not contain data.

The words which should have been used to assemble the output blocks are given in table 1; those which were used are given in table 2.

In the light of the above discussion the following procedure can be derived for matching the range data correctly to the erroneous IR data.

Having assembled the M blocks of length 386 in memory, define a set of "incorrect" block origins corresponding to those used in Watts' processing; since the M words are no longer present at the beginning of the data, this series is now 1 - M, 401 - M, ..., 400 * (M - 1) + 1 - M. Taking the length of these blocks as 387, a new set of M blocks, each of length 386, may be generated by selecting and
reassembling portions of the blocks as described in (a) above, and adding \(13 \times (M - 1)\) words of padding to the end of the last block.

The words used to assemble the blocks of modified range data are indicated in table 3.

A listing of program LUNACPY6 begins on page seven. Following the listing is an updated set of plots, designated SC12BM, produced by LUNAPLT5 from file SC12BM.
<table>
<thead>
<tr>
<th>Block</th>
<th>4 MHz</th>
<th>8.1 MHz</th>
<th>16 MHz</th>
<th>32.1 MHz</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3-388</td>
<td>5-390</td>
<td>9-394</td>
<td>14-399</td>
</tr>
<tr>
<td>2</td>
<td>389-774</td>
<td>391-776</td>
<td>395-780</td>
<td>400-785</td>
</tr>
<tr>
<td>3</td>
<td>777-1162</td>
<td>781-1166</td>
<td>786-1171</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>1163-1548</td>
<td>1167-1552</td>
<td>1172-1557</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>1553-1938</td>
<td>1558-1943</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>1939-2324</td>
<td>1944-2329</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>2325-2710</td>
<td>2330-2715</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>2711-3096</td>
<td>2716-3101</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td></td>
<td></td>
<td>3102-3487</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td></td>
<td></td>
<td>3488-3873</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td></td>
<td></td>
<td>3874-4259</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td></td>
<td></td>
<td>4260-4645</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td></td>
<td></td>
<td>4646-5031</td>
<td></td>
</tr>
</tbody>
</table>

Table 1 - Locations which should have been used to assemble blocks of dB data for file SCI2.
<table>
<thead>
<tr>
<th>Block</th>
<th>4 MHz.</th>
<th>8.1 MHz.</th>
<th>15 MHz.</th>
<th>32.1 MHz.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3-387</td>
<td>5-387</td>
<td>9-387</td>
<td>14-387</td>
</tr>
<tr>
<td></td>
<td>401</td>
<td>401-403</td>
<td>401-407</td>
<td>401-412</td>
</tr>
<tr>
<td>2</td>
<td>402-774</td>
<td>404-787</td>
<td>408-787</td>
<td>413-787</td>
</tr>
<tr>
<td></td>
<td>(775-787)</td>
<td>801-802</td>
<td>801-806</td>
<td>801-811</td>
</tr>
<tr>
<td>3</td>
<td>803-1187</td>
<td>807-1187</td>
<td>912-1187</td>
<td>1201-1205</td>
</tr>
<tr>
<td></td>
<td>1201</td>
<td>1201-1205</td>
<td>1201-1210</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>1202-1548</td>
<td>1206-1587</td>
<td>1211-1587</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(1549-1587)</td>
<td>1601-1604</td>
<td>1601-1609</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>1605-1987</td>
<td>1610-1987</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>2004-2387</td>
<td>2009-2387</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>2401-2402</td>
<td>2401-2407</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>2403-2787</td>
<td>2408-2787</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>2801</td>
<td>2801-2806</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>2802-3096</td>
<td>2807-3187</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>(3097-3187)</td>
<td>3201-3205</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>3206-3587</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>3631-3604</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>3605-3987</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>4001-4003</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>4004-4387</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>4401-4402</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>4403-4787</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>4801</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>4802-5031</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>(5032-5187)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 2 - Locations which were used to assemble blocks of dB data for file SJI2.
(Locations in parentheses contain meaningless information.)
### Table 3 - Locations used to assemble blocks of range data for file SCI2M. (Locations in parentheses contain padding.)

