Series analyzed
QC
807.5
.U6
W6
no.
200
c.2

NOAA Technical Memorandum ERL WPL-200

BETA: A PROGRAM FOR CALCULATING AND ARCHIVING BACKSCATTER PROFILES

A. Weickmann

Wave Propagation Laboratory
Boulder, Colorado
June 1991

noaa NATIONAL OCEANIC AND ATMOSPHERIC ADMINISTRATION / Environmental Research Laboratories

QC
807.5
.U6
W6
no.200
c.2

NOAA Technical Memorandum ERL WPL-200

BETA: A PROGRAM FOR CALCULATING AND ARCHIVING BACKSCATTER PROFILES

A. Weickmann

Cooperative Institute for Research in Environmental Sciences
University of Colorado
Boulder, Colorado

Wave Propagation Laboratory
Boulder, Colorado
June 1991

LIBRARY
AUG 9 1991
N.O.A.A.
U.S. Dept. of Commerce

UNITED STATES
DEPARTMENT OF COMMERCE

Robert A. Mosbacher
Secretary

NATIONAL OCEANIC AND
ATMOSPHERIC ADMINISTRATION

John A. Knauss
Under Secretary for Oceans
and Atmosphere/Administrator

Environmental Research
Laboratories

Joseph O. Fletcher
Director

NOTICE

Mention of a commercial company or product does not constitute an endorsement by NOAA Environmental Research Laboratories. Use for publicity or advertising purposes of information from this publication concerning proprietary products or the tests of such products is not authorized.

For sale by the National Technical Information Service, 5285 Port Royal Road
Springfield, VA 22161

QC
807.5
.U6
W6
no. 200
c. 2

Contents

Abstract 1

1.0 Introduction 1

2.0 Scientific Basis 4

3.0 Beta Program 9

3.1 Program Input 10

3.2 Program Output 12

3.3 Questions and Answers 16

4.0 Typical BETA Execution 21

5.0 Program Structure 40

6.0 Archive Structure 54

References 58

Appendix A Software Listings A-1

Appendix B Common Block Listings B-1

Appendix C Recent Software Changes C-1

BETA : A Program for Calculating and Archiving Backscatter Profiles

A. Weickmann

ABSTRACT

BETA is a FORTRAN program that processes signal data from the NOAA Coherent Lidar and creates and archives aerosol backscatter profiles. This document focuses on the program software and its execution.

1. INTRODUCTION

This document supersedes a previous memorandum written by Post et al. (1983), describing the BETA program. The introduction and scientific basis borrow heavily from that document but because the actual BETA software and execution are vastly different, new documentation is warranted. Over the years, the BETA program has been the collaborative effort of many software designers.

NOAA has developed a pulsed, infrared Doppler laser-radar (lidar) system (Post and Cupp, 1990) for measuring atmospheric winds to ranges of 10-20 km. Pulses of 3 μs duration are transmitted up to 20 Hz pulse repetition frequency (PRF), and a very small amount of energy from the outgoing pulse is scattered by aerosol particles back toward the lidar. By measuring the

frequency shift (Doppler shift) of this backscattered signal, one can deduce the velocity component of the aerosols which are in line with (radial to) the laser beam. Since the aerosols are very small, they move "exactly" with the winds; thus we obtain a measurement of the radial wind component. By scanning the lidar beam in a conical pattern, it is possible to deduce the average wind speed and direction within the pattern.

The whole process depends on having enough aerosols to backscatter energy from the transmitted pulse. Therefore, there is a genuine need to document the backscattering properties of the aerosols and their statistical fluctuations. The backscattering coefficient, beta, is a measure of the aerosol's backscattering ability. By definition, it is the effective total cross-sectional area in square meters of all the particles in 1 m^3 of air, divided by 4π. The division by 4π assumes the energy is scattered isotropically into a 4π steradian solid angle (sphere). Therefore, beta has units of $m^2/m^3/sr$ or $m^{-1}\ sr^{-1}$.

The most efficient way to measure beta versus altitude, or the beta profile, is to point the lidar beam vertically (elevation angle = 90° or zenith angle = 0°), but any elevation angle is allowed.

The BETA program uses recorded lidar signals and calculates the beta profile that had to exist in order to give rise to the

signals. Either raw signal data or moment data (zeroth moment of the frequency spectrum) can be used in the calculations, but raw data are the primary input. Moment data require a different signal decoding routine, but otherwise program functions remain the same. The program code is listed in Appendix A and is available from the Wave Propagation Laboratory, Atmospheric Lidar Group, on magnetic tape or PC-DOS format floppy disks.

2.0 SCIENTIFIC BASIS

The lidar's raw data-taking system digitizes an analog voltage originating from the heterodyne signal detector and records this signal on 9-track digital magnetic tape. The digitizer has 8-bit resolution, a minimum sampling interval of 100 ns (maximum rate of 10 MHz) and a sampling capacity of 2048 consecutive points. The data are complex; that is, for each data point, two numbers are generated (R and Q) which are 90° out of phase (hence they are in "quadrature" and represent the real and imaginary parts of a complex number).

Generally, signal power is computed by squaring both R and Q and taking the sum of the squares. However, due to infrequent digitizer problems, we have developed the ability to analyze signal power from either the R or the Q channel. In this case, signal from the single channel is squared and doubled. Amplifiers in the analog signal chain saturate at high signal levels, so nonlinear signal correction factors have been derived from measured input versus output curves. Third or fourth order polynomial correction factors are applied to recorded signal power values above 2500. Individual correction factors have been computed for $(R^2 + Q^2)$, $2R^2$, and $2Q^2$. For the single channel data, two sets of correction factors have been computed. The correction can be applied to either unaveraged data or time averaged data.

The gain of the receiving chain is adjusted so that when no signal is present, the mean background noise level uses only 1-3 bits of the dynamic range of the digitizer, leaving 7-5 bits for

the (additive) signal level. The maximum unsaturated signal plus noise is 128^2 or 16,384.

We know the relationship between the volume backscatter coefficient, β, and the system signal to noise ratio, SNR. It is

$$\beta(R) = \frac{8*h\nu*bw*R^2*snr(R)*A_1}{\pi*E_1*A_3*ej*C*D^2*A_2*E_2(R)*A_4}$$

A_1 = calibration factor $(10^{(dbcal/10)})$

A_2 = shot noise factor $(1.0 - 10^{-shtdat/10})$

A_3 = optical loss factor

A_4 = round trip atmospheric absorption factor, f(R)

E_1 = heterodyne detector quantum efficiency

E_2 = receiver efficiency, f(R)

h = Plank's constant (J s)

υ = frequency (s^{-1})

bw = bandwidth (s^{-1})

R = range (m)

snr = signal to noise ratio, f(R)

ej = pulse energy (J)

C = speed of light (ms^{-1})

D = telescope diameter (m)

The calibration factor, A_1, is computed using dbcal, an independent measurement of how well the system performs with

respect to theory. SHTDAT in A_2 is the system shot noise at the time of the data. The measured roundtrip optical power loss factor is 0.5307 and the heterodyne detector efficiency is assumed to be 0.40 for all wavelengths. The diameter of the telescope mirror is 0.28 m. The pulse energy is measured manually for each data run by averaging the laser's output power for several seconds and dividing by the PRF.

The round-trip atmospheric absorption parameter accounts for molecular absorption of CO_2 radiation and can be calculated by two methods. First, one of five pre-computed seasonal models accounting for the absorption of radiation by H_2O, CO_2, N_2O, CH_4, O_3, O_2, CO, and the H_2O continuum can be used. The second method calculates absorption from a radiosonde profile and accounts for H_2O, CO_2, and H_2O continuum. Fewer molecules are used in the interest of program speed and because the omitted molecules contribute a small fraction to the total absorption. Both methods use the AFGL Hitran Database (Rothman et al., 1987). Total absorption values are computed for round trips through 1 km layers up to 30 km. The absorption coefficients above 10 km in altitude become nearly constant, despite changes in density, owing to pressure and temperature effects on the CO_2 line shape and strength. When using a radiosonde profile, absorption coefficients between the maximum radiosonde height and 30 km are set equal to zero, so that round trip integrated absorption above the maximum height remains constant.

Average noise is computed as the average power within a

selected range or noise window. The noise window is an altitude range of at least 1 km where the received power is at a consistent minimum. This occurs at an altitude above any clouds where there are no signals from aerosols. The average signal power is the sum of the signal values within a selected range resolution, usually 0.5 km, divided by the number of data points within the range. The range resolution is centered about the range R. The SNR at each range R is then the average signal for the range resolution minus the average noise divided by the average noise.

Additional factors of noise beta and beta ratio describe the relationship between the pure noise region and backscatter. The minimum valid SNR, SNRMIN, is computed and replaces SNR in the beta equation to calculate the noise beta at every range. Beta values greater than the noise beta are accepted as valid. Beta ratio is the ratio of noise beta to beta in decibels (dB).

SNRMIN is computed as the quality factor threshold value, QUAL, divided by the square root of the number of records.

$$depsam = dgtzrt/bwmhz$$

$$qual = \sqrt{1/[(\frac{ninc}{depsam})-1]+1/[(\frac{nnois}{depsam})-1]}$$

$$snrmin = qual/\sqrt{nrec}$$

dgtzrt = digitizer rate

bwmhz = bandwidth in MHz

nnois = number of data points within the noise window

ninc = number of data points within the altitude resolution increment

nrec = number of records processed

These equations arise because there is confidence that a signal is truly present only if its level is above the root mean square statistical fluctuation expected in simultaneously determining both its level and the noise level, given a finite number of samples of signal and noise. Also confidence grows with the square root of the number of times the procedure is repeated (average), hence with the square root of NREC.

3. BETA PROGRAM

The backscatter program, BETA, has evolved over the last several years in conjunction with the CO_2 lidar system. Since the lidar is an operating research instrument, changes in hardware due to improvements or failures have resulted in varying techniques of data analysis. The software reflects lidar hardware changes by incorporating options such as a data deglitcher (filter), decoding of data from single or both digitizer channels, and data storage modifications on magnetic tape. The program can process raw or moment data, collected from 1986 to the present, but it is primarily used with raw data.

BETA has been developed on a Data General MV8000 using DG/UX (Unix), has an executable size of 280,400 bytes and is linked to an in-house tape library and a modified Tektronix Plot 10 graphics library. Half-inch 9 track magnetic tape is used for input and the output devices are a Digital LN03 Plus laser printer/plotter, a Printronix printer, and a graphics terminal running the Tektronix 4010 emulator.

BETA is an interactive program, designed to minimize keystrokes and operator error by assigning default values where appropriate. These default values are based on typical data collection methods. When prompted by a question with a default, a carriage return results in using the default. This decreases run time and helps to eliminate user mistakes. If an unacceptable

input is sensed, the prompting question is repeated.

3.1 Input

The information necessary to execute BETA is obtained from lidar signal data, data collection parameters, and predetermined system and atmospheric parameters. The data collection parameters include data shot noise, pulse energy, wavelength, bandwidth, beam elevation angle, physical location, and attenuation, if used. The predetermined information includes system dB down from expected (from system calibrations), receiver efficiency profile, and atmospheric absorption profile.

The lidar signal data are stored on magnetic tape, which holds a variable number of data files, and each file contains a variable number of records. For raw data, the typical vertical profile contains 1000 records or pulses. Each pulse generally has 2048 16-bit data values when the profile extends to near 30 km. Otherwise, the pulse length is set to 1024 or approximately 15 km. The BETA program can process a maximum of 5000 records.

The shot noise for the data is calculated using

$$\text{data shot noise} = 10.0 * \text{LOG}(\text{tel}/\text{det}) \qquad (3.1)$$

where 'det' is the average noise in detector blocked file and 'tel' is the average noise in telescope blocked file.

The pulse energy is recorded at the start of the data run. It

is a measure of the average energy in Joules for the outgoing pulse during the data run.

The wavelength in microns, bandwidth in MHz and the elevation angle in degrees are all entered by the lidar operator.

An attenuator is used when the signal is too high, preventing amplifier or digitizer saturation. The attenuation factor, in dB, is recorded and used in the program to normalize the attenuated data.

An ASCII text file containing lidar experiment locations and the corresponding altitudes above sea level is used by the beta program so that all output data is referenced to sea level.

The CO_2 lidar is calibrated off a hard target on a regular basis. The calibration information is stored in an archive for automatic retrieval by this and other programs. A record of equipment changes and the date on which they were implemented is also maintained. If an equipment change will affect a calibration, an equipment record is entered into the calibration archive. The calibration date closest to the date and time of the data is used for analysis, provided no equipment changes have intervened. The beta equation requires a calibration factor that is computed using dB down from expected, the difference between expected (theoretical) and measured SNR for the calibration target. These

values of SNR are retrieved from the calibration archive.

Receiver efficiency, that is, how efficiently a signal mixes with the local oscillator, or is sensed on the detector, changes with range so range profiles have been computed for wavelengths of 10.591 and 9.25 μm from 0 to 30 km at 0.5 km increments. If the data wavelength falls between these values, a receiver efficiency profile is linearly interpolated. The profile is then linearly interpolated in range to compute the beta value at a selected altitude.

Atmospheric absorption profiles for the five seasonal models at wavelengths 10.591, 9.569, and 9.282 μm are stored in ASCII file ABSORP.DAT. These profiles, extending from 0 to 30 km ASL at 1 km increments, have been computed for layers (0-1, 1-2 km ...) using the AFGL Hitran Database, wavenumber, and the seasonal atmospheric models. An absorption profile can also be created during the execution of the BETA program from an input ASCII radiosonde file, provided the file contains pressure in millibars, height in meters, and temperature and dew point in degrees Celsius. A maximum of 100 vertical levels are allowed. The file is read as a free format and must have pressure, height, temperature, and dew point in order, on each line.

3.2 Output

The only automatic output from BETA is the BETA.PRINT file,

which, at a minimum, contains data processing information. If requested, it also lists vertical profiles of altitude versus beta, snr, noise beta and beta ratio.

If output to plotter was selected, the file 'qqplot' will be created and output to the laser printer. Modification of the Tektronix Plot 10 package allows simultaneous output to a graphics screen and a plot file. Other changes allow better text output on the plots.

The program outputs three types of plots:

Signal power versus time - An average signal value for each pulse/record is computed and plotted.

Altitude versus signal power - Altitude versus an average signal value for each point along the vertical pulse, averaged over all pulse/records. All altitude data are referenced to sea level.

Altitude versus beta - The beta value is computed using the equation in Section 2 and plotted against altitude at a resolution determined by the user.

If the profile is being archived, the altitude resolution is automatically set to 0.5 km. If the user selects the running average option, beta will be calculated and plotted at every point along the profile. If both archiving and running average are selected, two beta plots will be generated, one with beta values at every 0.5 km and another with values at each point. The dashed line on the plot is the profile of beta standard deviation. Beta values above this line are considered valid.

Table 1 lists the data output files. Other files are also created by BETA, but those pertain to the program execution, and discussed in Section 5.

Table 1. BETA Output Files

File Name	Contents
ASCII Text Files:	
BETA.PRINT	data processing information vertical profile
BETA.RAWDAT	average signal power at each point
LOOKABSP	interpolated absorption profile for radiosonde
ABSPSONDE	absorption information at radiosonde heights
'filename'	profile information same as archive
Plot File:	
qqplot	relative signal power vs time
	altitude vs relative signal power
	altitude vs backscatter (beta)
Binary Archive Files:	
'filename'.DRCT	unique profile information and pointers into main data file.
'filename'	main data file for profile information

'filename' - Data file name is selected by the user during program execution.

3.3 Questions and Answers

When executing the BETA program, the user is asked a series of questions. Most questions have obvious answers; there are some, however, that require some prior knowledge and experience working with the data. The following is a question and answer section for those less obvious questions.

Is this a multiple file run?

Usually we calculate beta profiles from one file at a time, but the program can produce beta profiles for individual files and then average all the profiles into one profile. The maximum number of files you can average is 10. If you answer YES, be prepared to input the file numbers of all the files to be used. They need not be contiguous, but they should be entered in ascending order.

Do you want calibration data from the archive?

The normal answer is YES and the calibration archive will be searched for the appropriate information. You will be asked to confirm the value of dB down. If you answer YES and the program cannot find a value for the data or if you answer NO, you get the next question.

Enter dB down from expected?

This value depends on how well you feel the lidar was performing. A good default value for the NOAA CO_2 lidar is 4.0 dB.

Enter shot noise for the data (dB)?

Refer to Section 3.1 for shot noise calculation. A good default value for the NOAA CO_2 lidar is 5 dB, but the value is highly dependent on the detector used.

Enter the energy per pulse for the data (J)?

This value is on the operator's log sheet, noted at the time the data was taken.

Default plot types and scale?

This question references the first two plots, signal versus time and altitude versus signal. "Average Plot" produces one plot of all records averaged together over time. "Single Plot" produces a single plot for every record. There are other programs that analyze single records so the "Single Plot" option is rarely used. "Linear Scale" plots the signal power on a linear as opposed to a dB scale. "Medium Scale" plots the signal power on a scale of 0-2000 as opposed to "Low" of 0-400 or "High" of 0-20000.

Do you want the beta profile ARCHIVED?

If the data are of good quality, you may want to store the profile in a data archive. The profile can later be accessed by other programs for statistical analysis. The archive structure is described in Section 6.

Is the value of PRF correct?

PRF is an acronym for Pulse Repetition Frequency. Check the value displayed above this question (from the file header) against the value written on the log sheet. Occasionally the PRF is changed for data collection and the tape header information is not updated to reflect the change. The value on the log sheet is correct.

Do you want the data decoded using: ?

R2Q2 decodes data from both the real and imaginary channels, R and Q (Section 2.). This is the customary method of decoding the data. However, there have been problems with the signal digitizer in the past, so we have the ability to decode data coming from only one channel. After the signal has been decoded, coefficients are applied to correct for saturation and the signal is then passed on to other parts of the program. The coefficients correct for nonlinearities in the signal which generally occur in values above 2500. The coefficients have been derived empirically using a least squares fitting routine with either third or fourth order polynomials.

Do you want to De-Glitch the data?

The old A-D digitizer, before 15 November 1990, produced periodic spikes in the data. The spikes are not representative of the data, so we identify them as bad points, remove them, and insert an average value. Bad points are identified as being greater than or equal to 10 times the average of the previous

points at that height. A minimum of four points are used in the average. A bad point is replaced by the average of the previous points.

Enter the altitude of the last clear air (km) on an even 0.5 km interval?

When archiving, the operator is asked if clouds were present. If there was a cloud, it is important to know where the cloud occurs in the profile because a cloud attenuates the signal returning from above the cloud. For this reason, we want to know the highest altitude, at an 0.5 km increment, that is not affected by the cloud; this is called the "last clear air". Remember, the beta value at the 0.5 km interval is calculated using data ± 0.25 km.

Enter the beginning, ending altitude for the noise window?

The noise window is chosen by examining the altitude versus signal power plot. Choose an altitude range where there is a constant minimum signal for at least 1 km. This generally occurs at the top of the profile, after any clouds, where the signal becomes just the noise. If you have chosen the noise window correctly, approximately 16 percent of the beta values in the pure noise region will be above the noise beta sensitivity line. After viewing the beta profile that results from a noise window selection, the operator may chose a different noise window and iterate until satisfied.

Is the data attenuated in this file?

Check the log sheet for an attenuation value in dB associated with the file; usually the answer is NO.

Choose the averaging method?

The beta values are calculated using an average of the points within the chosen altitude resolution. If you are archiving, the resolution is automatically set to 0.5 km and the beta values are calculated centered at standard 0.5 km increments (5.0, 5.5, 6.0, etc.). Two averaging methods are available: block and running. Both methods use an equal number of points on either side of the altitude.

The block average computes a beta value at selected heights in the following manner. The beta calculated for 5.5 km uses the data points from 5.25 to 5.75 km, and the beta at 6.0 km uses data from 5.75 to 6.25 km, etc. In practice, no data point is used twice.

The running average computes a beta value for every point in the vertical profile. A template corresponding to the altitude resolution in length and centered about the desired beta value, defines the data points used in the averaging. The template is moved along the profile to compute beta values at each data point.

Would you like to process another tape file?

You can skip forward or backward to other files on the tape. You can also process other records within the current file.

4. TYPICAL BETA EXECUTION

There are many options in the BETA program, but most data collection methods allow for a standard usage of the program. Usually one file is processed, calibration constants are retrieved from the archive, and the profile is printed and archived. Plots are output to both the plotter and the screen. The example program execution shown in this section is followed by the standard outputs. In addition, there are samples of the optional types of printer and plotter output. All samples are based on the same data file.

Several pieces of information necessary for program execution are obtained by examining the operator's log sheet (Figure 4.1), which is filled out as the data are taken. These are : file number, number of records, wavelength, bandwidth, elevation angle, prf, pulse energy, attenuation, and location.

Figure 4.2 is the signal power versus time plot, which immediately follows a pause in the execution while the data tape is accessed. This plot gives the user an indication of the smoothness or consistency of the data-taking process. If the system sensitivity changes, or clouds come and go during the file, such changes will be obvious. After the user views the plot and presses the carriage return, the altitude versus signal power plot appears (Figure 4.3). This plot needs to be examined carefully to pick a noise window altitude interval of at least 1 km. In this example, an altitude interval of 18-21 km appears to contain a consistent minimum signal. If the profile is to be archived, the plot must

also be examined for the altitude of the last clear air before a cloud. This value is chosen to the closest 0.5 km interval. In this case, the cloud occurs between approximately 10.2 km and 11.8 km. Therefore, the altitude of the last clear air was chosen as 9.5 km. In this case, 10.0 km is not correct because the beta computation includes signal data from 9.75 to 10.25 km.

Since the profile is being archived, the data averaging method chosen was block averaging. Figure 4.4 is a plot of the block-averaged data. The dashed line is the calculated noise beta at each 0.5 km altitude increment. The mimimum altitude of the data is determined by the altitude of the lidar above sea level plus the minimum effective range of the lidar which is 1.5 km for the standard pulse length. When archiving, an additional altitude increment may be added so that the mimimum profile altitude coincides with an even 0.5 km altitude.

If, after pressing the carriage return to erase the plot, the user elects to exit the program, the BETA.PRINT file will be output (Figure 4.5). This printout is a record of the input variables used for execution and contains noise data information, data quality information, the vertical profiles of altitude, beta, and snr, and other calculated parameters.

Figure 4.6 is an example of a running average beta profile using the same input data. The plot resolution is one beta value for each of the 2048 points in the vertical profile. The averaging filter or template is 0.5 km, so that each beta value, except at the ends of the profile, is an average of the calculated beta

values within 0.25 km either side of the current beta value. The beta values at the ends of the profile are not averaged.

The optional ASCII output file (Figure 4.7), contains the vertical profile of altitude, beta, snr, and noise beta. Additional information on line 1 is file number, date, time, start record, and number of records processed. Line 2 is wavelength, bandwidth, and beginning and ending altitude of the noise window. This file option has been created so that a user can download a profile to a floppy disk.

The final type of output the user can choose is the ASCII file BETA.RAWDAT (Figure 4.8). This file contains the average power values for every data point in the vertical profile. This data also corresponds to the altitude versus signal power plot, (Figure 4.3).

If a radiosonde profile is used to calculate the absorption profile, there are other output files, as shown in Table 2. These files are created so that the user can verify program input when it is derived from a radiosonde profile.

BETA9010001201 TAPE #1 B + C ALIB

1.1K ALI
2.5K NOISE WINDOW

Tape I.D. CAL 90 100 0 12:01 Clouds/Weather/Visibility cirrus + low fog ~~toward foot hills~~

Date(Z) APRIL 10 1990 Day(Z) 100 Temperature 35°F 12:12:00 →

Location TABLE MT 10.6μ GRATING ISOLATORS IN TABLE FLOATING Dewpoint 34°F

Crew DICK TUESDAY

Laser	Digitizer		Processor	
FOC	DEM GAIN	BW	GATE ATT	WIDTH
∞	300	10	25	2

P = .657 = 844 mb

AD9 = Cn²
AD7 = PRESS

FILE	TIME(Z)	# REC	SAMP PUL	M/R	AZ	EL	PUL AVG	PRF	SCAN	J PUL 4.9KV	NOTES
0	12:01:51	500	500	R	11.5	1.07		8		.53	13.72 DET BLOCK AVE NOISE 12.55 DC 1.20 } SHOT NOISE 6.45
1	12:03:36	"	20.48	"	"	"		"		.53	56.32 TELE BLOCK Sh.#: 613 .97 55.45
LOCAL 06:23 2	12:07:30	1000	"	"	90	90		"	B	.50	9-10K 5K +2.6°C
3	12:23:17	1000	500	"	202.7	.27		"	CALIB	.44	FOR TOWARD FOOT HILLS 31°F WET 32°F DRY 40 db a = 300 BW=10
	: :										≈ 3×10⁻¹⁵
4	12:30:09	"	"	"	"	"		"	"	.40	40db "
	: :										
5	12:37:18	"	"	"	"	"		"	"	.40	SUN RISE " "
	: :										+2.8°C 5×10⁻¹⁵ 34°F DRY 32°F WET
6	12:49:40	"	"	"	"	"		"	"	.43	MIRROR IN 40db BW=10
	: :										ISOLATORS OUT
7	12:52:19	"	"	"	"	"		"	"	.43	+2.8°C 40db MIRROR OUT NO TUNING
	: :										24K SHOTS SOME CIRRUS CLOUDS IN FOOT HILLS
8	13:04:47	"	"	"	"	"		"	"	.43	LOOKS CLEAR NEAR 40db WIND 2M/S @ 350° 2×10⁻¹⁴ Cn² 32.5°F 32.0°F
	: :										
9	13:14:08	"	"	"	"	"		"	"	.41	Cn² starting down 40db 5×10⁻¹⁵

Figure 4.1 Sample log sheet from experiment site.

```
(boris) : beta

   BETA program is running

 Enter the operator's initials (2 characters)
 ? : aw


 Are you using the TEK 4051? (Y or N)
  (default = N)
 ? :


 Plot to screen (0), plotter (1), or both(2)
 ? : 1

 Output diverted to laser plotter *

     ~~~ INITIALIZATION CONSTANTS ~~~
  (These inputs are static in this run.)

 Enter the magnetic tape unit number
 ? : 0

Opening tape drive /dev/rmt/c22d0m 5
** Tape /dev/rmt/c22d0m is now read only**


 Choose the lidar site from the following list
 by the corresponding number.  If the site is
 not on the list enter 0

  1 T22 (Table Mountain)              2 Edwards AFB
  3 San Clemente                      4 RL-3 Parking Lot
  5 BAO                               6 CP-3 (Brighton)
  7 Radio Building                    8 Stapleton
  9 Miller Watch,  WSMR              10 Brush Creek
 11 Texex                            12 Mesogamma 86 (Front Range)
 13 Mesogamma 86 (Phase B-BAO)       14 FIRE 86 (Oshkosh,Wi)
 15 Chinook 87 (Betasso)             16 Erie Site #1
 17 FIFE  (Manhattan, Ks)            18 LASBEX87 (Moss Landing, Ca)
 19 JPL 87 (Pasadena, Ca)            20 DBC 87  (RMA, Co)
 21 Trail Ridge, Co                  22 Mauna Loa
 23 MOFFETT FIELD, CA                24 grand canyon
 25 Battersby, Canada                26 Rocky Flats
? : 1
```

```
~~~ CHANGEABLE INITIALIZATIONS ~~~

Is this a multiple file run  (Y or N)
  (default = N)
? :


Input the requested information for each file.

Enter the file number
? : 2


Enter the starting record number
? : 1


 Enter the number of records to process
? : 1000


Would you like to :
        Continue with program  -  C
        Modify recent inputs   -  M
   (default = C)
? :


Positioning tape to requested file ...


Select the data wavelength :
          10.591 microns  -  A
           9.569 microns  -  B
           9.282 microns  -  C
                 other    -  D
 (default = A)
? :


Select the data bandwidth :
                10.0 MHz  -  A
                 5.0 MHz  -  B
                 3.0 MHz  -  C
                   other  -  D
 (default = A)
? :
```

```
Do you want calibration data from the archive
(otherwise manual input)   Y or N
 (default = Y)
 ? :


 The archive will be searched for the nearest
 date to the data.  The values will be displayed
 before proceeding.

 Searching calibration file /usr/local/database/CALARC.DRCT

 Calibration on :  4/13/90   1:11:29
 Calibration : dB down from expected = 3.20185

 Is this value ok  (Y or N) ?
 (default = Y)
 ? :


 Enter shot noise for the data (DB)
 ? : 6.13


 Choose an atmospheric absorption model
   0 = No correction for absorption
   1 = Tropical
   2 = Midlatitude Summer
   3 = Midlatitude Winter
   4 = Subarctic Summer
   5 = Subarctic Winter
   7 = Derive from Radiosonde Profile
 ? : 3


 Is the Elevation angle of the scanner 90 degrees
  (Y or N)     (default = Y)
 ? :


  Enter the energy per pulse for the data (J)
 ? : .5


  Enter the initial range in km
    (default = 1.581)
 ? :
```

```
The number of points per pulse is 2048.
 2048 points - approx 30 km range.
 1024 points - approx 15 km range.

 Enter the end range in km
? : 22


Would you like to :
        Continue with program  -  C
        Modify recent inputs   -  M
   (default = C)
? :


The default plot types and scale are as follows:

 Average Plots,  Linear Scaling,  Medium Scale
 -------         ------           ------
Are these the values you want (Y or N)
 (default = Y)
? :


~~~~~ Printer Outputs ~~~~~

Do you want a  RAW  data printout,  (Y or N)
 (default = N)
? :


Do you want a  BETA PROFILE  printout,  (Y or N)
 (default = Y)
? :


~~~~~ Archive Output ~~~~~

Do you want the beta profile  ARCHIVED  (Y or N)
 (default = N)
? : y


Enter the archive file name (max of 25 characters)
? : samplearc
```

```
Would you like to :
        Continue with program  -  C
        Modify recent inputs   -  M
   (default = C)
? :


File  2
on  4/10/90
at 12: 7:30


      Data Type:                                Raw
 1    Digitizer Rate (.08 - 10MHz)            10.00
 2    Delay to 1st sample (US)                 1.05
 3    Number of samples per pulse           2048.00
 4    Trigger Mode                               .00
 5    PRF                                       8.00
10    Number of records digitized.          1000.00

Is the value for PRF correct  (Y or N)
(default = Y)
? :


Do you want the data decoded using :
                     R2Q2  -  B
                       R2  -  R
                       Q2  -  Q
(default = B)
? :


Do you want to De-Glitch the data   (Y or N)
 (default = N)
? : y


The data tape is being accessed...
```

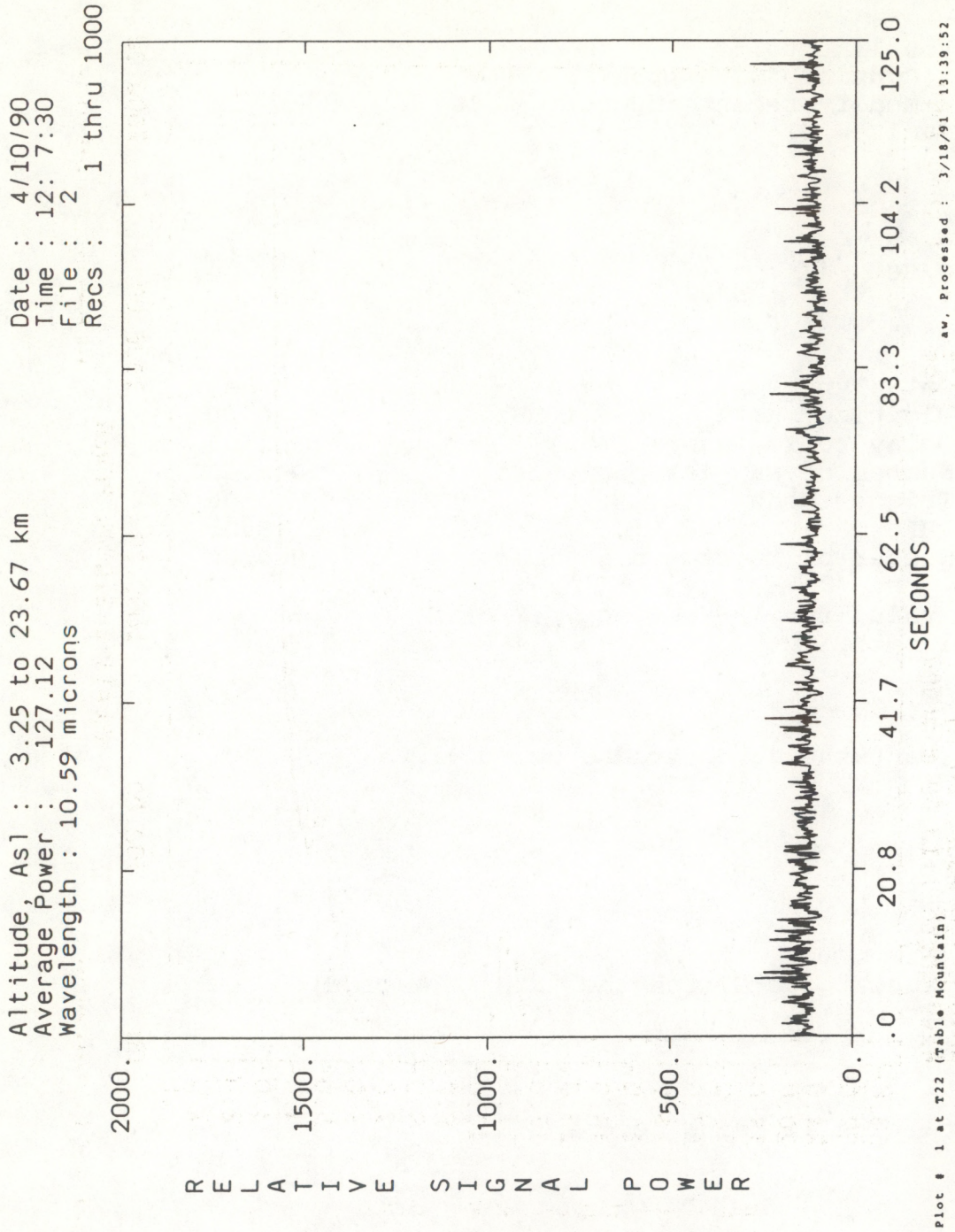

Figure 4.2 Signal power versus time. Scales are linear and medium (0-2000), and data is averaged for each pulse.

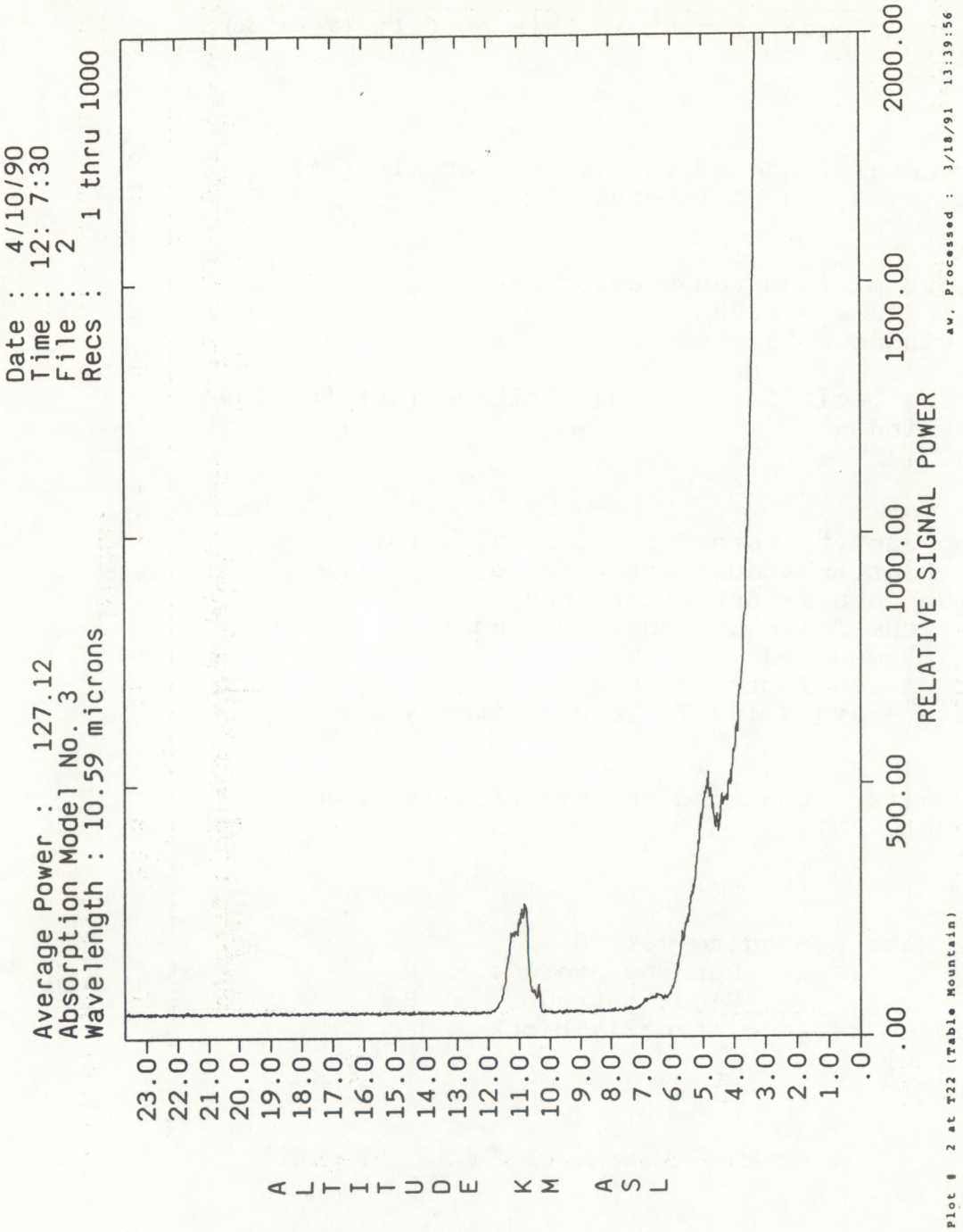

Figure 4.3 Altitude versus signal power. Scales are linear and medium (0-2000), and data is averaged over time.

```
Are there clouds present in this profile (Y or N)
 (default = N)
? : y


Enter the altitude of the last clear air (km)
  on an even 0.5 km interval.
? : 9.5

Data above minimum range occur at :
Start Altitude = 3.25
 End Altitude = 23.669

Enter the beginning, ending altitude (km) for the
 Noise Window
? : 18,21

  User :
 If you want to enter your own value for
       Unattenuated Average Noise
 and the data is not attenuated,
 answer the following questions as :
  Data attenuated ?      Y
  Attenuation factor ?   0.0
  Unatten. avg Noise ?   your desired value.


Is the data attenuated in this file (Y or N)
 (default = N)
? :


Choose data averaging method :
                  Running Average -  R
                  Block Average   -  B
(archives are automatically block
 averaged)
(default = B)
? :
```

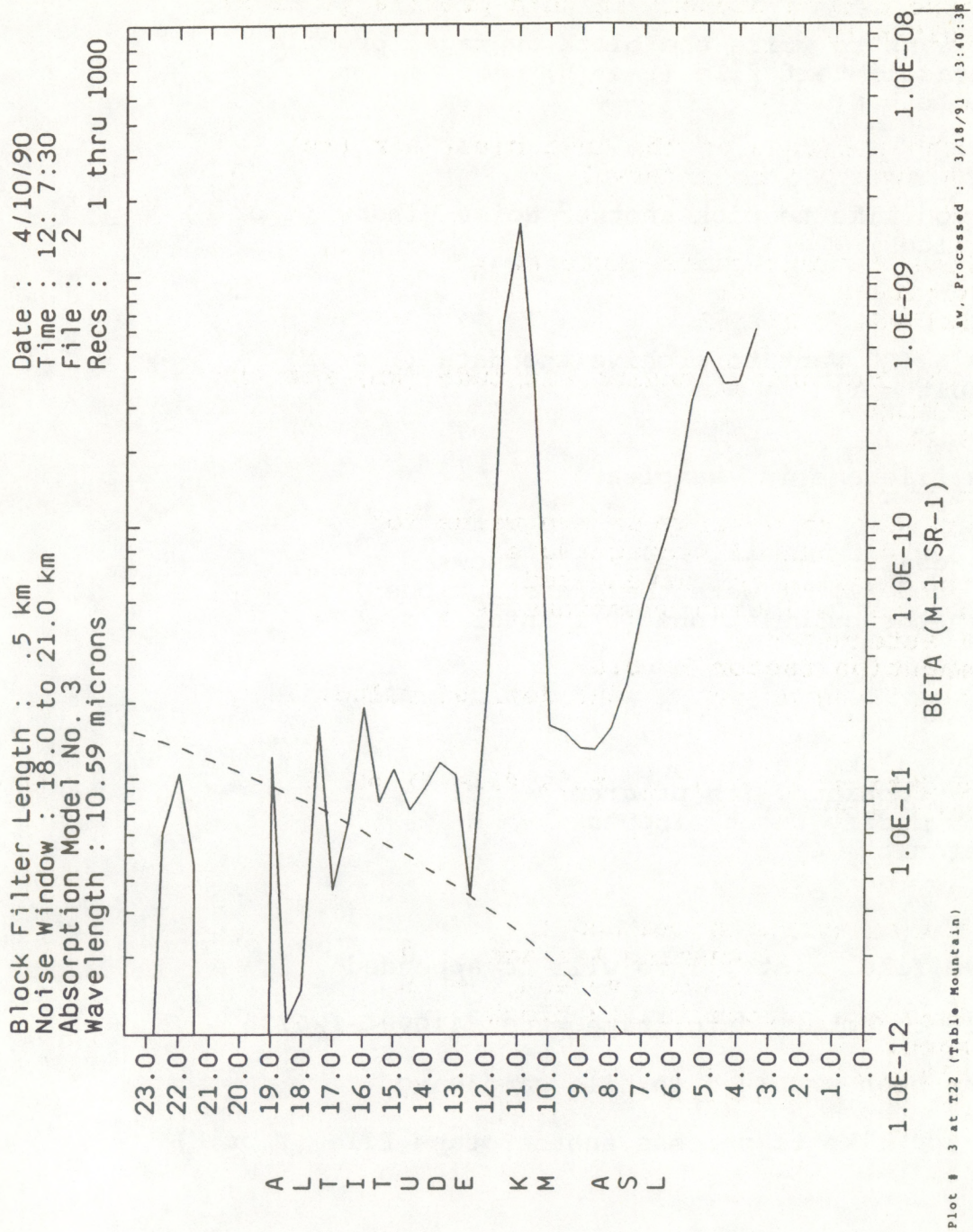

Figure 4.4 Altitude versus beta. Block averaged with 0.5 km altitude resolution.