<table>
<thead>
<tr>
<th>Block</th>
<th>4 MHz</th>
<th>8.1 MHz</th>
<th>16 MHz</th>
<th>32.1 MHz</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1-385</td>
<td>1-383</td>
<td>1-379</td>
<td>1-374</td>
</tr>
<tr>
<td></td>
<td>399</td>
<td>397-399</td>
<td>393-399</td>
<td>388-399</td>
</tr>
<tr>
<td>2</td>
<td>400-772</td>
<td>400-783</td>
<td>400-779</td>
<td>400-774</td>
</tr>
<tr>
<td></td>
<td>(773-785)</td>
<td>797-798</td>
<td>793-798</td>
<td>788-798</td>
</tr>
<tr>
<td>3</td>
<td>799-1183</td>
<td>799-1179</td>
<td>799-1174</td>
<td>799-1174</td>
</tr>
<tr>
<td></td>
<td>1197</td>
<td>1193-1197</td>
<td>1188-1197</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>1198-1544</td>
<td>1198-1579</td>
<td>1198-1574</td>
<td>1198-1574</td>
</tr>
<tr>
<td></td>
<td>(1545-1583)</td>
<td>1593-1596</td>
<td>1588-1596</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>1597-1979</td>
<td>1597-1574</td>
<td>1597-1574</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2393-2394</td>
<td>2388-2394</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>2395-2779</td>
<td>2395-2774</td>
<td>2395-2774</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2793</td>
<td>2788-2793</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>2794-3088</td>
<td>2794-3174</td>
<td>2794-3174</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(3089-3179)</td>
<td>3186-3192</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td></td>
<td>3193-3574</td>
<td>3193-3574</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>3598-3591</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td></td>
<td>3592-3974</td>
<td>3592-3974</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>3988-3990</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td></td>
<td>3991-4374</td>
<td>3991-4374</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>4388-4389</td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td></td>
<td>4390-4774</td>
<td>4788</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>4789-5018</td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td></td>
<td>4789-5018</td>
<td>(5019-5174)</td>
<td></td>
</tr>
</tbody>
</table>
**LUNACPY6**