```
Do you want to write the block averaged profile
 to an ascii text file (Y or N)
 (default = N)
? :


Would you like to pick another Noise Window (Y or N)
 (default=N)
? :


Do you STILL want to archive the data (Y or N)
 (default = Y)
? : y

Archive filename is  samplearc


Is the quality of the beta data known
to be good (1) or were there assumptions
made in the calculations (0). Enter 1 or 0
? : 1



Would you like to :
        Continue with program  -  C
        Modify recent inputs   -  M
   (default = C)
? :


Archive file exists, info will be appended

Profile : number, date, time =  3  41090  120730

Data has been archived in file samplearc

Would you like to process another tape file (Y or N)
   (default=N)
? :


Beta program finished
```

Beta Program Output (Processed on 3/18/91 at 14:43:44 by aw)

~~~ Data Information ~~~

Altitude and Location are 1.669 km asl at T22 (Table Mountain)

File	Date	Time	Starting Record	# Records
2	4/10/90	12: 7:30	1	1000

~~~ Calibration and Processing ~~~

Calibration dB down from expected is 3.20 (Archive: 4/13/90)
Shot Noise (dB) for the data is 6.13
Calibration factor is 2.764
Bandwidth (MHz) is 10.00
Wavelength (microns) is 10.591
Energy per pulse (J) is .50
A/D Average energy per pulse (J) is .37
Elevation angle (deg) is 90.00
Azimuth angle (deg) from tape is .00
Initial and end ranges (km) are 1.58 22.00
Atmospheric absorption model is No. 3 - Midlatitude Winter

~~~ Plot Setup ~~~

Average plots only
Scale of Signal Axis : Linear
: 0 - 2,000 (Medium)

~~~ Outputs ~~~

Plots : Altitude vs. Relative Signal Power (RSP)
RSP vs. Time, Altitude vs. Beta
Printouts : Raw data printed out in file BETA.RAWDAT
BETA profile printed out in file BETA.PRINT
This profile was archived to file samplearc

~~~ Noise Characteristics ~~~

Altitude of the Noise Window : 18.0 to 21.0 km
Altitude Resolution : .50 km
Mean Noise = 47.67
Noise Standard Deviation = 1.45
Normalized Noise Std Dev = .0304
Estimated Mean Noise Std Dev = .1018
Estimated Normalized Noise Std Dev = .214E-02
Estimated Normalized Noise Std Dev (dB) = -26.70

~~~ Data Quality ~~~

Quality Factor Threshold = .25 (2.6% bad data)
Quality Factor Threshold = .19 (16% bad data)
Clouds present, Altitude of last clear air is 9.5 km

Figure 4.5 The output file, BETA.PRINT, input data and noise information.

File	Date	Time	Start Record	# Records
2	4/10/90	12: 7:30	1	1000

Digitizer correction applied to R2Q2 data
Altitude of Noise Window : 18.0 to 21.0 km asl
Wavelength : 10.591 Bandwidth : 10.0 Data Good : Q-Factor GT .19

ALT (KM)	BETA(M-1 SR-1)	Q-FACTOR	SNR(DB)	BETA RATIO	NOISE BETA
3.500	6.04E-10	849.54	14.292	-36.50	1.35E-13
4.000	3.67E-10	363.41	10.604	-32.81	1.92E-13
4.500	3.64E-10	265.08	9.234	-31.44	2.62E-13
5.000	4.85E-10	266.93	9.264	-31.47	3.46E-13
5.500	3.10E-10	133.17	6.244	-28.45	4.44E-13
6.000	1.21E-10	41.32	1.162	-23.37	5.56E-13
6.500	7.44E-11	20.76	-1.827	-20.38	6.82E-13
7.000	4.93E-11	11.36	-4.447	-17.76	8.26E-13
7.500	2.34E-11	4.53	-8.436	-13.77	9.82E-13
8.000	1.57E-11	2.59	-10.864	-11.34	1.16E-12
8.500	1.29E-11	1.83	-12.374	-9.83	1.34E-12
9.000	1.30E-11	1.60	-12.969	-9.24	1.55E-12
9.500	1.51E-11	1.63	-12.868	-9.34	1.76E-12
10.000	1.59E-11	1.52	-13.189	-9.02	2.00E-12
10.500	3.57E-10	30.16	-.206	-22.00	2.25E-12
11.000	1.58E-09	119.68	5.780	-27.99	2.52E-12
11.500	6.44E-10	43.76	1.411	-23.62	2.80E-12
12.000	2.35E-11	1.44	-13.405	-8.80	3.10E-12
12.500	3.45E-12	.19	-22.172	-.03	3.42E-12
13.000	1.02E-11	.52	-17.877	-4.33	3.76E-12
13.500	1.14E-11	.53	-17.759	-4.45	4.11E-12
14.000	9.14E-12	.39	-19.114	-3.09	4.49E-12
14.500	7.47E-12	.29	-20.354	-1.85	4.88E-12
15.000	1.09E-11	.39	-19.077	-3.13	5.29E-12
15.500	8.00E-12	.27	-20.751	-1.45	5.72E-12
16.000	1.91E-11	.59	-17.298	-4.91	6.17E-12
16.500	6.87E-12	.20	-22.063	-.14	6.64E-12
17.000	3.63E-12	.10	-25.136	2.93	7.14E-12
17.500	1.63E-11	.40	-18.932	-3.27	7.65E-12
18.000	1.48E-12	.03	-29.637	7.43	8.18E-12
18.500	1.11E-12	.02	-31.177	8.97	8.74E-12
19.000	1.21E-11	.25	-21.066	-1.14	9.32E-12
19.500	1.00E-20	-.05	.000	.00	9.92E-12
20.000	1.00E-20	-.22	.000	.00	1.05E-11
20.500	1.00E-20	-.03	.000	.00	1.12E-11
21.000	1.00E-20	-.11	.000	.00	1.19E-11
21.500	4.56E-12	.07	-26.601	4.40	1.25E-11
22.000	1.05E-11	.15	-23.232	1.03	1.33E-11
22.500	6.09E-12	.08	-25.820	3.61	1.40E-11
23.000	1.45E-13	.00	-42.299	20.09	1.48E-11
23.500	1.00E-20	-4.04	.000	.00	1.56E-11

Figure 4.5 (continued.) BETA.PRINT file, vertical profile information.

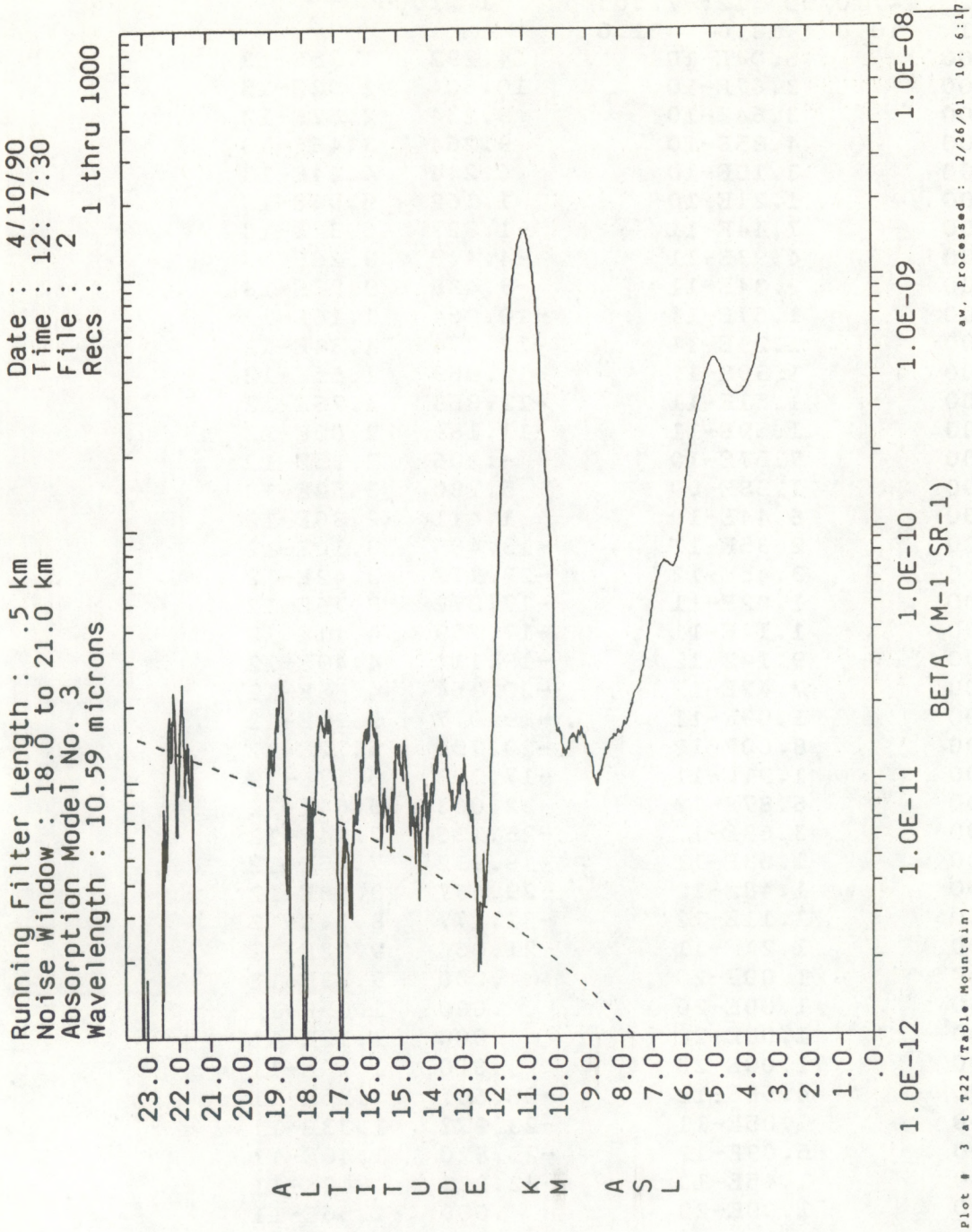

Figure 4.6 Altitude versus beta. Running average with 0.5 km altitude resolution. Optional output.

```
     2   4/10/90  12: 7:30        1  1000
10.591   10.0   18.0   21.0
  3.500        6.04E-10        14.292   1.35E-13
  4.000        3.67E-10        10.604   1.92E-13
  4.500        3.64E-10         9.234   2.62E-13
  5.000        4.85E-10         9.264   3.46E-13
  5.500        3.10E-10         6.244   4.44E-13
  6.000        1.21E-10         1.162   5.56E-13
  6.500        7.44E-11        -1.827   6.82E-13
  7.000        4.93E-11        -4.447   8.26E-13
  7.500        2.34E-11        -8.436   9.82E-13
  8.000        1.57E-11       -10.864   1.16E-12
  8.500        1.29E-11       -12.374   1.34E-12
  9.000        1.30E-11       -12.969   1.55E-12
  9.500        1.51E-11       -12.868   1.76E-12
 10.000        1.59E-11       -13.189   2.00E-12
 10.500        3.57E-10         -.206   2.25E-12
 11.000        1.58E-09         5.780   2.52E-12
 11.500        6.44E-10         1.411   2.80E-12
 12.000        2.35E-11       -13.405   3.10E-12
 12.500        3.45E-12       -22.172   3.42E-12
 13.000        1.02E-11       -17.877   3.76E-12
 13.500        1.14E-11       -17.759   4.11E-12
 14.000        9.14E-12       -19.114   4.49E-12
 14.500        7.47E-12       -20.354   4.88E-12
 15.000        1.09E-11       -19.077   5.29E-12
 15.500        8.00E-12       -20.751   5.72E-12
 16.000        1.91E-11       -17.298   6.17E-12
 16.500        6.87E-12       -22.063   6.64E-12
 17.000        3.63E-12       -25.136   7.14E-12
 17.500        1.63E-11       -18.932   7.65E-12
 18.000        1.48E-12       -29.637   8.18E-12
 18.500        1.11E-12       -31.177   8.74E-12
 19.000        1.21E-11       -21.066   9.32E-12
 19.500        1.00E-20          .000   9.92E-12
 20.000        1.00E-20          .000   1.05E-11
 20.500        1.00E-20          .000   1.12E-11
 21.000        1.00E-20          .000   1.19E-11
 21.500        4.56E-12       -26.601   1.25E-11
 22.000        1.05E-11       -23.232   1.33E-11
 22.500        6.09E-12       -25.820   1.40E-11
 23.000        1.45E-13       -42.299   1.48E-11
 23.500        1.00E-20          .000   1.56E-11
```

Figure 4.7 Ascii text file with altitude versus block averaged beta, snr, and noise beta.

File : 2
Records : 1 - 1000
Date : 4/10/90 Time : 12: 7:30

Average Power vs. Range :

1	2379.34	2	2172.33	3	2075.55	4	1967.37	5	1845.50
6	1838.79	7	1777.74	8	1696.16	9	1653.14	10	1550.84
11	1499.86	12	1419.85	13	1385.17	14	1327.18	15	1285.42
16	1233.68	17	1211.96	18	1170.34	19	1140.59	20	1111.27
21	1101.32	22	1039.49	23	1016.95	24	984.34	25	1011.29
26	984.92	27	930.83	28	875.74	29	850.98	30	842.39
31	842.22	32	808.80	33	798.69	34	759.34	35	724.77
36	720.30	37	732.59	38	689.96	39	687.47	40	698.69
41	691.40	42	689.56	43	659.25	44	633.91	45	601.51
46	606.12	47	623.75	48	608.98	49	577.97	50	581.90
51	563.60	52	563.30	53	563.80	54	571.08	55	543.89
56	524.62	57	517.42	58	502.96	59	527.90	60	513.73
61	496.76	62	510.31	63	507.36	64	505.78	65	467.77
66	481.67	67	466.02	68	472.27	69	479.65	70	472.74
71	468.80	72	466.80	73	465.20	74	474.41	75	462.02
76	456.32	77	461.75	78	478.36	79	481.78	80	451.99
81	426.35	82	427.19	83	427.70	84	439.68	85	418.37
86	405.48	87	427.02	88	426.04	89	442.16	90	435.33
91	434.10	92	426.44	93	412.66	94	420.18	95	427.87
96	448.38	97	443.09	98	462.71	99	449.72	100	477.17
101	478.30	102	476.96	103	492.09	104	488.86	105	473.41
106	480.37	107	492.43	108	524.86	109	498.71	110	510.48
111	508.55	112	488.95	113	494.41	114	479.57	115	485.85
116	475.90	117	472.33	118	469.94	119	455.91	120	451.33
121	443.09	122	431.04	123	427.62	124	413.30	125	439.21
126	403.79	127	394.87	128	382.32	129	388.73	130	366.49
131	360.64	132	357.26	133	342.88	134	338.11	135	318.73
136	319.13	137	300.93	138	296.41	139	287.87	140	301.47
141	282.69	142	287.69	143	280.33	144	278.21	145	278.98
146	263.32	147	263.43	148	253.40	149	236.99	150	241.85
151	233.37	152	224.50	153	233.90	154	228.37	155	217.14
156	226.71	157	227.75	158	204.60	159	204.13	160	194.72
161	188.74	162	194.10	163	204.30	164	209.14	165	195.86
166	180.18	167	173.14	168	171.98	169	157.24	170	157.81
171	151.17	172	139.31	173	135.79	174	144.10	175	131.05
176	132.24	177	127.67	178	118.08	179	111.67	180	106.09
181	104.61	182	105.33	183	98.52	184	95.79	185	99.75
186	95.02	187	91.90	188	91.42	189	85.88	190	87.03
191	82.10	192	83.75	193	82.91	194	81.38	195	78.40
196	77.62	197	76.23	198	76.45	199	77.01	200	80.86
201	77.62	202	82.18	203	76.31	204	76.43	205	73.23
206	75.77	207	77.38	208	80.35	209	79.12	210	82.69
211	80.14	212	83.73	213	81.54	214	81.89	215	82.79

Figure 4.8 Subset of the output file, BETA.RAWDAT, averaged raw data for approximately 3.2 km range. The complete data file extends to the maximum profile range, 22 km.

5. PROGRAM STRUCTURE

The BETA program, written in FORTRAN 77, consists of 44 modules: the main program, beta.f, and 43 subroutines. There are nine external common blocks for data transfer between the program modules. It is linked to a modified Plot 10 Tektronix Graphics Library and a magnetic tape access library.

Figure 5.1 is a diagram of the program flow. Box numbers correspond to boxes in Figure 5.2, which is a diagram of the subroutine calling sequence. If upon reaching the end of the program, box 23, the user chooses to process another file, the program returns to box 2.

BETA accesses many external data files. The necessary input files are an atmospheric absorption data file containing vertical profiles for each of the five seasonal models at the three wavelengths used in the program and a lidar location file with an altitude above sea level and a character name for each location where data have been collected. If the user selects calibration data from the archive, then the calibration archive files must be present. If the atmospheric absorption profile is to be derived from a radiosonde profile, two ASCII data files are needed. One contains a radiosonde profile of pressure (mb), height (m), temperature (°C), and dew point (°C). This file is accessed with a free-formatted read. The other, an AFGL Hitran database contains line strength data for the wavenumber (cm^{-1}) ± 10 cm^{-1} corresponding the wavelength in use.

The user selects various output files during program

execution. The beta archive files are binary and the other information files are ASCII. A plot file containing graphics commands may also be generated. The primary output text file is BETA.PRINT and the graphics file name is qqplot. Table 2 describes all input and output files.

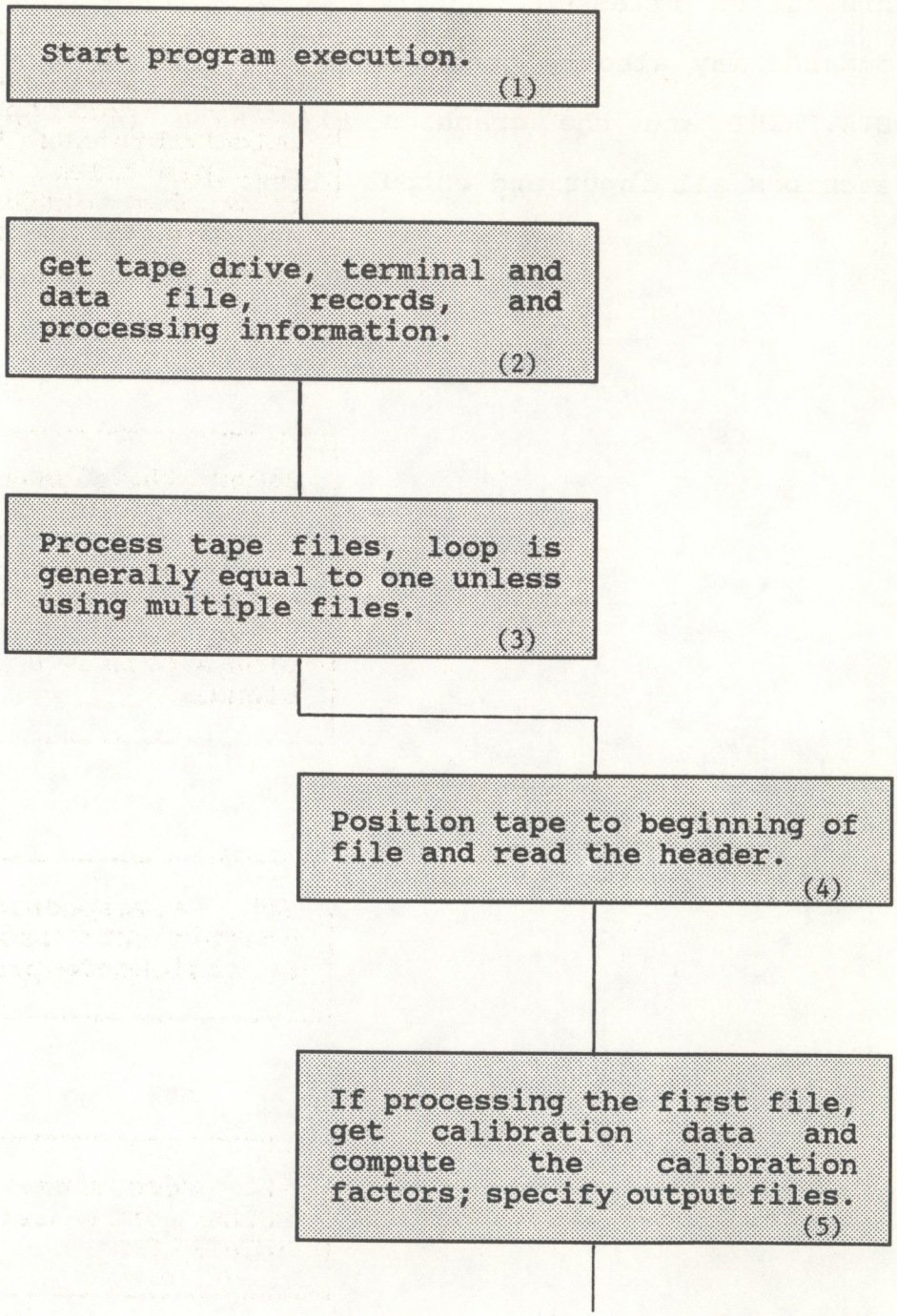

Figure 5.1 BETA Program Flow Diagram

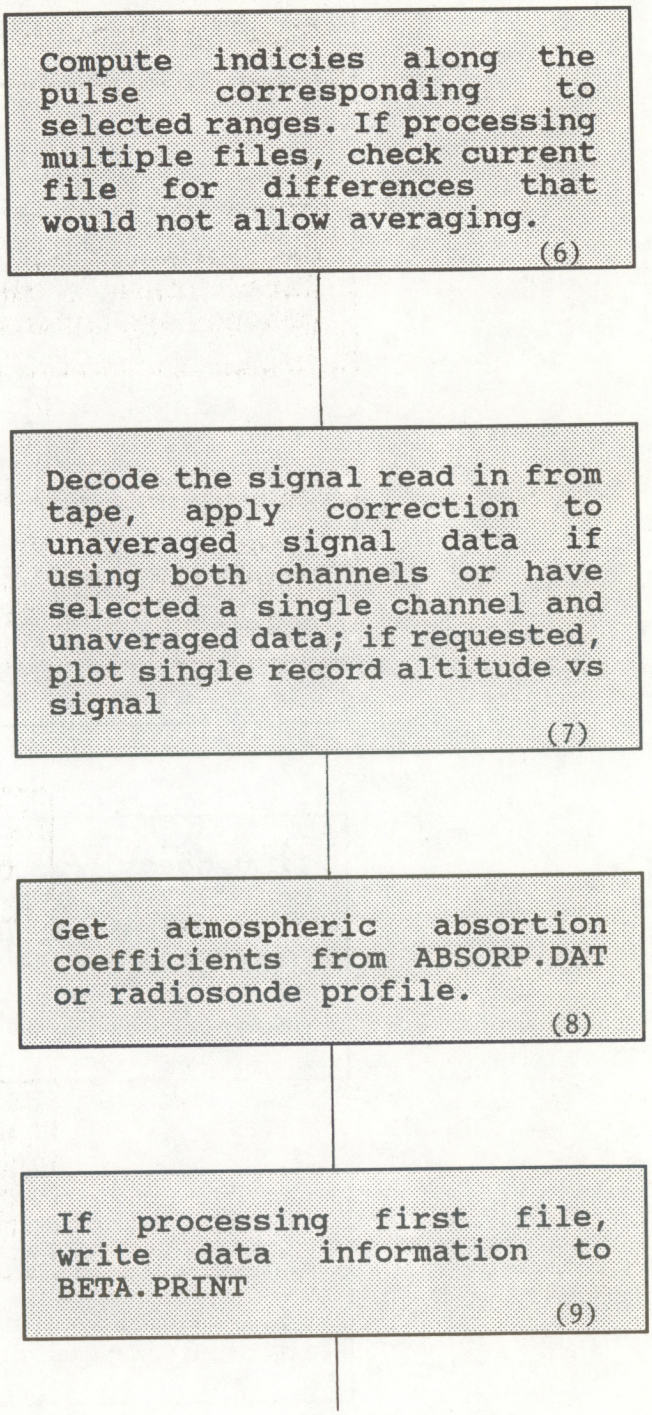

Figure 5.1 continued.

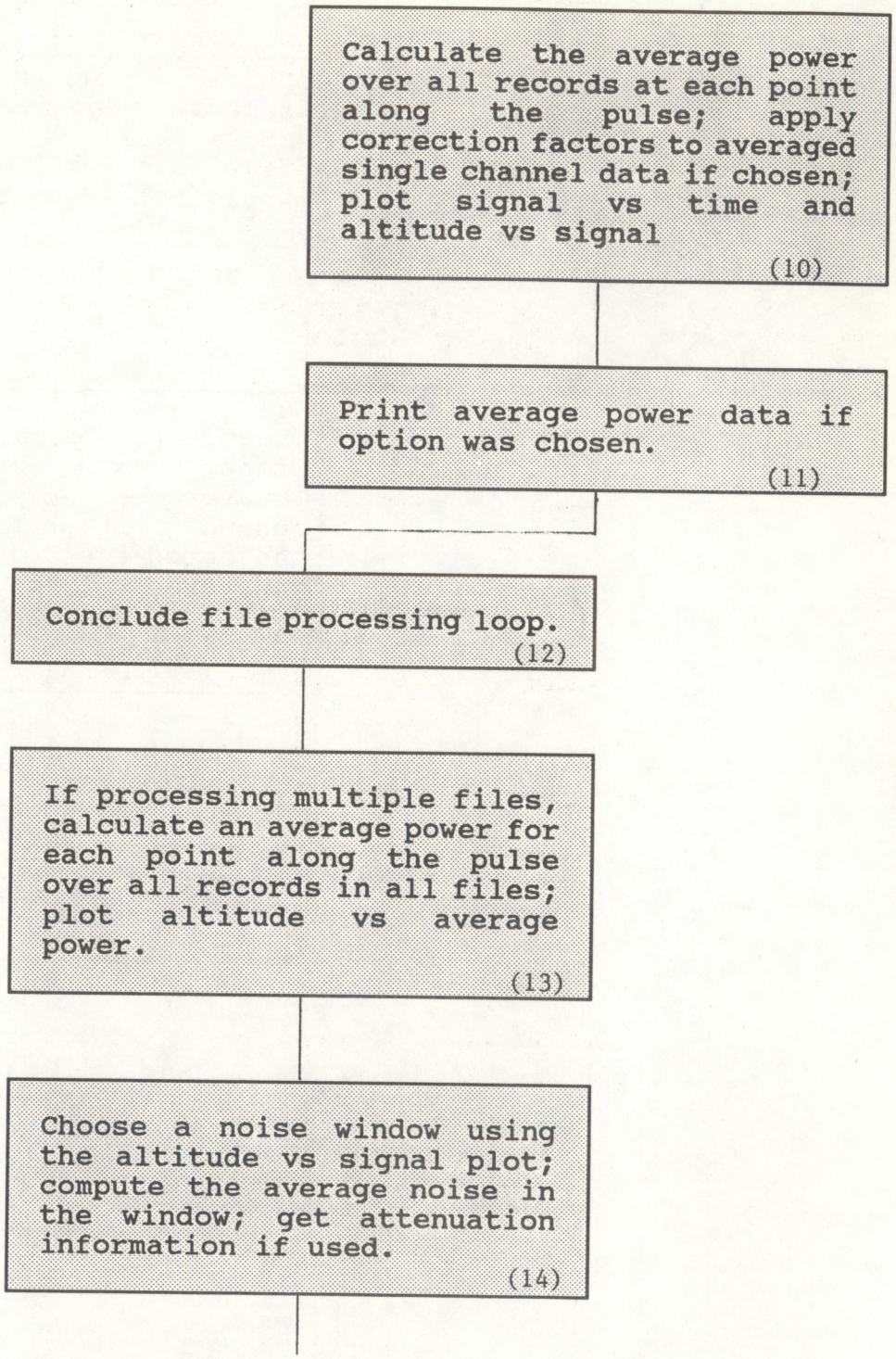

Figure 5.1 continued.

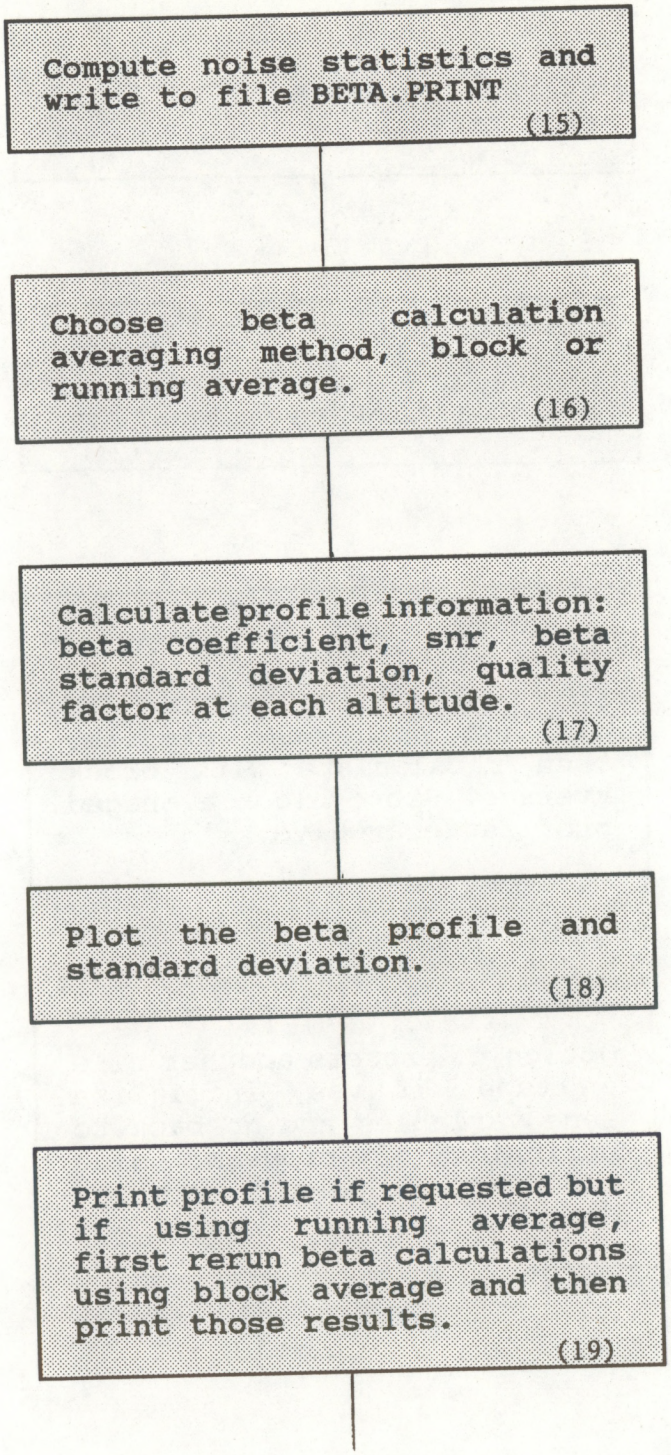

Figure 5.1 continued.

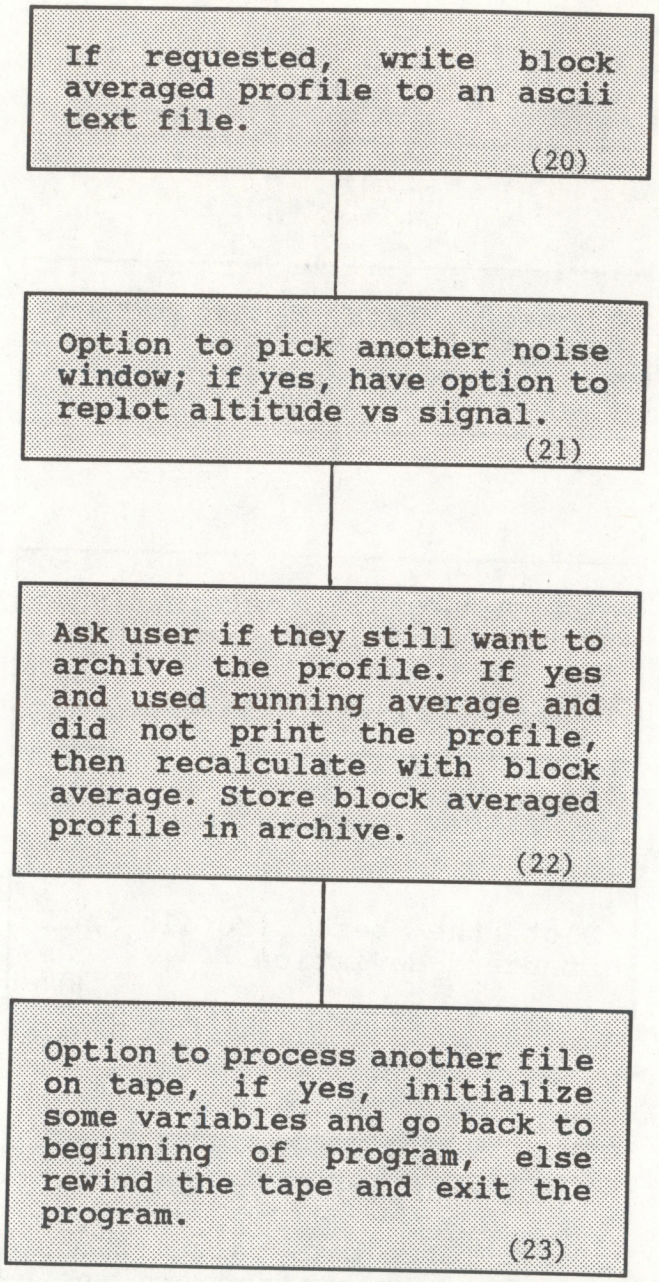

Figure 5.1 continued.

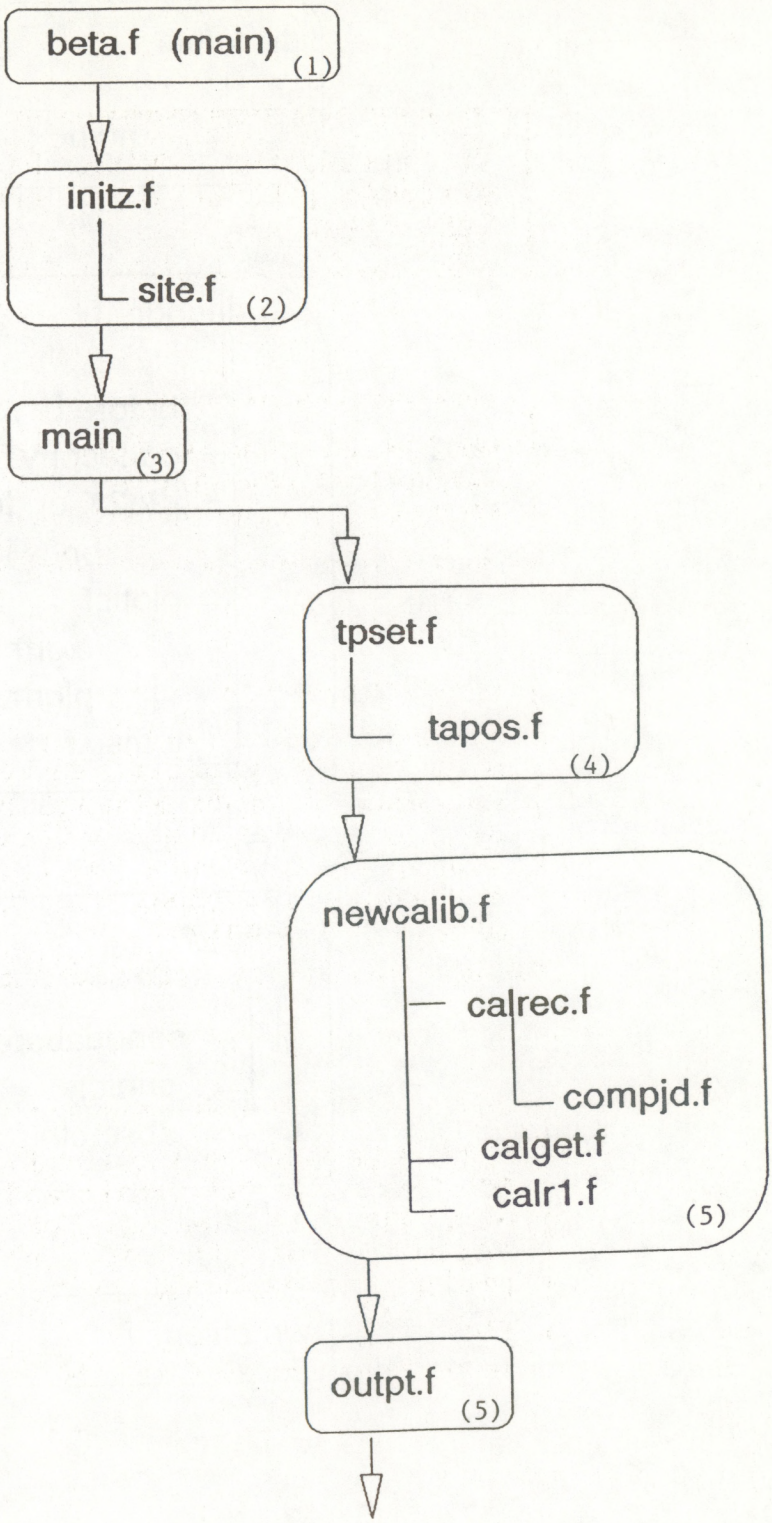

Figure 5.2 Subroutine calling sequence from main program - BETA.

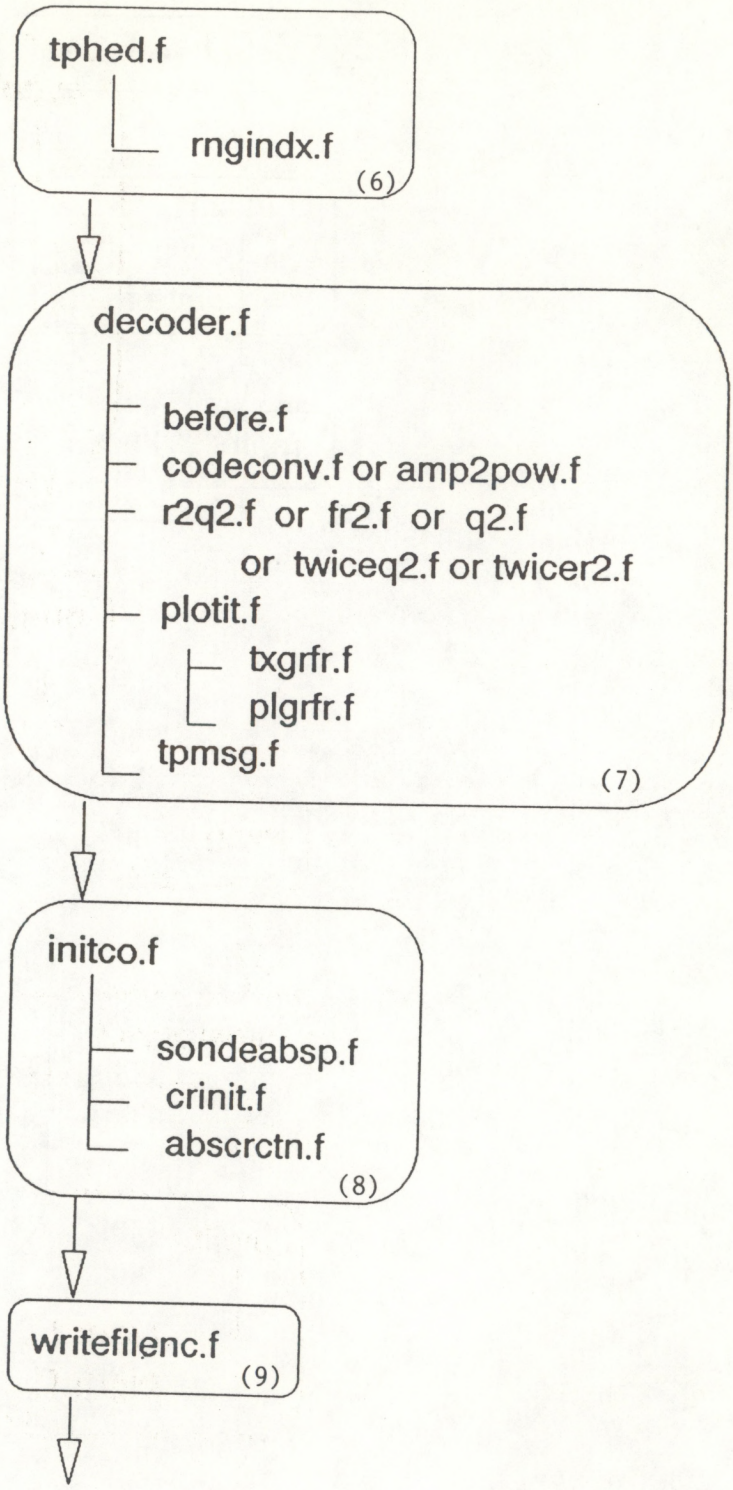

Figure 5.2 continued.

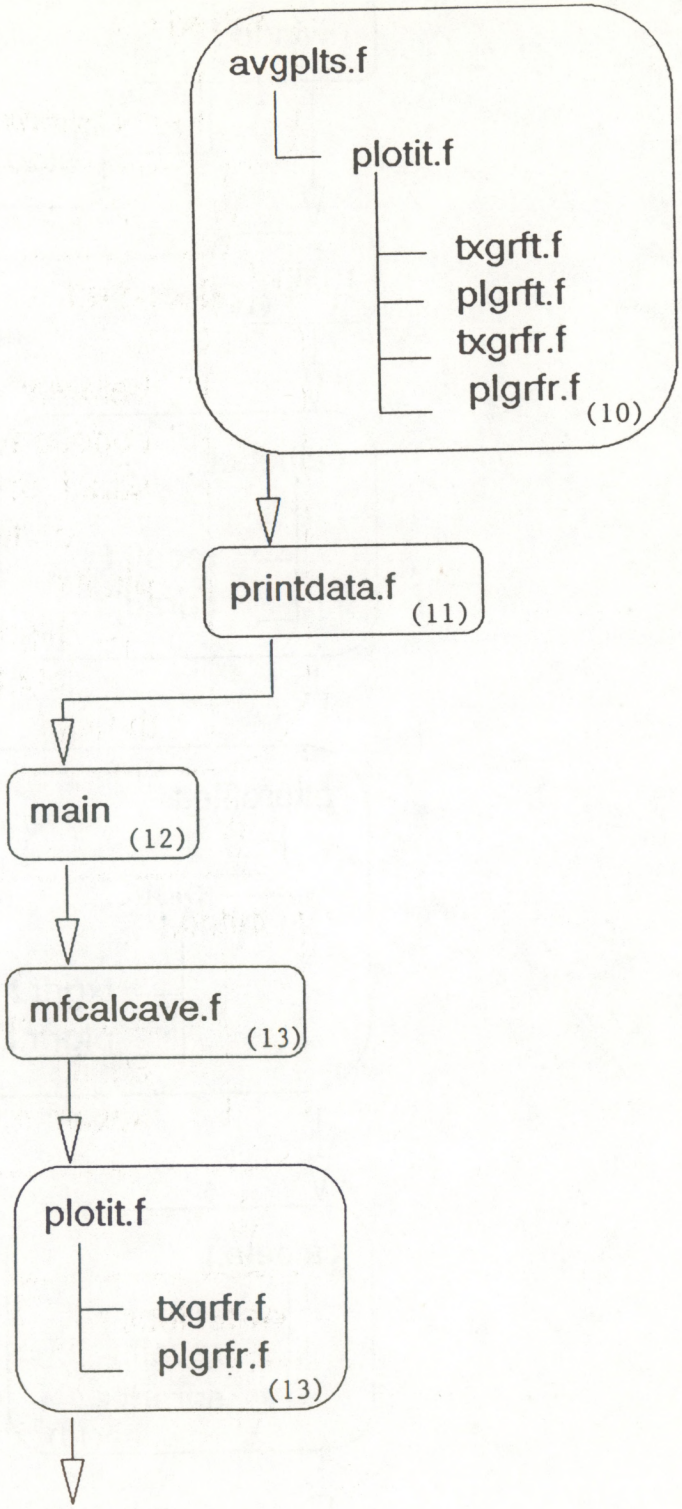

Figure 5.2 continued.

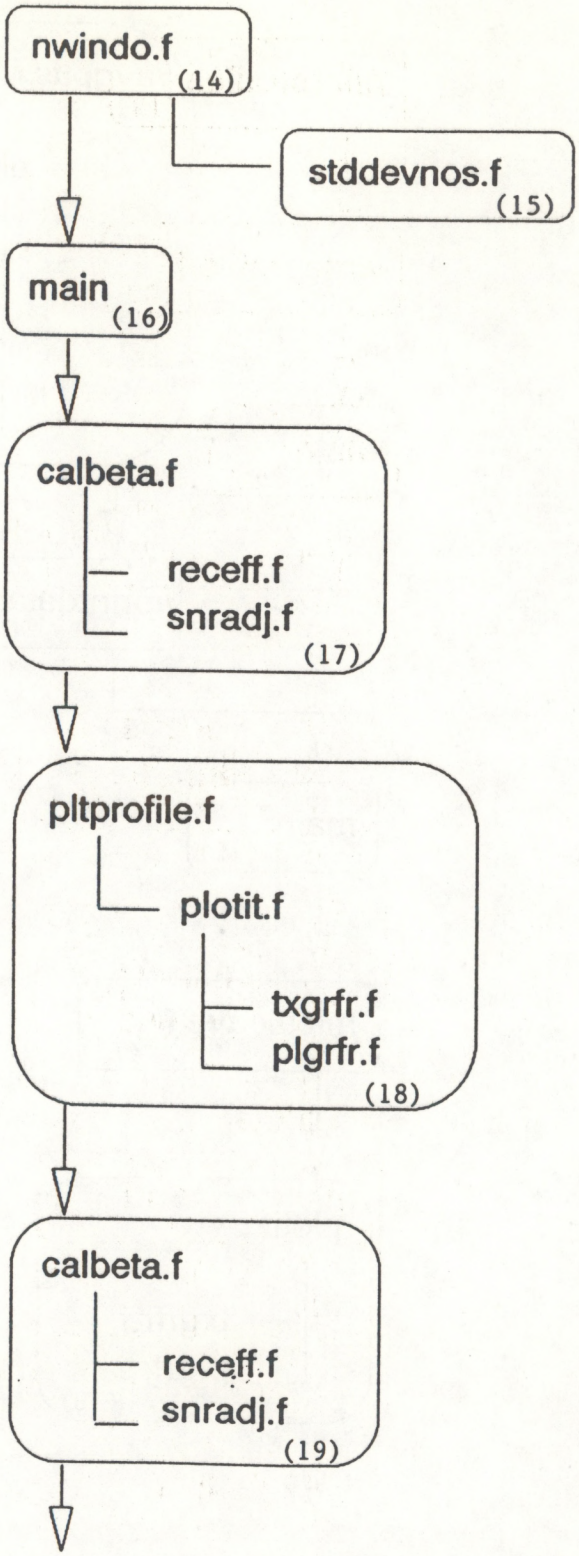

Figure 5.2 continued.

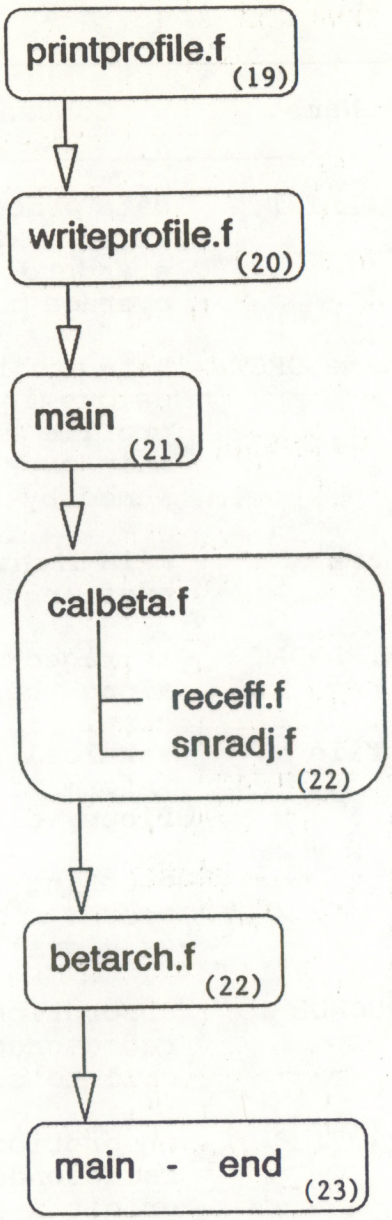

Figure 5.2 continued.

Table 2 BETA Data Files

I/O	Unit	Type	File Name	Contents/Function
Out	40	ASCII	BETA.PRINT	data processing information and vertical profiles of beta, snr, standard deviation of each, average power and quality factor
Out	50	binary	arcname.DRCT	beta archive directory, contains unique information for each profile and pointers into the main archive file, arcname. Named by user.
Out	52	binary	arcname	main archive file containing the profile data. Named by user.
Out	65	ASCII	BETA.RAWDAT	averaged raw data for each point along the profile.
Out	42	ASCII	textfile	vertical profile with header information for output to floppy disk. Named by user.
Out		graph	qqplot	plotting instructions for the plots: signal vs time; altitude vs signal; altitude vs beta.
Out	22	ASCII	ABSPSONDE	absorption parameters at radiosonde heights. Output from call to sondeabsp.f
Out	24	ASCII	ABSPROFILE	absorption coefficients at radiosonde heights. Input into crinit.f and abscrctn.f
Out	70	ASCII	LOOKABSP	absorption coefficients interpolated to 1 km heights. This replaces data from ABSORP.DAT
In	30	binary	CALARC.DRCT	calibration archive directory
In	32	binary	CALARC.INFO	calibration archive profiles

Table 2 continued

I/O	Unit	Type	File Name	Contents/Function
In	15	ASCII	ABSORP.DAT	absorption coefficient profiles for the 5 seasonal models at wavelengths 10.591, 9.569, 9.282 microns.
In	60	ASCII	lidarlocs	contains altitude in km, asl and character name of all lidar data collection sites.
In	20	ASCII	radiosonde	vertical radiosonde profile of height (m), pressure (mb), temperature (C), and relative humidity. One line per height, free formatted read. Named by user.
In	66	ASCII	HIT944_1081	portion of the AFGL Hitran Database for wavenumbers 934.6 to 1091.6 cm^{-1}.

6. ARCHIVE STRUCTURE

The vertical profiles of altitude, beta, and average power along with pertinent data parameters are stored in an archive. The archive consists of two files, a directory file and a data file. The directory file contains information that uniquely identifies a profile and data structure pointers to the first and last records of the profile in the data file, referred to as head and tail pointers. Other programs use information in the directory file to sort profiles for data analysis. Each profile in the data file contains three header lines and a variable number of vertical profile lines.

Both files are direct access, binary and unformatted. The directory file has record lengths of 46 bytes and the data file has record lengths of 48 bytes. The data file is named by the user and the directory file is the same name with the suffix '.DRCT' automatically attached. Accessing the archive requires the existence of both files.

Each profile is numbered in the directory file. When a profile is deleted, the number is set to its negative (-number) to signify a deleted profile. The data are not removed nor are the other profiles renumbered. When a new profile is added to the archive, its length is determined (number of lines) and the directory file is searched for deleted profiles that have an equal or greater number of lines (tail-head+1 of deleted profile ≥ new profile length). If a space can be found, the new profile is inserted into the space and assigned the absolute value of the

profile number. If space cannot be found, the profile is appended to the end of the data file and numbered sequentially. The data structure pointers, head and tail, are written to the directory file with other profile information.

The first line in the directory file contains a code for archive type (a beta archive is 1) and a counter for the number of profiles in the archive. The counter includes deleted profiles so that all directory entries are searched.

Subroutine betarch.f writes the information to the archive. Some information, particularly for the directory, is stored in the archive in a different format than in the common blocks. Table 3 describes the information stored in the archive and its type and location in common. When the archive data have been constructed from common block data, an asterisk appears next to the variable name. Some information is not contained in the common blocks, such as archive type, process date and time, and number of profiles. In this case the common block column is left blank.

Four byte integer and real dummy data fields have been included in the archive so that future additions will not change the file structure.

Table 3. Archive Contents

Variable Description	Size	Name	Common
Directory File, Record 1:			
archive type	int*4	arctyp	
number of profiles	int*4	numbet	
Directory File, Records 2 - N :			
profile number	int*4	numo	
date of profile, MMDDYY	int*4	idate*	header
time of profile, HHMMSS	int*4	itime*	header
altitude of last clear air, km	real*4	clrair	vars
head pointer into data file	int*4	begrec	
tail pointer into data file	int*4	endrec	
profile data quality flag	int*4	bqc*	calc
site number	int*4	isite	vars
wavelength, microns	real*4	wave*	vars
bandwidth, MHz	real*4	band*	vars
Data File, Header Line 1 :			
beginning tape record	int*2	irec	vars
ending tape record	int*2	irend	vars
number of data points in the profile	int*2	icount	calc
atmospheric model number	int*2	model	vars
first altitude of the profile, km	real*4	alt(1)	signl
last altitude of the profile,km	real*4	alt(icount)	signl
profile altitude resolution, km	real*4	dalt	calc
altitude of lidar, km	real*4	altlid	vars
Data File, Header Line 2 :			
begin altitude of noise window, km	real*4	begalt	calc
end altitude of noise window, km	real*4	endalt	calc
average noise	real*4	avnois	calc
standard deviation of noise	real*4	sdnois	calc
calibration date, MMDDYY	int*4	caldat	calc
db down from expected	real*4	dbcal	calc
calibration shot noise, no longer used	real*4	shtref	calc
data shot noise	real*4	shtdat	calc
raw (1) versus moment (2) data flag	int*2	ihead(8)	header

Table 3 continued.

Variable Description	Size	Name	Common
Data File, Header Line 3 :			
elevation angle, degrees	real*4	eleva	vars
azimuth angle, degrees	real*4	azimth	vars
data process date, MMDDYY	int*4	pdate	
data process time, HHMMSS	int*4	ptime	
operator initials	int*2	op	vars
quality factor threshold value	real*4	qual	signl
real dummy	real*4	adum	
integer dummy	int*4	idum	
integer dummy	int*4	idum	
integer dummy	int*4	idum	
integer dummy	int*4	idum	
Data File, Profile Lines (1-icount) :			
altitude, km	real*4	alt( )	signl
beta coefficient, m^{-1} s^{-1}	real*4	betaco( )	signl
average power at each altitude	real*4	avsgnl( )	signl
real dummy	real*4	adum	
real dummy	real*4	adum	
real dummy	real*4	adum	
real dummy	real*4	adum	
real dummy	real*4	adum	
real dummy	real*4	adum	
real dummy	real*4	adum	

References

Post, M.J., J.F. Morrow, and D.B. Jensen, 1983: BETA: A Program for Calculating and Archiving Backscattering Profiles taken with the NOAA Coherent Lidar System. NOAA Tech. Memo. ERL WPL-113, NOAA Environmental Research Laboratories, Boulder, Colo., 44 pp.

Post, M.J., and R.E. Cupp, 1990: Optimizing a pulsed Doppler lidar. Applied Optics, 29, 4145-4158.

Rothman, L.S., R.R. Gamache, A. Goldman, L.R. Brown, R.A. Toth, H.M. Pickett, R.L. Poynter, J.M. Flaud, C. Camy-Peyret, A. Barbe, N. Husson, C.P. Rinsland, and H.A.H. Smith, 1987. The HITRAN database: 1986 edition. Applied Optics, 26, 4058-4097.

Appendix A

Software Listings

Table of Contents

abscrctn.f A-52
amp2pow.f A-4
avgplts.f A-4
before.f A-5
beta.f A-1
betarch.f A-5
calbeta.f A-8
calget.f A-9
calr1.f A-10
calrec.f A-10
codeconv.f A-13
compjd.f A-13
crinit.f A-51
decoder.f A-14
fr2.f A-16
initco.f A-17
initz.f A-19
mfcalcave.f A-20
newcalib.f A-21
nwindo.f A-23
outpt.f A-26
plgrfr.f A-27
plgrft.f A-28
plotit.f A-29
pltprofile.f A-32
printdata.f A-32
printprofile.f A-32
q2.f A-33
r2q2.f A-34
receff.f A-34
rngindx.f A-34
site.f A-35
snradj.f A-36
sondeabsp.f A-38
stddevnos.f A-40
tapos.f A-53
tphed.f A-41
tpmsg.f A-42
tpset.f A-44
twiceq2.f A-44
twicer2.f A-44
txgrfr.f A-45
txgrft.f A-47
writefilenc.f A-47
writeprofile.f A-49

```
      PROGRAM BETA
C-----------------------------------------------------------------
C     This program looks at raw signals and computes back-
C     scattering for data collected with the NOAA COHERENT
C     DOPPLER LIDAR.  Maximums:  5000 records/file, 2048
C     points/record, 30 KM range.
C-----------------------------------------------------------------
C
C     This program was written by personnel from
C     NOAA/ERL/Wave Propagation Laboratory/Atmospheric Lidar Group
C
C     Software Responsibility : A. Weickmann    (303) 497-6595
C     Scientific Responsibility : M.J. Post     (303) 497-6048
C
C     Modified :   8/10/89   by  AMW
C     Modified :   8/29/90   by  RPM - added CALBETA
C     Modified :   1/07/91   by  amw - improve inputs
C-----------------------------------------------------------------
C      INPUTS
C Variable  type,common        Description
C --------  -----------  ----------------------------------------
C
C      OUTPUTS
C Variable  type,common        Description
C --------  -----------  ----------------------------------------
C REINIT  1,setup    Re-initialization flag
C-----------------------------------------------------------------

      INCLUDE "CB.CALC"
      INCLUDE "CB.SETUP"
      INCLUDE "CB.HEADER"
      INCLUDE "CB.MULTFL"
      INCLUDE "CB.PLOT"
      INCLUDE "CB.SIGNL"
      INCLUDE "CB.VARS"

      INTEGER * 2 IERR
      REAL*4 AVPWDB, ARAY(2048)
      CHARACTER*1 CANS, GRFTYP

 1111 FORMAT(A1)

C     Open the BETA.PRINT text file

      OPEN(40,FILE='BETA.PRINT',CARRIAGECONTROL='FORTRAN',
     +        STATUS='NEW')

      REINIT = .FALSE.

 7    MFERRR = .FALSE.
      MF1PAS = .TRUE.
      MFNMPL = 0
      IPLTN=0

 10   CONTINUE

C     Get mag tape, terminal, and file information
      CALL INITZ

      DO 99 I = 1,MFNMFI

         IFN = MFIFN(I)
         IREC = MFIREC(I)
         NREC = MFNREC(I)
         IREND = IREC + NREC - 1
         CALL TPSET

         IF(IHEAD(3).LT.86) THEN
                CALL NEWPAG
                PRINT *
                PRINT *,' This version of beta does not process data prior'
                PRINT *,'  to 1986.  Contact M.J. Post or A. Weickmann'
                PRINT *,'  if you have pre-1986 data.'
                RETURN
         ENDIF

C     Put date and time of each file into arrays
```

```
         MFDATE(I,1)=IHEAD(1)
         MFDATE(I,2)=IHEAD(2)
         MFDATE(I,3)=IHEAD(3)
         MFTIME(I,1)=IHEAD(4)
         MFTIME(I,2)=IHEAD(5)
         MFTIME(I,3)=IHEAD(6)

         IF (MF1PAS) THEN
                CALL NEWCALIB
                CALL OUTPT
         ENDIF

 15      CONTINUE

C     Calculate range indicies

         CALL TPHED

C        If multiple files, and this file has critical differences
C                from the 1st file, skip it.
         IF (MFERRR) THEN
                MFERRR = .FALSE.
                GOTO 99
         ENDIF

C     Decode the signal read in from tape

         CALL DECODER

C     Read in absorption coefficients for selected wavelength.

         CALL INITCO

C     Write to the BETA.PRINT text file.

         IF ((MULTFI .AND. MF1PAS) .OR. (.NOT. MULTFI)) THEN
                CALL WRITEFILENC
         ENDIF

C     Calculate average power and scales and plot the altitude vs.
C      power plot and the power vs time plot.

         CALL AVGPLTS(PRF,ARCDAT)
         IF (PRNDAT .EQ. "Y ")  CALL PRINTDATA(AVGPWR)

         MF1PAS = .FALSE.

 99      CONTINUE

      IF (MULTFI) THEN
         CALL MFCALCAVE(ARAY)
C     Set IFN to -1 to signal plotting of all files averaged together
         IFN=-1
         GRFTYP='R'
         CALL PLOTIT(ARAY,NPTS,GRFTYP)
         IF(IPLTA.EQ.0 .or. IPLTA.EQ.2) READ(*,1111) CANS
         CALL NEWPAG
      ENDIF

 20   CONTINUE

      NREC = 0
      DO 22 I = 1, MFNMFI
         NREC = NREC + MFNREC(I)
 22    CONTINUE
      IF (MULTFI) THEN
         IREC = 1
      ENDIF
      IREND = IREC + NREC - 1

C     Enter noise window and compute indicies in array
      CALL NWINDO(COSRAD,ALTLID)

C     Calculate SNR, Beta, and related statistics.
 30   PRINT *
      print *,' Choose data averaging method :'
```

```
      print *,'                          Running Average -  R '
      print *,'                          Block Average   -  B '
      print *,' (archives are automatically block'
      print *,'  averaged)'
      print *,' (default = B) '
      PRINT *,' ? : '
      READ (*,1111,err=30) CANS
      IF (CANS.EQ.' ' .OR. CANS.EQ.'b' .OR. CANS.EQ.'B') THEN
         IF (ARCDAT .EQ. 'Y ') THEN
                BTYPE = 'A '
         ELSE
                BTYPE = 'B '
         ENDIF
      ELSE IF (CANS.EQ.'r' .OR. CANS.EQ.'R') THEN
         BTYPE = 'R '
      ELSE
         print *
         print *,' Your answer was not an option, try again.'
         GOTO 30
      ENDIF
      CALL CALBETA

C     Plot the beta profile.
      CALL PLTPROFILE

C     If requested, print the beta profile.
      IF (PRNPRF.EQ.'Y ') THEN

C     Rerun calbeta with block avg if did running average last time.
         IF (BTYPE.EQ.'R ' .AND. ARCDAT.EQ.'Y ') THEN
            BTYPE = 'A '
            CALL CALBETA
         ELSE IF (BTYPE.EQ.'R ' .AND. ARCDAT.NE.'Y ') THEN
            BTYPE = 'B '
            CALL CALBETA
         ENDIF

         CALL PRINTPROFILE
      ENDIF

C     Write the profile to a text file if desired

      CALL NEWPAG
 35   print *
      print *,' Do you want to write the block averaged profile'
      print *,'  to an ascii text file (Y or N)'
      print *,'  (default = N)'
      print *,' ? : '
      read (*,1111,err=35) cans

      if(cans.eq.'Y' .or. cans.eq.'y') then
         call writeprofile
      else if(cans.ne.' ' .and. cans.ne.'N' .and. cans.ne.'n') then
         print *
         print *,' Your answer was not an option, try again.'
         goto 35
      endif


C     Option to choose another noise window and replot the raw data
C      so that a new noise window can be chosen.

      CALL NEWPAG
 37   PRINT *
      PRINT *,' Would you like to pick another Noise Window (Y or N)'
      PRINT *,'  (default=N)'
      PRINT *,' ? : '
      READ (*,1111,err=37) CANS


      if(cans.eq.' ' .or. cans.eq.'N' .or. cans.eq.'n') then
         goto 60
      else if(cans.eq.'Y' .or. cans.eq.'y') then
         IF(IPLTA.EQ.1) GOTO 20
 38      PRINT *
         PRINT *,' Do you want to see the raw data plot  (Y or N)'
         PRINT *,'    (default = Y)'
         PRINT *,' ? : '
         READ (*,1111,err=38) CANS

         IF(CANS.EQ.'N' .OR. CANS.EQ.'n') GOTO 20
         if(cans.ne.'Y' .and. cans.ne.'y' .and. cans.ne.' ') then
                print *
                print *,' Your answer was not an option, try again.'
                goto 38
         endif

         BETAFL='N '
         IF(LINDB.EQ.'D ') THEN
C     Convert power to DB.
                DO 40 I=1,NPTS
                   AVPWDB = 10.0 * ALOG10(AVGPWR(I))
                   ARAY(I) = AVPWDB
 40             CONTINUE
         ELSE
                DO 50 I=1,NPTS
                   ARAY(I)=AVGPWR(I)
 50             CONTINUE
         ENDIF

C      Plot Average power vs alitiude.  R1 and R2 are ranges
C     corrected for elevation angle and lidar altitude, in effect
C     they are altitudes.

         YMIN = 0.0
         YMAX = R2
         XMIN=0.0

         IF (LINDB .EQ. "D ") THEN
                IF (LOHI .EQ. "H ") THEN
                        XMAX = 40.0
                ELSE IF (LOHI .EQ. "M ") THEN
                                XMAX = 20.0
                ELSE IF (LOHI .EQ. "L ") THEN
                                XMAX = 12.0
                ENDIF
         ELSE
                IF (LOHI .EQ. "H ") THEN
                        XMAX = 20000.0
                ELSE IF (LOHI .EQ. "M ") THEN
                        XMAX = 2000.0
                ELSE IF (LOHI .EQ. "L ") THEN
                        XMAX = 400.0
                ENDIF
         ENDIF

C     Plot average power vs altitude

                GRFTYP = 'R'
                CALL PLOTIT (ARAY, NPTS,  GRFTYP)
                IF(IPLTA.EQ.0 .OR. IPLTA.EQ.2) READ(*,1111) CANS
         CALL NEWPAG
         GOTO 20
      else
         print *
         print *,' Your answer was not an option, try again.'
         goto 37
      ENDIF

 60   continue

      IF (ARCDAT .EQ. "Y ") THEN
 70      PRINT *
         PRINT *,' Do you STILL want to archive the data (Y or N) '
         PRINT *,'  (default = Y)'
         PRINT *,' ? : '
         READ(*,1111,err=70) CANS

         IF(CANS.EQ.'N'.OR.CANS.EQ.'n') GOTO 1045
         if(cans.eq.'Y' .or. cans.eq.'y' .or. cans.eq.' ') then

                CALL NEWPAG
C  (If did running average recently, must do block average for archive.)
                IF (BTYPE.EQ.'R ' .AND. PRNPRF.NE.'Y ') THEN
                   BTYPE = 'A '
                   CALL CALBETA
```

```
                ENDIF
                IF (BTYPE.EQ.'R ') CALL PLTPROFILE
                   CALL BETARCH
         else
                print *
                print *,' Your answer was not an option, try again.'
                goto 70
         endif
      ENDIF

1045  CONTINUE

80    PRINT *
      PRINT *,' Would you like to process another tape file (Y or N)'
      PRINT *,'   (default=N)'
      PRINT *,' ? : '
      READ (*,1111,err=80) CANS

      IF(CANS.EQ.'Y' .OR. CANS.EQ.'y') THEN
         REINIT=.TRUE.
         MF1PAS=.TRUE.
         CALL NEWPAG
         GOTO 10
      else if(cans.eq.' ' .or. cans.eq.'N' .or. cans.eq.'n') then
         close(40)
         call mtrew(ipack)
         print *
         print *,' Beta program finished'
         print *
      else
         print *
         print *,' Your answer was not an option, try again.'
         goto 80
      ENDIF

      RETURN
      END
```

```
      REAL*8 FUNCTION AMP2POW(IGT)
C-----------------------------------------------------------------
C   This function accepts an amplitude value obtained from the moment
C   data and converts it to power.
C-----------------------------------------------------------------
C
C         This program was written by personnel from
C         NOAA/ERL/Wave Propagation Laboratory/Atmospheric Lidar Group
C
C         Software Responsibility : A. Weickmann    (303) 497-6595
C         Scientific Responsibility : M.J. Post     (303) 497-6048
C
C         Last Modified :   8/09/89   by  AMW
C-----------------------------------------------------------------
C         INPUTS
C Variable  type,common      Description
C --------  -----------  ----------------------------------------
C  IGT      i,passed     Amplitude. A 12-bit positive logarithmic int.
C
C         OUTPUTS
C Variable  type,common      Description
C --------  -----------  ----------------------------------------
C  AMP2POW  r,function   Relative power corresponding to IGT, real*8
C-----------------------------------------------------------------

      INTEGER*2 IGT

      AMP2POW = IGT * 0.046875
      AMP2POW = 196. - AMP2POW
      AMP2POW = 10.0 ** (AMP2POW/10.)
      AMP2POW = AMP2POW / 1.E9
      RETURN
      END


      SUBROUTINE AVGPLTS(PRF,ARCDAT)
C-----------------------------------------------------------------
C
C   Calculate the average power and plot: time (x) vs. average power (y)
C     and average power (x) vs. altitude (y).
C-----------------------------------------------------------------
C
C         This program was written by personnel from
C         NOAA/ERL/Wave Propagation Laboratory/Atmospheric Lidar Group
C
C         Software Responsibility : A. Weickmann    (303) 497-6595
C         Scientific Responsibility : M.J. Post     (303) 497-6048
C
C         Last Modified :   8/21/89   by  AMW
C         Last Modified :   1/07/91   by  AMW  Improve inputs
C-----------------------------------------------------------------
C         INPUTS
C Variable  type,common      Description
C --------  -----------  ----------------------------------------
C   ARCDAT    i,passed   Option to archive the beta profile data Y or N
C   AVGPWR( ) r,signl    Average power array
C   IPLTA     i,vars     Plot to screen =0 or plotter =1 or both =2
C   LINDB     i,plot     Linear or decibel scaling
C   LOHI      i,plot     Low to high scaling flag
C   NREC      i,vars     # of records to process
C   NPTS      i,header   Number of points (or samples) to be processed
C   PRF       r,passed   Pulse Repetition Frequency in Hz
C   PWR( )    r,signl    Signal power array
C   R1        r,header   Begin range, adjusted for lidar altitude and
C                                  elevation angle
C   R2        r,header   End range, adjusted for lidar altitude and
C                                  elevation angle
C   SATCOR    i,vars     Digitizer saturation correction flag
C
C         OUTPUTS
C Variable  type,common      Description
C --------  -----------  ----------------------------------------
C   AVGPWR( ) r,signl    Average power array
C   CLRAIR    r,vars     Altitude of the last clear air (km)
C   XMAX      r,plot     Maximum x defining the graphic data window
C   XMIN      r,plot     Minimum x defining the graphic data window
C   YMAX      r,plot     Maximum y defining the graphic data window
C   YMIN      r,plot     Minimum y defining the graphic data window
C-----------------------------------------------------------------

      INCLUDE "CB.HEADER"
      INCLUDE "CB.PLOT"
      INCLUDE "CB.SIGNL"
      INCLUDE "CB.VARS"

      INTEGER*2 I, ARCDAT
      REAL*4 AVPWDB, PRF, ARAY(2048)
      REAL*4 C0,C1,C2,C3, APWR,COR
      CHARACTER*1 GRFTYP, CANS

 1111 FORMAT(A1)
 3333 format(f20.0)

C     These are the digitizer correction factors to be applied to
C      averaged power  if SATCOR = 11 OR 21   11/8/89

      C0 = 1.27525E+00
      C1 = -3.06986E-04
      C2 = -2.01185E-08
      C3 = 5.54436E-11

C     Compute the average power, thus far the AVGPWR array has
C      contained the sum of the power, must divide by NREC to get
C      the average power.

      DO 20 I=1,NPTS
         AVGPWR(I) = AVGPWR(I) / NREC
         IF(SATCOR.EQ.11 .OR. SATCOR.EQ.21) THEN
                APWR = AVGPWR(I)
                COR = C0 + (C1*APWR) + (C2*(APWR**2)) + (C3*(APWR**3))
                AVGPWR(I) = 2.0 * COR * APWR
         ENDIF
         ARAY(I) = AVGPWR(I)
 20   CONTINUE

      IF(LINDB.EQ.'D ') THEN
C     Convert power to DB.
         DO 40 I=1,NPTS
            AVPWDB = 10.0 * ALOG10(AVGPWR(I))
            ARAY(I) = AVPWDB
 40      CONTINUE
      ENDIF


C     Set up the axis scaling for linear or dB.

C     (linear)
      IF (LINDB .EQ. "L ") THEN
            IF (LOHI .EQ. "H ") THEN
                YMAX = 20000.0
            ELSE IF (LOHI .EQ. "M ") THEN
                YMAX = 2000.0
            ELSE IF (LOHI .EQ. "L ") THEN
                YMAX = 400.0
            ENDIF

C     (deciBell)
      ELSE
            IF (LOHI .EQ. "H ") THEN
                YMAX = 40.0
            ELSE IF (LOHI .EQ. "M ") THEN
                YMAX = 20.0
            ELSE IF (LOHI .EQ. "L ") THEN
                YMAX = 12.0
            ENDIF
      ENDIF

      XMAX = NREC / PRF

C      Plot time vs. Average power.

      GRFTYP = 'T'
      CALL PLOTIT (PWR,NREC,  GRFTYP)
      IF(IPLTA.EQ.0 .OR. IPLTA.EQ.2) READ(*,1111) CANS
      CALL NEWPAG
```

```
C       Plot Average power vs alitiude.  R1 and R2 are ranges
C       corrected for elevation angle and lidar altitude, in effect
C       they are altitudes.

        YMIN = 0.0
        YMAX = R2

        IF (LINDB .EQ. "D ") THEN
              IF (LOHI .EQ. "H ") THEN
                  XMAX = 40.0
              ELSE IF (LOHI .EQ. "M ") THEN
                  XMAX = 20.0
              ELSE IF (LOHI .EQ. "L ") THEN
                  XMAX = 12.0
              ENDIF
        ELSE
              IF (LOHI .EQ. "H ") THEN
                  XMAX = 20000.0
              ELSE IF (LOHI .EQ. "M ") THEN
                  XMAX = 2000.0
              ELSE IF (LOHI .EQ. "L ") THEN
                  XMAX = 400.0
              ENDIF
        ENDIF

        if(iplta.eq.1) goto 50
        IF(ARCDAT.EQ.'Y ') THEN
              CALL NEWPAG
45            WRITE(*,900) OP
              WRITE(*,*) 'Take note of the altitude of the last clear air'
              WRITE(*,*) ' You will be asked for it right after this plot'
              WRITE(*,*)
              print *,'   Press return to continue '
              read (*,1111,err=45) cans
        ENDIF
900     FORMAT(/,' ****  ',A2,'   ****',/)

50      continue

C       Plot average power vs altitude

        GRFTYP = 'R'
        CALL PLOTIT (ARAY,NPTS,  GRFTYP)
        IF(IPLTA.EQ.0 .OR. IPLTA.EQ.2) READ(*,1111) CANS
        CALL NEWPAG
        IF(ARCDAT.EQ.'Y ') THEN
60            PRINT *
              PRINT *,' Are there clouds present in this profile (Y or N)'
              PRINT *,'  (default = N)'
              PRINT *,' ? : '
              READ (*,1111,err=60) CANS

              if(cans.eq.'N' .or. cans.eq.'n' .or. cans.eq.' ') then
                  clrair = 0.0
              else if(cans.eq.'Y' .or. cans.eq.'y') then
65                PRINT *
                  PRINT *,' Enter the altitude of the last clear air (km)'
                  PRINT *,'   on an even 0.5 km interval.'
                  PRINT *,' ? : '
                  READ (*,3333,err=65) CLRAIR
              else
                  print *
                  print *,' Your answer was not an option, try again.'
                  goto 60
              endif
        endif
        CALL NEWPAG

        RETURN
        END
```

```
        LOGICAL FUNCTION BEFORE(MO,DAY,YR,IHEAD)
C-----------------------------------------------------------------------
C  This routine determines if this file on the tape was made before
C  The date specified in YR,MO, and DAY.  IHEAD is the header record
C  of the tape file.  BEFORE will return true if the date on the tape
C  is before the date specified.
C-----------------------------------------------------------------------
C
C         This program was written by personnel from
C         NOAA/ERL/Wave Propagation Laboratory/Atmospheric Lidar Group
C
C         Software Responsibility : A. Weickmann    (303) 497-6595
C         Scientific Responsibility : M.J. Post     (303) 497-6048
C
C         Last Modified :   8/09/89   by  AMW
C-----------------------------------------------------------------------
C         INPUTS
C Variable  type,common      Description
C --------  ----------  ----------------------------------------------
C  IHEAD( )  i,passed    Integer header from tape
C  MO        i,passed    Month (1 - 12)
C  DAY       i,passed    Day (1 - 31)
C  YR        i,passed    Year (1 - 99)
C
C         OUTPUTS
C Variable  type,common      Description
C --------  ----------  ----------------------------------------------
C  BEFORE    l,function  .true. or .false. as described above
C-----------------------------------------------------------------------

          INTEGER *4 MO,DAY,YR
          INTEGER *2 IHEAD(200)

          IF(IHEAD(3) .LT. YR) GOTO 10
          IF(IHEAD(3) .GT. YR) GOTO 20
          IF(IHEAD(1) .LT. MO) GOTO 10
          IF(IHEAD(1) .GT. MO) GOTO 20
          IF(IHEAD(2) .LT. DAY) GOTO 10
          IF(IHEAD(2) .GE. DAY) GOTO 20

10        BEFORE = .TRUE.
          GOTO 30

20        BEFORE = .FALSE.

30        RETURN
          END
```

```
        SUBROUTINE BETARCH
C-----------------------------------------------------------------------
C         Archive a beta profile into a beta archive file.
C-----------------------------------------------------------------------
C
C         This program was written by personnel from
C         NOAA/ERL/Wave Propagation Laboratory/Atmospheric Lidar Group
C
C         Software Responsibility : A. Weickmann    (303) 497-6595
C         Scientific Responsibility : M.J. Post     (303) 497-6048
C
C         Modified :   8/09/89   by amw
C         Modified :   1/07/91   by amw  improve inputs
C-----------------------------------------------------------------------
C         INPUTS
C Variable  type,common      Description
C --------  ----------  ----------------------------------------------
C   ALT( )     r,signl    Altitude array, km
C   ALTLID     r,vars     Lidar altitude in km ASL
C   ARCNAM     c,charac   Archive disk filename
C   AVNOIS     r,calc     Average noise in noise window
C   AVSGNL( )  r,signl    Average power at each altitude
C   AZIMTH     r,vars     Azimuth angle of scanner, deg w.r.t north
C   BEGALT     r,calc     Beginning altitude of the noise window, km asl
C   BETACO( )  r,signl    Array to store the BETA coefficients
C   BW         r,vars     Bandwidth of signals in MHz
C   CALDAT     i,calc     Calibration date MMDDYY
```

```
C   CLRAIR    r,vars     Altitude of the last clear air (km)
C   DALT      r,calc     Requested altitude increment for averaging
C                         processed data (as opposed to ZINC)
C   DBCAL     r,calc     Continuous wave calibration SNR, dB
C   ELEVA     r,vars     Elevation angle of scanner, deg above horizon
C   ENDALT    r,calc     Ending altitude for the noise window, km asl
C   ICOUNT    i,calc     Number of altitude increments in the profile
C   IHEAD( )  i,header   Integer header from tape
C   IREC      i,vars     Start record number
C   IREND     i,vars     End record number
C   ISITE     i,vars     Site number from lidarsite file
C   MODEL     i,vars     Atmospheric absorption model
C   NNOIS     i,calc     Number of noise samples in the noise window
C   NQC       i,calc     Quality control for calibration and beta
C   OP        i,vars     Operator's initials
C   QUAL      r,signl    The quality factor threshold value.
C   RLAM      r,vars     Wavelength,  meters
C   SDNOIS    r,calc     Standard deviation of noise in noise window
C   SHTDAT    r,calc     Shot noise for the data run, (new lidar)
C   SHTREF    r,calc     Shot noise for the calibration run, (new)
C
C      OUTPUTS
C Variable  type,common      Description
C --------  -----------  ----------------------------------------
C   ARCNAM    c,charac   Archive disk filename
C-----------------------------------------------------------------

      INCLUDE "CB.VARS"
      INCLUDE "CB.CALC"
      INCLUDE "CB.HEADER"
      INCLUDE "CB.SIGNL"
      INCLUDE "CB.CHARAC"

      LOGICAL EX
      CHARACTER*1 CANS
      INTEGER*4 NUMBET, JDATE, JTIME, MDATE(3), MTIME(3)
      INTEGER*4 BEGREC,ENDREC,BLAREC,BQC,PDATE,PTIME,ARCTYP
      INTEGER*4 IDUM, IDATE, ITIME, IYR, IT,NUMREC, IREAD,NUMO
      REAL*4 ADUM,CLOUD, WAVE, BAND
      CHARACTER*20 CBLANK
      CHARACTER*30 CIDIRF
      CHARACTER*25 CIPROF

1111  FORMAT(A1)
2222  format(I20)

      ARCTYP=1
      ADUM=999.99
      IDUM=99999
      CBLANK='
      CIDIRF=CBLANK//CBLANK(1:10)
      CIPROF=CBLANK//CBLANK(1:5)

      BQC=1
      WRITE(*,*) "Archive filename is  ",ARCNAM

      IF (IHEAD(8) .EQ. 2) THEN
2        PRINT *,' This is MOMENT DATA, i.e. increment of .3 km'
         PRINT *,' Do you want it archived to this file? (Y or N)'
         PRINT *,'  (default = Y)'
         PRINT *,' ? : '
         READ (*,1111,err=2) CANS
         IF (CANS .EQ. 'N' .OR. CANS .EQ. 'n') THEN
                ARCNAM= CBLANK//CBLANK(1:5)
5               PRINT *
                PRINT *, "Enter the new file name (max 25 chars) : "
                READ (*,'(A25)',err=5) ARCNAM
            else if(cans.ne.'Y' .and. cans.ne.'y' .and. cans.ne.' ') then
                print *
                print *,' Your answer was not an option, try again.'
                goto 2
            endif
      ENDIF

      IF(DALT.NE.0.5 .and. IHEAD(8) .NE. 2)THEN
            PRINT *, "Archived data must be at 0.5 km increments,"
            PRINT *, " or be moment. This data will not be archived."
            RETURN
      ENDIF

      CLOUD=CLRAIR

3     IF(NQC.EQ.0) THEN
            BQC=0
      ELSE
            CALL ERASE
            PRINT *
4           PRINT *
            PRINT *, " Is the quality of the beta data known "
            PRINT *, " to be good (1) or were there assumptions "
            PRINT *, " made in the calculations (0). Enter 1 or 0 "
            PRINT *,' ? : '
            READ (*,2222,err=4) IQC
            IF(IQC.EQ.0) BQC=0
      ENDIF

6     PRINT *
      PRINT *
      PRINT *,' Would you like to : '
      PRINT *,'          Continue with program  -  C '
      PRINT *,'          Modify recent inputs   -  M '
      PRINT *,'     (default = C) '
      PRINT *,' ? : '
      READ (*,1111,err=6) CANS

      if(cans.eq.'C' .or. cans.eq.'c' .or. cans.eq.' ') then
            goto 8
      else if(cans.eq.'M' .or. cans.eq.'m') then
            CALL ERASE
            GOTO 3
      else
            print *
            print *,' Your answer was not an option, try again.'
            goto 6
      endif

8     continue

C     Compute number of records, numrec for incoming data
      NUMREC=ICOUNT+3
      IDATE=IHEAD(1)*10000 + IHEAD(2)*100 + IHEAD(3)
      ITIME=IHEAD(4)*10000 + IHEAD(5)*100 + IHEAD(6)

      I=INDEX(ARCNAM,' ')
      CIPROF(1:I-1)=ARCNAM(1:I-1)
      CIDIRF(1:I+4)=ARCNAM(1:I-1)//'.DRCT'