Program to generate a modified version of file #2, in which the range data corresponding to missing DR information have been deleted.

```c
INTEGER*4 MM(4) / 2, 4, 8, 13 / REAL*4 DATA(6000)

COPY ALL THE DATA WHICH REQUIRE NO MODIFICATION - I. E. THE LABEL RECORD THROUGH THE 2 MHZ. DATA.

DO 10 I = 1, 29
  READ (1, 1000) (DATA(J), J = 1, 579)
  WRITE (2, 1000) (DATA(J), J = 1, 579)
10 CONTINUE

LOOP OVER THE FOUR FREQUENCIES WHICH REQUIRE MODIFICATION.

DO 150 I = 1, 4
  M = MM(I)
  IORG = 1
  IEND = 386

READ THE M BLOCKS OF RANGE DATA INTO MEMORY.

DO 20 J = 1, M
  READ (1, 2000) (DATA(K), K = IORG, IEND)
  ICPG = IORG + 386
  IFND = IEND + 386
20 CONTINUE

ID IS INITIALIZED AS THE FIRST WORD OF THE FIRST GROUP OF 13 WORDS TO BE DELETED.

ID = 383 - M

MM1 IS THE NUMBER OF GROUPS OF 13 WORDS TO BE DELETED.

MM1 = M - 1

N IS THE NUMBER OF WORDS TO BE MOVED FROM THE BEGINNING OF BLOCK J + 1 TO THE END OF BLOCK J. (INITIALLY M - 1)
\[ V = \text{MM1} \]

Look over the set of 13-word groups.

\[ \text{DO 60 } J = 1, \text{MM1} \]

Transfer \( N \) words from block \( J + 1 \) to block \( J \).

\[ \text{DO 40 } K = 1, N \]
\[ \text{JD} = \text{JD} + K - 1 \]
\[ \text{JS} = \text{JD} + 13 \]
\[ \text{DATA(JD)} = \text{DATA(JS)} \]
\[ \text{CONTINUE} \]

For group \( J + 1 \), \( N \) is decremented by 1.

\[ N = N - 1 \]

The beginning of group \( J + 1 \) is 400 words from the beginning of group \( J \).

\[ \text{IL} = \text{ID} + 400 \]

\( \text{JD} \) is currently the index of the 36th word of output block \( J \); set \( \text{JS} \) to index the 1st word, and then write the block on the output file.

\[ \text{JS} = \text{JD} + 135 \]
\[ \text{WRITE}(2, 2000) (\text{DATA}(K), K = \text{JS}, \text{JD}) \]
\[ \text{WRITE}(6, 1800) (\text{DATA}(K), K = \text{JS}, \text{JD}) \]
\[ \text{CONTINUE} \]

Complete output block \( N \) by placing the last correct range value (containing 13 word \( N \))

\[ N = 366 \times M \]

in the \( MM \) locations beginning at location \( N + 1 \).

\[ \text{MM} = 13 \times \text{MM1} \]
\[ \text{DO 80 } J = 1, \text{MM} \]
\[ \text{JD} = \text{JD} + J \]
\[ \text{DATA(JD)} = \text{DATA(M)} \]
\[ \text{CONTINUE} \]

The first word of the output block is computed as above, and the blocks written on the output file.
C
JS = 131 - 325
C
137 F (J, 2000) (DATA(K), K = JS, JD)
C
137 F (J, 2000) (DATA(K), K = JS, JD)
C
REDEFINE N AS THE LAST VALID WORD (WITHIN THE LAST BLOCK)
OF DB DATA
C
N = 136 - NN
C
FOR EACH COMPONENT,
C
DO 140 IC = 1, G
C
READ M BLOCKS OF DB DATA.
C
DO 120 J = 1, M
C
READ (1, 2000) (DATA(K), K = 1, 796)
C
WRITE BLOCKS 1 THROUGH M - 1 ON THE OUTPUT FILE IMMEDIATELY.
C
110 IF (J .NE. K) GO TO 110
C
FILL THE LAST NN WORDS OF BLOCK K WITH TABBING
BEFORE WRITING IT ON THE OUTPUT FILE.
C
DO 100 K = 1, NN
C
JD = N + K
C
DATA(JD) = -135.
C
100 CONTINUE
C
110 CONTINUE
C
120 CONTINUE
C
140 CONTINUE
C
150 CONTINUE
C
RETURN
C
1200 FORMAT (12D8, 100F4, 170A8)
C
2300 FORMAT (200E6.1, 180F6.1)
C
3000 FORMAT (1C / ' ', 15Fr.1 / 25 (1X, 15Fr.1/))
END
TO: Distribution

FROM: J. C. Blyaardsam

SUBJECT: Comparison of SFP Range Data and Data from the VLBI Experiment

The VLBI data used for this study were obtained from tape number G2TMS (Goddard Space Flight Centre) as a set of x-y coordinate pairs; associated with the first pair in each group of five was a time (Greenwich Mean) expressed in hours, minutes, and seconds. These times were converted to seconds, and times for the four remaining pairs in each group were generated by adding values of one, two, three, and four seconds to the initial value. The x-y pairs were converted to distances.

The SFP data consisted of the sixteen megahertz range values from tape number SFP009. The time (GM) corresponding to the first datum was set at 1427.4 seconds; times for succeeding values were generated by repeated addition of 0.81 second, the time interval between samples.

For each VLBI range datum, a corresponding SFP range value was computed by linear interpolation, using the two arrays of time data; the difference between these ranges was calculated as the VLBI range minus the interpolated SFP range. The differences obtained are plotted in figure 1; in this plot, the data are grouped into ten-second intervals, and the maximum and minimum differences over each interval are displayed. The mean of these differences was found to be -23.94 metres (indicating a lag in the VLBI data), and the standard deviation of the differences was effectively zero, considering the limits of precision of the calculations.

A more direct visual comparison is provided by figure 2, in which both sets of data are plotted on a single set of axes.
MEMO TO: D.W. Strangway
FROM: James Rossiter
RE: SEP Antenna Patterns Reconstructed from EP-4 Turn

Introduction

During EVA II of Apollo 17, the Lunar Roving Vehicle (LRV) made a complete 360° turn around the deployment site of Seismic Explosion Package 4 (EP-4), about 525 m. from the SEP transmitter site. This turn provided an opportunity to estimate the directional characteristics of the three orthogonal SEP loop receiving antennas, as mounted on the LRV, over a dielectric earth.

Ideally, any signal received by the $H_r$ (radial) antenna should smoothly interchange with the signal received by the $H_\phi$ (tangential) antenna as the LRV goes through each 90° of the turn. The $H_z$ (vertical) signal should remain constant throughout the turn. If the turn were of zero radius, any deviations from the above could then be attributed to interference by the Rover and/or mount.

Data Reduction

Data were taken from Watt's lunar tape SEPDO9, which included the error noted in his memo (July 2, 1974). Details of the organization of the data after their removal from tape are given by Rylaarsdam ("Apollo 17 SEP Data Processing", July 1974). The following steps were then taken in order to
get antenna pattern plots.