      INQUIRE(FILE=CIDIRF,EXIST=EX)

      OPEN(UNIT=50,FILE=CIDIRF,ACCESS="DIRECT",RECL=46,
     +     FORM="UNFORMATTED",STATUS="UNKNOWN")

      OPEN(UNIT=52,FILE=CIPROF,ACCESS='DIRECT',RECL=48,
     +     FORM='UNFORMATTED',STATUS='UNKNOWN')

      IF(EX) THEN
            PRINT *
            PRINT *, " Archive file exists, info will be appended"
            READ(50,REC=1)ARCTYP,NUMBET
            IF(ARCTYP.NE.1) THEN
                PRINT *,' Your are trying to write to a non-beta'
                PRINT *,'  archive file.  This can not be done.'
                GOTO 100
            ENDIF

C     Read thru directory to find empty record gaps

        DO 20 I=2,NUMBET+1
            READ(50,REC=I) NUMO,JDATE,JTIME,CLD,BEGREC,ENDREC,JQC,
     +                     JSITE,WAVE,BAND
            IF(NUMO.GT.0) GOTO 20
            BLAREC=ENDREC-BEGREC+1
            IF(NUMREC.LE.BLAREC) THEN
                    ENDREC=BEGREC+NUMREC-1
                    IREAD=I
                    NUMO=ABS(NUMO)
```

```
                    GOTO 50
                ENDIF
 20           CONTINUE

C         NO BLANK RECORD SPACES OF LARGE ENOUGH SIZE WERE FOUND
C         SO ADD TO END OF DATABASE.
C         SET POINTERS TO WRITE INTO DATA BASE THEN WRITE

              BEGREC=ENDREC+1
              ENDREC=BEGREC+NUMREC-1
              NUMBET=NUMBET+1
              IREAD=NUMBET+1
              NUMO=ABS(NUMO)+1

          ELSE
C             File is new, must set value of numbet
              PRINT *
              PRINT *, " Archive file is new "

 30           NUMBET=1
              IREAD=NUMBET+1
              BEGREC=1
              ENDREC=BEGREC+NUMREC-1
              NUMO=1
          ENDIF

 50       CONTINUE
          WAVE=RLAM*1.E6
          BAND=BW/1.E6
          CALL DATE(MDATE)
          CALL TIME(MTIME)
          IYR=MDATE(1)-1900
          PDATE=MDATE(2)*10000 + MDATE(3)*100 + IYR
          PTIME=MTIME(1)*10000 + MTIME(2)*100 + MTIME(3)

          WRITE(52,REC=BEGREC) IREC,IREND,ICOUNT,MODEL,ALT(1),
     +     ALT(ICOUNT),DALT,ALTLID
          WRITE(52,REC=BEGREC+1) BEGALT,ENDALT,AVNOIS,SDNOIS,
     +     CALDAT,DBCAL,SHTREF,SHTDAT,IHEAD(8)
          WRITE(52,REC=BEGREC+2) ELEVA,AZIMTH,PDATE,PTIME,OP,
     +     QUAL,ADUM,IDUM,IDUM,IDUM,IDUM
          IT=1
          DO 70 IR=BEGREC+3,ENDREC
              WRITE(52,REC=IR) ALT(IT), BETACO(IT), AVSGNL(IT),
     +                         ADUM,ADUM,ADUM,ADUM,ADUM,ADUM,ADUM
              IT=IT+1
 70       CONTINUE

C         NOW WRITE INTO THE DIRECTORY AND UPDATE NUMBET

          PRINT *
          PRINT *,' Profile : number, date, time =  ',NUMO,IDATE,ITIME
          WRITE(50,REC=1) ARCTYP, NUMBET
          WRITE(50,REC=IREAD) NUMO,IDATE,ITIME,CLOUD,BEGREC,ENDREC,BQC,
     +     ISITE,WAVE,BAND

          PRINT *
          WRITE(*,*) "Data has been archived in file "//ARCNAM

 100      CONTINUE

          CLOSE(50)
          CLOSE(52)

          RETURN
          END
```

```
      SUBROUTINE CALBETA
C-----------------------------------------------------------------------
C
C  This routine calculates the BETA array and Signal to Noise (SNR)
C        array.  Beta and SNR minimum and standard deviation arrays are
C        also computed.
C
C        BTYPE = R -> do running average
C              =     B -> do block average
C              =     A -> do archive run
C              (i.e., block average with 0.5 km altitude increment, and
C               start at altitude ending in .0 or .5.)
C-----------------------------------------------------------------------
C
C        This program was written by personnel from
C        NOAA/ERL/Wave Propagation Laboratory/Atmospheric Lidar Group
C
C        Software Responsibility : A. Weickmann    (303) 497-6595
C        Scientific Responsibility : M.J. Post     (303) 497-6048
C
C        Modified :   7/31/89   by  AMW
C        Modified :   8/29/90   by  RPM - comments fixed/added
C        Modified :   9/05/90   by  RPM - running averaging added
C-----------------------------------------------------------------------
C      INPUTS
C Variable  type,common      Description
C --------  ----------  ----------------------------------------------
C   ABSRP( )  r,signl    Absorption array, in units of km-1
C   ALTLID    r,vars     Lidar altitude in km ASL
C   AVGPWR( ) r,signl    Average power array
C   AVNOIS    r,calc     Average noise in noise window
C   BTYPE     i,vars     Beta calc. type:  A=archive, B=block, R=running avg.
C   BW        r,vars     Bandwidth of signals in MHz
C   CAL       r,calc     Calib factor for adjusting beta retrieval
C   COSRAD    r,vars     Cosine of zenith angle of scanner (radians)
C   DALT      r,calc     Requested altitude increment for averaging
C                        processed data (as opposed to ZINC)
C   DGTZRT    r,header   Digitizer rate, MHz
C   DIGALT    r,header   Altitude increment per digitizer sample
C   IHEAD( )  i,header   Integer header from tape
C   IPULSE    i,panel    Number of pulses averaged in processor
C   NNOIS     i,calc     Number of noise samples in the noise window
C   NPTS      i,header   Number of points (or samples) to be processed
C   NREC      i,vars     # of records to process
C   P1          r,calc     P1 and P3 are used in the BETA equation, please
C   P3        r,calc       see BETA.F for more details.
C   R1        r,header   Begin range, adjusted for lidar altitude and
C                          elevation angle
C   RLAM      r,vars     Wavelength,  meters
C
C      OUTPUTS
C Variable  type,common      Description
C --------  ----------  ----------------------------------------------
C   ALT( )    r,signl    Altitude array, km
C   AVSGNL( ) r,signl    Average power at each altitude
C   BETACO( ) r,signl    Array to store the BETA coefficients
C   BETAMN( ) r,signl    Array to store minimum BETA values
C   BETASD( ) r,signl    Array to store BETA std. dev. values
C   DALT      r,calc     Requested altitude increment for averaging
C                        processed data (as opposed to ZINC);
C                      for moment data, DALT is set to DIGALT;
C                      note that this is changed to 0.5 if BTYPE='A '
C   ICOUNT    i,calc     Number of altitude increments in the profile
C   QFAC( )   r,signl    Q-factor array
C   QUAL      r,signl    The quality factor threshold value.
C   SNR( )    r,signl    Array to store the Signal to Noise Ratios
C   SNRSDD( ) r,signl    Array to store SNR std.dev. in DB
C   ZINC      r,calc     Actual altitude increment for averaging
C                        processed data (as opposed to DALT)
C
C-----------------------------------------------------------------------

      INCLUDE "CB.CALC"
      INCLUDE "CB.HEADER"
      INCLUDE "CB.PANEL"
      INCLUDE "CB.SIGNL"
      INCLUDE "CB.VARS"
```

```
      REAL*4  DEPSAM, RNREC, QUALST, QUALLN, BEAM(60)
      REAL*4 TEMP1, BWMHZ, ADJST, DBADJ, DBSNR, SIGNOS, AVSIG
      REAL*4 ALTK, ARCRNG, RNG, R, DIFRAN, DRANG, RECEF, ABSP
      REAL*4 TMPSNR, SNRMIN, SNRSD, DALTX
      INTEGER*4 IRA, IRA1, IRNG, IRNG1, LOOP
      INTEGER*2 NINC, ITEMP1, ITEMP2, ICM1

      character*1 cans

C     DEPSAM, The number of dependent samples.
C     RNREC, Float(NREC), So the sqrt of nrec can be taken
C     QUALST, The stringent threshold value.
C     QUALLN, The lenient threshold value, (used in comparisons).


C     Fill the BEAM array with receiver efficiencies knowing the
C     wavelength of the data.

      CALL RECEFF(RLAM,BEAM)

C  Begin calculating BETA Signal to Noise Ratio's (SNR'S) for given
C  altitude averaging intervals.

      IF (IHEAD(8) .EQ. 2) DALT = DIGALT
      NINC = DALT / DIGALT
      ZINC = NINC * DIGALT
      TEMP1 = NNOIS * DGTZRT / 10.
      RNREC = NREC
      BWMHZ = BW / 1.E6

C     For now (7/89) the qual factor can not be computed for moment
C     data.

      IF(IHEAD(8).EQ.2) GOTO 13

C    MJ decided that both calculations of QUAL were correct and that
C    both should be displayed on the printout, but that the more
C    lenient should be used in the calculation of beta minimum and
C    standard deviation.

      DEPSAM = 1.
      IF ( DGTZRT .GT. BWMHZ) DEPSAM = DGTZRT/BWMHZ
      QUALST = 1/SQRT(NINC/DEPSAM-1) + 1/SQRT(NNOIS/DEPSAM-1)
      QUALLN = SQRT(1/(NINC/DEPSAM-1) + 1/(NNOIS/DEPSAM-1))
      QUAL = QUALLN

      IF (BTYPE.NE.'R ') THEN
            WRITE(40,900) QUALST
            WRITE(40,901) QUALLN
      ENDIF
 900  FORMAT(" Quality Factor Threshold = ",F8.2,"   (2.6% bad data)")
 901  FORMAT(" Quality Factor Threshold = ",F8.2,"   (16% bad data)")

C     Compute altitude array first then compute the index
C     into the data arrays, K,  from the altitude.  This can be
C     done knowing that
C        ALT( ) = R1 + (K*DELALT)  + ARCRNG
C    If the data is to be archived, alt values should be at even
C     0.5 km increments.  ARCRNG is computed as the difference
C     between R1 + DALT/2 and the next 0.5 increment.
C     The value of RANGE1 has been checked in NEWCALIB.F for a
C     value greater than the minimum range of the lidar.  If it is
C     less than the minimum range, it is set equal to the minimum
C     range. The data less than the minimum range is no good and
C     should not be used for computations.
C
 13   ARCRNG=0.0

C  Calculate number of altitude increments (output points).
      IF (BTYPE.EQ.'R ') THEN
            LOOP = NPTS - NINC + 1
      ELSE
            LOOP = NPTS / NINC
      ENDIF

C  Calculate ARCRNG -- when archiving raw data only.
      IF(IHEAD(8).EQ.2) GOTO 16
```

```
      IF (BTYPE.EQ.'A ') THEN
            IF(DALT.NE.0.5) DALT=0.5
            ALTA=R1+0.25

            DO 15 IJ=1,LOOP
                  ALTB=0.5*FLOAT(IJ)
                  IF(ALTB-ALTA .LT. 0.0) GOTO 15
                  ARCRNG=ALTB-ALTA
                  GOTO 16
 15         CONTINUE
      ENDIF

C  Calculate altitudes.
 16   ALT(1) = R1 + ARCRNG + (DALT/2.)

      IF (BTYPE.EQ.'R ') THEN
            DALTX = DIGALT
      ELSE
            DALTX = DALT
      ENDIF

      DO 18 IJ=2,LOOP
            ALT(IJ) = ALT(IJ-1) + DALTX
 18   CONTINUE

C     ----------------------------
C     Main loop to calculate betas
C     ----------------------------

      DO 100 ICOUNT=1,LOOP

            ALTK=ALT(ICOUNT)-(DALT/2.)
            K=((ALTK - R1 - ARCRNG)/DIGALT)

            AVSIG = 0.
            DO 30 I = 1, NINC
                  AVSIG = AVSIG + AVGPWR(K + I)
 30         CONTINUE
            AVSIG = AVSIG / NINC

C     if attenuation is equal to 0.0 use unattenuated average noise
C      to calculate the SNR. This hook enables noise to be entered
C      when the noise from the noise window is no good, as in low
C      elevation scans.  5/24/90 amw as per mj post.

            IF(ATTEN.GE.0.0) THEN
                  SIGATN = AVSIG - AVNOSA
                  MFACTR = 10 ** (ATTEN/10.)
                  AVSGNL(ICOUNT) = SIGATN * MFACTR
                  SIGNOS = AVNOIS / SQRT(RNREC)
                  QFAC(ICOUNT) = AVSGNL(ICOUNT) / SIGNOS
                  SNR(ICOUNT) = AVSGNL(ICOUNT) / AVNOIS
                  if(atten.eq.0.0) snr(icount) = avsgnl(icount)/avnosa
            ELSE
                  AVSGNL(ICOUNT) = AVSIG
                  SIGNOS = AVNOIS / SQRT(RNREC)
                  QFAC(ICOUNT) = (AVSIG - AVNOIS) / SIGNOS
                  SNR(ICOUNT) = (AVSIG - AVNOIS) / AVNOIS
            ENDIF

C - Adjust SNR & QFAC according to # of pulses avg, if this is moment
C - data.

            IF (IHEAD(8) .NE. 1) THEN
                  IF (SNR(ICOUNT) .GT. 0) THEN
                        DBSNR = 10. * LOG10(SNR(ICOUNT))
                        ADJST = SNRADJ(DBSNR, IPULSE)
                        DBADJ = DBSNR - ADJST
                        SNR(ICOUNT) = 10. ** (DBADJ/10.)
                        QFAC(ICOUNT) = ( SNR(ICOUNT) * SQRT(RNREC) * AVNOIS*
     +                                   10. ** (ADJST/10.) ) / SIGNOS
                  ENDIF
            ENDIF

            IF (SNR(ICOUNT) .LT. 0.0) SNR(ICOUNT) = 0.0
            RNG  =  (ALT(ICOUNT) - ALTLID) / COSRAD
            R = RNG * 1000.

C           Receiver efficiency data is at 0.5 km intervals from 0.5 km
C           to 30.0 km range.  Altitude is converted to range knowing
C           the  altitude of the lidar and receiver efficiencies are
C           interpolated between the known range values.
C           Molecular Absorption data is at 1 km intervals from 1 km
C           to 30.0 km altitude ASL.  Absorption coeff for altitudes other
C           than at the even altitude increments must be interpolated.

C           Compute range index for receiver efficiencies.

            IRNG = INT(RNG)
            DIFRAN = RNG - IRNG

            IF(DIFRAN.GE.0.5) THEN
                  DIFRAN = (DIFRAN - 0.5) / 0.5
                  IRNG = (IRNG*2) + 1
                  IRNG1 = IRNG + 1
                  GOTO 50
            ENDIF
            IRNG = IRNG*2
            IRNG1 = IRNG+1
            DIFRAN = DIFRAN / 0.5

 50         CONTINUE

C           Compute altitude index for absorption coefficients.

            IRA = ALT(ICOUNT)
            IF( IRA .LE. 0) IRA = 1
            IRA1 = IRA + 1
            DRANG = ALT(ICOUNT) - IRA

            IF (IRNG .GE. 59) THEN
                  RECEF = BEAM(60)
            ELSE
                  RECEF = (BEAM(IRNG1) - BEAM(IRNG)) * DIFRAN + BEAM(IRNG)
            ENDIF
            IF (IRA .GE. 29) THEN
                  ABSP = ABSRP(30)
            ELSE
                  ABSP = (ABSRP(IRA1) - ABSRP(IRA)) * DRANG + ABSRP(IRA)
            ENDIF

            P4 = (P1 * (R ** 2))  / (P3 * RECEF * ABSP)

            BETACO(ICOUNT) = P4 * SNR(ICOUNT) * CAL


C           This section of code calculates parameters BETAMN and
C           BETASD, which are, respectively, the minimum acceptable
C           BETA values and the BETA standard deviation. The BETASD
C           is then graphed on the BETA plot.

            SNRMIN = QUAL / SQRT(RNREC)
C                 SNRSD = SNRMIN + SNR(ICOUNT) / SQRT(TEMP1 * RNREC)
            SNRSD = SNRMIN
            IF (SNR(ICOUNT) .NE. 0) THEN
                  TMPSNR = SNRSD / SNR(ICOUNT)
            ELSE
                  TMPSNR = 1
            ENDIF
            IF ( TMPSNR .LT. 0.0 ) TMPSNR = 1.0E-10
            SNRSDD(ICOUNT) = 10  * ALOG10(TMPSNR)
            BETAMN(ICOUNT) = P4 * SNRMIN * CAL
            BETASD(ICOUNT) = P4 * SNRSD * CAL

100   CONTINUE

      ICOUNT=LOOP

      RETURN
      END


      SUBROUTINE CALGET(IGREC,DBCAL)
```

```
C-----------------------------------------------------------------
C         This routine computes the dB down from the measured versus
C         expected power SNR.
C-----------------------------------------------------------------
C
C         This program was written by personnel from
C         NOAA/ERL/Wave Propagation Laboratory/Atmospheric Lidar Group
C
C         Software Responsibility : A. Weickmann    (303) 497-6595
C         Scientific Responsibility : M.J. Post     (303) 497-6048
C
C         Modified : 8/09/89 by AMW
C         Modified : 8/16/90 by RPM - added dummy character, C, to READ
C                    of CALARC.DRCT file.
C-----------------------------------------------------------------
C          INPUTS
C Variable  type,common      Description
C --------  -----------  ----------------------------------------
C  IGREC      i,passed   Record number in calibration data file
C
C      OUTPUTS
C Variable  type,common      Description
C --------  -----------  ----------------------------------------
C   DBCAL     r,passed  Continuous wave calibration SNR, dB
C-----------------------------------------------------------------

          INTEGER*2 DATIME(6)
          INTEGER*4 BREC,EREC,IGREC,IQUAL
          REAL*4 DBCAL
          REAL*4 RBW,WL,RALT,TE,SN,ATT,AVSIG,PSP,E,MES,PP,SNRMEA
          REAL*4 SIGH,SIGC,EES,SNREXP,SNRMDB,SNREDB
          CHARACTER*31 CFILE,CFILE1
          CHARACTER*15 SYSFOC
          CHARACTER*1 C

          CFILE='/usr/local/database/CALARC.DRCT'
          CFILE1='/usr/local/database/CALARC.INFO'

          OPEN(30,FILE=CFILE,ACCESS='DIRECT',RECL=40,FORM='UNFORMATTED',
     +                STATUS='OLD')
          OPEN(32,FILE=CFILE1,ACCESS='DIRECT',RECL=46,FORM='UNFORMATTED',
     +                STATUS='OLD')

          READ(30,REC=IGREC) BREC,EREC,C,(DATIME(I),I=1,6),RBW,WL,IQUAL

C         Read parameters from this calibration

C         The measured SNR is the 6th item on the 6th line.
C         The expected SNR is the 10th item on the 6th line.

          READ(32,REC=BREC+5) AVSIG,PSP,E,MES,PP,SNRMEA,
     +                        SIGH,SIGC,EES,SNREXP

          CLOSE(30)
          CLOSE(32)

C         Compute the DBCAL =  EXPECTED - MEASURED
          SNRMDB=ALOG10(SNRMEA)*10.
          SNREDB=ALOG10(SNREXP)*10.

          DBCAL=SNREDB-SNRMDB

          RETURN
          END


          SUBROUTINE CALR1(BEGRAN, MINRNG)
C-----------------------------------------------------------------
C
C   This subroutine calculates the suggested beginning range.
C
C-----------------------------------------------------------------
C
C         This program was written by personnel from
C         NOAA/ERL/Wave Propagation Laboratory/Atmospheric Lidar Group
C
C         Software Responsibility : A. Weickmann    (303) 497-6595
C         Scientific Responsibility : M.J. Post     (303) 497-6048
C
C         Last Modified : 8/20/90 by RPM
C-----------------------------------------------------------------
C      INPUTS
C Variable  type,common      Description
C --------  -----------  ----------------------------------------
C  ALTLID     r,vars   Lidar altitude in km ASL
C  MINRNG     r,passed       Minimum range in km
C
C      OUTPUTS
C Variable  type,common      Description
C --------  -----------  ----------------------------------------
C  BEGRAN     r,passed       Suggested beginning range
C
C-----------------------------------------------------------------

          INCLUDE "CB.VARS"

          REAL*4 BEGRAN, MINRNG, ALTMIN, ALTBEG, TRUNC, FRAC

C  Calculate minimum altitude at which can get data.
          ALTMIN = ALTLID + MINRNG

C  Want to start at the next increment ending in .25 or .75, so
C  for altitude resolution of 0.5 km, centers will end in .0 or .5
C  (for archiving at 0.5 km).
          TRUNC = AINT(ALTMIN)
          FRAC = ALTMIN - TRUNC

          IF (FRAC .GE. 0.0 .AND. FRAC .LE. 0.25) THEN
               ALTBEG = TRUNC + 0.25
          ELSE IF (FRAC .GT. 0.25 .AND. FRAC .LE. 0.75) THEN
               ALTBEG = TRUNC + 0.75
          ELSE
               ALTBEG = TRUNC + 1.25
          ENDIF

C  Subtract off the altitude of the lidar to get range
C  (i.e., range 0 is at the altitude of the lidar).
          BEGRAN = ALTBEG - ALTLID

C  Make sure the range is >= the minimum range.
          IF (BEGRAN .LT. MINRNG) BEGRAN = MINRNG

          RETURN
          END


          SUBROUTINE CALREC(DATE,TIME,NQC,CDATE,CTIME,ICREC,ICERR)
C-----------------------------------------------------------------
C         This subroutine searches the directory in the calibration
C         archive for the date, time and record number for
C         the nearest data to the incoming date and time.
C         In general, the calibration chosen is the closest to
C         the data date.  The exception occurs when an equipment
C         change has occurred then the previous calibration to
C         the data date is used.
C-----------------------------------------------------------------
C
C         This program was written by personnel from
C         NOAA/ERL/Wave Propagation Laboratory/Atmospheric Lidar Group
C
C         Software Responsibility : A. Weickmann    (303) 497-6595
C         Scientific Responsibility : M.J. Post     (303) 497-6048
C
C         Last Modified :   8/09/89   by  AMW
C-----------------------------------------------------------------
C      INPUTS
C Variable  type,common      Description
C --------  -----------  ----------------------------------------
```

```
C   BW        r,vars      Bandwidth of signals in MHz
C   DATE      i,passed    Data date, MMDDYY
C   RLAM      r,vars      Wavelength,  meters
C   TIME      i,passed    Data time, HHMMSS
C
C      OUTPUTS
C Variable  type,common      Description
C --------  -----------  ------------------------------------------
C  CDATE     i,passed    MMDDYY calibration date
C  CTIME     i,passed    HHMMSS calibration time
C  ICREC     i,passed    Record number in calibration archive directory
C                           for valid calibration.
C  ICERR     i,passed    Error code =0 no error`in database access
C                                                        =-1 only equip data brackets
DATE
C                                                        =-2 equip and data on same
day and
C                                                                no   calib   on
correct side of DATE
C                                                        =-3  no  calib  records in
file, or no
C                                                                calib available.
C   NQC       i,passed    Quality control for calibration and beta
C                                                          1 = good      0 = bad
C-----------------------------------------------------------------------
```

```
      INCLUDE "CB.VARS"

      INTEGER*2 DATIME(6)
      INTEGER*4 IIDATE, ICREC, ICERR, BREC, EREC
      INTEGER*4 JD,JYR,JMO,JDY,JHR,JMN
      INTEGER*4 ARCTYP, DATE,TIME,CDATE,CTIME,IQUAL
      REAL*4 BAND,BAND1, WAVE, WAVE1,PRCISN
      REAL*4 TIME1,TIME2,TIME3,FJD
      CHARACTER*1 TYPE, JTYPE, LTYPE
      CHARACTER*31 CFILE
      CHARACTER*1 C


      ICERR=0
      JDATE=0
      LDATE=999999
      IFLAG=0
      LREC=0
      JREC=0

C     Convert incoming date to YYMMDD for comparisons
      IMM=DATE/10000
      IDD=MOD(DATE,10000)/100
      IYY=MOD(DATE,100)
      IIDATE=IYY*10000+IMM*100+IDD


      CFILE='/usr/local/database/CALARC.DRCT'
      PRINT *
      PRINT *,' Searching calibration file ',CFILE
      PRINT *

      OPEN(30,FILE=CFILE,ACCESS='DIRECT',RECL=40,FORM='UNFORMATTED',
     +     STATUS='OLD',ERR=800)


      READ(30,REC=1) ARCTYP,NCAL
      IF(NCAL.EQ.0) THEN
            ICERR=-3
            RETURN
      ENDIF
      WAVE1 = RLAM*1.0E6
      BAND1 = BW

C     Wavelengths stored in the calibration database are stored at
C      2 decimal place precision (i.e. 10.59). Beta uses 3 decimal
C      places so we look for wave values plus/minus prcisn to
C      encompass the value in the 3rd decimal place.  It is the
C      nature of the line strength database that 2 usable CO2 lines
C      are no closer than 0.01 microns apart.
      PRCISN=0.035
```

```
      DO 40 I=2,NCAL+1
            READ(30,REC=I) BREC,EREC,TYPE,(DATIME(M),M=1,6),BAND,
     +                     WAVE,IQUAL
            IF(DATIME(1).EQ.99) GOTO 40
            IF(BAND.EQ.-1.0 .AND. WAVE.EQ.-1.0) GOTO 10
            IF(BAND.NE.BAND1) GOTO 40
            IF(WAVE.LT.(WAVE1-PRCISN).OR.WAVE.GT.(WAVE1+PRCISN)) GOTO 40
            IDATE=DATIME(3)*10000 + DATIME(1)*100 + DATIME(2)
10          ITIME=DATIME*(4)*10000 + DATIME(5)*100 + DATIME(6)

C           Incoming record date coincides with an existing calibration
C           data record.
C           If calibration and equipment dates are same then:
C           If data taken before equipment change use previous
C           calibration.  If data taken after equipment change use
C           next calibration.
            IF(IDATE.EQ.IIDATE)THEN
                  IF(TYPE.EQ.'E') THEN
                        IFLAG=1
                        KDATE=IDATE
                        KTIME=ITIME
                        GOTO 40
                  ELSE
                        ICREC=I
                        CDATE=IDATE
                        CTIME=ITIME
                        NQC=IQC
                        RETURN
                  ENDIF
            ENDIF

          IF(IDATE.LT.IIDATE) THEN
                  IF(IDATE.GE.JDATE) THEN
                        IF(ITIME.LT.JTIME .AND. IDATE.EQ.JDATE) GOTO 40
                        JDATE=IDATE
                        JTIME=ITIME
                        JREC=I
                        JTYPE=TYPE
                        JQC=IQC
                  ENDIF
                  GOTO 40
            ENDIF
            IF(IDATE.GT.IIDATE) THEN
                  IF(IDATE.LE.LDATE) THEN
                        IF(ITIME.GT.LTIME .AND. IDATE.EQ.LDATE) GOTO 40
                        LDATE=IDATE
                        LTIME=ITIME
                        LREC=I
                        LTYPE=TYPE
                        LQC=IQC
                  ENDIF
            ENDIF
40    CONTINUE

C     If there were calibration entries bracketing the
C     incoming date, JDATE and LDATE contain the before and
C     after dates.  Now must check for record type. If data
C     is bracketed by two equipment records, routine will
C     return with a negative record number. The closest
C     non-equipment record will be used as the calibration.
C     Special case is when the data and equipment fall on the
C     same date.  Then times have to be checked and closest
C     calibration relative to the data but on the correct side
C     of the equipment change is selected


80    CONTINUE

      IF (JTYPE.EQ.'E'.AND.LTYPE.EQ.'E') THEN
            ICERR=-1
            RETURN
      ENDIF

C     Equipment and data on same day
      IF(IFLAG.EQ.1) THEN
            IF(TIME.GT.KTIME)THEN
                  IF(LDATE.EQ.999999) THEN
                        ICERR=-2
```

```
                RETURN
            ENDIF
            GOTO 60
        ELSE
            IF(JDATE.EQ.0) THEN
                ICERR=-2
                RETURN
            ENDIF
            GOTO 70
        ENDIF
    ENDIF

C   Bracketed by one equipment record

    IF(LTYPE.EQ.'E') GOTO 70

    IF(JTYPE.EQ.'E') GOTO 60

C   Check to make sure a calibration exists on both sides
C   of the incoming date, if not take the one that does.
    IF (LDATE .EQ. 999999)   GOTO 70
    IF (JDATE .EQ. 0) GOTO 60

C   Choose closest calibration
    READ(30,REC=JREC) BREC,EREC,C,(DATIME(I),I=1,5)
    JYR=INT4(DATIME(3))
    JMO=INT4(DATIME(1))
    JDY=INT4(DATIME(2))
    JHR=INT4(DATIME(4))
    JMN=INT4(DATIME(5))
    JD=0
    FJD=0

    CALL COMPJD(JYR,JMO,JDY,JHR,JMN,JD,FJD)
   TIME1=FLOAT(JD)+FJD

    READ(30,REC=LREC) BREC,EREC,C,(DATIME(I),I=1,5)
    JYR=INT4(DATIME(3))
    JMO=INT4(DATIME(1))
    JDY=INT4(DATIME(2))
    JHR=INT4(DATIME(4))
    JMN=INT4(DATIME(5))
    JD=0
    FJD=0

    CALL COMPJD(JYR,JMO,JDY,JHR,JMN,JD,FJD)
    TIME3=FLOAT(JD)+FJD


    CLOSE(30)

    IMM=DATE/10000
    IDD=MOD(DATE,10000)/100
    IYY=MOD(DATE,100)
    IHH=TIME/10000
    IMIN=MOD(TIME,10000)/100
    CALL COMPJD(IYY,IMM,IDD,IHH,IMIN,JD,FJD)
    TIME2=FLOAT(JD)+FJD

    IF((TIME3-TIME2).GE.(TIME2-TIME1)) GOTO 70

C   Use calibration following the incoming date if one exists.
 60 IF (LDATE .EQ. 999999) THEN
        ICERR = -3
        RETURN
    ENDIF
    ICREC=LREC
    IYY=LDATE/10000
    IMM=MOD(LDATE,10000)/100
    IDD=MOD(LDATE,100)
    CDATE=IMM*10000 + IDD*100 + IYY
    CTIME=LTIME
    NQC=LQC
    RETURN

C   Use calibration before the incoming date if one exists.
70  IF (JDATE .EQ. 0) THEN
        ICERR = -3
        RETURN
    ENDIF
    ICREC=JREC
    IYY=JDATE/10000
    IMM=MOD(JDATE,10000)/100
    IDD=MOD(JDATE,100)
    CDATE=IMM*10000 + IDD*100 + IYY
    CTIME=JTIME
    NQC=JQC
    RETURN

800 PRINT *,' Calibration archive file can not be accessed.'
    ICERR=-3
    RETURN

    END
```

```
      SUBROUTINE CODECONV (IPAN1, IPAN2, IPAN3, PRF,
     &          IPULSE, ILAG, IWIDTH, IATTEN, RDELAY, TINC, GATEIN)
C -----------------------------------------------------------------
C       This program converts the code from the 3 processor front panel
C       switches as per Page # 29,SP10B Doppler Lider Signal Processor
C       Operating manual.
C-----------------------------------------------------------------
C
C       This program was written by personnel from
C       NOAA/ERL/Wave Propagation Laboratory/Atmospheric Lidar Group
C
C       Software Responsibility : A. Weickmann   (303) 497-6595
C       Scientific Responsibility : M.J. Post    (303) 497-6048
C
C       Last Modified :   8/10/89 by  AMW
C-----------------------------------------------------------------
C       INPUTS
C Variable  type,common    Description
C --------  -----------    ------------------------------------
C       IPAN1   i,passed  I*2 variables corresponding to words 1-3 from
C       IPAN2   i,passed   the processor front panel switches. As of now,
C       IPAN3   i,passed   these are obtained from words 4-6 of lidar
C                                                  moment data records.
C       PRF    i,passed   Pulse Repetition Frequency in Hz
C
C       OUTPUTS
C Variable  type,common    Description
C --------  -----------    ------------------------------------
C       GATEIN       r,panel       Range increment for processor data, km
C       IATTEN       i,panel       Attenuation shown on processor panel, dB
C       ILAG         i,panel       Lag as it appears on processor panel
C       IPULSE       i,panel       Number of pulses averaged in processor
C       IWIDTH       i,panel       Range gate width as it appears on panel, us
C       RDELAY       r,panel       Delay as it appears on panel (us = microsec)
C       TINC         r,panel       Time increment between pulse (s)

C-----------------------------------------------------------------

      INTEGER *2 IPAN1, IPAN2, IPAN3
      INTEGER *2 IPULSE,ILAG,IATTEN
      INTEGER *2 ITEM,IPAV,IIP1,IIP2,IIP3,IP2,IP3
      INTEGER *2 IILAG
      INTEGER *2 IIAT,IIAT1,IIAT2,IAT,IAT2
      INTEGER *2 IDEL,IKD2,ID1,ID2,IID2,IID3,IKD1
      INTEGER *2 ID2M,ID3,IFD1,IFD2,IK
      INTEGER *2 ISEV,ITSE,ITSX,IFT,IHEX
      INTEGER *2 IFN,IEN,ISK,ISTK
      INTEGER *2 IOSK,IF4,IOK,INN,ISN
      INTEGER *2 IATT
      INTEGER *2 IIWIDTH,IWIDTH,ITN

      REAL *4 TINC,RDELAY
      REAL *4 GATEIN

C -----------------------------------------------------------------

C     Speed of light (xE5 km/sec)
      SPDLGT = .299792458

      IFN   = -4
      IEN   = -8
      ISK   = 1600K
      ITN   = -10
      ISTK    = 6000K
      IOSK    = 160K
      IF4   = 400K
      IOK   = 1000K
      INN   = -9
      ISN   = -6

      ISEV    = 1777K
      ITSE    = 17K
      ITSX    = 360K
      IFT   = 1400K
      ITEM    = IPAN1
      IPAV    = iand2 (ISEV,ITEM)
      IIP1    = iand2 (ITSE,IPAV)
      IIP2    = iand2 (ITSX,IPAV)
      IIP3    = iand2 (IFT ,IPAV)
      IP2   = ishft2( IIP2,IFN)
      IP3   = ishft2( IIP3,IEN)
      IPULSE      = IIP1+10*IP2+100*IP3

      IF (PRF .GT. 0) THEN
          TINC =  IPULSE  / PRF
      ELSE
          TINC = 0
      END IF

      IHEX = 16000K
      IILAG  = iand2 (IHEX,ITEM)
      ILAG    = ishft2(IILAG,ITN)+1

      ITEM    = IPAN2
      IIWIDTH  = iand2 (ISTK,ITEM)
      IWIDTH      = ishft2 (IIWIDTH,ITN)+1

      GATEINC = FLOAT(IWIDTH) * SPDLGT / 2.

      ITEM    = IPAN3
      IATT    = iand2(ISEV,ITEM)
      IIAT1  = iand2(  ITSE,IATT)
      IIAT2  = iand2( IOSK,IATT)
      IAT2    = ishft2(IIAT2,IFN)
      IATTEN      = IIAT1+10*IAT2

      ITEM    = IPAN2
      IDEL    = iand2 (ISEV,ITEM)
      IKD2    = iand2 (ISK,IATT)
      ID1   = iand2 (  ITSE,IDEL)
      IID2    = iand2 ( ITSX,IDEL)
      IID3    = iand2 ( IF4,IDEL)
      IKD1    = iand2 (IOK,IDEL)
      ID2   = ishft2( IID2,IFN)
      ID3   = ishft2( IID3,IEN)
      IFD1    = ishft2( IKD1,INN)
      IFD2    = ishft2( IKD2,ISN)
      IK     = ior2  ( IFD1,IFD2)

      RDELAY      = FLOAT(ID1+10*ID2+100*ID3)+FLOAT(IK)/16.0

      RETURN
      END
```

```
      SUBROUTINE COMPJD(JYR,JMNTH,JDAY,JHR,JMN,JD,FJD)
C-----------------------------------------------------------------
C       Computes julian day and fraction from year, month,
C       day, and time in Universal time (ie. 0 Z to 23 Z)
C       Accurate only between March 1,1900 and February 28, 2100
C       based on julian calendar corrected to correspond to
C       gregorian calendar during this period.
C-----------------------------------------------------------------
C
C       This program was written by personnel from
C       NOAA/ERL/Wave Propagation Laboratory/Atmospheric Lidar Group
C
C       Software Responsibility : A. Weickmann   (303) 497-6595
C       Scientific Responsibility : M.J. Post    (303) 497-6048
C
C       Last Modified :   8/10/89 by  AMW
C-----------------------------------------------------------------
C       INPUTS
C Variable  type,common    Description
C --------  -----------    ------------------------------------
C       JYR         i,passed       Year 0 to 99
C       JMNTH       i,passed       Month 1 to 12
C       JDAY      i,passed  Day 1 to 31
C       JHR         i,passed       Hour  0 to 23
C       JMN         i,passed       Minute  0 to 59
```

```
C
C         OUTPUTS
C Variable  type,common     Description
C --------  -----------  ----------------------------------------
C         JD                  i,passed      Julian Day
C         FJD            r,passed        Fractional Julian Day
C-----------------------------------------------------------------

          INTEGER*4 NDM(12)
          INTEGER*4 JDOR,JYR19,JD
          INTEGER*4 JYR,JMNTH,JDAY,JHR,JMN
          REAL*4 FJD

C         JDOR=JD OF DECEMBER 30, 1899 AT 12 HOURS UT
          DATA JDOR/2415019/, JYR19/1900/
          DATA NDM/0,31,59,90,120,151,181,212,243,273,304,334/

          JD=JDOR
          JYRM9=JYR
          LP=JYRM9/4
          IF(LP.EQ.0) GOTO 4
          JD=JD+1461*LP
 4        NY=JYRM9-4*LP
          IC=0
          IF(NY.GT.0) GOTO 5
          IF(JMNTH.GT.2) IC=1
          GOTO 6
 5        JD=JD+365*NY+1
 6        JD=JD+NDM(JMNTH)+JDAY+IC
          IF(JHR.GE.12) GOTO 7
          JD=JD-1
          FJD= 0.5+0.041666667*FLOAT(JHR)+0.00069444444*FLOAT(JMN)
          RETURN
 7        FJD=0.041666667*FLOAT(JHR-12)+0.00069444444*FLOAT(JMN)
          RETURN
          END


          SUBROUTINE DECODER
C-----------------------------------------------------------------
C
C         This routine reads the IDAT array from tape, decodes the
C          data into R and Q components, sets scaling for plots
C          and plots single plots if requested.
C
C-----------------------------------------------------------------
C
C         This program was written by personnel from
C         NOAA/ERL/Wave Propagation Laboratory/Atmospheric Lidar Group
C
C         Software Responsibility : A. Weickmann   (303) 497-6595
C         Scientific Responsibility : M.J. Post     (303) 497-6048
C
C         Modified :  8/01/89        by  AMW
C          Modified : 8/21/90        by  TG  - A/D avg energy modification added
C         Modified :  1/07/91        by  amw  improve inputs
C-----------------------------------------------------------------
C         INPUTS
C Variable  type,common     Description
C --------  -----------  ----------------------------------------
C         IHEAD( )  i,header         Integer header from tape
C         IPACK()       i,setup      MTAPE array     * DO NOT CHANGE *
C         IPLTA         i,vars       Plot to screen =0 or plotter =1 or both =2
C         IREC          i,vars       Requested begin record number
C         ISTART        i,header     Starting sample index for data within a record
C         ISTOP         i,header     Stopping sample index for data within a record
C         LINDB         i,plot       Linear or decibel scaling
C         LOHI          i,plot       Low to high scaling flag
C         NPTS          i,header     Number of points (or samples) to be processed
C         NPTSPP        r,header     Number of points (or samples) per pulse
C         NREC          i,vars       # of records to process
C         PARM( )       r,header     Real header from tape
C         PLTTYP        i,plot       Flag for single or average plots
C
C         OUTPUTS
C Variable  type,common     Description
```



```
C --------  -----------  ----------------------------------------
C         ARAY( )       r,passed     Array used in graphing single plots
C         AVGPWR( ) r,signl Average  power array
C         AVEENG        r,vars       A/D Average energy per pulse
C         AZIMTH        r,vars       Azimuth angle of scanner, deg w.r.t north
C         GATEIN        r,panel      Range increment for processor data, km
C         IATTEN        i,panel      Attenuation shown on processor panel, dB
C         ICURR         i,setup      Current record number
C         IDAT( )       i,signl      Array to store data from tape
C         ILAG          i,panel      Lag as it appears on processor panel
C         IPULSE        i,panel      Number of pulses averaged in processor
C         IWIDTH        i,panel      Range gate width as it appears on panel, us
C         MFAVPW( ) r,multfl         Power avgd over multfiles
C         MFNMPL        i,multfl     Number of pulses in MFAVPW array
C         PWR( )        r,signl      Signal power array
C         RDELAY        r,panel      Delay as it appears on panel (us = microsec)
C         SATCOR        i,vars       Digitizer saturation correction flag
C         TINC          r,panel      Time increment between pulse (s)
C         XMAX          r,plot       Maximum x defining the graphic data window
C-----------------------------------------------------------------

          INCLUDE "CB.HEADER"
          INCLUDE "CB.MULTFL"
          INCLUDE "CB.PANEL"
          INCLUDE "CB.PLOT"
          INCLUDE "CB.SETUP"
          INCLUDE "CB.SIGNL"
          INCLUDE "CB.VARS"

          REAL*8 RQ
          CHARACTER*1 CANS, GRFTYP
          REAL*4 ENERGMON,DUM,THRESH, TKM1,ARAY(2048), PPWR(2048)
          INTEGER * 4  IBYTES, IER, NBYTES, yymmdd
          INTEGER * 2 IERFLG, IPASS, JJ, K, KM1, L, GLITCH
          LOGICAL MOMENT, LENERR, ANSWER, BEFORE

1111      FORMAT(A1)

2         PRINT *
          PRINT *,' Do you want the data decoded using : '
          PRINT *,'                          R2Q2  -  B '
          PRINT *,'                          R2  -  R '
          PRINT *,'                          Q2  -  Q '
          PRINT *,' (default = B)'
          PRINT *,' ? : '
          READ (*,1111,err=2) CANS

          if(cans.eq.' ' .or. cans.eq.'B' .or. cans.eq.'b') then
               satcor = 0
4              print *
               print *,' Do you want to De-Glitch the data  (Y or N)'
               print *,'  (default = N)'
               print *,' ? : '
               read (*,1111,err=4) cans

               if(cans.eq.' ' .or. cans.eq.'N' .or. cans.eq.'n') then
                    glitch = 0
               else if(cans.eq.'Y' .or. cans.eq.'y') then
                    glitch = 1
               else
                    print *
                    print *,' Your answer was not an option, try again.'
                    goto 4
               endif
          else if(cans.eq.'R' .or. cans.eq.'r') then
               satcor = 20
          else if(cans.eq.'Q' .or. cans.eq.'q') then
               satcor = 10
          else
               print *
               print *,' Your answer was not an option, try again.'
               goto 2
          endif

          if(satcor.eq.0) goto 6

5         PRINT *
          PRINT *,' Do you want to apply a saturation correction to '
```

```
      PRINT *,'                    Unaveraged data  -  U '
      PRINT *,'                Time averaged data  -  T '
      PRINT *,' (default = U) '
      PRINT *,' ? : '
      READ (*,1111,err=5) CANS

      if(cans.eq.' ' .or. cans.eq.'U' .or. cans.eq.'u') goto 6
      if(cans.eq.'T' .or. cans.eq.'t') then
            satcor = satcor + 1
      else
            print *
            print *,' Your answer was not an option, try again.'
            goto 5
      endif

6     continue

      LENERR = .TRUE.
      IF (IHEAD(8) .EQ. 1) THEN
            MOMENT = .FALSE.
      ELSE
            MOMENT = .TRUE.
      ENDIF

      IERFLG = 1

8     IF (MOMENT) THEN
            IBYTES = 304*2
      ELSE
            IBYTES = (NPTSPP + 16) * 2
      END IF

      ANSWER = BEFORE(5,24,88,IHEAD)

      PRINT *
      PRINT *,' The data tape is being accessed... '

      yymmdd = (ihead(3)*10000) + ihead(1)*100 + ihead(2)

      ENERGMON=0.0
      DO 90 K = 1, NREC
            IF (IERFLG .EQ. 0) GO TO 9
            CALL MTREAD(IPACK, IDAT, IBYTES, NBYTES, IER)
            GO TO (9, 9, 120, 125, 130, 130, 140, 130, 150, 140,
     +            130, 130, 140, 140) IER

9           CONTINUE

C - If this is moment data, we need to re-pack IDAT array such that it
C - contains only amplitude values for all of the range gates.
C       Beginning on 5/24/88 the azimuth angle was written to tape
C       differently.