(1) The values of all 36 components over the entire range of the turn (493 to 538 m. from the SEP site) were removed from Rylaarsdam's file SCI3 (which included no pre-processing of the turn by Watts). These values were stored in a new file called EP4, listed using program LUNACPY4, and plotted using program LUNAPLT4 (and routine GAPLOT). The points were spaced according to the time scale implicit in the data; an example is shown in Figure 1.

(2) Odometer counts (one count = 0.49 m. of wheel turn), received from both the right front and left rear wheels of the LRV, are available for each 1.02 seconds of the traverse (see memo by Redman, July 16/73). Ideally, given a high density of odometer pulses, and assuming no wheel slippage or sticking, LRV speed and rate of turn could be completely determined. However, the coarseness of the odometer pulses prevented this detailed reconstruction (see Figures 2 and 3). Antenna patterns plotted using the navigation data (see Rylaarsdam's report) were far less consistent from component to component than were those plotted assuming the LRV speed to be constant during the non-stationary portions of the turn.

Therefore a template with three pairs of bounds was set up to separate the points that were recorded while the LRV was actually turning from those recorded while the LRV was either
on its traverse leg or stopped. By using components that had a good deal of character, the times during which the LRV was stationary were easily distinguished on the set of plots like Figure 1. The end-points of the turn were more difficult to estimate, and consistency from component to component was the only criterion available.

Unfortunately the 16 and 32 MHz data could not be used to construct the template, since both of these frequencies contain a drop-out due to Watt's spooling error during the turn.

(3) The total angle through which the LRV turned was calculated in the following way:

Assume there is no net slippage or sticking of either wheel over the turn. Then, for each wheel,

\[ c = n\pi r, \]  

where \( c \) = the circumference of the turn made by the wheel

\[ r = \text{total number of counts} \times 0.49 \text{ meters}; \]

\( n\pi = \text{number of radians of the turn}; \) and,

\[ r = \text{radius of the turn (m.)}. \]

Therefore,

\[ n\pi = \frac{(c_o - c_i) \times 0.49}{r_o - r_i} \]

where \( c_o - c_i = \) the difference in odometer counts between the two wheels over the turn (see Figure 3); and,

\[ r_o - r_i = \text{the distance between the two wheels} \]

\[ = 1.73 \text{ m. (Apollo 17 LRV Manual)}. \]
For the turn, $c_o - c_i = 21\pm2$, 
therefore, $n\pi = 6.0\pm0.5$ radians.

Although this is evidently a fairly crude estimate, it indicates that the turn was close to $360^\circ$.

(4) The portions of file EP4 determined by step 2 to be actually in the turn proper were plotted as a function of angle using program ANTENNAO (and routine ANTPAT). A complete set of patterns is shown in Figure 4. The angles start along the negative x-axis, and increment uniformly clockwise over $2\pi$ radians.

Discussion

Basically the plots show the expected type of behaviour. The vertical components are fairly smooth (except those which have very low signals), with few lobes, while the $H_\phi$ and $H_\theta$ components do interchange. It must be pointed out that the 16 and 32 MHz plots do not contain any angular correction for the missing points, and this will certainly create some amount of distortion in the patterns.

Several of the plots do not align well with the north-south and east-west axes - e.g. 4MHz $H_\phi$ endfire. This is possibly due to an incorrect choice of either the bounds or of the total angle.

The major obstacles in obtaining good patterns from this
analysis are as follows:

(1) Very poor sampling for the lower frequencies (as few as 8 points for a complete turn at 1 and 2 MHz), giving virtually no resolution of any lobe structure.

(2) Non-constant range of the LRV through the turn for the higher frequencies. The turn had a diameter of approximately 15 m - or about 1.5 wavelengths at 32 MHz. Therefore the signal received during the turn could have changed substantially quite independently of LRV rotation.

(3) Lack of direct knowledge of a) the exact position of the turn in the data stream, b) the complete angle of rotation, or c) the speed of rotation. These could only be estimated, and compatibility from component to component used to improve the estimate. The problem was particularly severe because of the drop-outs at 16 and 32 MHz.

(4) Unknown source signal. It is evidently not a plane wave, since the SEP transmitter was used. Reflections and scattering from the subsurface may well have had important influences on the type of pattern.

Conclusions

Considering the above problems, the amount of distortion of the patterns is within the error of the analysis. $H_z$ appears non-directional at all frequencies; $H_\rho$ and $H_\phi$ interchange smoothly through the turn. It is therefore not possible to attribute any large degree of interference to the
LRV or mount. This does not imply that such interferences did not exist - only that this analysis was not able to detect it.

It would probably be worth while in the future to analyse the data without the drop-outs at 16 and 32 MHz. These frequencies have both the highest resolution and are most likely to be susceptible to interference from the LRV or mount. A systematic attempt to use a number of different possible bounds and rotation angles may locate the turn in the data stream better. If so, such a study could be more definitive.
LRV WHEEL - EP-L TURN.

Bounds used.
Figure 4. SEP antenna radiation patterns from EP-4 turn for all components. Since choice of the exact position of the turn is somewhat arbitrary (see text), these patterns are only approximate.

16 and 32 MHz each suffer a 13-point data dropout during the turn in these plots. This has not been corrected for in any way.
4.0 MHz. Hzrd
16.0 MHZ. HP 6650