            ENERGMON=ENERGMON+REAL(IDAT(NPTSPP+3))
            IF (MOMENT) THEN
                  IF (K .EQ. 1) THEN

C                 Get the azimuth angle from tape.
                        IF (ANSWER) THEN
                              AZIMTH  = IDAT(2)/10
                        ELSE
                              AZIMTH = IDAT(2)/100
                              IF (AZIMTH .LT. 0) THEN
                                    AZIMTH = AZIMTH + 655.36
                              ENDIF
                        ENDIF

                        CALL CODECONV (IDAT(4), IDAT(5), IDAT(6), PRF,
     +                IPULSE, ILAG, IWIDTH, IATTEN, RDELAY, TINC, GATEIN)
                        WRITE(*,20) IPULSE
20                      FORMAT(/,9X,"Number of pulses averaged: ", T45, I8)
                        WRITE(*,21) ILAG
21                      FORMAT(9X,"Lag: ", T45, I8)
                        WRITE(*,22) IWIDTH
22                      FORMAT(9X,"Gate Width (us): ",T45, I8)
                        WRITE(*,23) IATTEN
23                      FORMAT(9X,"Attenuation: ", T45, I8)
                        WRITE(*,24) RDELAY
24                      FORMAT(9X,"Delay (us): ", T45, F8.2)
                        WRITE(*,25) TINC
25                      FORMAT(9X,"Time increment per sample: ", T45, F8.2)
                        WRITE(*,26) GATEIN
26                      FORMAT(9X,"Gate Width (km): ", T45, F8.3,/)
                  ENDIF

                  L = 0
                  DO 30 I = 14,302,3
                        L = L + 1
                        IDAT(L) = IDAT(I)
30                CONTINUE

            ELSE

C           Get the azimuth angle from tape.
                  IF (K .EQ. 1) then
                        AZIMTH = IDAT(PARM(3) + 1)/100
                        IF (AZIMTH .LT. 0) THEN
                              AZIMTH = AZIMTH + 655.36
                        ENDIF
                  ENDIF

            ENDIF

C       Set flag to MTREAD next time thru loop

            IERFLG = 1
            KM1 = K - 1
            TKM1 = KM1 / 10.
            JJ  =  1
            DO 10 I = ISTART, ISTOP
            IPASS = I

            IF (MOMENT) THEN
C - 2.50 scales the power returned from AMP2POW to be comparable to
C - power values returned from R2Q2.
                  RQ = 2.50 * AMP2POW(IDAT(IPASS))
            ELSE
                  IF(SATCOR.EQ.0) RQ = R2Q2(IPASS)
                  IF(SATCOR.EQ.10) RQ = TWICEQ2(IPASS)
                  IF(SATCOR.EQ.11) RQ = Q2(IPASS)
                  IF(SATCOR.EQ.20) RQ = TWICER2(IPASS)
                  IF(SATCOR.EQ.21) RQ = FR2(IPASS)
            END IF

C       if the data was taken on or after 11/15/90 new digitizers were
C        used.  This changes the unaveraged data correction values for
C        R2 + Q2 data.  Changes made pursuant to 12/13/90 Post et al
C        memo.      amw 12/13/90

            IF(SATCOR.EQ.0.AND.RQ.GT.2500) THEN
                  RQTEM=RQ
                  if(yymmdd.ge.901115) then
                        rq = 2.18416E03 +
     +                        (rqtem * (-1.01756)) +
     +                        ((rqtem**2) * 6.20471E-04) +
     +                        ((rqtem**3) * (-7.14828E-08)) +
     +                        ((rqtem**4) * (3.07629E-12))

                  else
                        RQ = -7.575E03 + (RQTEM*6.2225) +
     +                        ((RQTEM**2)* (-1.1157E-03)) +
     +                        ((RQTEM**3)* (8.8696E-08))
                  endif
            ENDIF

            IF(SATCOR.EQ.0 .AND. GLITCH.EQ.1) THEN
                  IF (K .GT. 4) THEN
                        THRESH = AVGPWR(JJ) / TKM1
                        IF (RQ .GE. THRESH)  RQ = AVGPWR(JJ) / KM1
                  ENDIF
            ENDIF

            PWR(K) = PWR(K) + RQ
            ARAY(JJ) = RQ
            AVGPWR(JJ) = AVGPWR(JJ) + RQ
            MFAVPW(JJ) = MFAVPW(JJ) + RQ
```

```
                JJ = JJ + 1
10        CONTINUE

                MFNMPL = MFNMPL + 1

C  Set scaling and plot single records if requested

                IF (PLTTYP .EQ. "A ") GO TO 80
                GRFTYP = 'R'
                IF (LINDB .EQ. "L ") GO TO 60
                DO 50 I = 1, NPTS
                      PPWR(I) = ARAY(I)
                      PPWR(I) = 10.0 * ALOG10(PPWR(I))
                      ARAY(I) = PPWR(I)
                      IF (PPWR(I) .EQ. 1.) ARAY(I) = 1.
50              CONTINUE
                GO TO 70

60              CONTINUE

C         Linear scaling

                IF (LOHI .EQ. "H ") THEN
                      XMAX = 20000.0
                ELSE IF (LOHI .EQ. "M ") THEN
                      XMAX = 2000.0
                ELSE IF (LOHI .EQ. "L ") THEN
                      XMAX = 400.0
                ENDIF

                CALL PLOTIT(ARAY,NPTS, GRFTYP)
                IF (IPLTA .EQ. 0 .OR. IPLTA .EQ. 2) READ(*,1111) CANS
                GO TO 80

70              CONTINUE

C         dB scaling

                IF (LOHI .EQ. "H ") THEN
                      XMAX = 40.0
                ELSE IF (LOHI .EQ. "M ") THEN
                      XMAX = 20.0
                ELSE IF (LOHI .EQ. "L ") THEN
                      XMAX = 12.0
                ENDIF

                CALL PLOTIT(ARAY,NPTS,GRFTYP)
                IF (IPLTA .EQ. 0 .OR. IPLTA .EQ. 2) READ(*,1111) CANS

80              CONTINUE
                PWR(K) = PWR(K) / NPTS

90        CONTINUE

C         Update current tape record position after nrec number of reads

          icurr = irec + nrec

          ENERGMON=ENERGMON/REAL(NREC)
          DUM=1.0/1500.0
C
C          A/D avg energy modification added 8/21/90
C          DUM is a variable to hold the calibration factor and ENERGMON
C          is the averaged sum of the idat array from tape.
C
          AVEENG=ENERGMON*(-1)*DUM
91        CONTINUE

          IF (LINDB .EQ. "L ") GO TO 110
          DO 100 K = 1, NREC
                PWR(K) = 10.0 * ALOG10(PWR(K))
100       CONTINUE

110       CONTINUE

          GO TO 160

C************** ERROR MESSAGES ****************
```

```
C          ~~ TAPE ERROR = 3 ~~
120       CONTINUE
          WRITE(*,*) "LENGTH OR PARITY ERROR"
          IF (NBYTES .EQ. 2048) THEN
                IF (LENERR) THEN
                      CALL TPMSG (IER, NBYTES)
                      WRITE(*,*) "THIS IS PROBABLY AN OLD TAPE"
                      LENERR = .FALSE.
                      STOP
                ENDIF
          ELSEIF (NBYTES .EQ. 604) THEN
                IF (LENERR) THEN
                      CALL TPMSG (IER, NBYTES)
                      WRITE(*,*) "MOMENT DATA, I'LL PROCESS IT THAT WAY."
                      MOMENT = .TRUE.
                      STOP
                ENDIF
          ELSE
                      CALL TPMSG (IER, NBYTES)
                      STOP
          ENDIF
          STOP

C          ~~ TAPE ERROR = 4 ~~
125        CONTINUE
           IF (NBYTES .EQ. 2048) THEN
                IF (LENERR) THEN
                CALL TPMSG (IER, NBYTES)
                WRITE(*,*) "THIS IS PROBABLY AN OLD TAPE"
                STOP
                ENDIF
           ELSEIF (NBYTES .EQ. 604) THEN
                IF (LENERR) THEN
                CALL TPMSG (IER, NBYTES)
                WRITE(*,*) "MOMENT DATA, I'LL PROCESS IT THAT WAY."
                MOMENT = .TRUE.
                GOTO 8
                ENDIF
           ELSE
                CALL TPMSG (IER, NBYTES)
                STOP
           ENDIF

C          ~~ TAPE ERROR = 5, 6, 8, 11, 12  ~~
130       CONTINUE
          CALL TPMSG (IER, NBYTES)
          IERFLG = 1
            NREC = K - 1
            GOTO 91

C          ~~ TAPE ERROR = 7,10,13,14  ~~          Exit the subroutine
140       CONTINUE
          CALL TPMSG (IER, NBYTES)
          GO TO 160

C          ~~ TAPE ERROR = 9  ~~    Logical end of tape, rewind and exit.
150       CONTINUE
          CALL TPMSG (IER, NBYTES)
          CALL MTREW (IPACK)

160       CONTINUE
          RETURN
          END


          FUNCTION FR2(I)
C-----------------------------------------------------------------------
C
C  This function decodes the raw data from the lidar digitizer
C  and returns in R2 the power for the point "I".
C
C-----------------------------------------------------------------------
C
C         This program was written by personnel from
C         NOAA/ERL/Wave Propagation Laboratory/Atmospheric Lidar Group
```

```
C
C         Software Responsibility : A. Weickmann   (303) 497-6595
C         Scientific Responsibility : M.J. Post     (303) 497-6048
C
C         Modified :  8/08/89        by  AMW
C         Modified :  9/12/90        by  RPM - fixed comments
C-----------------------------------------------------------------
C         INPUTS
C Variable  type,common     Description
C --------  -----------  ------------------------------------------
C         I                 i,passed    Index into the IDAT raw signal array.
C         IDAT( )      i,signl        Array to store data from tape
C
C         OUTPUTS
C Variable  type,common     Description
C --------  -----------  ------------------------------------------
C         FR2    r,function            Real portion of signal squared plus
C                                                 imaginary portion of signal squared.
C
C-----------------------------------------------------------------

          INCLUDE "CB.SIGNL"

          INTEGER*2 RBYTE, LBYTE
          INTEGER*2 MASK, SHIFT, ITEMP, I

          MASK = 377K
          RBYTE = IAND (IDAT(I), MASK)
          ITEMP = NOT2(MASK)
          LBYTE = IAND (IDAT(I), ITEMP)
          SHIFT = -8
          LBYTE = ISHFT2(LBYTE, SHIFT)
          SHIFT = 7
          IF (BTEST2(RBYTE, SHIFT)) RBYTE = RBYTE - 256
          IF (BTEST2(LBYTE, SHIFT)) LBYTE = LBYTE - 256
          FR2 = (float(RBYTE)+0.7) * (float(RBYTE)+0.7)

          RETURN
          END


          SUBROUTINE INITCO
C-----------------------------------------------------------------
C
C  This routine initializes  the absorption coefficient array for
C  the given lidar altitude (ALTLID) above sea level (ASL) and
C  scanner zenith angle (90 - ELEVA).
C
C-----------------------------------------------------------------
C
C         This program was written by personnel from
C         NOAA/ERL/Wave Propagation Laboratory/Atmospheric Lidar Group
C
C         Software Responsibility : A. Weickmann   (303) 497-6595
C         Scientific Responsibility : M.J. Post     (303) 497-6048
C
C         Modified :  7/27/89        by  AMW
C         Modified :  8/23/90        by  RPM - changed capitilization
C         Modified :  1/16/91        by  amw - added radiosonde profile
C-----------------------------------------------------------------
C         INPUTS
C Variable  type,common     Description
C --------  -----------  ------------------------------------------
C         ALTLID        r,vars        Lidar altitude in km ASL
C         DIGTIM        r,header      Time between digitizer samples
C         ELEVA         r,vars        Elevation angle of scanner, deg above horizon
C         MODEL      i,vars  Atmospheric absorption model
C         RANGE1        r,vars        Initial range, km
C         RANGE2        r,vars        End range, km
C         RLAM          r,vars        Wavelength,  meters
C
C         OUTPUTS
C Variable  type,common     Description
C --------  -----------  ------------------------------------------
C         ABSRP( )  r,signl Absorption array, in units of km-1
C         COSRAD        r,vars        Cosine of zenith angle of scanner (radians)
C         DIGALT        r,header      Altitude increment per digitizer sample
C         R1                r,header    Begin range, adjusted for lidar altitude and
C                                                 elevation angle
C         R2                r,header    End range, adjusted for lidar altitude and
C                                                 elevation angle
C         RSDATA        c,charac      Name of radiosonde data file
C         SONDHT        r,vars        Altitude, m, asl of highest radiosonde data
C-----------------------------------------------------------------

          INCLUDE "CB.HEADER"
          INCLUDE "CB.SIGNL"
          INCLUDE "CB.VARS"
          INCLUDE "CB.CHARAC"

          INTEGER*4 NWAVE, IALT1, IALT2
          integer*2 nwv, modl, ipts
          real*4 wn, zeroht, hgtasl(30), absrpt(30)
          REAL*4 DIFALT, THETAR, DELRNG
          REAL*4 CUMK, WAVELN, WAVE, XABSRP(30,5)

          THETAR = (90 - ELEVA) * 0.017453293
          COSRAD = COS(THETAR)
          WAVELN = RLAM

C          Calculate the altitude offset for the delay
           DELRNG = .149896 * DIGTIM
           DIGALT = DELRNG * COSRAD

C         Derive an atmospheric model from a radiosonde profile.

          if(model.eq.7) then
               modl = 0
               ipts = 30
               wn = (1./rlam) / 100.
               zeroht = 0.0
               if(rlam.eq.1.0591E-05) nwv = 0
               if(rlam.eq.9.282E-06) nwv = 1

               call sondeabsp(wn,sondht,rsdata)
               call crinit('ABSPROFILE',zeroht,zeroht,modl,nwv)
               do 4 i=1,ipts
                    hgtasl(i) = float(i)
 4             continue

               call abscrctn(hgtasl,ipts,eleva,absrpt)

C         Set  absorption values to zero above SONDHT,
C          the maximum radiosonde height.  100mb approximates
C          16 km so the data from 16 km (if its a full radiosonde)
C          to 30 km should be set to zero, which would approximate
C          a normal absorption profile. After the round-trip losses
C          are computed, the value of absrp( ) should equal 1.0
C               exp(0) = 1.0

               open (70,file='LOOKABSP',status='unknown')
               write(70,940)

               do 6 i=1,ipts
                    absrp(i) = (1/absrpt(i))
                    if(i .gt. int(sondht/1000.)) absrp(i) = 1.0
                    write(70,950) hgtasl(i), absrp(i)
 6             continue

 940      format(' altitude  round-trip',/,'        (km)      absorption')
 950      format(f8.2,e13.5)
               close(70)
               goto 90
          endif

C         The absorption data is stored in the text file ABSORP.DAT.
C         The first line contains the number of wavelengths represented
C         in the file. Line 2 has a wavelength number, Lines 3-32
C         contain the absorption coefficients for the 5 atmospheric
C         models at that wavelength. Line 33 has wavelength # 2,
C         Lines 34-63 the absorption coeff, etc for as many wavelengths
C         as specified in the first line of the file.  When the user
C         uses a new wavelength they must first calculate the
C         absorption coefficients and enter them into the file -
```

```
C          ABSORP.DAT

           OPEN(UNIT=15,FILE='/usr/local/database/ABSORP.DAT',
             +      STATUS='OLD',ERR=700)

           READ(15,900) NWAVE

           DO 20 I=1,NWAVE
                 READ(15,910) WAVE
                 DO 10 J=1,30
                 READ(15,920)  (XABSRP(J,K),K=1,5)
 10              CONTINUE
                 IF(WAVE.EQ.WAVELN) GOTO 30
 20        CONTINUE

           CALL NEWPAG
           PRINT *
           PRINT *,' Absorption coefficients for this wavelength are'
           PRINT *,'  not present in the ABSORP.DAT file.  This program'
           PRINT *,'  is terminating, use the ATMABS program to'
           PRINT *,'  generate the coefficients, put them in the data'
           PRINT *,'  file and try again.'
           PRINT *,' Wavelength = ',WAVELN
           CLOSE(15)
           STOP

 30        CONTINUE
           CLOSE(15)

C          Use NO atmospheric model

             IF (MODEL .EQ. 0) THEN
                  DO 35 I=1,30
                 ABSRP(I) = 1.0
 35         CONTINUE
                  GO TO 90
            END IF

           IALT = ALTLID
           IALT1 = IALT + 1
           DIFALT = IALT1 - ALTLID
           IF (IALT .LE. 0) GO TO 50
           DO 40 I = 1, IALT
                 ABSRP(I) = 0.0
 40        CONTINUE

 50        CONTINUE
           IF (IALT1 .LE. 0) GO TO 60
           ABSRP(IALT1) = XABSRP(IALT1, MODEL) * DIFALT

 60        CONTINUE
           IALT2 = IALT1 + 1
           DO 70 KL = IALT2, 30
                 ABSRP(KL) = XABSRP(KL, MODEL)
 70        CONTINUE
           CUMK = 0.0

C  Integrate the absorption coefficients for round-trip
C  loss to all ranges.


           DO 80 I = 1, 30
                 CUMK = CUMK + ABSRP (I) / COSRAD
                 ABSRP(I) = EXP(-2. * CUMK)
 80        CONTINUE

C           Calculate and show to the user the start and end altitudes
C           for the data.
C          Good data does not occur within the minimum range of the lidar
C          so the beginning range should be at least equal to if not
C          greater than the minimum range.
C
 90        R1 = RANGE1 * COSRAD  + ALTLID
           R2 = RANGE2 * COSRAD  + ALTLID

           GOTO 100

 700       PRINT *,' Error opening the ABSORP.DAT file, does it exist ? '
           STOP
 900       FORMAT(2X,I2)
 910       FORMAT(2X,1PE10.4)
 920       FORMAT(5(2X,F6.5))

 100       CONTINUE
           RETURN
           END
```

```
      SUBROUTINE INITZ
C-----------------------------------------------------------------------
C
C       This subroutine initializes variables and the tektronix
C       plotting package.
C-----------------------------------------------------------------------
C
C       This program was written by personnel from
C       NOAA/ERL/Wave Propagation Laboratory/Atmospheric Lidar Group
C
C       Software Responsibility : A. Weickmann   (303) 497-6595
C       Scientific Responsibility : M.J. Post    (303) 497-6048
C
C       Modified :  7/20/89        by  AMW
C       Modified :  8/21/90        by  RPM - fixed typo's in comments
C       Modified :  9/04/90        by  RPM - upped loop 30 from 350 to 2048
C       Modified :  1/04/91        by  AMW - improve inputs
C                   :              2/8/91          by  amw - stop on tape drive error
C-----------------------------------------------------------------------
C
C                 INPUTS
C Variable  type,common              Description
C --------  ----------  ----------------------------------------
C REINIT    1,setup     Re-initialization flag
C
C                 OUTPUTS
C Variable  type,common              Description
C --------  ----------  ----------------------------------------
C ALT ()        r,signal    Altitude array, km
C ALTLID        r,vars      Lidar altitude in km ASL
C AVGPWR ()     r,signal    Average power array
C BETACO ()     r,signal    Array to store the BETA coefficients
C BETAFL        i,calc      Flag to plot the BETA profile Y or N
C ICURF         i,setup     Current file number
C ICURR         i,setup     Current record number
C IPACK()           i,setup     MTAPE array     * DO NOT CHANGE *
C IPLTA         i,vars      Plot to screen =0 or plotter =1 or both =2
C ISITE         i,vars      Site number from lidarsite file
C MFAVPW ()     r,multfl    Power avgd over multfiles
C MFIFN ()      i,multfl    Multi file numbers.
C MFIREC ()     i,multfl    Multi file beg rec number.
C MFNMFI        i,multfl    Number of files averaged.
C MFNMPL        i,multfl    Number of pulses in MFAVGPWR array
C MFNREC ()     i,multfl    Multi file number of recs.
C MULTFI        l,multfl    TRUE if multifiles averaging in effect
C OP                i,vars      Operator's initials
C PWR ()        r,signal    Signal power array
C QFAC ()           r,signal    Q-factor array
C SITNAM        c,charac    Name of lidar site
C SNR ()        r,signal    Array to store the Signal to Noise Ratios
C
C-----------------------------------------------------------------------

      INCLUDE "CB.CALC"
      INCLUDE "CB.CHARAC"
      INCLUDE "CB.PLOT"
      INCLUDE "CB.MULTFL"
      INCLUDE "CB.SETUP"
      INCLUDE "CB.SIGNL"
      INCLUDE "CB.VARS"

      INTEGER*2  I, IANS, ICHARR, INUM, MODE
      INTEGER*4  IERR, INTER,NUMS, NS, ITU, NSITE(50)
      REAL*4 SITALT(50)
      CHARACTER*30 SITES(50)
      CHARACTER*1 CANS

C     Check to see if this is a re-initialization  or
C           the first call to INIT.

      IF (REINIT) GO TO 30

 1111 FORMAT(A1)
 2222 FORMAT(I20)

C  Initialize the TEKTRONIX plot package.
```

```
      PRINT *
      PRINT *,'   BETA program is running  '
 2    PRINT *
      PRINT *," Enter the operator's initials (2 characters)
      PRINT *,' ? : '
      READ (*, 910,err=2) OP
910   FORMAT(A2)

 4    PRINT *
      PRINT *,' Are you using the TEK 4051? (Y or N) '
      PRINT *,'  (default = N)'
      PRINT *,' ? : '
      READ(*, 1111,err=4) CANS

      if(cans.eq.'Y' .or. cans.eq.'y') then
           icharr = 240
      else if(cans.eq.'N' .or. cans.eq.'n' .or. cans.eq.' ') then
           icharr = 960
      else
           print *
           print *,' Your answer was not an option, try again.'
           goto 4
      endif

 8    PRINT *
      PRINT *,' Plot to screen (0), plotter (1), or both(2) '
      PRINT *,' ? : '
      READ(*, 2222,err=8) IPLTA

      IF (IPLTA.LT.0.OR.IPLTA.GT.2) then
           print *
           print *,' Your answer was not an option, try again.'
           goto 8
      endif

      IF(IPLTA.EQ.1) ICHARR=0

      CALL INITT(ICHARR,IPLTA)

C  Initialize the constant information.

      CALL ERASE
      PRINT *
      PRINT *,'      ~~~ INITIALIZATION CONSTANTS ~~~'
      PRINT *,'  (These inputs are static in this run.) '
 10   PRINT *
      PRINT *,' Enter the magnetic tape unit number      '
      PRINT *,' ? : '
      READ (*,2222,err=10) ITU

      CALL MTINIT(IPACK, ITU, IERR)
      if(ierr.ne.0) then
           print *
           print *,' Tape drive error, is drive on-line ? '
           stop
      endif

      CALL NEWPAG

C     Select site of lidar and retrieve the altitude of lidar.

 14   PRINT *
      PRINT *,' Choose the lidar site from the following list'
      PRINT *,' by the corresponding number.  If the site is '
      PRINT *,' not on the list enter 0 '
      CALL SITE('LIS',NUMS,NSITE,SITES,SITALT)
      NS=NUMS-1
      IF(MOD(NUMS,2).EQ.0) NS=NUMS
      WRITE(*,*)
      DO 15 IK=1,NS,2
           IF(NSITE(IK).GT.0 .AND. NSITE(IK+1).GT.0) THEN
                WRITE(*,940)NSITE(IK),SITES(IK),NSITE(IK+1),SITES(IK+1)
           ELSE
                IF(NSITE(IK).GT.0) WRITE(*,930) NSITE(IK),SITES(IK)
                IF(NSITE(IK+1).GT.0) WRITE(*,935) NSITE(IK+1),SITES(IK+1)
           ENDIF
 15   CONTINUE
```

```
      IF(MOD(NUMS,2).NE.0 .AND. NSITE(NUMS).NE.-1)
     +          WRITE(*,930) NSITE(NUMS),SITES(NUMS)
      PRINT *, '? : '
      READ (*,2222,err=14) ISITE

      IF(ISITE.GT.NUMS) GOTO 14
      IF(ISITE.EQ.0) THEN
            CALL SITE('ADD',NUMS,NSITE,SITES,SITALT)
            ISITE=NUMS
      ENDIF
      ALTLID=SITALT(ISITE)
      SITNAM=SITES(ISITE)

930   FORMAT(1X,I3,1X,A30)
935   FORMAT(T35,1X,I3,1X,A30)
940   FORMAT(1X,I3,1X,A30,1X,I3,1X,A30)

      CALL NEWPAG

C     Initialize current file, record, and number of pulses in multifile
C           average power array.

      ICURF = 0
      ICURR = 1
      MFNMPL = 0


C     This information may be re-initialized.

      PRINT *
      PRINT *
      PRINT *,'    ~~~ CHANGEABLE INITIALIZATIONS ~~~ '
      PRINT *
30      CONTINUE

      BETAFL = "N "

      MFNMFI = 1
34    PRINT *
      PRINT *,' Is this a multiple file run  (Y or N) '
      PRINT *,'    (default = N) '
      PRINT *,' ? : '
      READ (*,1111,err=34)  CANS

      IF (CANS.EQ.'Y ' .OR. CANS.EQ.'y') THEN
36          PRINT *
            PRINT *,' How many files for average ? (max 10) '
            PRINT *,' ? : '
            READ (*,2222,err=36)  MFNMFI
            MULTFI = .TRUE.
      else if(cans.eq.'N' .or. cans.eq.'n' .or. cans.eq.' ') then
            MULTFI = .FALSE.
      else
            print *
            print *,' Your answer was not an option, try again.'
            goto 34
      ENDIF

      IF (MULTFI) THEN
            WRITE(*,*) "REMINDER: when processing multiple files, there "
            WRITE(*,*) "are a number of parameters that must be"
            WRITE(*,*) "consistant from file to file, Eg. digitizer rate"
            WRITE(*,*) "The only ones checked by this program are"
            WRITE(*,*) "number pts/pulse, the digitizer rate"
            WRITE(*,*) "and delay to first sample."
            WRITE(*,*)
            WRITE(*,*)
            PAUSE
            CALL ERASE
      ENDIF

      PRINT *
      PRINT *, 'Input the requested information for each file.'

      DO 2000 I = 1,MFNMFI

            J = I
            IF (MOD(J,3).EQ.0) CALL ERASE

40          PRINT *
            PRINT *, ' Enter the file number '
            PRINT *,' ? : '
            READ (*,2222,err=40)  MFIFN(I)
42          PRINT *
            PRINT *, ' Enter the starting record number '
            PRINT *,' ? : '
            READ (*,2222,err=42)  MFIREC(I)
44          PRINT *
            PRINT *,'  Enter the number of records to process  '
            PRINT *,' ? : '
            READ (*,2222,err=44)  MFNREC(I)

2000      CONTINUE

48      PRINT *
        PRINT *
        PRINT *,' Would you like to : '
        PRINT *,'    Continue with program  -  C '
        PRINT *,'    Modify recent inputs       -  M '
        PRINT *,'    (default = C) '
        PRINT *,' ? : '
        READ (*,1111,err=48) CANS

        IF(CANS.EQ.'M' .OR. CANS.EQ.'m') THEN
              CALL ERASE
              GOTO 30
        else if(cans.ne.'C' .and. cans.ne.'c' .and. cans.ne.' ') then
              print *
              print *,' Your answer was not an option, try again.'
              goto 48
        ENDIF

        print *
        print *
        print *,' Positioning tape to requested file ...'
        print *

C  Initialize power and beta arrays

        DO 60 I = 1, 2048
              AVGPWR(I)  =  0.0
              MFAVPW(I) = 0.0
              ALT(I) = 0.0
              BETACO(I) = 0.0
              QFAC(I) = 0.0
              SNR(I) = 0.0
60      CONTINUE
        DO 70 I=1,5000
              PWR(I) = 0.0
70      CONTINUE

        RETURN
        END


        SUBROUTINE MFCALCAVE (ARAY)
C-----------------------------------------------------------------------
C       Calculates the average power for the multiple file case.
C-----------------------------------------------------------------------
C
C       This program was written by personnel from
C       NOAA/ERL/Wave Propagation Laboratory/Atmospheric Lidar Group
C
C       Software Responsibility : A. Weickmann   (303) 497-6595
C       Scientific Responsibility : M.J. Post    (303) 497-6048
C
C       Last Modified :   8/03/89 by  AMW
C-----------------------------------------------------------------------
C             INPUTS
C Variable  type,common            Description
C --------  ------------  ------------------------------------------
C       AVGPWR( ) r,signl Average power array
C       LINDB        i,plot       Linear or decibel scaling
```

```
C         MFAVPW( ) r,multfl          Power avgd over multfiles
C         MFNMPL        i,multfl      Number of pulses in MFAVPW array
C         NPTS          i,header      Number of points (or samples) to be processed
C
C                OUTPUTS
C Variable  type,common               Description
C --------  -----------  ------------------------------------------
C         AVGPWR( ) r,signl Average power array
C         MFAVPW( ) r,multfl          Power avgd over multfiles
C-----------------------------------------------------------------

          INCLUDE "CB.HEADER"
          INCLUDE "CB.PLOT"
          INCLUDE "CB.MULTFL"
          INCLUDE "CB.SIGNL"

          INTEGER * 2 I
          REAL*4 AVPWDB, ARAY(2048)

C         When running beta with multiple files the power vs alt
C         and time vs power plots are done for each file.  The beta
C         plot is only done on the averaged data from all files.
C         So, the AVGPWR( ) is set equal to MFAVPW( ).

          DO 10 I = 1, NPTS
               MFAVPW(I) = MFAVPW(I) / MFNMPL
               AVGPWR(I) = MFAVPW(I)
               ARAY(I) = MFAVPW(I)
10        CONTINUE

          IF (LINDB .EQ. "D ") THEN
               DO 20  I = 1, NPTS
                    AVPWDB = 10.0 * ALOG10(MFAVPW(I))
                    ARAY(I) = AVPWDB
20             CONTINUE
          ENDIF

          RETURN
          END



          SUBROUTINE NEWCALIB
C-----------------------------------------------------------------
C
C         This subroutine retrieves the calibration factors based on
C          wavelength and bandwidth.  Other system parameters are entered
C          at this point.
C
C-----------------------------------------------------------------
C
C         This program was written by personnel from
C         NOAA/ERL/Wave Propagation Laboratory/Atmospheric Lidar Group
C
C         Software Responsibility : A. Weickmann   (303) 497-6595
C         Scientific Responsibility : M.J. Post     (303) 497-6048
C
C         Modified : 8/15/89 by AMW
C         Modified : 8/20/90 by RPM - added CALR1 for suggesting range
C         Modified : 1/07/91 by AMW - improved inputs
C-----------------------------------------------------------------
C                INPUTS
C Variable  type,common               Description
C --------  -----------  ------------------------------------------
C         IHEAD( )  i,header          Integer header from tape
C         NPTSPP        r,header      Number of points (or samples) per pulse
C         NQC           i,calc        Quality control for calibration and beta
C         REINIT        l,setup       Re-initialization flag
C
C                OUTPUTS
C Variable  type,common               Description
C --------  -----------  ------------------------------------------
C         BW                  r,vars  Bandwidth of signals in MHz
C         CAL           r,calc        Calib factor for adjusting beta retrieval
C         CALDAT        i,calc        Calibration date MMDDYY
C         DBCAL         r,calc        Continuous wave calibration SNR, dB
C         EJ                  r,vars  Energy/pulse (J)
C         ELEVA         r,vars        Elevation angle of scanner, deg above horizon
C         MODEL         i,vars        Atmospheric absorption model
C         P1                  r,calc  P1 and P3 are used in the BETA equation, please
C         P3                  r,calc        see beta.f for more details
C         RANGE1        r,vars        Initial range, km
C         RANGE2        r,vars        End range, km
C         RLAM          r,vars        Wavelength,  meters
C         RSDATA        c,charac      Name of radiosonde data file
C         SHTDAT        r,calc        Shot noise for the data run, (new lidar)
C         SHTREF        r,calc        Shot noise for the calibration run, (new)
C-----------------------------------------------------------------

          INCLUDE "CB.CALC"
          INCLUDE "CB.SETUP"
          INCLUDE "CB.VARS"
          INCLUDE "CB.HEADER"
          INCLUDE "CB.CHARAC"

          REAL*4 MINRNG, BEGRAN, PTEMP, HNU, OPTLOS, HETEFF
          REAL*4 F3, F4, C, PI, D
          INTEGER*4  CDATE, CTIME, DATIME(6)
          CHARACTER*1 CANS

C  Initialize the Signal to Noise Ratio equation constants

 1111     FORMAT(A1)
 2222     FORMAT(I20)
 3333     format(F20.0)

C         D, Diameter of the telescope, meters ; C, speed of light
          D = 0.28
          PI = 3.14159
          C = 2.998E8
          PLANKS=6.6262E-34

C         The HNU factor is wavelength dependent and varies linearly
C          with wavelength.
C         The calculation of ETA has been ETA = 0.1592 * REFFAC.
C          The constant has been broken down into 2 components,
C          the Heterodyne quantum efficiency, HETEFF,  and
C          the Optical Loss, OPTLOS.  These 2 components are wavelength
C          dependent but the values are extremely difficult to derive
C          and are assumed to be close to the existing values.
C          Therefore, the program is set up to use different values if
C          ever obtained, but for now the same values will be used.
C           A. Weickmann as per M.J. Post  12/20/88

               CALL NEWPAG
 5        PRINT *
          PRINT *,' Select the data wavelength : '
          PRINT *,'          10.591 microns  -  A '
          PRINT *,'           9.569 microns  -  B '
          PRINT *,'           9.282 microns  -  C '
          PRINT *,'           other        -  D '
          PRINT *,'  (default = A)'
          PRINT *,' ? : '
          READ(*,1111,err=5) CANS

          if(cans.eq.'A' .or. cans.eq.'a' .or. cans.eq.' ') then
               RLAM = 1.0591E-5
               HNU = (c * planks)/rlam
               HETEFF = 0.40
               OPTLOS = 0.5307
          else IF(CANS.EQ.'B' .OR. CANS.EQ.'b') THEN
               RLAM = 9.569E-6
               HNU =  (c*planks)/rlam
               HETEFF = 0.40
               OPTLOS = 0.5307
          else IF(CANS.EQ.'C' .OR. CANS.EQ.'c') THEN
               RLAM = 9.282E-6
               HNU = (c*planks)/rlam
               HETEFF = 0.40
               OPTLOS = 0.5307
          else IF(CANS.EQ.'D' .OR. CANS.EQ.'d') THEN
 8             PRINT *
               PRINT *,' Enter the wavelength in microns '
```

```
            PRINT *,' ? : '
            READ (*,3333,err=8)  RLAM
            IF(RLAM.GT.10.591 .OR. RLAM.LT.9.250) THEN
                  PRINT *
                  PRINT *,' **** WARNING **** This wavelength is greater'
                  PRINT *,' than 10.591 or less than 9.25 microns.  The'
                  PRINT *,' HNU factor and Receiver Efficiencies will be'
                  PRINT *,' set to the values associated with the nearest'
                  PRINT *,' standard wavelength.'
                  IF(RLAM.GT.10.591) rlam = 10.591
                  IF(RLAM.LT.9.250) rlam = 9.250
            ENDIF
            RLAM = RLAM * 1.0E-6
            hnu= (c*planks)/rlam
            HETEFF = 0.40
            OPTLOS = 0.5307
      else
            print *
            print *,' Your answer was not an option, try again.'
            goto 5
      endif

C  Initialize the CALIBration constants.
C  The REINIT variable is used when processing multiple but
C        unlike files on the tape.  Certain parameters remain
C        the same, like dB down and shot noise (both calibration
C        and data) and atmospheric model so these questions
C        need not be asked again.  Other parameters do change,
C        therefore, if REINIT is true the program skips down to
C        ask the additional questions.

 10   CONTINUE
      IF (REINIT) THEN
            BW=BW/1.E6
            GOTO 1060
      ENDIF

 15   PRINT *
      PRINT *,' Select the data bandwidth : '
      PRINT *,'                10.0 MHz  -  A '
      PRINT *,'                   5.0 MHz  -  B '
      PRINT *,'                   3.0 MHz  -  C '
      PRINT *,'                     other  -  D '
      PRINT *,'  (default = A)'
      PRINT *,' ? : '
      READ (*,1111,err=15) CANS

      if(cans.eq.'a' .or. cans.eq.'A' .or. cans.eq.' ') then
            bw = 10.0
      else if(cans.eq.'b' .or. cans.eq.'B') then
            bw = 5.0
      else if(cans.eq.'c' .or. cans.eq.'C') then
            bw = 3.0
      else if(cans.eq.'d' .or. cans.eq.'D') then
 20         PRINT *
            PRINT *,' Enter the Bandwidth in MHz '
            PRINT *,' ? : '
            READ (*,3333,err=20) BW
      else
            print *
            print *,' Your answer was not an option, try again.'
            goto 15
      endif

      CALL NEWPAG
 24   PRINT *, "Do you want calibration data from the archive"
      PRINT *, "(otherwise manual input)       Y or N "
      PRINT *,' (default = Y)'
      PRINT *,' ? : '
      READ (*,1111,err=24) CANS

      if(cans.eq.'N' .or. cans.eq.'n') then
            goto 40
      else if(cans.ne.'Y' .and. cans.ne.'y' .and. cans.ne.' ') then
            print *
            print *,' Your answer was not an option, try again.'
            goto 24
      endif

      PRINT *
      PRINT *," The archive will be searched for the nearest"
      PRINT *," date to the data.  The values will be displayed"
      PRINT *," before proceeding."

      MDATE=IHEAD(1)*10000 + IHEAD(2)*100 + IHEAD(3)
      MTIME=IHEAD(4)*10000 + IHEAD(5)*100 + IHEAD(6)
      CALL CALREC(MDATE,MTIME,NQC,CDATE,CTIME,ICREC,ICERR)
      CALDAT =CDATE
      IF(ICERR.NE.0) THEN
            IC=IABS(ICERR)
            GOTO (30,32,34) IC
 30   PRINT *, " Only equipment data brackets date"
            GOTO 40
 32   PRINT *, " Equipment and data date  on same date and"
            PRINT *, " there is no calibration data before/after"
            PRINT *, " equipment date."
            GOTO 40
 34   PRINT *, " There is no calibration data"
            GOTO 40
      ENDIF

      CALL CALGET(ICREC,DBCAL)

      DATIME(1)=CDATE/10000
      DATIME(2)=MOD(CDATE,10000)/100
      DATIME(3)=MOD(CDATE,100)
      DATIME(4)=CTIME/10000
      DATIME(5)=MOD(CTIME,10000)/100
      DATIME(6)=MOD(CTIME,100)

      IF(NQC.EQ.0) THEN
            PRINT *, "NOTE : Some of the calibration information is "
            PRINT *, "              assumed."
            PRINT *
      ENDIF
      WRITE(*,900) (DATIME(I),I=1,6)
 36   PRINT *,' Calibration : dB down from expected = ',DBCAL
      PRINT *
      PRINT *,' Is this value ok  (Y or N) ? '
      PRINT *,' (default = Y)'
      PRINT *,' ? : '
      READ (*,1111,err=36) CANS

      IF(CANS.EQ.'N' .OR. CANS.EQ.'n') GOTO 40
      if(cans.ne.'y' .and. cans.ne.'Y' .and. cans.ne.' ') then
            print *
            print *,' Your answer was not an option, try again.'
            goto 36
      endif

      GOTO 50

 40   PRINT *, "Enter the desired values manually"
      PRINT *
      CALDAT =0

      NQC=1
 44     PRINT *,' Enter  CALIBRATION : dB Down From Expected '
      PRINT *,' ? : '
      READ (*,3333,err=44) DBCAL

C The following code was added to account for the variable shot noise
C during data taking.
 50    PRINT *
      PRINT *,' Enter shot noise for the data (DB) '
      PRINT *,' ? : '
      READ (*,3333,err=50) SHTDAT

      CAL = (10.** (dbcal/10.)) / (1. - (10.**(-shtdat/10.)))

 54   PRINT *
      PRINT *,' Choose an atmospheric absorption model '
```

```
      PRINT *,'   0 = No correction for absorption'
      PRINT *,'   1 = Tropical '
      PRINT *,'   2 = Midlatitude Summer '
      PRINT *,'   3 = Midlatitude Winter '
      PRINT *,'   4 = Subarctic Summer '
      PRINT *,'   5 = Subarctic Winter          '
      print *,'   7 = Derive from Radiosonde Profile '
      PRINT *,' ? : '
      READ (*,2222,err=54) MODEL

      if(model.lt.0 .or. model.gt.7) then
            print *
            print *,' Your answer was not an option, try again.'
            goto 54
      endif
      if(model.eq.7) then
            call newpag
 55         print *
            print *,' Enter the radiosonde file name (max 25 char)'
            print *
            print *,'  File must have  altitude (m), pressure (mb),'
            print *,'  temperature (C), relative humidity (%) on'
            print *,'  each data line.'
            print *
            print *,' ? : '
            read (*,'(a25)',err=55) rsdata
      endif

      CALL NEWPAG

1060   CONTINUE

 56   PRINT *
      PRINT *,' Is the Elevation angle of the scanner 90 degrees'
      PRINT *,'   (Y or N)          (default = Y)'
      PRINT *,' ? : '
      READ (*,1111,err=56) CANS

      IF(CANS.EQ.'N' .OR. CANS.EQ.'n') THEN
 58         PRINT *
            PRINT *,' Enter the elevation angle of the scanner (deg.) '
            PRINT *,' ? : '
            READ (*,3333,err=58) ELEVA
      else if(cans.eq.'Y' .or. cans.eq.'y' .or. cans.eq.' ') then
            eleva = 90.
      else
            print *
            print *,' Your answer was not an option, try again.'
            goto 56
      endif

 59   PRINT *
            PRINT *,'  Enter the energy per pulse for the data (J) '
      PRINT *,' ? : '
      READ (*,3333,err=59) EJ

      BW = BW * 1.E6
      P1 = 8. * HNU * BW

      ETA = HETEFF * OPTLOS

      P3 = PI * ETA * EJ * C * D * D

      MINRNG=1.5

C  Get the suggested initial range.
      CALL CALR1(BEGRAN, MINRNG)

C  Prompt user for initial range.
 60   PRINT *
      PRINT *,'  Enter the initial range in km '
      PRINT *,'     (default = ',BEGRAN, ')'
      PRINT *,' ? : '
      READ(*,3333,err=60) RANGE1
      IF (RANGE1.EQ.0.) THEN
            RANGE1 = BEGRAN
      ELSE IF (RANGE1.LT.0.) THEN
            GO TO 60
      ELSE IF (RANGE1.LT.MINRNG) THEN
            PRINT *
            PRINT *, '  Minimum range of ', MINRNG, ' will be used.'
            PRINT *
            RANGE1 = MINRNG
      ENDIF

      PRINT *
      PRINT *,' The number of points per pulse is ', NPTSPP
      PRINT *,'  2048 points - approx 30 km range.'
      PRINT *,'  1024 points - approx 15 km range.'


C  Prompt user for end range.
 70   PRINT *
      PRINT *,'  Enter the end range in km '
      PRINT *,' ? : '
      READ(*,3333,err=70) RANGE2

      if(range2.le.0) then
            print *
            print *,' Your answer was not valid, try again.'
            goto 70
      else if(range2.le.range1) then
            PRINT *
            PRINT *, '  End range must be greater than initial range of ',
     &      RANGE1
            PRINT *
            GO TO 70
      ENDIF

C  Check if user wants to modify inputs.
 80   PRINT *
      PRINT *
      PRINT *,' Would you like to : '
      PRINT *,'     Continue with program  -  C '
      PRINT *,'     Modify recent inputs       -  M '
      PRINT *,'     (default = C) '
      PRINT *,' ? : '
      READ (*,1111,err=80) CANS

      IF(CANS.EQ.'M' .OR. CANS.EQ.'m') THEN
            CALL NEWPAG
            goto 10
      else if(cans.ne.'c' .and. cans.ne.'C' .and. cans.ne.' ') then
            print *
            print *,' Your answer was not an option, try again.'
            goto 80
      ENDIF

 900  FORMAT(2X,"Calibration on : ",2(I2,"/"),I2,2X,2(I2,":"),I2)

      RETURN
      END


      SUBROUTINE NWINDO(COSRAD,ALTLID)
C-----------------------------------------------------------------------
C     This routine gets the user's request for the altitude range of the
C     desired noise window, converts the ranges to the actual points in
C     the pulse, and calculates the average power, (signal return), over
C     that range.  It then calls the routine to calculate the standard
C     deviation of the noise, passing it the sum of squares, and square
C     of sums, of the average power over the noise window.
C-----------------------------------------------------------------------
C
C     This program was written by personnel from
C     NOAA/ERL/Wave Propagation Laboratory/Atmospheric Lidar Group
C
C     Software Responsibility : A. Weickmann   (303) 497-6595
C     Scientific Responsibility : M.J. Post    (303) 497-6048
C
C     Last Modified :  8/07/89 by  AMW
C-----------------------------------------------------------------------
C              INPUTS
```

```
C Variable  type,common              Description
C --------  -----------  ----------------------------------------
C         ARCDAT        i,calc       Option to archive the beta profile data Y or N
C         ALTLID        r,passed     Lidar altitude in km ASL
C         AVGPWR( ) r,signl Average  power array
C         COSRAD        r,passed     Cosine of zenith angle of scanner (radians)
C         DELAY         r,header     Corrected delay to 1st sample (data point) us
C         DGTZRT        r,header     Digitizer rate, MHz
C         IHEAD( )  i,header         Integer header from tape
C         ISTART        i,header     Starting sample index for data within a record
C         NPTS          i,header     Number of points (or samples) to be processed
C         R1                  r,header   Begin range, adjusted for lidar altitude and
C                                                    elevation angle
C         R2                  r,header   End range, adjusted for lidar altitude and
C                             elevation angle
C
C               OUTPUTS
C Variable  type,common              Description
C --------  -----------  ----------------------------------------
C         ATTEN         r,calc       Attenuation factor if present
C         AVNOIS        r,calc       Average noise in noise window
C         AVNOSA        r,calc       Attenuated average noise in noise windo
C         BEGALT        r,calc       Beginning altitude of the noise window, km asl
C         DALT          r,calc       Requested altitude increment for averaging
C                                                    processed data (as opposed
to ZINC)
C         ENDALT        r,calc       Ending altitude for the noise window, km asl
C         NNOIS         i,calc       Number of noise samples in the noise window
C-----------------------------------------------------------------------
```

```
      INCLUDE "CB.CALC"
      INCLUDE "CB.HEADER"
      INCLUDE "CB.SIGNL"

      REAL*4 XTEMP,AVNOIS2,TNOS1,TNOS2,RNOS1,RNOS2,DR
      INTEGER*2 I, MAXPTS, NOS1, NOS2
      CHARACTER*1 CANS

      CALL ERASE
1111  FORMAT(A1)
2222  format(I20)
3333  format(f20.0)
3334  format(2f5.2)

      PRINT *, 'Data above minimum range occur at : '
      PRINT *, 'Start Altitude = ',R1
      PRINT *,' End Altitude = ',R2

C     Get the users request for the noise window altitudes.
10    print *
      PRINT *,' Enter the beginning, ending altitude (km) for the '
      PRINT *,'  Noise Window '
      PRINT *,' ? : '
      READ (*,*,err=10) BEGALT, ENDALT

      IF(IHEAD(8).EQ.2) THEN
            DALT=0.3
            GOTO 100
      ENDIF
      IF(ARCDAT.EQ.'Y ' .OR. ARCDAT.EQ.'y ') THEN
            DALT=0.5
      ELSE
15          PRINT *,' Enter the altitude resolution in km '
            PRINT *,' ? : '
            READ (*,3333,err=15) DALT
      ENDIF

C     if atten = -1.0  no attenuation, use  computed unatten average
C                      noise to compute snr (calbeta.f)
C           if atten = 0.0      no attenuation, use 'user supplied' unatten
C                      average noise to compute snr (calbeta.f)
C     if atten gt 0.0  is attenuation, use 'user supplied' unatten
C                      average noise to compute snr (calbeta.f)

      call newpag
      print *,'   User : '
      print *,'  If you want to enter your own value for '
      print *,'   Unattenuated Average Noise '
      print *,'  and the data is not attenuated, '
      print *,'  answer the following questions as :'
      print *,'   Data attenuated ?    Y '
      print *,'   Attenuation factor ?  0.0'
      print *,'   Unatten. avg Noise ?  your desired value.'
      print *
20    PRINT *
      PRINT *,' Is the data attenuated in this file (Y or N)'
      PRINT *,'   (default = N) '
      PRINT *,' ? : '
      READ (*,1111,err=20) CANS

C     Temporarily hold the Unattenuated average noise value in
C     the variable AVNOSA.  The value will later be swapped.

      if(cans.eq.' ' .or. cans.eq.'N' .or. cans.eq.'n') then
            atten = -1.0
      else if(cans.eq.'Y' .or. cans.eq.'y') then
            call newpag
30          PRINT *
            PRINT *,' Enter the attenuation factor (dB) '
            PRINT *,' ? : '
            READ (*,3333,err=30) ATTEN

35          PRINT *
            PRINT *,' Enter a value for Unattenuated average noise '
            PRINT *,' ? : '
            READ (*,3333,err=35) AVNOSA
      else
            print *
            print *,' Your answer is not an option, try again.'
            goto 20
      endif

C     Get the range gates associated with the altitudes requested.

100   RNOS1 = (BEGALT-ALTLID) / COSRAD
      RNOS2 = (ENDALT-ALTLID) / COSRAD

      DR = 1.0 / DGTZRT

C     Get the actual pulse points that correspond to the
C     requested altitudes.

      TNOS1 = RNOS1 / .149896 - DELAY
      NOS1 = INT(TNOS1 / DR + 0.5)
      TNOS2 = RNOS2 / .149896 - DELAY
      NOS2 = INT(TNOS2 / DR + 0.5)

C     if nos1 is less than one, set to 1
C     if nos2 is gt NPTS, set to npts

      MAXPTS=NPTS

      NOS1  =  NOS1 - ISTART
      NOS2  =  NOS2 - ISTART
      IF (NOS1 .LT. 1)  NOS1 = 1
      IF (NOS2 .GT. MAXPTS)  NOS2 = MAXPTS

      NNOIS = NOS2 - NOS1 + 1
      IF (NNOIS .GT. MAXPTS) THEN
            NOS1 = NOS2 - MAXPTS + 1
            NNOIS = MAXPTS
      ENDIF

C     Initialize the values to be determined

      AVNOIS = 0.
      XTEMP = 0.
      AVNOIS2 = 0.

C     Get the sum and sum of squares of the average power over the
C     noise gates.
      DO 120 I = NOS1, NOS2
            AVNOIS = AVNOIS + AVGPWR(I)
            AVNOIS2 = AVNOIS2 + AVGPWR(I)**2
120   CONTINUE
```

```
C       Get the square of sums of average power.
        XTEMP = AVNOIS**2

C       Get the actual average power over the noise gates.
        AVNOIS = AVNOIS / NNOIS

C       Get the standard deviation of the average noise
        CALL STDDEVNOS(XTEMP,AVNOIS2)

C       Put the noise values in the proper variables
        IF(ATTEN.GT.0.0) THEN
              XTEMP = AVNOIS
              AVNOIS = AVNOSA
              AVNOSA = XTEMP
        ENDIF

        RETURN
        END
```

```
      SUBROUTINE OUTPT
C-----------------------------------------------------------------------
C
C         Designate plotter and printer output for this Beta run.
C
C-----------------------------------------------------------------------
C
C         This program was written by personnel from
C         NOAA/ERL/Wave Propagation Laboratory/Atmospheric Lidar Group
C
C         Software Responsibility : A. Weickmann   (303) 497-6595
C         Scientific Responsibility : M.J. Post    (303) 497-6048
C
C         Last Modified :   7/25/89 by  AMW
C-----------------------------------------------------------------------
C                 INPUTS
C Variable  type,common            Description
C --------  ----------  ----------------------------------------------
C
C
C                 OUTPUTS
C Variable  type,common            Description
C --------  ----------  ----------------------------------------------
C         ARCDAT      i,calc       Option to archive the beta profile data Y or N
C         ARCNAM      c,charac     Archive disk filename
C         BFLAG       i,plot       Flag for labeling the graphs
C         LINDB       i,plot       Linear or decibel scaling
C         LOHI        i,plot       Low to high scaling flag
C         PLTTYP      i,plot       Flag for single or average plots
C         PRNDAT      i,calc       Option to print out the raw data, Y or N
C         PRNPRF      i,calc       Option to print out the BETA profile Y or N
C
C-----------------------------------------------------------------------

          INCLUDE "CB.CALC"
          INCLUDE "CB.CHARAC"
          INCLUDE "CB.PLOT"

          CHARACTER*1 CANS

 1111     FORMAT(A1)

 10       CONTINUE
          CALL NEWPAG

C         BFLAG is a flag to indicate 1st 2 plots are not betas
          BFLAG = 0

C
C                    SET SCALING OF PLOTS.
C
               CALL NEWPAG
          PLTTYP = 'A '
          LINDB = 'L '
          LOHI = 'M '

 15       PRINT *
          PRINT *,' The default plot types and scale are as follows:'
          PRINT *
          PRINT *,'  Average Plots,  Linear Scaling,  Medium Scale'
          PRINT *,'  -------                                      '
          PRINT *,' Are these the values you want (Y or N)'
          PRINT *,'  (default = Y)'
          PRINT *,' ? : '
          READ (*,1111,err=15) CANS

          IF(CANS.EQ.'N' .OR. CANS.EQ.'n') GOTO 20
          if(cans.ne.'Y' .and. cans.ne.'y' .and. cans.ne.' ') then
               print *
               print *,' Your answer was not an option, try again.'
               goto 15
          endif
          GOTO 100

 20       CONTINUE
          PRINT *
               PRINT *,' ~~~  PLOT SETUP  ~~~  '
```

```
          PRINT *
          PRINT *,' Choose the type of plots :'
          PRINT *,'      average plots    - A '
          PRINT *,'      single record plots  - S '
          PRINT *,'  (default = A)'
          PRINT *,' ? : '
          READ (*,1111,err=20) CANS

          if(cans.eq.'S' .or. cans.eq.'s') then
               plttyp = 'S '
          else if(cans.ne.'A' .and. cans.ne.'a' .and. cans.ne.' ') then
               print *
               print *,' Your answer was not an option, try again.'
               goto 20
          endif

 30         PRINT *
          PRINT *,' Choose the type of scaling : '
          PRINT *,'            linear scaling       - L '
          PRINT *,'            dB scaling                    - D '
          PRINT *,' (default = L) '
          PRINT *,' ? : '
          READ (*,1111,err=30) CANS

          if(cans.eq.'D' .or. cans.eq.'d') then
               lindb = 'D '
          else if(cans.ne.'L' .and. cans.ne.'l' .and. cans.ne.' ') then
               print *
               print *,' Your answer was not an option, try again.'
               goto 30
          endif

 40       PRINT *
          PRINT *,' Choose the axis scaling : '
          IF(LINDB.EQ.'L ') THEN
               PRINT *,'                    0 to 20,000  - H '
               PRINT *,'                    0 to  2,000  - M '
               PRINT *,'                    0 to      400  - L '
          ELSE
               PRINT *,'                    0 to 40  - H '
               PRINT *,'                    0 to 20  - M '
               PRINT *,'                    0 to 12  - L '
          ENDIF
          PRINT *,'  (default = M)'
          PRINT *,' ? : '
          READ (*,1111,err=40) CANS

          if(cans.eq.'L' .or. cans.eq.'l') then
               lohi = 'L '
          else if(cans.eq.'H' .or. cans.eq.'h') then
               lohi = 'H '
          else if(cans.ne.'M' .and. cans.ne.'m' .and. cans.ne.' ') then
               print *
               print *,' Your answer was not an option, try again.'
               goto 40
          endif

 50       PRINT *
          PRINT *
          PRINT *,' Would you like to : '
          PRINT *,'    Continue with program  -  C '
          PRINT *,'    Modify recent inputs       -  M '
          PRINT *,'    (default = C) '
          PRINT *,' ? : '
          READ (*,1111,err=50) CANS

          IF(CANS.EQ.'M' .OR. CANS.EQ.'m') GOTO 10
          if(cans.ne.'c' .and. cans.ne.'C' .and. cans.ne.' ') then
               print *
               print *,' Your answer was not an option, try again.'
               goto 50
          endif

 100      CONTINUE

          CALL NEWPAG
```

```
      PRNDAT='N '
      PRNPRF='Y '
      ARCDAT='N '
      PRINT *
      PRINT *,' ~~~~~ Printer Outputs ~~~~~'

110   PRINT *
      PRINT *,' Do you want a  RAW  data printout,  (Y or N) '
      PRINT *,'  (default = N) '
      PRINT *,' ? : '
      READ (*,1111,err=110) CANS

      if(cans.eq.'Y' .or. cans.eq.'y') then
            prndat = 'Y '
      else if(cans.ne.'N' .and. cans.ne.'n' .and. cans.ne.' ') then
            print *
            print *,' Your answer was not an option, try again.'
            goto 110
      endif

120   PRINT *
      PRINT *,' Do you want a  BETA PROFILE  printout,  (Y or N)'
      PRINT *,'  (default = Y)'
      PRINT *,' ? : '
      READ(*,1111,err=120) CANS

      if(cans.eq.'N' .or. cans.eq.'n') then
            prnprf = 'N '
      else if(cans.ne.'Y' .and. cans.ne.'y' .and. cans.ne.' ') then
            print *
            print *,' Your answer was not an option, try again.'
            goto 120
      endif

      PRINT *
      PRINT *,' ~~~~~ Archive Output ~~~~~'
130   PRINT *
      PRINT *,' Do you want the beta profile  ARCHIVED  (Y or N)'
      PRINT *,'  (default = N)'
      PRINT *,' ? : '
      READ(*,1111,err=130) CANS

      if(cans.eq.'Y' .or. cans.eq.'y') then
            arcdat = 'Y '
135   PRINT *
            PRINT *,' Enter the archive file name (max of 25 characters)'
            PRINT *,' ? : '
            READ(*,1113,err=135) ARCNAM
      else if(cans.ne.'N' .and. cans.ne.'n' .and. cans.ne.' ') then
            print *
            print *,' Your answer was not an option, try again.'
            goto 130
      endif

1113  FORMAT(A25)

140   PRINT *
      PRINT *
      PRINT *,' Would you like to : '
      PRINT *,'    Continue with program  -  C '
      PRINT *,'    Modify recent inputs       -  M '
      PRINT *,'    (default = C) '
      PRINT *,' ? : '
      READ (*,1111,err=140) CANS

      IF(CANS.EQ.'M' .OR. CANS.EQ.'m') GOTO 100
      if(cans.ne.'C' .and. cans.ne.'c' .and. cans.ne.' ') then
            print *
            print *,' Your answer was not an option, try again.'
            goto 140
      endif

      RETURN
      END
```

```
      SUBROUTINE PLGRFR(YARRY,XARRY1,XARRY2,IPTS,NCURVE,CLABX,
     +                  CLABY)
C-----------------------------------------------------------------------
C       Plot the data profiles for the alt vs power or alt vs beta.
C       Plot the graph frame and the x,y points of up to 2 curves
C        and label the axis.  The graphics frame is already open
C        by a previous call to TXGRFR.F
C-----------------------------------------------------------------------
C
C       This program was written by personnel from
C       NOAA/ERL/Wave Propagation Laboratory/Atmospheric Lidar Group
C
C       Software Responsibility : A. Weickmann   (303) 497-6595
C       Scientific Responsibility : M.J. Post    (303) 497-6048
C
C       Modified :  7/20/89        by  AMW
C       Modified :  8/21/90        by  RPM - added to INPUTS comments
C-----------------------------------------------------------------------
C                 INPUTS
C Variable  type,common          Description
C ---------  ----------  ----------------------------------------------
C       XMAX        r,plot       Maximum x defining the graphic data window
C       XMIN        r,plot       Minimum x defining the graphic data window
C       YMAX        r,plot       Maximum y defining the graphic data window
C       YMIN        r,plot       Minimum y defining the graphic data window
C       YARRY( )  r,passed
C       XARRY1( ) r,passed
C       XARRY2( ) r,passed
C       IPTS        i,passed
C       NCURVE      i,passed
C       CLABX       r,passed
C       CLABY       r,passed
C
C                 OUTPUTS
C Variable  type,common          Description
C ---------  ----------  ----------------------------------------------
C                 none
C-----------------------------------------------------------------------
      INCLUDE "CB.PLOT"

      REAL*4 YARRY(2048), XARRY1(2048), XARRY2(2048)
      REAL*4 XX, XL, HA, XINC, YINC, YY, X, Y
      INTEGER*2 IPTS, NCURVE, IX,IY,LENS,ICHR
      INTEGER*2 IUP, IXX, IYY, IDASH
      INTEGER*2 IXMIN, IXMAX, IYMIN, IYMAX
      CHARACTER*8 CLBX, CLBY
      CHARACTER*25 CLABX, CLABY
      CHARACTER*80 CSTR,CEMPTY
      CHARACTER*20 CBLANK

      CBLANK='

      ICHR=2
      IXMIN=175
      IXMAX=975
      IYMIN=80
      IYMAX=670

      CALL TWINDO(IXMIN,IXMAX,IYMIN,IYMAX)
      CALL DWINDO(XMIN,XMAX,YMIN,YMAX)

C     Draw Axis

      CALL MOVEA(XMIN,YMIN)
      CALL DRAWA(XMIN,YMAX)
      CALL DRAWA(XMAX,YMAX)
      CALL DRAWA(XMAX,YMIN)
      CALL DRAWA(XMIN,YMIN)

C     Draw X axis ticks and label them

      XX=XMIN
      NX=4
      NX1=NX+1
      XINC=(XMAX-XMIN)/NX
      DO 20 J=1,NX1
            call movea(xx,YMAX)
```

```
            ix=0
            iy=-10
            call drwrel(ix,iy)
            CALL MOVEA(XX,YMIN)
            CALL DRWREL(IX,IY)
            IX=-80
            IY=-25
            IF(NCURVE.EQ.2) THEN
                  XL=10.**((XX-120.)/10.)
                  WRITE(CLBX,900) XL
            ELSE
                  WRITE(CLBX,910) XX
            ENDIF
            CALL STSREL(IX,IY,CLBX,ICHR)
            if(ncurve.eq.2) then
                  do 15 in=2,9
                        xadd = alog10(float(in)) * 10.
                        call movea(xx+xadd,ymin)
                        ix=0
                        iy=-7
                        call drwrel(ix,iy)
                        call movea(xx+xadd,YMAX)
                        call drwrel(ix,iy)
 15               continue
            endif
            XX=XX+XINC
 20     CONTINUE

 900    FORMAT(1PE7.1)
 910    FORMAT(F8.2)
 920    FORMAT(F4.1)

C       Draw Y axis ticks and label them

        YY=YMIN
        MNYY=INT(YMIN)
        MXYY=INT(YMAX+0.5)
        NY=MXYY-MNYY
        NY=NY+1
        IF(NY.LT.1) NY=1
        YINC=1.0
        DO 40 J=1,NY
            call movea(xmax,yy)
            ix=-10
            iy=0
            call drwrel(ix,iy)
            CALL MOVEA(XMIN,YY)
            CALL DRWREL(IX,IY)
            IY=-7
            IX=-55
            WRITE(CLBY,920) YY
            CALL STSREL(IX,IY,CLBY,ICHR)
            YY=YY+YINC
            IF(YY.GT.YMAX) GOTO 50
 40     CONTINUE

 50     CONTINUE

C       Label Y axis

        HA = (YMAX-YMIN)*0.5
        CALL MOVEA(XMIN,YMAX-HA)
        LENS=STRLEN(CLABY)
        IX=2
        IUP=LENS/IX
        IF(MOD2(LENS,IX).NE.0) IUP = IUP+1
        IYY=388
        IXX=22
        IYY=IYY+(IUP*IXX)
        IX=50
        CALL MOVABS(IX,IYY)
        DO 55 J=1,LENS
            IY=J*IXX
            CALL STSABS(IX,IYY-IY,CLABY(J:J),ICHR)
 55     CONTINUE

C       Label X axis
```

```
        IX=420
        IY=20
        CALL STSABS(IX,IY,CLABX,ICHR)

C       Plot the data arrays, if ncurve=1 plot is alt vs power with
C        one curve. If ncurve=2 plot is alt vs beta with 2 curves,
C        yarry1 is a solid line and yarry2 is a dashed line.

        Y=YARRY(1)
        X=XARRY1(1)
        IF(X.GT.XMAX) X=XMAX
        CALL MOVEA(X,Y)
        DO 80 I=2,IPTS
              Y=YARRY(I)
              X=XARRY1(I)
              CALL DRAWA(X,Y)
 80     CONTINUE

C       Draw the dashed curve if ncurve=2

        IF(NCURVE.EQ.2) THEN
              IDASH=34
              Y=YARRY(1)
              X=XARRY2(1)
              CALL MOVEA(X,Y)
              DO 100 I=2,IPTS
                    Y=YARRY(I)
                    X=XARRY2(I)
                    CALL DASHA(X,Y,IDASH)
 100          CONTINUE
        ENDIF

        IX=10
        IY=7o0
        CALL MOVABS(IX,IY)
        CALL TSEND
        RETURN
        END
```

```
        SUBROUTINE PLGRFT (ARRY,KREC)
C-----------------------------------------------------------------------
C       Plot the time vs power graph, the data only.  A previous call
C        to TXGRFT.F77 has opened the graphics window and
C        written text to the graph.
C-----------------------------------------------------------------------
C
C       This program was written by personnel from
C       NOAA/ERL/Wave Propagation Laboratory/Atmospheric Lidar Group
C
C       Software Responsibility : A. Weickmann   (303) 497-6595
C       Scientific Responsibility : M.J. Post     (303) 497-6048
C
C       Last Modified :   8/07/89 by  AMW
C-----------------------------------------------------------------------
C                 INPUTS
C Variable  type,common            Description
C --------  ----------  ----------------------------------------------
C       ARRY        r,passed      Data array to plot
C       KREC        i,passed      Number of points to plot from ARRY.
C       XMAX        r,plot        Maximum x defining the graphic data window
C       XMIN        r,plot        Minimum x defining the graphic data window
C       YMAX        r,plot        Maximum y defining the graphic data window
C       YMIN        r,plot        Minimum y defining the graphic data window
C
C                 OUTPUTS
C Variable  type,common            Description
C --------  ----------  ----------------------------------------------
C         none
C-----------------------------------------------------------------------

        INCLUDE "CB.PLOT"

        INTEGER*2 KREC
        REAL*4 ARRY(KREC)
```

```
      REAL*4 HA, XINC,YINC,XX,YY,DELX,X,Y
      INTEGER*2 IXMIN,IYMIN,IXMAX,IYMAX
      INTEGER*2 IX,IY,IUP,LENS,IXX,IYY,ICHR
      CHARACTER*80 CSTR,CEMPTY
      CHARACTER*20 CBLANK
      CHARACTER*7 CLBX,CLBY

      CBLANK='
      CEMPTY=CBLANK//CBLANK//CBLANK//CBLANK

      ICHR=2
      DELX=XMAX/KREC

      IXMIN=175
      IXMAX=975
      IYMIN=80
      IYMAX=670

      CALL TWINDO(IXMIN,IXMAX,IYMIN,IYMAX)
      CALL DWINDO(XMIN,XMAX,YMIN,YMAX)

C     Draw axis

      CALL MOVEA(XMIN,YMIN)
      CALL DRAWA(XMIN,YMAX)
      CALL DRAWA(XMAX,YMAX)
      CALL DRAWA(XMAX,YMIN)
      CALL DRAWA(XMIN,YMIN)

C     Draw Y ticks and label

      YY=YMIN
      NY=4
      NY1=NY+1
      YINC=YMAX/NY
      DO 20 J=1,NY1
            call movea(xmax,yy)
            ix=-10
            iy=0
            call drwrel(ix,iy)
            CALL MOVEA(XMIN,YY)
            CALL DRWREL(IX,IY)
            IX=-80
            IY=-7
            WRITE(CLBY,900) YY
            CALL STSREL(IX,IY,CLBY,ICHR)
            YY=YY+YINC
            IF(YY.GT.YMAX) YY=YMAX
 20   CONTINUE
 900  FORMAT(F6.0)

C     Label Y axis

      CSTR='RELATIVE SIGNAL POWER'
      HA= (YMAX-YMIN)*0.5
      CALL MOVEA(XMIN,YMAX-HA)
      LENS=STRLEN(CSTR)
      IX=2
      IUP=LENS/IX
      IF(MOD2(LENS,IX).NE.0) IUP=IUP+1
      IYY=383
      IXX=22
      IYY=IYY+(IUP*IXX)
      IX=50
      CALL MOVABS(IX,IYY)
      DO 40 J=1,LENS
            IY=J*IXX
            CALL STSABS(IX,IYY-IY,CSTR(J:J),ICHR)
 40   CONTINUE

C     Draw X ticks and label

      XX=0.0
      NX=KREC/5
      IF(NX.LT.1) NX=1
 50   IF(NX.LE.10) GOTO 60
      NX=NX/2
      GOTO 50
 60   XINC= XMAX/NX
      NX1=NX+1
      DO 70 J=1,NX1
            call movea(xx,YMAX)
            ix=0
            iy=-10
            call drwrel(ix,iy)
            CALL MOVEA(XX,YMIN)
            CALL DRWREL(IX,IY)
            IX=-45
            IY=-25
            WRITE(CLBX,910) XX
            CALL STSREL(IX,IY,CLBX,ICHR)
            XX=XX+XINC
            IF(XX.GT.XMAX) XX=XMAX
 70   CONTINUE
 910  FORMAT(F5.1)

C     X axis label

      CSTR=CEMPTY
      CSTR='SECONDS'
      IX=480
      IY=20
      CALL STSABS(IX,IY,CSTR,ICHR)

C     Plot the data

      X=XMIN
      Y=ARRY(1)
      IF(Y.GT.YMAX) Y=YMAX
      CALL MOVEA(X,Y)
      DO 100 I=2,KREC
            X=X+DELX
            Y=ARRY(I)
            CALL DRAWA(X,Y)
 100  CONTINUE

      IX=10
      IY=760
      CALL MOVABS(IX,IY)
      CALL TSEND
      RETURN
      END


      SUBROUTINE PLOTIT (ARRY,KPTS, GRFTYP)
C-----------------------------------------------------------------
C     Arrange the calls to the various types of plots :
C     range vs power,  alt vs. beta,  power vs. time.
C-----------------------------------------------------------------
C
C     This program was written by personnel from
C     NOAA/ERL/Wave Propagation Laboratory/Atmospheric Lidar Group
C
C     Software Responsibility : A. Weickmann   (303) 497-6595
C     Scientific Responsibility : M.J. Post     (303) 497-6048
C
C     Modified :  8/03/89        by  AMW
C     Modified :  8/21/90        by  RPM - adding INPUT comments
C-----------------------------------------------------------------
C            INPUTS
C Variable  type,common            Description
C --------  -----------  ----------------------------------------
C     ALT( )        r,signl      Altitude array, km
C     BETAFL        i,calc       Flag to plot the BETA profile Y or N
C     DIGALT        r,header     Altitude increment per digitizer sample
C     ICOUNT        i,calc       Number of altitude increments in the profile
C     IFN           i,vars       File number
C     IHEAD( )      i,header     Integer header from tape
C     IPLTN         i,vars       Consecutive plot number
C     MFIFN( )      i,multfl     Multi file numbers.
C     MFNMFI        i,multfl     Number of files averaged.
C     MFTIME( ) i,multfl         Time for each file
C     MULTFI        l,multfl     TRUE if multifiles averaging in effect
```

```
C         NREC          i,vars       # of records to process
C         NPTS          i,header     Number of points (or samples) to be processed
C         PLBTSD( ) r,signl Array for plotting beta std. dev.
C         R1                  r,header   Begin range, adjusted for lidar altitude and
C                                                  elevation angle
C         ZINC          r,calc       Actual altitude increment for averaging
C                                                  processed data (as opposed to DALT)
C         ARRY          r,passed
C         KPTS          i,passed
C         GRFTYP        c,passed
C
C              OUTPUTS
C Variable  type,common             Description
C --------  -----------  ----------------------------------------
C         IPLTN         i,vars       Consecutive plot number
C         XMIN          r,plot       Minimum x defining the graphic data window
C         YMIN          r,plot       Minimum y defining the graphic data window
C-----------------------------------------------------------------
          INCLUDE "CB.CALC"
          INCLUDE "CB.HEADER"
          INCLUDE "CB.MULTFL"
          INCLUDE "CB.SIGNL"
          INCLUDE "CB.VARS"

          INTEGER*2 NCURVE,KPTS
          INTEGER*2 I, IST1, ISTP, ISTRT, NUMF, JTIME(9), FILES(10)
          REAL*4  ARRY(KPTS),RANGE(2048),DUM(2048)
          REAL*4 AVGP, SUM, EXTRA
          CHARACTER*1 GRFTYP
          CHARACTER*25 CLABX, CLABY

C         Increment the consecutive plot number

          IPLTN=IPLTN+1
C
C         FILL THE DUM ARRAY
C
          DO 5 II=1,NPTS
               DUM(II)=0.0
 5        CONTINUE
          EXTRA = 0.0
C
          IF (BETAFL .EQ. "Y ") THEN
               ISTRT = 1
               ISTP = ICOUNT
               DELX = ZINC
          ELSE
               ISTRT = 1
               ISTP = NPTS
               DELX = DIGALT
          ENDIF

C         Get the data time from the tape header information
C
          DO 15 I=1,6
               JTIME(I)=IHEAD(I)
 15        CONTINUE

          XMIN=0.0
          YMIN=0.0

C         Plot time vs. power for single file only.

          IF(GRFTYP.EQ.'T') THEN
               SUM = 0.0
               DO 17 I=1,NREC
                    SUM = SUM + ARRY(I)
 17            CONTINUE
               AVGP = SUM / NREC
               CALL TXGRFT(JTIME,AVGP)
               CALL PLGRFT(ARRY,NREC)
               GOTO 100
          ENDIF

C         Plot range vs power or alt vs beta for single or multiple files.

          RANGE(1)=R1
          IST1=ISTRT+1
          DO 20 I=IST1,ISTP
               RANGE(I)=RANGE(I-1)+DELX
 20       CONTINUE
C         IFN is set to -1 in beta to designate multifile range or alt
C         plots else the MULTFI flag will always produce multifile plots
C
          IF(MULTFI .AND. (IFN.EQ.-1)) THEN
               JTIME(4) = MFTIME(1,1)
               JTIME(5) = MFTIME(1,2)
               JTIME(6) = MFTIME(1,3)
               JTIME(7) = MFTIME(MFNMFI,1)
               JTIME(8) = MFTIME(MFNMFI,2)
               JTIME(9) = MFTIME(MFNMFI,3)
          ENDIF

C         Single file, Range vs Power.

          IF(IFN.GE.0 .AND. BETAFL.EQ.'N ') THEN
               FILES(1) = IFN
               NUMF = 1
               SUM = 0.0
               DO 30 I=1,NPTS
                    SUM = SUM + ARRY(I)
 30            CONTINUE
               AVGP = SUM / NPTS
               CALL TXGRFR(JTIME,FILES,NUMF,AVGP)
               CLABX = 'RELATIVE SIGNAL POWER'
               CLABY = 'ALTITUDE KM ASL'
               GOTO 75
          ENDIF

C         Single file, Altitude vs Beta.

          IF(IFN.GE.0 .AND. BETAFL.EQ.'Y ') THEN
               FILES(1) = IFN
               NUMF = 1
               CALL TXGRFR(JTIME,FILES,NUMF,EXTRA)
               CLABX = 'BETA (M-1 SR-1)'
               CLABY = 'ALTITUDE KM ASL'
               GOTO 90
          ENDIF

C         Multiple files, Range vs. Power.

          IF(IFN.EQ.-1 .AND. BETAFL.EQ.'N ') THEN
               DO 50 I=1,MFNMFI
                    FILES(I) = MFIFN(I)
 50            CONTINUE
               NUMF = MFNMFI
               SUM = 0.0
               DO 55 I=1,NPTS
                    SUM = SUM + ARRY(I)
 55            CONTINUE
               AVGP = SUM / NPTS
               CALL TXGRFR(JTIME,FILES,NUMF,AVGP)
               CLABX = 'RELATIVE SIGNAL POWER'
               CLABY = 'ALTITUDE KM ASL'
               GOTO 75
          ENDIF

C         Multiple files, Altitude vs. Beta.

          IF(IFN.EQ.-1 .AND. BETAFL.EQ.'Y ') THEN
               DO 60 I=1,MFNMFI
                    FILES(I) = MFIFN(I)
 60            CONTINUE
               NUMF = MFNMFI
               CALL TXGRFR(JTIME,FILES,NUMF,EXTRA)
               CLABX = 'BETA (M-1 SR-1)'
               CLABY = 'ALTITUDE KM ASL'
               GOTO 90
          ENDIF

 75       CONTINUE
          NCURVE=1
          CALL PLGRFR(RANGE,ARRY,DUM,NPTS,NCURVE,CLABX,CLABY)
          GOTO 100
```

```
90      CONTINUE
        NCURVE = 2
        CALL PLGRFR(ALT,ARRY,PLBTSD,ICOUNT,NCURVE,CLABX,CLABY)

100     CONTINUE

        RETURN
        END
```

```
      SUBROUTINE PLTPROFILE
C-----------------------------------------------------------------------
C
C  This routine sets up the variables needed to plot the BETA profile.
C
C-----------------------------------------------------------------------
C
C         This program was written by personnel from
C         NOAA/ERL/Wave Propagation Laboratory/Atmospheric Lidar Group
C
C         Software Responsibility : A. Weickmann   (303) 497-6595
C         Scientific Responsibility : M.J. Post     (303) 497-6048
C
C         Modified :  8/07/89        by  AMW
C         Modified :  9/04/90        by  RPM - upped DUMBET dimension to 2048
C-----------------------------------------------------------------------
C                INPUTS
C Variable  type,common              Description
C --------  -----------  ----------------------------------------
C         BETACO( ) r,signl Array to store the BETA coefficients
C         BETASD( ) r,signl Array to store BETA std. dev. values
C         ICOUNT       i,calc        Number of altitude increments in the profile
C         IPLTA        i,vars        Plot to screen =0 or plotter =1 or both =2
C
C                OUTPUTS
C Variable  type,common              Description
C --------  -----------  ----------------------------------------
C         BETAFL       i,calc        Flag to plot the BETA profile Y or N
C         PLBTSD( ) r,signl Array for plotting beta std. dev.
C         XMAX         r,plot        Maximum x defining the graphic data window
C-----------------------------------------------------------------------

          INCLUDE "CB.CALC"
          INCLUDE "CB.PLOT"
          INCLUDE "CB.SIGNL"
          INCLUDE "CB.VARS"

          REAL*4 DUMBET(2048)
          INTEGER*2 I
          CHARACTER*1 GRFTYP, CANS

 1111     FORMAT(A1)

C  Plot BETA profile

          DO 10 I = 1, ICOUNT
                IF (BETACO(I) .LE. 0.) BETACO(I) = 1.E-20
                IF (BETASD(I) .LE. 0.) BETASD(I) = 1.E-20
                DUMBET(I) = 10. * ALOG10(BETACO(I)) + 120.
                PLBTSD(I) = 10. * ALOG10(BETASD(I)) + 120.
10        CONTINUE

          XMAX = 40.
          BETAFL = 'Y '
          GRFTYP = 'R'
          CALL PLOTIT(DUMBET,ICOUNT,GRFTYP)
          IF(IPLTA.EQ.0 .OR. IPLTA .EQ. 2) READ(*,1111) CANS

          RETURN
          END



          SUBROUTINE PRINTDATA(AVGPWR)
C-----------------------------------------------------------------------
C
C         Print out the averaged raw data for every point between the
C          beginning and ending ranges. The output file is BETA.RAWDAT
C
C-----------------------------------------------------------------------
C
C         This program was written by personnel from
C         NOAA/ERL/Wave Propagation Laboratory/Atmospheric Lidar Group
C
C         Software Responsibility : A. Weickmann   (303) 497-6595
C         Scientific Responsibility : M.J. Post     (303) 497-6048
C
C         Last Modified :   8/02/89 by  AMW
C-----------------------------------------------------------------------
C                INPUTS
C Variable  type,common              Description
C --------  -----------  ----------------------------------------
C         AVGPWR( ) r,passed         Average power array
C         IFN          i,vars        File number
C         IHEAD( )  i,header         Integer header from tape
C         IREC         i,vars        Start record number
C         IREND        i,vars        End record number
C         NPTS         i,header      Number of points (or samples) to be processed
C
C                OUTPUTS
C Variable  type,common              Description
C --------  -----------  ----------------------------------------
C           none
C-----------------------------------------------------------------------

          INCLUDE "CB.HEADER"
          INCLUDE "CB.VARS"

          INTEGER*2 I
          REAL*4 AVGPWR(2048)

          OPEN(UNIT=65,FILE='BETA.RAWDAT',STATUS='UNKNOWN')

          WRITE(65,1000)
 1000     FORMAT(1H1)

          WRITE (65, 1020) IFN, IREC, IREND, (IHEAD(I), I = 1, 6)
1020      FORMAT(" File : ", I2,/, " Records : ", I5, " - ", I5,/,
            +                 " Date : ", I2,2("/",I2), "      Time : ",I2,2(":",I2))

          WRITE(65,*)
          WRITE(65,1030)
1030      FORMAT(" Average Power vs. Range : ", //)
          WRITE (65, 1040) (I, AVGPWR(I), I = 1, NPTS)
1040      FORMAT(5(6X, I5, 1X, F10.2))

          CLOSE(65)

          RETURN
          END



          SUBROUTINE PRINTPROFILE
C-----------------------------------------------------------------------
C         Print out the beta profile to file BETA.PRINT
C-----------------------------------------------------------------------
C
C         This program was written by personnel from
C         NOAA/ERL/Wave Propagation Laboratory/Atmospheric Lidar Group
C
C         Software Responsibility : A. Weickmann   (303) 497-6595
C         Scientific Responsibility : M.J. Post     (303) 497-6048
C
C         Modified :  8/02/89        by  AMW
C         Modified :  8/29/90        by  RPM - fixed INPUTS comments
C-----------------------------------------------------------------------
C                INPUTS
C Variable  type,common              Description
C --------  -----------  ----------------------------------------
C         ALT( )       r,signl       Altitude array, km
C         ARCDAT       i,calc        Option to archive the beta profile data Y or N
C         BEGALT       r,calc        Beginning altitude of the noise window, km asl
C         BETACO( ) r,signl Array to store the BETA coefficients
C         BETAMN( ) r,signl Array to store minimum BETA values
C         BETASD( ) r,signl Array to store BETA std. dev. values
C         BW                 r,vars  Bandwidth of signals in MHz
C         ENDALT       r,calc        Ending altitude for the noise window, km asl
C         IFN          i,vars        File number
```

```
C         IHEAD( )  i,header        Integer header from tape
C         IREC          i,vars      Start record number
C         MFDATE( ) i,multfl        Date for each file
C         MFIFN( )  i,multfl        Multi file numbers.
C         MFIREC( ) i,multfl        Multi file beg rec number.
C         MFNMFI        i,multfl    Number of files averaged.
C         MFNREC( ) i,multfl        Multi file number of recs.
C         MFTIME( ) i,multfl        Time for each file
C         MULTFI        l,multfl    TRUE if multifiles averaging in effect
C         NREC          i,vars      # of records to process
C         QFAC( )       r,signl     Q-factor array
C         QUAL          r,signl     The quality factor threshold value.
C         RLAM          r,vars      Wavelength,  meters
C         SATCOR        i,vars      Digitizer saturation correction flag
C         SNR( )        r,signl     Array to store the Signal to Noise Ratios
C         SNRSDD( ) r,signl Array to store SNR std.dev. in DB
C
C                OUTPUTS
C Variable  type,common             Description
C --------  ----------  ------------------------------------
C           none
C-----------------------------------------------------------------

          INCLUDE "CB.CALC"
          INCLUDE "CB.HEADER"
          INCLUDE "CB.MULTFL"
          INCLUDE "CB.SIGNL"
          INCLUDE "CB.VARS"

          INTEGER*2 I, J, JPASS, LREC
          REAL*4 RLAMM, BWW

          IF(ARCDAT.EQ.'Y ') THEN
                IF(CLRAIR.EQ.0.0) WRITE(40,900)
                IF(CLRAIR.GT.0.0) WRITE(40,910) CLRAIR
          ENDIF
900       FORMAT(' No clouds present in this profile')
910       FORMAT(' Clouds present, Altitude of last clear air is ',F4.1,
            +                  ' km')

          WRITE(40,1100)
1010      FORMAT(1X,'File',4X,'Date',6X,'Time',5X,'Start Record',2X,
            +                  '# Records')
1020      FORMAT(1X,I3,3X,I2,2('/',I2),2X,I2,2(':',I2),6X,I4,9X,I4)

          IF (.NOT. MULTFI) THEN
                WRITE(40,1010)
                WRITE(40,1020) IFN, (IHEAD(I), I = 1,6), IREC, NREC
          ELSE
                WRITE(40,1010)
                DO 20 I=1,MFNMFI
                      WRITE(40,1020) MFIFN(I), (MFDATE(I,J),J=1,3),
            +         (MFTIME(I,J),J=1,3), MFIREC(I), MFNREC(I)
20              CONTINUE
          ENDIF

          IF(SATCOR.EQ.0) WRITE(40,1022)
          IF(SATCOR.EQ.10) WRITE(40,1023)
          IF(SATCOR.EQ.11) WRITE(40,1024)
          IF(SATCOR.EQ.20) WRITE(40,1025)
          IF(SATCOR.EQ.21) WRITE(40,1026)

1022      FORMAT(/,' Digitizer correction applied to R2Q2 data')
1023      FORMAT(/,' Digitizer correction applied to Q2 Unaveraged data')
1024      FORMAT(/,' Digitizer correction applied to Q2 Time averaged',
            +   ' data')
1025      FORMAT(/,' Digitizer correction applied to R2 Unaveraged data')
1026      FORMAT(/,' Digitizer correction applied to R2 Time averaged',
            +   ' data')

          WRITE(40, 1030) BEGALT,ENDALT
1030      FORMAT(' Altitude of Noise Window : ',F4.1,' to ',F4.1,
            +                  ' km asl')
          RLAMM = RLAM * 1.0E6
          BWW = BW / 1.0E6
          WRITE(40,1040) RLAMM,BWW,QUAL
1040      FORMAT(' Wavelength : ',F6.3,'            Bandwidth : ',F4.1,
            +                '        Data Good :  Q-Factor GT ',F5.2)
```

```
          WRITE(40,*)


          WRITE(40,1060)
1060      FORMAT ("  ALT (KM)       BETA(M-1 SR-1)  Q-FACTOR              SNR(DB)"
            +     "  SNRSD(DB)      BMIN                        BSD", /)
          DO 50 J = 1, ICOUNT
                JPASS = J
                IF (SNR(JPASS) .NE. 0) THEN
                      SNRDB = (10. * ALOG10(SNR(JPASS)))
                ELSE
                      SNRDB = 0.
                ENDIF
                WRITE (40,1070)  ALT(JPASS), BETACO(JPASS),
            +           QFAC(JPASS), SNRDB, SNRSDD(JPASS),
            +           BETAMN(JPASS), BETASD(JPASS)
 50       CONTINUE
1070       FORMAT(F7.3, 7X, 1PE9.2, 2X, 0PF11.2, F12.3,
            +                  2X, F8.2, 2(2X, 1PE9.2) )

          WRITE(40, 1100)
1100      FORMAT(1H1)

          RETURN
          END



          FUNCTION Q2(I)
C-----------------------------------------------------------------
C
C  This function decodes the raw data from the lidar digitizer
C  and returns in Q2 the power for the point "I".
C
C-----------------------------------------------------------------
C
C         This program was written by personnel from
C         NOAA/ERL/Wave Propagation Laboratory/Atmospheric Lidar Group
C
C         Software Responsibility : A. Weickmann   (303) 497-6595
C         Scientific Responsibility : M.J. Post    (303) 497-6048
C
C         Last Modified :   8/08/89 by  AMW
C-----------------------------------------------------------------
C                INPUTS
C Variable  type,common             Description
C --------  ----------  ------------------------------------
C         I                 i,passed    Index into the IDAT raw signal array.
C         IDAT( )     i,signl       Array to store data from tape
C
C                OUTPUTS
C Variable  type,common             Description
C --------  ----------  ------------------------------------
C         Q2                r,function  Real portion of signal squared plus
C                                                imaginary portion of signal squared.
C-----------------------------------------------------------------

          INCLUDE "CB.SIGNL"

          INTEGER*2 RBYTE, LBYTE
          INTEGER*2 MASK, SHIFT, ITEMP, I

          MASK = 377K
          RBYTE = IAND (IDAT(I), MASK)
          ITEMP = NOT2(MASK)
          LBYTE = IAND (IDAT(I), ITEMP)
          SHIFT = -8
          LBYTE = ISHFT2(LBYTE, SHIFT)
          SHIFT = 7
          IF (BTEST2(RBYTE, SHIFT)) RBYTE = RBYTE - 256
          IF (BTEST2(LBYTE, SHIFT)) LBYTE = LBYTE - 256
          Q2 = (float(LBYTE)+0.7) * (float(LBYTE)+0.7)

          RETURN
          END
```

```
      FUNCTION R2Q2(I)
C-----------------------------------------------------------------
C
C  This function decodes the raw data from the lidar digitizer
C  and returns in R2Q2 the power for the point "I".
C
C-----------------------------------------------------------------
C
C        This program was written by personnel from
C        NOAA/ERL/Wave Propagation Laboratory/Atmospheric Lidar Group
C
C        Software Responsibility : A. Weickmann   (303) 497-6595
C        Scientific Responsibility : M.J. Post     (303) 497-6048
C
C        Last Modified :   8/08/89 by  AMW
C-----------------------------------------------------------------
C              INPUTS
C Variable  type,common            Description
C --------  -----------  ----------------------------------------
C        I                 i,passed    Index into the IDAT raw signal array.
C        IDAT( )     i,signl       Array to store data from tape
C
C              OUTPUTS
C Variable  type,common            Description
C --------  -----------  ----------------------------------------
C        R2Q2        r,function    Real portion of signal squared plus
C                                                imaginary portion of signal squared.
C-----------------------------------------------------------------

      INCLUDE "CB.SIGNL"

      INTEGER*2 RBYTE, LBYTE
      INTEGER*2 MASK, SHIFT, ITEMP, I

      MASK = 377K
      RBYTE = IAND (IDAT(I), MASK)
      ITEMP = NOT2(MASK)
      LBYTE = IAND (IDAT(I), ITEMP)
      SHIFT = -8
      LBYTE = ISHFT2(LBYTE, SHIFT)
      SHIFT = 7
      IF (BTEST2(RBYTE, SHIFT)) RBYTE = RBYTE - 256
      IF (BTEST2(LBYTE, SHIFT)) LBYTE = LBYTE - 256
      R2Q2 = float(RBYTE)*float(RBYTE)+float(LBYTE)*float(LBYTE)

      RETURN
      END
```

```
      SUBROUTINE RECEFF(RLAM,BEAM)
C-----------------------------------------------------------------
C        Return the receiver efficiencies for the input wavelength.
C-----------------------------------------------------------------
C
C        This program was written by personnel from
C        NOAA/ERL/Wave Propagation Laboratory/Atmospheric Lidar Group
C
C        Software Responsibility : A. Weickmann   (303) 497-6595
C        Scientific Responsibility : M.J. Post     (303) 497-6048
C
C        Last Modified :   7/20/89 by  AMW
C-----------------------------------------------------------------
C              INPUTS
C Variable  type,common            Description
C --------  -----------  ----------------------------------------
C RLAM                r,passed      Wavelength, meters
C
C              OUTPUTS
C Variable  type,common            Description
C --------  -----------  ----------------------------------------
C BEAM( )             r, passed     Receiver efficiencies for wavelength
C-----------------------------------------------------------------
C
C        The numbers in the data statement are receiver efficiencies
C        accounting for stigmatism in the optics for wavelengths
C        10.6 and 9.25.  The numbers were computed by Yan.
C        Return the receiver efficiencies for the input wavelength.
C        They can be linearly interpolated from 10.591 to 9.26 microns
C        If the input wavelength is larger than 10.591 notify the
C        user and send back rec. eff. of 10.591, likewise for values
C        less than 9.25 microns, send 9.25 micron rec. eff.
C
C        These values at at 0.5 km increments from 0.5 km to 30.0 km
C
C-----------------------------------------------------------------

      REAL*4 BEAM(60), RLAM
      REAL*4  BEAM10(60), BEAM9(60)

      DATA BEAM10 /
     & .15780E-01,.50200E-01,.79050E-01,.10000E+00,.11520E+00,
     & .12660E+00,.13530E+00,.14220E+00,.14800E+00,.15250E+00,
     & .15610E+00,.15920E+00,.16180E+00,.16410E+00,.16600E+00,
     & .16770E+00,.16920E+00,.17050E+00,.17170E+00,.17270E+00,
     & .17370E+00,.17450E+00,.17530E+00,.17600E+00,.17670E+00,
     & .17730E+00,.17790E+00,.17840E+00,.17880E+00,.17930E+00,
     & .17970E+00,.18010E+00,.18050E+00,.18080E+00,.18110E+00,
     & .18140E+00,.18170E+00,.18200E+00,.18220E+00,.18250E+00,
     & .18270E+00,.18290E+00,.18310E+00,.18330E+00,.18350E+00,
     & .18370E+00,.18390E+00,.18400E+00,.18420E+00,.18440E+00,
     & .18450E+00,.18460E+00,.18480E+00,.18490E+00,.18500E+00,
     & .18520E+00,.18530E+00,.18540E+00,.18550E+00,.18560E+00/
      DATA BEAM9 /
     & .1287421E-01,.4502914E-01,.7406126E-01,.9521150E-01,
     & .1102182E+00,.1213645E+00,.1298761E+00,.1365298E+00,
     & .1422725E+00,.1466156E+00,.1502009E+00,.1532021E+00,
     & .1557456E+00,.1579248E+00,.1598103E+00,.1614557E+00,
     & .1629029E+00,.1641847E+00,.1653272E+00,.1663514E+00,
     & .1672744E+00,.1681102E+00,.1688703E+00,.1695644E+00,
     & .1702005E+00,.1707856E+00,.1713254E+00,.1718249E+00,
     & .1722884E+00,.1727196E+00,.1731218E+00,.1734976E+00,
     & .1738497E+00,.1741801E+00,.1744907E+00,.1747834E+00,
     & .1750595E+00,.1753204E+00,.1755674E+00,.1758015E+00,
     & .1760237E+00,.1762348E+00,.1764357E+00,.1766271E+00,
     & .1768096E+00,.1769839E+00,.1771505E+00,.1773098E+00,
     & .1774623E+00,.1776086E+00,.1777488E+00,.1778835E+00,
     & .1780128E+00,.1781372E+00,.1782569E+00,.1783722E+00,
     & .1784833E+00,.1785904E+00,.1786937E+00,.1787935E+00/

      IF(RLAM.GE.1.0591E-5) THEN
            DO 10 I=1,60
                  BEAM(I) = BEAM10(I)
 10         CONTINUE
            GOTO 200
      ENDIF

      IF(RLAM.LE.9.250E-6) THEN
            DO 20 I=1,60
                  BEAM(I) = BEAM9(I)
 20         CONTINUE
            GOTO 200
      ENDIF

      DELRLM = 1.0591E-5 - RLAM
      DELWV =  1.0591E-5 - 9.250E-6
      DO 40 I=1,60
            XX = ((BEAM10(I) - BEAM9(I)) * DELRLM ) / DELWV
            BEAM(I) = BEAM10(I) - XX
 40   CONTINUE

 200  CONTINUE
      RETURN
      END
```

```
      SUBROUTINE RNGINDX(JYR,JWV,PARM1,PARM2,PARM3,RNGMIN,
     &                   RNG1,RNG2,DELTAT,DELTAR,RR1,JSTRT,JSTP,JPTS)
C-----------------------------------------------------------------
C
C        This program was written by Kathleen Healy from
```

```
C         NOAA/ERL/Wave Propagation Laboratory/Atmospheric Lidar Group
C
C         Software Responsibility : A. Weickmann   (303) 497-6595
C         Scientific Responsibility : M.J. Post     (303) 497-6048
C
C         Last Modified :   8/10/89 by  AMW
C-----------------------------------------------------------------------
C                 INPUTS
C Variable  type,common              Description
C --------  -----------  ------------------------------------------
C         JYR          i,passed      2 digit year from tape
C         JWV          i,passed      = 0 if wavelength is 10.59 um
C                                          = 1 if wavelength is 9.25 um
C                                          = 2 if wavelength is 9.56
C         PARM1        r,passed      Digitizer rate(MHz) or 1/range gate spacing(us)
C                                          (for moment)
C         PARM2        r,passed      Delay to first sample (microseconds)
C         PARM3        r,passed      Number of samples per pulse
C         RNGMIN       r,passed      Minimum range that RNG1 must not go below (km)
C         RNG1         r,passed      Beginning range (km)
C         RNG2         r,passed      Ending range (km)
C
C                OUTPUTS
C Variable  type,common              Description
C --------  -----------  ------------------------------------------
C         RNG1         r,passed      Input RNG1 recalculated if necessary
C         RNG2         r,passed      Input RNG2 recalculated if necessary
C         DELTAT       r,passed      Range gate spacing (us) i.e. spacing of
C                                          samples in microseconds
C         DELTAR       r,passed      Range gate spacing (km) i.e. spacing of samples
C         RR1          r,passed      Range value that corresponds to index of 1
C         JSTRT        i,passed      Index value corresponding to RNG1
C         JSTP         i,passed      Index value corresponding to RNG2
C         JPTS         i,passed      Number of points
C-----------------------------------------------------------------------

C         OUTADV = some sort of delay supplied by M.J.Post
C               =1.80 for 10.59 microns, new lidar system
C               =2.47 for 9.25  microns, new lidar system
C               =1.93 for 9.56 microns,  new lidar system
C               =2.0  for lidar data before 1986 (left in as reference.)

          REAL*4 DELTAR,DELTAT,RNG1,RNG2,RR1,DELAY,RNGMIN,OUTADV
          REAL*4 STTIM,SPTIM,PARM1,PARM2,PARM3
          INTEGER*2 JSTRT, JSTP, JYR, JWV, MAXPTS, JPTS

          IF (RNG1.LT.RNGMIN) RNG1 = RNGMIN
          MAXPTS = NINT2(PARM3)
          IF (MAXPTS.GT.2048) MAXPTS = 2048
          OUTADV=1.80
          IF (JWV.EQ.1) OUTADV = 2.47
          IF (JWV.EQ.2) OUTADV = 1.93
          IF (JYR.LT.86) OUTADV=2.0
          DELAY = PARM2-OUTADV
          DELTAT = 1.0/PARM1
          DELTAR = .149896*DELTAT
          RR1=DELAY*.149896
          STTIM = RNG1 /.149896 - DELAY
          JSTRT = INT2(STTIM/DELTAT + 0.5)
          SPTIM = RNG2/.149896 - DELAY
          JSTP = INT2(SPTIM/DELTAT + 0.5)
          IF (JSTP.GT.MAXPTS) JSTP = MAXPTS
          IF (JSTP-JSTRT+1 .GT. MAXPTS) JSTRT = JSTP - MAXPTS + 1
          IF (JSTRT.LT.1) JSTRT = 1
          JPTS = JSTP - JSTRT + 1
          IF (JSTRT.EQ.1) RNG1=RR1
          IF (JSTP.EQ.MAXPTS) RNG2 = RNG1 + (JPTS-1) * DELTAR

          RETURN
          END


          SUBROUTINE SITE(CFUNC,NUM,MSITE,CSITES,SITALT)
C-----------------------------------------------------------------------
C         Accesses a lidar site file containing a site number,
C         location and altitude. User can add, delete or list sites.
C         Sites are not renumbered when one is deleted unless
C         deleted site is last site.  This way archived data is
C         always referring to the same site.
C         The site file is /usr/local/database/LIDARLOCS.
C-----------------------------------------------------------------------
C
C         This program was written by personnel from
C         NOAA/ERL/Wave Propagation Laboratory/Atmospheric Lidar Group
C
C         Software Responsibility : A. Weickmann   (303) 497-6243
C         Scientific Responsibility : M.J. Post     (303) 497-6048
C
C         Modified :  8/08/89        by  AMW
C         Modified :  1/07/91        by amw          improve inputs
C-----------------------------------------------------------------------
C                 INPUTS
C Variable  type,common              Description
C --------  -----------  ------------------------------------------
C  CFUNC        c,passed     ADD, LIS or DEL  specify function to occur
C
C                 OUTPUTS
C Variable  type,common              Description
C --------  -----------  ------------------------------------------
C  NUM                 i,passed      Number of sites in site file
C  MSITE( )     i,passed     Array of site numbers
C  CSITES( )    c,passed     Array of site locations
C  SITALT( ) r,passed        Array of site altitudes
C-----------------------------------------------------------------------

          PARAMETER (MAXSIT=50)

          REAL*4 SALT,SITALT(MAXSIT)
          INTEGER*4 MST,MSITE(MAXSIT)
          CHARACTER*3 CFUNC
          CHARACTER*30 CSITE,CSITES(MAXSIT)

 1111     format(a1)
 2222     format(I20)
 3333     format(f20.0)

          OPEN (UNIT=60,FILE='/usr/local/database/LIDARLOCS',STATUS='OLD')

C         Put list of site numbers, sites and altitudes in arrays
C         A deleted site is set to -1 but not removed from the list.

          READ(60,900) NUM
          DO 25 I=1,NUM
               READ(60,910) MSITE(I),SITALT(I),CSITES(I)
 25       CONTINUE
 910      FORMAT(I3,1X,F6.3,1X,A30)

          IF(CFUNC.EQ.'LIS') THEN
               CLOSE(60)
               RETURN
          ENDIF

          IF(CFUNC.EQ.'DEL') GOTO 50

C         Add a site and altitude to the list

          IF(NUM.EQ.MAXSIT) THEN
               PRINT *, 'Contact a programmer:  the number of sites'
               PRINT *, 'allowed in the site.f routine has been exceeded.'
               CLOSE(60)
               RETURN
          ENDIF

          IF(MSITE(NUM).EQ.-1) THEN
               CSITES(NUM)='
               GOTO 40
          ENDIF
          NUM=NUM+1
 40       PRINT *
          PRINT *, 'Enter a site name (max 30 characters) '
          PRINT *,' ? : '
```

```
      READ(*,'(A30)',err=40) CSITE

 45   PRINT *
      PRINT *,' Enter the altitude in km for the site'
      PRINT *,' ? : '
      READ (*,3333,err=45) SALT

      MSITE(NUM)=NUM
      SITALT(NUM)=SALT
      CSITES(NUM)=CSITE
      GOTO 100

C     Delete sites, set site number to -1

 50   CONTINUE
      PRINT *
      PRINT *,' Enter the site number you want deleted'
      PRINT *,' ? : '
      READ (*,2222,err=50) MST

      DO 70 I=1,NUM
            READ(60,910) MSITE(I),SITALT(I),CSITES(I)
            IF(MSITE(I).EQ.MST) THEN
                  MSITE(I)=-1
                  GOTO 100
            ENDIF
 70   CONTINUE

      PRINT *
      PRINT *,' Could not find the site number you wanted to '
      PRINT *,'  delete. Try again.'
      CLOSE(60)
      RETURN

 100  CONTINUE
      REWIND (60)
      WRITE(60,900) NUM
      DO 30 I=1,NUM
            WRITE(60,910) MSITE(I),SITALT(I),CSITES(I)
 30   CONTINUE

 900  FORMAT(I3)
      CLOSE(60)
      RETURN

      END


      FUNCTION SNRADJ (SNR, NUMAVG)
C ------------------------------------------------------------
C     This function computes the moment data SNR adjustment (dB).
C ------------------------------------------------------------
C
C     This program was written by personnel from
C     NOAA/ERL/Wave Propagation Laboratory/Atmospheric Lidar Group
C
C     Software Responsibility : A. Weickmann   (303) 497-6595
C     Scientific Responsibility : M.J. Post     (303) 497-6048
C
C     Last Modified :   7/20/89 by  AMW
C-------------------------------------------------------------
C            INPUTS
C Variable  type,common          Description
C --------  -----------  ----------------------------------
C     SNR         r,passed      Moment data SNR (dB) value for which
C                                       adjustment is to be calculated.
C     NUMAVG      i,passed      No. of pulses averaged by moment data
C                                       processor (NUMAVG is from front panel
decode.)
C
C            OUTPUTS
C Variable  type,common          Description
C --------  -----------  ----------------------------------
C  SNRADJ             r,function    Adjusted value for the SNR
C-------------------------------------------------------------
```

```
      DIMENSION SNRCOR(4,9) /
     &          2.3, 2.3, 2.3, 2.3, 2.3,
     &     2.3, 2.3, 2.4, 2.5,
     &     2.4, 2.5, 2.8, 3.0,
     &     3.0, 3.5, 4.3, 4.7,
     &     4.2, 5.2, 6.2, 6.9,
     &     5.6, 6.7, 7.8, 8.6,
     &     6.3, 7.4, 8.7, 9.5,
     &     6.5, 7.7, 9.0, 9.9,
     &     6.5, 7.8, 9.0, 10.0 /

      DIMENSION NPAVG(4) / 1, 2, 4, 8 /
      DIMENSION SNRMOM(9) / -20.,-15.,-10.,-5.,0.,5.,10.,15.,20. /

      INTEGER*2 I, J, NAVG, NUMAVG

C - If # pulses averaged is less than 1, something is terribly wrong.
C - Return SNRADJ = -9999.00

      IF (NUMAVG .LT. 1) THEN
            SNRADJ = -9999.00
            RETURN
      END IF


C - If SNR is less than -20 DB, adjusted SNR is the same for any number
C - of pulses averaged (curves collapse to SNRCOR(1,1) ).
      IF (SNR .LT. SNRMOM(1)) THEN
            SNRADJ = SNRCOR(1,1)
            RETURN
      END IF


C - If number of pulses averaged is greater than 8, set it to 8.
      IF (NUMAVG .GT. NPAVG(4)) THEN
            NAVG = NPAVG(4)
      ELSE
            NAVG = NUMAVG
      END IF

      DO I = 1,4
            IF (NAVG .LT. NPAVG(I)) GO TO 50
      END DO
      I = I - 1


C - If SNR is greater than 20 DB, then interpolation between columns of
C - SNRCOR is not necessary.  The curves are horizontal lines beyond
C - 20 DB.
      50    CONTINUE
      IF (SNR .GT. SNRMOM(9)) THEN
            IF (NAVG .GE. NPAVG(4)) THEN
                  SNRADJ = SNRCOR(4,9)
                  RETURN
            ELSE
                  CALL INTERP (FLOAT(NPAVG(I-1)), SNRCOR(I-1,9),
     &              FLOAT(NPAVG(I)), SNRCOR(I,9), FLOAT(NAVG), ADJ)
                  SNRADJ = ADJ
                  RETURN
            END IF
      END IF


      DO J = 1,9
            IF (SNR .LT. SNRMOM(J)) GO TO 100
      END DO
      J = J - 1

 100  CONTINUE

C - First, interpolate across neighboring columns of two rows.
      CALL INTERP (SNRMOM(J-1),SNRCOR(I-1,J-1),
     &                  SNRMOM(J),SNRCOR(I-1,J), SNR, ADJIM1)     !Row 1
      CALL INTERP (SNRMOM(J-1),SNRCOR(I,J-1),
     &                  SNRMOM(J),SNRCOR(I,J), SNR, ADJI)                 !Row
2

C - Now, interpolate across the two interpolated points from above.
```

```
CALL INTERP (FLOAT(NPAVG(I-1)),ADJIM1,
  &                FLOAT(NPAVG(I)), ADJI, FLOAT(NAVG), ADJ)

SNRADJ = ADJ
RETURN
END

SUBROUTINE INTERP (X1, Y1, X2, Y2, X, YNEW)

YNEW = Y1 + (Y2 - Y1) * ((X - X1) / (X2 - X1))
RETURN
END
```

```
      subroutine sondeabsp(wn,sondht,rsdata)
C-----------------------------------------------------------------
C
C       Calculate an atmospheric absorption profile from a radiosonde
C       profile.  The Hitran database and a subset of the
C       software written in the atmabs.f program are used here.
C       Molecules of water, carbon dioxide, and water continuum
C       are used to account for absorption.
C-----------------------------------------------------------------
C
C       This program was written by personnel from
C       NOAA/ERL/Wave Propagation Laboratory/Atmospheric Lidar Group
C
C       Software Responsibility : A. Weickmann   (303) 497-6595
C       Scientific Responsibility : M.J. Post    (303) 497-6048
C
C       Modified:  2/27/91        amw  add comments and clean text
C-----------------------------------------------------------------
C       program output is at one frequency/wavelength.
C       The atmosphere is modelled by equations.
C       Line profiles are assumed to be Lorentzian even at high
C       altitudes and are corrected for temperature and pressure.
C       Maximum input of 4000 line strengths.
C
C       The number-molecule correlation in the program is:
C               1  -     H2O     5  -  CO
C               2  -     CO2     6  -  CH4
C               3  -     O3      7  -  O2
C               4  -     N2O     8  -  Water Continuum
C             Right now only molecules 1,2 and 8 are included.
C
C  Program Variables :
C  EW - saturation vapor pressure (mbars)
C  N1 - molecular density of H2O (molecules cm**-2 m**-1 *path of 1000m)
C  N2 - molecular density of CO2 (molecules cm**-2 m**-1 *path of 1000m)
C  N8 - equal to N1, molecular density of H2O used to find H2O continuum
C  PD - pressure (mbars)
C  RH - relative humidity (%)
C  RSDATA - filename for radiosonde data
C  T - temperature (Kelvin)
C  V1 - volumetric mixing ratio of H2O (no units)
C  V2 - volumetric mixing ratio of CO2 (no units)
C  WN - wavenumber (cm**-1)
C  Z - height (meters)
C::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C  ABUND - relative natural abundaces of isotopes
C  ALF0 - air broadened halfwidth (HWHM) (cm**-1 per atmos) at 296K
C  CAY - absorption coefficient
C  CO2ABS - absorption coefficient for CO2 (km**-1)
C  EN - coefficient of temperature dependence of ALF0
C  ENDSCN - ending V of output
C  EPP - lower state energy (cm**-1)
C  GNU - frequency (cm**-1)
C  GNUBGN,GNUEND -
C  H2OABS - absorption coefficient for H2O (km**-1)
C  ISO - isotope number
C  MOL - molecule index # of line parameter data
C  NOUT - number of frequencies in scan to output = 1
C  NUMLIN - total number of molecular lines affecting the calculations
C  PH - H2O continuum (km**-1)
C  S - line strength (cm**-1 per (molecules cm-2)) at 296K
C  TOTAL - summed absorp. coeff. for all molecules (km**-1)
C  TT - corrected temperature
C  V - frequency of absorption (cm**-1). Wavelength (um) = 1E4/V.
C  WINGS - frequency both above and below V out to which lines are
C          included when calculating contributions from wings.
C::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::

      DIMENSION ABUND(7,10), ALF0(4000), ALFA0(4000), CAY(7), CS2(7)
      DIMENSION C2(7), EN(4000), EPP(4000), GAS(8), GNU(4000)
      DIMENSION ISO(4000), MOL(4000), MOLL(8), PH(1000)
      DIMENSION S(4000), SS(4000), TOTAL(1000)
      DIMENSION H2OABS(1000), CO2ABS(1000)
      DIMENSION EW(1000), N1(1000), N2(1000), N8(1000)
      DIMENSION PD(1000), RH(1000), T(1000), V1(1000), Z(1000)
      REAL*8 CA, CAY, CS1, C0, EPP, SS, TTEMP, V
      REAL  N1, N2, N3, N4, N5, N6, N7, N8
      CHARACTER  RSDATA*25, MSTRING*35
```

```
C  CONSTANTS -
      G = 9.81      !(m s**-2 , gravatational acceleration)
      V2 = 3.48E-04       !(volumetric mixing ratio of CO2)

      NORDER = 1


C  PROFILE VERSION 0 : SONDE ::::::::::::::::::::::::::::::::::::::
100       CONTINUE

C  open radiosonde data file and read in altitude (m), pressure (mbars),
C  temperature (deg C) , and relative humidity (%)

107       FORMAT(A14)
      OPEN (UNIT=20,FILE=RSDATA,STATUS='OLD',err=6030)
      NPTS = 0
      DO 120 I=1,1000
       READ(20,*,END=125) Z(I), PD(I), T(I), RH(I)
       NPTS = NPTS + 1
120       CONTINUE
125       CLOSE(20)

C  find order of input altitudes i.e., descending or ascending

      IF (Z(2) .LT. Z(1)) NORDER = -1
      if(norder.eq.1) then
            sondht = z(npts)
      else
            sondht = z(1)
      endif

C  calculate saturation vapor pressure (EW), and H2O volumetric mixing
C  ratio after converting temperature to Kelvin

      DO 150 I = 1,NPTS
       T(I) = T(I) + 273.15
       EW(I) = FEW(T(I))
       V1(I) = FV1RH(EW(I),PD(I),RH(I))
150       CONTINUE

      GO TO 1000  !END VERSION 0


1000      CONTINUE

C  calculate molecular density of H2O (N1) and molecular density
C  of CO2 (N2) with units of (molecules cm**-2 km**-1)
c  or # molecules in cm**2 column 1 km long.  this gives absorption
c  coefficients units of km**-1.

      PATH = 1000.                  !1000 meter path or 1 km
      DO 1100 I=1,NPTS
       N1(I) = FN(PD(I),T(I),V1(I)) * PATH
       N2(I) = FN(PD(I),T(I),V2) * PATH
       N8(I) = N1(I)
1100      CONTINUE

C  The following are concentrations for molecules with index 3 - 7
C  as described above.  At this time they are not used. If used
C  these terms should be put into array form and the code containing
C  these variables should be changed accordingly.
      N3 = 0
      N4 = 0
      N5 = 0
      N6 = 0
      N7 = 0

C  COMPUTE ABSORPTION HERE ::::::::::::::::::::::::::::::::::::::::

C  Flag to indicate which molecules to find absorption for.
      GAS(1) = 1    !H2O
      GAS(2) = 1    !CO2
      GAS(3) = 0
      GAS(4) = 0
      GAS(5) = 0
      GAS(6) = 0
      GAS(7) = 0
```

```
      GAS(8) = 1     !H2O CONTINUUM

C  The C2 array contains the power to which temperature is raised in
C  correcting line strengths.  Array index is by molecule number.

      C2(1)=1.5
      C2(2)=1.0
      C2(3)=1.5
      C2(4)=1.0
      C2(5)=1.0
      C2(6)=1.5
      C2(7)=1.0

C Initialize other constants

      PI=3.14159265359
      WINGS=10.
      TEMP0=296.
      NOUT=1
      DO 1200 I=1,NPTS
        PH(I)=0.
1200   CONTINUE

C Relative natural abundances of isotopes by molecule indices:
C (molecule #, isotope #)

      DO 1210 J=1,10
        DO 1210 I=1,7
            ABUND(I,J)=0.0
1210   CONTINUE
      ABUND(1,1)=.9973
      ABUND(1,2)=.002
      ABUND(1,3)=.0004
      ABUND(1,4)=.0003
      ABUND(2,1)=.9842
      ABUND(2,2)=.0110
      ABUND(2,3)=.0039
      ABUND(2,4)=.0008
      ABUND(2,5)=.000044
      ABUND(2,6)=.000009
      ABUND(2,7)=.000004
      ABUND(2,8)=.000002
      ABUND(3,1)=.9928
      ABUND(3,2)=.004
      ABUND(3,3)=.002
      ABUND(4,1)=.9904
      ABUND(4,2)=.0036
      ABUND(4,3)=.0036
      ABUND(4,4)=.002
      ABUND(4,5)=.0004
      ABUND(5,1)=.9865
      ABUND(5,2)=.011
      ABUND(5,3)=.002
      ABUND(5,4)=.0004
      ABUND(6,1)=.9883
      ABUND(6,2)=.0111
      ABUND(6,3)=.00059
      ABUND(7,1)=.9952
      ABUND(7,2)=.004
      ABUND(7,3)=.0008


C Read in line parameter data from disk file. Data is a subset of the
C 1986 AFGL compilation

      OPEN(66,FILE='/usr/local/database/HIT944_1081',status='old',
     +     err=6020)

      GNUBGN=WN-WINGS
      GNUEND=WN+WINGS
      DO 1264 JJ=1,8
        MOLL(JJ) = 0
1264   CONTINUE
      I=1
1270   READ(66,1280,end=1295) MOL(I),ISO(I),GNU(I),S(I),ALF0(I),EPP(I),
     +                       EN(I)
1280   FORMAT(I2,I1,F12.6,E10.3,10X,F5.4,5X,F10.4,F4.2)
      IF(GNU(I).LT.GNUBGN) GO TO 1270
      IF(GNU(I).GT.GNUEND) GO TO 1274
      ML=MOL(I)
      MOLL(ML)=1
      I = I + 1
      IF(I.LE.2000) GO TO 1270
      ENDSCN = GNU(I) - WINGS
      WRITE(6,1290) ENDSCN
1290   FORMAT(' MEMORY FILLED,  ENDSCN = ',F10.2)
      STOP
1295  write(6,1296) gnubgn, gnuend
1296  format(' Could not find hitran data between these',
     +        ' wavenumbers :  ',2e13.4)
      stop

C Begin absorption calculations between pressure levels

1274   NUMLIN=I-1


      DO 1300 K=1,NPTS

C Temperature corrections for layer
C Convert pressure from mb to atmospheres          1013 mb = 1 atm

1368   CS1 = (TEMP0 - T(K)) / (TEMP0 * T(K) * 0.695008)
       TT = TEMP0 / T(K)
       DO 1390 I=1,7
          CS2(I) = TT**C2(I)
1390     CONTINUE
       DO 1392 I=1,NUMLIN
          ML = MOL(I)
          SS(I) = S(I) * CS2(ML) * DEXP(-EPP(I) * CS1)
          ALFA0(I) = ALF0(I) * (PD(K)/1013.)*TT**EN(I)
1392     CONTINUE

C Calculate absorption coefficient CAY for each V (NOUT times)

       V=WN
          TOTAL(K)=0.0
1396   DO 1398 I=1,7
             CAY(I) = 0.0
1398   CONTINUE
          VLO = V - WINGS
          VHI = V + WINGS
          I5=1
          I6=1
          DO 1400 I=I5,NUMLIN
             IF (VLO .GT. GNU(I)) GO TO 1400
             I5 = I
             GO TO 1405
1400   CONTINUE
          I5 = NUMLIN
1405   DO 1410 J=I6,NUMLIN
             IF (VHI .GT. GNU(J)) GO TO 1410
             I6 = J
             GO TO 1415
1410   CONTINUE
          I6 = NUMLIN

C Apply LORENTZIAN lineshape and adjust for isotopic abundance

1415   DO 1420 I=I5,I6
             ML = MOL(I)
             ISOT=ISO(I)
             ZF = (V - GNU(I))**2 + ALFA0(I) * ALFA0(I)
             CAY(ML) = CAY(ML)+SS(I)*ABUND(ML,ISOT) * ALFA0(I) / ZF
1420   CONTINUE

C H2O continuum calculations start here. See ROBERTS ET AL, APPL OPT 15,
C P 2085

          PH(K)=0.
          IF(V.LT.800.OR.V.GT.1250) GO TO 1425
          MOLL(8) = 1
          IF(GAS(8).EQ.0.) GO TO 1425
          TTEMP = DBLE(T(K))
          C0 = (1.25E-22 + 2.34E-19 * DEXP(-V * 8.3E-3 ))*
     1         DEXP(1800.0 * (296.0 - TTEMP) / (TTEMP * 296.0))
```

```
                PW = N8(K) * 1.362E-27 * T(K)
                PH(K) = C0 * (PW + 0.002 * ((PD(K)/1013.) - PW)) * N8(K)

C H2O continuum calculations end here

1425      H2OABS(K) = N1(K) * CAY(1)
                CO2ABS(K) = N2(K) * CAY(2)

                TOTAL(K) = (1./PI) * (H2OABS(K) + CO2ABS(K) + CAY(3)*N3
     +                + CAY(4)*N4 + CAY(5)*N5 + CAY(6)*N6 + CAY(7)*N7) + PH(K)

C Temporarily keep track of wate and carbon dioxide contributions
C separately and write them to a text file for checking.

                H2OABS(K) = (1./PI) * H2OABS(K)
                CO2ABS(K) = (1./PI) * CO2ABS(K)
1300       CONTINUE

          CLOSE (66)

C OUTPUT RESULTS ::::::::::::::::::::::::::::::::::::::::::::::::::::

          L = 1
          M = NPTS
          N = 1

          IF (MVERSN .EQ. 0 .AND. NORDER .EQ. -1) THEN
           L = NPTS
           M = 1
           N = -1
          END IF

          MSTRING = 'SONDE PROFILE'

           OPEN(22,FILE='ABSPSONDE',status='unknown',err=6000)
           open(24,file='ABSPROFILE',status='new',err=6010)

C         write to text file, ABSPSONDE

           WRITE(22,5050) WN, WINGS, MSTRING
           WRITE(22,5054) RSDATA
           WRITE(22,5062)
           DO 5020 I=L,M,N
                WRITE(22,5064) Z(I), TOTAL(I), PD(I), T(I), N1(I), N2(I),
            &            H2OABS(I), PH(I), CO2ABS(I)
5020        CONTINUE

C         write to absorption profile data file

           DO 5025 I=L,M,N
                WRITE(24,5066) Z(I), TOTAL(I)
5025        CONTINUE

C PUT FORMATS FOR OUTPUT HERE

5050      FORMAT(/,' Wave number (cm**-1) =  ',F9.3,/,
            & ' Spectral reach (cm**-1) =  ',F9.3,/,' Used ',A35)
5054      FORMAT(' Radiosonde data obtained from file  ',A32)
5062      FORMAT(/,'  HEIGHT','        TOTAL          ',' PRESSURE ',' TEMPERATURE',
            & '    H2O    ','    CO2       ','          H2O          ','          H2O          ',
            & '   CO2    ',/,
            & ' (meter)','  ABSORP COEF ','  (mbars)  ','  (Kelvin)  ',
            & ' MOLEC DENS  ','MOLEC DENS  ','ABSORP COEF ',' CONTINUUM  ',
            & 'ABSORP COEF ',/,
            & ' ',' (km**-1)  ',' ',                                  ',
            & '  (mol/cm**2 in',' 1 km column',') (km**-1)  ',' (km**-1)  ',
            & '  (km**-1)  ',/)
5064      FORMAT(1X,F7.0,1PE12.3,0P,2F11.3,1PE13.3,4E12.3,0P)
5066      FORMAT(1X,F7.0,1PE12.3,0P)

          goto 7010

6000      print *,' Error opening output absorption text file.'
          goto 7010

6010      print *,' Error opening output absorption data file.'
          goto 7000

6020      print *,' Error opening line data file - HIT944_1081'
          goto 7000

6030      print *,' Error opening radiosonde data file'
          goto 7000

7000      print *,' File open error is fatal - program is terminating'
          stop

7010      continue
          return
          end

          FUNCTION FEW(T)
C  Calculates the saturation vapor pressure at temp T(K) in mbars
           REAL LOGEW
           TS = 373.16                    !(K , steam point temperature)
           P1 = -7.90298 * ((TS/T) - 1)
           P2 = 5.02808 * ALOG10(TS/T)
           P3 = -1.3816E-07 * (10.**(11.344*(1-(T/TS)))-1)
           P4 = 8.1328E-03 * (10.**(-3.49149*((TS/T)-1))-1)
           P5 = ALOG10(1013.246)
           LOGEW = P1 + P2 + P3 + P4 + P5
           FEW = 10.** (LOGEW)
           RETURN
          END

          FUNCTION FN(PD,T,V)
C  Calculates molecular density (molecules cm**-2 m**-1) from pressure
C  PD (mbars), temperature T(kelvin) and volumetric mixing ratio  V
           REAL NA
           CP = 100.    !(Newton m**-2/mbars , converts mbars to MKS)
           R = 8.3143 !(J/(mole K) , ideal gas constant)
           NA = 6.0225E23     !(molecules/mole , AVOGADRO'S number)
           FN = (PD * NA * V * CP * 1.E-04) / (R * T)
           RETURN
          END

          FUNCTION FV1RH(EW,PD,RH)
C  Calculates volumetric mixing ratio from relative humidity RH(%),
C  and also pressure PD(mbars) and saturation vapor pressure EW(mbars)
           FV1RH = (EW * RH) / (PD * 100.)
           RETURN
          END


          SUBROUTINE STDDEVNOS(XTEMP,AVNOIS2)
C-----------------------------------------------------------------------
C         This subroutine calculates the estimated standard deviation
C         of the average noise (AVNOIS).
C-----------------------------------------------------------------------
C
C         This program was written by personnel from
C         NOAA/ERL/Wave Propagation Laboratory/Atmospheric Lidar Group
C
C         Software Responsibility : A. Weickmann   (303) 497-6595
C         Scientific Responsibility : M.J. Post    (303) 497-6048
C
C         Last Modified :   8/07/89  by  AMW
C-----------------------------------------------------------------------
C                  INPUTS
C Variable  type,common                 Description
C --------  ------------  ----------------------------------------------
C         AVNOIS       r,calc       Average noise in noise window
C         AVNOIS2      r,passed     Sum of the squares of the average power.
C         BEGALT       r,calc       Beginning altitude of the noise window, km asl
C         DALT         r,calc       Requested altitude increment for averaging
C                                              processed data (as opposed to ZINC)
C         ENDALT       r,calc       Ending altitude for the noise window, km asl
C         NNOIS        i,calc       Number of noise samples in the noise window
C         XTEMP        r,passed     Square of the sums of average power.
C
C                  OUTPUTS
C Variable  type,common                 Description
```

```
C --------  -----------  ----------------------------------------
C         SDNOIS        r,calc        Standard deviation of noise in noise window
C-----------------------------------------------------------------

          INCLUDE "CB.CALC"

          REAL*4 ESTSDN, SDNORM, SIGMA, SIGMADB
          REAL*4 XTEMP, AVNOIS2

          SDNOIS = SQRT(FLOAT((NNOIS*AVNOIS2-XTEMP)/(NNOIS*(NNOIS-1))))
          ESTSDN = SDNOIS/SQRT(FLOAT(NNOIS))
          SDNORM = SDNOIS/AVNOIS
          SIGMA = ESTSDN/AVNOIS
          SIGMADB = ALOG10(SIGMA) * 10

          IF(ATTEN.le.0.0) THEN
                WRITE(40,900)
900             FORMAT(' ~~~ Noise Characteristics ~~~',/)
          ELSE
                WRITE(40,905)
905             FORMAT(' ~~~ Attenuated Noise Characteristics ~~~',/)
          ENDIF
          WRITE(40,910) BEGALT,ENDALT
910       FORMAT(' Altitude of the Noise Window : ',F4.1, ' to ',F4.1,
            +       ' km')
          WRITE(40,920) DALT
920       FORMAT(' Altitude Resolution : ',F5.2,' km')

          WRITE(40,930) AVNOIS,SDNOIS,SDNORM,ESTSDN,SIGMA,SIGMADB
930             FORMAT(' Mean Noise  = ',F7.2,/,
            +     ' Noise Standard Deviation = ',F7.2,/,
            +     ' Normalized Noise Std Dev = ',F7.4,/,
            +     ' Estimated Mean Noise Std Dev = ',F7.4,/,
            +     ' Estimated Normalized Noise Std Dev = ',E10.3,/,
            +     ' Estimated Normalized Noise Std Dev (dB) = ',F7.2,/)
          IF(ATTEN.GT.0.0) THEN
                WRITE(40,935) AVNOSA
935             FORMAT(' Unattenuated Average Noise = ',F7.2,/)
          else if(atten.eq.0.0) then
                write(40,937)
937             format(' User supplied the average noise',/)
          ENDIF

          WRITE(40,950)
950       FORMAT(/,' ~~~ Data Quality ~~~',/)

          RETURN
          END


          SUBROUTINE TPHED
C-----------------------------------------------------------------
C
C  This routine prints the tape header and calculates the start and
C  stop indices from the header info.
C
C-----------------------------------------------------------------
C
C         This program was written by personnel from
C         NOAA/ERL/Wave Propagation Laboratory/Atmospheric Lidar Group
C
C         Software Responsibility : A. Weickmann   (303) 497-6595
C         Scientific Responsibility : M.J. Post    (303) 497-6048
C
C         Modified :  7/26/89        by  AMW
C         Modified :  8/23/90        by  RPM - put DIGTIM in OUTPUTS comments
C-----------------------------------------------------------------
C                INPUTS
C Variable  type,common              Description
C --------  -----------  ----------------------------------------
C         DL1SAM       r,header      Delay to the first sample, us (microseconds)
C         DGTZRT       r,header      Digitizer rate, MHz
C         IHEAD( )     i,header      Integer header from tape
C         MF1PAS       l,multfl      True if this is 1st file in multifile process
C         MULTFI       l,multfl      TRUE if multifiles averaging in effect
C         NPTSPP       r,header      Number of points (or samples) per pulse
C         NRECDG       r,header      Number of records digitized per file in raw
C                                                        data mode
C         PARM( )      r,header      Real header from tape
C         PRF          r,calc        Pulse Repetition Frequency in Hz
C         RLAM         r,vars        Wavelength,  meters
C         TRGMOD       r,header      Trigger mode
C
C                OUTPUTS
C Variable  type,common              Description
C --------  -----------  ----------------------------------------
C         DIGTIM       r,header      Time between digitizer samples
C         ISTART       i,header      Starting sample index for data within a record
C         ISTOP        i,header      Stopping sample index for data within a record
C         MFDGTZ       r,multfl      Digitizing rate for multifiles
C         MFDL1S       r,multfl      Delay to 1st sample for multifiles
C         MFERRR       l,multfl      True if some file won't work in multifile
C                                                        processing.
C         MFNPTS       i,multfl      Value of NPTS for multifiles
C         NPTS         i,header      Number of points (or samples) to be processed
C         PRF          r,calc        Pulse Repetition Frequency in Hz
C         RANGE1       r,vars        Initial range, km
C         RANGE2       r,vars        End range, km
C-----------------------------------------------------------------

          INCLUDE "CB.CALC"
          INCLUDE "CB.HEADER"
          INCLUDE "CB.MULTFL"
          INCLUDE "CB.VARS"

          INTEGER*4  FFLAG
          INTEGER*2  JWV, ISTA, ISTO
          REAL*4 DELTAR, RNGMIN, RR1, FNPTPP
          CHARACTER*8 MODE
          CHARACTER*1 CANS

1111      FORMAT(A1)
3333      format(f20.0)

          IF (IHEAD(8) .EQ. 1) THEN
                MODE = "       Raw"
          ELSE
                MODE = "  Moment"
          END IF

C  Print tape header

          CALL NEWPAG
          IF (DGTZRT .GT. 10.01 .OR. DGTZRT .LT. 0.08) THEN
20              PRINT *,' Enter the digitizer rate '
                PRINT *,'  ? : '
                READ (*,3333,err=20) DGTZRT
          ENDIF

          WRITE(*, 1010) IFN, (IHEAD(I), I = 1, 6)
1010      FORMAT(/, " File ", I2,
            *  /, " on ", I2,2( "/", I2),
            *  /, " at ", I2,2( ":", I2),//)
          WRITE(*, 1015) MODE
1015      FORMAT(1X, "  ",5X, "Data Type: ", T45, A8)
          WRITE(*, 1020) DGTZRT
1020      FORMAT(1X, " 1", 5X, "Digitizer Rate (.08 - 10MHz) ", T45, F8.2)
          WRITE(*, 1030) DL1SAM
1030      FORMAT(1X, " 2", 5X, "Delay to 1st sample (US) "T45, F8.2)
          WRITE(*, 1040) NPTSPP
1040      FORMAT(1X, " 3", 5X, "Number of samples per pulse ", T45, F8.2)
          WRITE(*, 1050) TRGMOD
1050      FORMAT(1X, " 4", 5X, "Trigger Mode ", T45, F8.2)
          WRITE(*, 1060) PRF
1060      FORMAT(1X, " 5", 5X, "PRF ", T45, F8.2)
          WRITE(*, 1070) NRECDG
1070      FORMAT(1X, "10", 5X, "Number of records digitized. ", T45, F8.2)

          FFLAG=0
          IF (MULTFI) THEN
```

```
            IF(MF1PAS) THEN
 30               PRINT *
                  PRINT *,' Is the PRF value the same for all files used '
                  PRINT *,' in this run        (Y or N)  : '
                  PRINT *,'  (default = Y)'
                  READ (*,1111,err=30) CANS

                  if(cans.eq.' ' .or. cans.eq.'Y' .or. cans.eq.'y') then
                        fflag = 0
                  else if(cans.eq.'N' .or. cans.eq.'n') then
                        fflag = 1
                        goto 100
                  else
                        print *
                        print *,' Your answer was not an option, try again.'
                        goto 30
              endif
            ELSE
              IF(FFLAG.EQ.0) GOTO 120
            ENDIF
        ENDIF

 100    PRINT *
        PRINT *, ' Is the value for PRF correct  (Y or N) '
        PRINT *,' (default = Y)'
        PRINT *,' ? : '
        READ(*,1111,err=100) CANS

        IF (CANS .EQ. 'N' .OR. CANS .EQ. 'n') THEN
 110          print *
              PRINT *, "Enter a new prf value  "
              PRINT *,' ? : '
              READ (*,3333,err=110) PRF
        else if(cans.ne.' ' .and. cans.ne.'Y' .and. cans.ne.'y') then
              print *
              print *,' Your answer was not an option, try again.'
              goto 100
        ENDIF

C  Calculate start, stop indices from tape header information.

 120    CONTINUE
        IF(RLAM.EQ.1.0591E-5) JWV=0
        IF(RLAM.EQ.9.250E-6) JWV=1
        IF(RLAM.EQ.9.536E-6) JWV=2
        RNGMIN=1.5

        CALL RNGINDX(IHEAD(3),JWV,DGTZRT,DL1SAM,NPTSPP,RNGMIN,
          +          RANGE1,RANGE2,DIGTIM,DELTAR,RR1,ISTA,
          +          ISTO,NPTS)
        ISTART=INT4(ISTA)
        ISTOP=INT4(ISTO)

        IF (MULTFI) THEN
              IF (MF1PAS) THEN
                    MFDGTZ = DGTZRT
                    MFNPTS = NPTS
                    MFDL1S = DL1SAM
              ELSE IF ((DGTZRT .NE. MFDGTZ)
          *         .OR. (NPTS .NE. MFNPTS)
          *         .OR. (DL1SAM .NE. MFDL1S)) THEN
                    WRITE(*,*)
                    WRITE(*,*) "Multifile error nonequivalence in this file"
                    WRITE(*,*) "1st Digitizing Rate: ",MFDGTZ," this file: ",DGTZRT
                    WRITE(*,*) "1st Npts: ",MFNPTS," this file: ",NPTS
                    WRITE(*,*) "1st Delay to 1st Samp: ",MFDL1S," this file: ",DL1SAM
                    MFERRR  = .TRUE.
              ENDIF
        ENDIF
        RETURN
        END


        SUBROUTINE TPMSG(NUM,LEN)
C-----------------------------------------------------------------------
C
C  This routine prints the appropriate error message for
C  errors that occur due to tape request.
C
C-----------------------------------------------------------------------
C
C         This program was written by personnel from
C         NOAA/ERL/Wave Propagation Laboratory/Atmospheric Lidar Group
C
C         Software Responsibility : A. Weickmann   (303) 497-6595
C         Scientific Responsibility : M.J. Post    (303) 497-6048
C
C         Last Modified :   8/07/89 by  AMW
C-----------------------------------------------------------------------
C                INPUTS
C Variable  type,common              Description
C ---------  ------------  ----------------------------------------
C NUM                 i,passed       Error number
C LEN                 i,passed       Length in words of words retrieved

C                OUTPUTS
C Variable  type,common              Description
C ---------  ------------  ----------------------------------------
C                none
C-----------------------------------------------------------------------

        INTEGER * 4  LEN, NUM

        GO TO (210, 10, 30, 50, 70, 90, 110, 210, 130, 150, 210, 210,
          &   170, 190) NUM

 10           CONTINUE
        WRITE (*, 20)
 20           FORMAT (" PARITY ERROR")
        GO TO 210

 30           CONTINUE
        WRITE (*, 40) LEN
 40           FORMAT (" LENGTH ERROR,  LENGTH =", I5)
        GO TO 210

 50           CONTINUE
        WRITE (*, 60) LEN
 60           FORMAT (" PARITY & LENGTH ERROR,  LENGTH =", I5)
        GO TO 210

 70           CONTINUE
        WRITE (*, 80)
 80           FORMAT (" END-OF-FILE (EOF)")
        GO TO 210

 90           CONTINUE
        WRITE (*, 100)
 100      FORMAT (" BEGINNING-OF-TAPE (BOT)")
        GO TO 210

 110      CONTINUE
        WRITE (*, 120)
 120      FORMAT (" END-OF-TAPE (PHYSICAL)")
        GO TO 210

 130      CONTINUE
        WRITE (*, 140)
 140      FORMAT (" END-OF-TAPE (LOGIC)")
        GO TO 210

 150      CONTINUE
        WRITE (*, 160)
 160      FORMAT (" TOO MANY MAGTAPE RETRIES")
        GO TO 210

 170      CONTINUE
        WRITE (*, 180)
 180      FORMAT (" NEEDS WRITE RING")
        GO TO 210

 190      CONTINUE
        WRITE (*, 200)
```

```
200       FORMAT (" DISASTER, PHYSICAL UNIT OFF LINE, ETC.")

210       CONTINUE
        RETURN
        END
```

```
      SUBROUTINE TPSET
C-----------------------------------------------------------------------
C
C       This subroutine positions the tape at the correct file
C       and record (GETFILE)  and then uses the header information
C       to fill program variables.
C
C-----------------------------------------------------------------------
C
C       This program was written by personnel from
C       NOAA/ERL/Wave Propagation Laboratory/Atmospheric Lidar Group
C
C       Software Responsibility : A. Weickmann   (303) 497-6595
C       Scientific Responsibility : M.J. Post     (303) 497-6048
C
C       Last Modified :   7/20/89 by  AMW
C-----------------------------------------------------------------------
C                INPUTS
C Variable  type,common              Description
C --------  ----------  ------------------------------------------------
C       IFN          i,vars        Requested File number
C       IPACK( )     i,setup       MTAPE array  *DO NOT CHANGE*
C       IREC         i,vars        Requested Record Number
C
C                OUTPUTS
C Variable  type,common              Description
C --------  ----------  ------------------------------------------------
C       DGTZRT       r,header      Digitizer rate, MHz
C       DL1SAM       r,header      Delay to the first sample, us (microseconds)
C       ICURF        i,setup       Current file number
C       ICURR        i,setup       Current record number
C       IHEAD( )     i,header      Integer header from tape
C       NRECDG       r,header      Number of records digitized per file in raw
C                                                   data mode
C       NPTSPP       r,header      Number of points (or samples) per pulse
C       PARM( )      r,header      Real header from tape
C       PRF          r,calc        Pulse Repetition Frequency in Hz
C       TRGMOD       r,header      Trigger mode
C-----------------------------------------------------------------------

      INCLUDE "CB.CALC"
      INCLUDE "CB.HEADER"
      INCLUDE "CB.VARS"
      INCLUDE "CB.SETUP"

C  Position tape at correct file and record

      call tapos(ipack,ifn,irec,icurf,icurr,ihead)

C - Check contents of IHEAD(8).  If it is 1, we have Raw data.  If it
C - is 2, we have Moment data and we adjust digitizer rate, delay to 1st
C - sample, and number of points per pulse to reflect Moment data.
C       Moment data - 2 micro-second gate spacing should be read
C                          from front panel of processor.

      IF (IHEAD(8) .EQ. 1) THEN
            DGTZRT = PARM(1)
            DL1SAM = PARM(2)
            NPTSPP = PARM(3)
      ELSE
            DGTZRT = 0.5
            DL1SAM = 0.0
            NPTSPP = 96
      ENDIF

      TRGMOD = PARM(4)
      PRF = 1.E6/PARM(5)
      NRECDG = PARM(10)

      IF(IHEAD(7) .NE. IFN) THEN
        WRITE (*,*) "File # read does not match file requested"
      ENDIF

      RETURN
      END
```

```
      FUNCTION TWICEQ2(I)
C-----------------------------------------------------------------------
C
C  This function decodes the raw data from the lidar digitizer
C  and returns in  2Q2 the power for the point "I".  A digitizer
C  correction is applied to 2Q2 before returning the value.  The
C  current correction is for the CLARET time period.
C
C-----------------------------------------------------------------------
C
C       This program was written by personnel from
C       NOAA/ERL/Wave Propagation Laboratory/Atmospheric Lidar Group
C
C       Software Responsibility : A. Weickmann   (303) 497-6595
C       Scientific Responsibility : M.J. Post     (303) 497-6048
C
C       Last Modified :  10/16/89 by  AMW
C-----------------------------------------------------------------------
C                INPUTS
C Variable  type,common              Description
C --------  ----------  ------------------------------------------------
C       I                    i,passed     Index into the IDAT raw signal array.
C       IDAT( )      i,signl       Array to store data from tape
C
C                OUTPUTS
C Variable  type,common              Description
C --------  ----------  ------------------------------------------------
C  TWICEQ2      r,function  Real portion of signal squared plus
C                                                 imaginary portion of signal squared.
C-----------------------------------------------------------------------

      INCLUDE "CB.SIGNL"

      INTEGER*2 RBYTE, LBYTE
      INTEGER*2 MASK, SHIFT, ITEMP, I
      REAL*4 COR, TWICEQ2,Q2,C0,C1,C2,C3

      MASK = 377K
      RBYTE = IAND (IDAT(I), MASK)
      ITEMP = NOT2(MASK)
      LBYTE = IAND (IDAT(I), ITEMP)
      SHIFT = -8
      LBYTE = ISHFT2(LBYTE, SHIFT)
      SHIFT = 7
      IF (BTEST2(RBYTE, SHIFT)) RBYTE = RBYTE - 256
      IF (BTEST2(LBYTE, SHIFT)) LBYTE = LBYTE - 256
C     R2Q2 = float(RBYTE)*float(RBYTE)+float(LBYTE)*float(LBYTE)
      Q2 =    (FLOAT(LBYTE)+0.7) * (FLOAT(LBYTE)+0.7)

C     These are the coefficients for point by point correction
C      along the pulse on unaveraged data.  11/8/89

      C0 = 1.27534E+00
      C1 = -3.64037E-04
      C2 = -1.65251E-08
      C3 = 2.48780E-11

      COR = C0 + (C1*Q2) + (C2*(Q2**2)) + (C3*(Q2**3))

      TWICEQ2 = 2. * COR * Q2
      RETURN
      END
```

```
      FUNCTION TWICER2(I)
C-----------------------------------------------------------------------
C
C  This function decodes the raw data from the lidar digitizer
C  and returns in  2Q2 the power for the point "I".  A digitizer
C  correction is applied to 2Q2 before returning the value.  The
C  current correction is for the CLARET time period.
C
C-----------------------------------------------------------------------
C
```

```
C         This program was written by personnel from
C         NOAA/ERL/Wave Propagation Laboratory/Atmospheric Lidar Group
C
C         Software Responsibility : A. Weickmann   (303) 497-6595
C         Scientific Responsibility : M.J. Post     (303) 497-6048
C
C         Last Modified :  10/16/89 by  AMW
C-----------------------------------------------------------------------
C                INPUTS
C Variable  type,common          Description
C ---------  -----------  ------------------------------------------
C         I                 i,passed    Index into the IDAT raw signal array.
C         IDAT( )      i,signl        Array to store data from tape
C
C                OUTPUTS
C Variable  type,common          Description
C ---------  -----------  ------------------------------------------
C  TWICEQ2      r,function  Real portion of signal squared plus
C                                          imaginary portion of signal squared.
C-----------------------------------------------------------------------

      INCLUDE "CB.SIGNL"

      INTEGER*2 RBYTE, LBYTE
      INTEGER*2 MASK, SHIFT, ITEMP, I
      REAL*4 COR, TWICEQ2,Q2,C0,C1,C2,C3
      REAL*4 TWICER2,R2

      MASK = 377K
      RBYTE = IAND (IDAT(I), MASK)
      ITEMP = NOT2(MASK)
      LBYTE = IAND (IDAT(I), ITEMP)
      SHIFT = -8
      LBYTE = ISHFT2(LBYTE, SHIFT)
      SHIFT = 7
      IF (BTEST2(RBYTE, SHIFT)) RBYTE = RBYTE - 256
      IF (BTEST2(LBYTE, SHIFT)) LBYTE = LBYTE - 256
C     R2Q2 = float(RBYTE)*float(RBYTE)+float(LBYTE)*float(LBYTE)
      R2 =    (FLOAT(RBYTE)+0.7) * (FLOAT(RBYTE)+0.7)

C     These are the coefficients for point by point correction
C      along the pulse on unaveraged data.  11/8/89

      C0 = 1.27534E+00
      C1 = -3.64037E-04
      C2 = -1.65251E-08
      C3 = 2.48780E-11

      COR = C0 + (C1*R2) + (C2*(R2**2)) + (C3*(R2**3))

      TWICER2 = 2. * COR * R2
      RETURN
      END


      SUBROUTINE TXGRFR(JTIME,FILES,NUMF,VAR1)
C-----------------------------------------------------------------------
C         Open the graphics window, plot the text for the range vs.
C          power plot and the altitude vs. beta plot for both
C          single and multiple files.
C-----------------------------------------------------------------------
C
C         This program was written by personnel from
C         NOAA/ERL/Wave Propagation Laboratory/Atmospheric Lidar Group
C
C         Software Responsibility : A. Weickmann   (303) 497-6595
C         Scientific Responsibility : M.J. Post     (303) 497-6048
C
C         Modified :  8/03/89        by  AMW
C         Modified :  9/10/90        by  RPM - writing filter length
C-----------------------------------------------------------------------
C                INPUTS
C Variable  type,common          Description
C ---------  -----------  ------------------------------------------
C         BEGALT        r,calc       Beginning altitude of the noise window, km asl
C         BETAFL        i,calc       Flag to plot the BETA profile Y or N
C         BTYPE         i,vars       Beta calc. type: A=archive, B=block, R=running avg.
C         ENDALT        r,calc       Ending altitude for the noise window, km asl
C         IFN           i,vars       File number
C         IPLTN         i,vars       Consecutive plot number
C         IREC          i,vars       Start record number
C         IREND         i,vars       End record number
C         MODEL         i,vars       Atmospheric absorption model
C         OP                  i,vars Operator's initials
C         RLAM          r,vars       Wavelength,  meters
C         SITNAM        c,charac     Name of lidar site
C
C                OUTPUTS
C Variable  type,common          Description
C ---------  -----------  ------------------------------------------
C                none
C-----------------------------------------------------------------------

      INCLUDE "CB.CHARAC"
      INCLUDE "CB.VARS"
      INCLUDE "CB.CALC"

      INTEGER*2 NUMF, JTIME(9), FILES(10)
      INTEGER*2 IX,IY,ICHR
      INTEGER*2 IXMIN,IXMAX, IYMIN,IYMAX
      INTEGER*4 IDATE(3), ITIME(3)
      REAL*4 VAR1, RLAMM
      CHARACTER*80 CSTR, CEMPTY
      CHARACTER*20 CBLANK

      CBLANK = '                    '
      CEMPTY = CBLANK//CBLANK//CBLANK//CBLANK

      ICHR = 2
      IX=2
      IY=1024
      IXMIN=0
      IXMAX=1023
      IYMIN=0
      IYMAX=767
      CSTR = CEMPTY

      CALL NEWPAG
      CALL TERM(IX,IY)
      CALL TWINDO(IXMIN,IXMAX,IYMIN,IYMAX)

C  Write block/running filter length.
      IF(BETAFL.EQ.'Y ') THEN
            IX=175
            IF (IFN.EQ.-1) THEN
                  IY = 767
            ELSE
                  IY = 747
            ENDIF
            IF (BTYPE.EQ.'R ') THEN
                  WRITE(CSTR,1040) DALT
            ELSE
                  WRITE(CSTR,1030) DALT
            ENDIF
            CALL STSABS(IX,IY,CSTR,ICHR)
            CSTR=CEMPTY
      ENDIF

C  Write avgpwr/beta multiple files.
      IF(IFN.EQ.-1) THEN
            IF(BETAFL.EQ.'N ') THEN
                  WRITE(CSTR,900) (FILES(I),I=1,NUMF)
            ELSE
                  WRITE(CSTR,910) (FILES(I),I=1,NUMF)
            ENDIF
            IX=175
            IY=747
            CALL STSABS(IX,IY,CSTR,ICHR)
            CSTR=CEMPTY
      ENDIF

C  Write noise window or average power.
      IF(BETAFL.EQ.'Y ') THEN
```

```
            WRITE(CSTR,920) BEGALT, ENDALT
        ELSE
            WRITE(CSTR,930) VAR1
        ENDIF
        IY=727
        IX=175
        CALL STSABS(IX,IY,CSTR,ICHR)
        CSTR=CEMPTY

C  Write absorption model no.
        WRITE(CSTR,940) MODEL
        IY=IY-20
        CALL STSABS(IX,IY,CSTR,ICHR)
        CSTR=CEMPTY

C  Write wavelength.
        RLAMM=RLAM*1.0E6
        WRITE(CSTR,950) RLAMM
        IY=IY-20
        CALL STSABS(IX,IY,CSTR,ICHR)
        CSTR=CEMPTY

C  Write date.
        IF(IFN.EQ.-1) GOTO 50
        WRITE(CSTR,960) (JTIME(K),K=1,3)
        IX=700
        IY=747
        CALL STSABS(IX,IY,CSTR,ICHR)
        CSTR=CEMPTY

C  Write time.
        WRITE(CSTR,970) (JTIME(K),K=4,6)
        IY=IY-20
        CALL STSABS(IX,IY,CSTR,ICHR)
        CSTR=CEMPTY

        WRITE(CSTR,980) FILES(1)
        IY=IY-20
        CALL STSABS(IX,IY,CSTR,ICHR)
        CSTR=CEMPTY

        WRITE(CSTR,990) IREC,IREND
        IY=IY-20
        CALL STSABS(IX,IY,CSTR,ICHR)
        CSTR=CEMPTY

        GOTO 100

50      CONTINUE

C  Multifile date and time.
        CSTR=CEMPTY
        WRITE(CSTR,960) (JTIME(K),K=1,3)
        IY=727
        IX=700
        CALL STSABS(IX,IY,CSTR,ICHR)
        CSTR=CEMPTY

        WRITE(CSTR,1000) JTIME(4),JTIME(5),JTIME(7),JTIME(8)
        IY=IY-20
        CALL STSABS(IX,IY,CSTR,ICHR)
        CSTR=CEMPTY

100     CONTINUE

C  Put plot number and site name in lower left corner.
        ICHR = 4
        IX=20
        IY=5
        WRITE(CSTR,1010) IPLTN, SITNAM
        CALL STSABS(IX,IY,CSTR,ICHR)
        CSTR=CEMPTY

C  Put op initials and current date and time in lower right corner.
        CALL DATE(IDATE)
        CALL TIME(ITIME)
        IDATE(1) = IDATE(1)-1900
        IX=730
        IY=5
        WRITE(CSTR,1020) OP,IDATE(2),IDATE(3),IDATE(1),(ITIME(I),I=1,3)
        CALL STSABS(IX,IY,CSTR,ICHR)
        CSTR=CEMPTY
        ICHR=2

C       Format Statements

900     FORMAT('Avgpwr Multiple Files : ',10I3)
910     FORMAT('Beta Multiple Files : ',10I3)
920     FORMAT('Noise Window : ',F4.1,' to ',F4.1,' km')
930     FORMAT('Average Power : ',F8.2)
940     FORMAT('Absorption Model No. ',I1)
950     FORMAT('Wavelength : ',F5.2,' microns')
960     FORMAT('Date : ',I2,2('/',I2))
970     FORMAT('Time : ',I2,2(':',I2))
980     FORMAT('File : ',I2)
990     FORMAT('Recs : ',I4,' thru ',I4)
1000    FORMAT('Time : ',I2,':',I2,' - ',I2,':',I2)
1010    FORMAT('Plot # ',I2,' at ',A30)
1020    FORMAT(A2,', Processed : ',I2,2('/',I2),2X,I2,2(':',I2))
1030    FORMAT('Block Filter Length :',F4.1,' km')
1040    FORMAT('Running Filter Length :',F4.1,' km')

        RETURN
        END
```

```
      SUBROUTINE TXGRFT(JTIME,VAR1)
C-----------------------------------------------------------------------
C     Open graphics window and put text on time vs. power plot.
C-----------------------------------------------------------------------
C
C     This program was written by personnel from
C     NOAA/ERL/Wave Propagation Laboratory/Atmospheric Lidar Group
C
C     Software Responsibility : A. Weickmann   (303) 497-6595
C     Scientific Responsibility : M.J. Post    (303) 497-6048
C
C     Last Modified :   8/04/89 by  AMW
C-----------------------------------------------------------------------
C                INPUTS
C Variable  type,common              Description
C --------  -----------  ----------------------------------------------
C     IFN         i,vars          File number
C     IPLTN       i,vars          Consecutive plot number
C     IREC        i,vars          Start record number
C     IREND       i,vars          End record number
C     OP                  i,vars  Operator's initials
C     R1                  r,header    Begin range, adjusted for lidar altitude and
C                                           elevation angle
C     R2                  r,header    End range, adjusted for lidar altitude and
C                                           elevation angle
C     RLAM        r,vars          Wavelength,  meters
C     SITNAM      c,charac        Name of lidar site
C
C                OUTPUTS
C Variable  type,common              Description
C --------  -----------  ----------------------------------------------
C                none
C-----------------------------------------------------------------------

      INCLUDE "CB.CHARAC"
      INCLUDE "CB.HEADER"
      INCLUDE "CB.VARS"

      INTEGER*2 IX,IY,ICHR, JTIME(9)
      INTEGER*2 IXMIN,IYMIN,IXMAX,IYMAX
      INTEGER*4 IDATE(3), ITIME(3)
      CHARACTER*80 CSTR, CEMPTY
      CHARACTER*20 CBLANK
      REAL*4 RLAMM,VAR1

      CBLANK='
      CEMPTY=CBLANK//CBLANK//CBLANK//CBLANK

      ICHR=2
      IX=2
      IY=1024
      IXMIN=0
      IXMAX=1023
      IYMIN=0
      IYMAX=767

      CALL NEWPAG
      CALL TERM(IX,IY)
      CALL TWINDO(IXMIN,IXMAX,IYMIN,IYMAX)

      IX=175
      IY=747
      WRITE(CSTR,900) R1,R2
      CALL STSABS(IX,IY,CSTR,ICHR)
      CSTR=CBLANK

      IY=IY-20
      WRITE(CSTR,910) VAR1
      CALL STSABS(IX,IY,CSTR,ICHR)
      CSTR=CEMPTY

      RLAMM = RLAM*1.0E6
      IY=IY-20
      WRITE(CSTR,920) RLAMM
      CALL STSABS(IX,IY,CSTR,ICHR)
      CSTR=CEMPTY

      IX=740
      IY=747
      WRITE(CSTR,930) (JTIME(K),K=1,3)
      CALL STSABS(IX,IY,CSTR,ICHR)
      CSTR=CEMPTY

      IY=IY-20
      WRITE(CSTR,940) (JTIME(K),K=4,6)
      CALL STSABS(IX,IY,CSTR,ICHR)
      CSTR=CEMPTY

      IY=IY-20
      WRITE(CSTR,950) IFN
      CALL STSABS(IX,IY,CSTR,ICHR)
      CSTR=CEMPTY

      IY=IY-20
      WRITE(CSTR,960) IREC,IREND
      CALL STSABS(IX,IY,CSTR,ICHR)
      CSTR=CEMPTY

C     Put plot number and location in lower left corner

      ICHR=4
      IX=20
      IY=5
      WRITE(CSTR,970) IPLTN, SITNAM
      CALL STSABS(IX,IY,CSTR,ICHR)
      CSTR=CEMPTY

C     Put op initials and current date and time in lower right corner

      CALL DATE(IDATE)
      CALL TIME(ITIME)
      IDATE(1) = IDATE(1)-1900
      IX=730
      IY=5
      WRITE(CSTR,980) OP,IDATE(2),IDATE(3),IDATE(1),(ITIME(I),I=1,3)
      CALL STSABS(IX,IY,CSTR,ICHR)
      CSTR=CEMPTY

C     Format Statements

 900  FORMAT('Altitude, Asl : ',F5.2,' to ',F5.2,' km')
 910  FORMAT('Average Power : ',F8.2)
 920  FORMAT('Wavelength : ',F5.2,' microns')
 930  FORMAT('Date : ',I2,2('/',I2))
 940   FORMAT('Time : ',I2,2(':',I2))
 950  FORMAT('File : ',I2)
 960  FORMAT('Recs : ',I4,' thru ',I4)
 970  FORMAT('Plot # ',I2,' at ',A30)
 980  FORMAT(A2,', Processed : ',I2,2('/',I2),2X,I2,2(':',I2))

      RETURN
      END


      SUBROUTINE WRITEFILENC
C-----------------------------------------------------------------------
C
C     This routine writes all processing information to a processing log
C     file called  BETA.PRINT.
C
C-----------------------------------------------------------------------
C
C     This program was written by personnel from
C     NOAA/ERL/Wave Propagation Laboratory/Atmospheric Lidar Group
C
C     Software Responsibility : A. Weickmann   (303) 497-6595
C     Scientific Responsibility : M.J. Post    (303) 497-6048
C
C     Last Modified :   8/02/89 by  AMW
C-----------------------------------------------------------------------
C                INPUTS
C Variable  type,common              Description
C --------  -----------  ----------------------------------------------
```

```
C          ALTLID        r,vars        Lidar altitude in km ASL
C          ARCDAT        i,calc        Option to archive the beta profile data Y or N
C          ARCNAM        c,charac      Archive disk filename
C          AVEENG        r,vars        A/D Average energy per pulse
C          AZIMTH        r,vars        Azimuth angle of scanner, deg w.r.t north
C          BW                  r,vars  Bandwidth of signals in MHz
C          CAL           r,calc        Calib factor for adjusting beta retrieval
C          CALDAT        i,calc        Calibration date MMDDYY
C          DBCAL         r,calc        Continuous wave calibration SNR, dB
C          EJ                  r,vars  Energy/pulse (J)
C          ELEVA         r,vars        Elevation angle of scanner, deg above horizon
C          IFN           i,vars        File number
C          IHEAD( )      i,header      Integer header from tape
C          IREC          i,vars        Start record number
C          LINDB         i,plot        Linear or decibel scaling
C          LOHI          i,plot        Low to high scaling flag
C          MFDATE( ) i,multfl          Date for each file
C          MFIFN( )  i,multfl          Multi file numbers.
C          MFIREC( ) i,multfl          Multi file beg rec number.
C          MFNREC( ) i,multfl          Multi file number of recs.
C          MFTIME( ) i,multfl          Time for each file
C          MODEL         i,vars        Atmospheric absorption model
C          MULTFI        l,multfl      TRUE if multifiles averaging in effect
C          NREC          i,vars        # of records to process
C          OP                  i,vars  Operator's initials
C          PLTTYP        i,plot        Flag for single or average plots
C          PRNDAT        i,calc        Option to print out the raw data, Y or N
C          PRNPRF        i,calc        Option to print out the BETA profile Y or N
C          RANGE1        r,vars        Initial range, km
C          RANGE2        r,vars        End range, km
C          RLAM          r,vars        Wavelength,  meters
C          RSDATA        c,charac      Name of radiosonde data file
C          SHTDAT        r,calc        Shot noise for the data run, (new lidar)
C          SHTREF        r,calc        Shot noise for the calibration run, (new)
C          SITNAM        c,charac      Name of lidar site
C          SONDHT        r,vars        Altitude, m, asl, of highest radiosonde data
C
C                OUTPUTS
C Variable  type,common                 Description
C --------  -----------  ----------------------------------------
C          none
C-----------------------------------------------------------------

      INCLUDE "CB.CALC"
      INCLUDE "CB.CHARAC"
      INCLUDE "CB.HEADER"
      INCLUDE "CB.MULTFL"
      INCLUDE "CB.PLOT"
      INCLUDE "CB.VARS"

      INTEGER * 4  IDATE(3), ITIME(3), NUMREC
      INTEGER*4 IMM,IDD,IYY
      CHARACTER*20 CSTR
      CHARACTER*30 CMODEL(0:5)

      CMODEL(0) = 'No. 0  - No absorption correction'
      CMODEL(1) = 'No. 1  - Tropical'
      CMODEL(2) = 'No. 2  - Midlatitude Summer'
      CMODEL(3) = 'No. 3  - Midlatitude Winter'
      CMODEL(4) = 'No. 4  - Subarctic Summer'
      CMODEL(5) = 'No. 5  - Subarctic Winter'


      CALL DATE (IDATE)
      CALL TIME (ITIME)
      IDATE(1) = IDATE(1) - 1900
      WRITE(40, 1000) IDATE(2), IDATE(3), IDATE(1),
     +    (ITIME(I), I = 1, 3), OP
1000  FORMAT(/,' Beta Program Output  ( Processed on ', 2(I2, '/'),
     +   I2,' at ', 2(I2, ':'), I2,' by ', A2,')')

      WRITE(40, 1010)
1010  FORMAT( /,' ~~~ Data Information ~~~ ',/)

      WRITE(40,1060) ALTLID,SITNAM
1060  FORMAT(' Altitude and Location are ',F7.3,' km asl          at ',
     +     A30)
```

```
      WRITE(40,1030)
1030  FORMAT(1X,'File',4X,'Date',6X,'Time',4X,'Starting Record',
     +    2X,'# Records')
1040  FORMAT(1X,I3,3X,I2,2('/',I2),2X,I2,2(':',I2),6X,I4,9X,I4)

      IF (MULTFI) THEN
            NUMREC = 0
            DO 1038 I=1,MFNMFI
                  WRITE(40,1040) MFIFN(I), (MFDATE(I,J),J=1,3),
     +                        (MFTIME(I,J),J=1,3),MFIREC(I),MFNREC(I)
                  NUMREC = NUMREC + MFNREC(I)
1038  CONTINUE
            WRITE(40,*)

            WRITE(40,1050) NUMREC
1050  FORMAT(' Total number of records : ',I5)

      ELSE
            WRITE(40, 1040) IFN, (IHEAD(I),I=1,6),IREC,NREC
      ENDIF


      WRITE(40, 1170)
1170  FORMAT( /,' ~~~ Calibration and Processing ~~~ ',/)
      IF(CALDAT.EQ.0) THEN
            CSTR=' (Operator Entered) '
      ELSE
            IMM=CALDAT/10000
            IDD=MOD(CALDAT/100,100)
            IYY=MOD(CALDAT,100)
            WRITE(CSTR,1175) IMM,IDD,IYY
      ENDIF

1175  FORMAT('(Archive: ',I2,2('/',I2),' )')

      WRITE(40, 1180) DBCAL,CSTR
1180  FORMAT(' Calibration  dB down from expected is '
     *    ,F6.2,6X,A20)

C     WRITE(40, 1210) SHTREF,CSTR
C1210 FORMAT(' Shot Noise (dB) for the calibration is  ',F5.2,
C    +     6X,A20)

      WRITE(40, 1215) SHTDAT
1215  FORMAT(' Shot Noise (dB) for the data is  ',F5.2)

      WRITE(40, 1220) CAL
1220  FORMAT(' Calibration factor is            ',F5.3)

      BWMHZ = BW / 1.0E6
      WRITE(40, 1230) BWMHZ
1230  FORMAT(' Bandwidth (MHz) is ', F5.2)

      RLAMM = RLAM * 1.0E6
      WRITE(40,1235) RLAMM
1235  FORMAT(' Wavelength (microns) is ', F8.3)

      WRITE(40, 1240) EJ
1240  FORMAT(' Energy per pulse (J) is ', F4.2)

      WRITE(40,1245) AVEENG
1245   FORMAT(' A/D Average energy per pulse (J) is ',F5.2)

      WRITE(40, 1250) ELEVA
1250   FORMAT(' Elevation angle (deg) is ', F6.2)

      WRITE(40, 1255) AZIMTH
1255  FORMAT(' Azimuth angle (deg) from tape is ',F8.2)

      WRITE(40, 1260) RANGE1, RANGE2
1260   FORMAT(' Initial and end ranges (km) are ',F6.2, 3X, F6.2)

      if(model.eq.7) then
            write(40,1275) rsdata
            write(40,1277) sondht/1000.
1275        format(' Atmospheric absorption profile derived from',
     +                  ' radiosonde profile in file  ',a25)
1277        format(' Maximum altitude of radiosonde profile (km) ',F6.2)
```

```
      else
              WRITE(40,1270) CMODEL(MODEL)
1270        FORMAT(' Atmospheric absorption model is ',A30)
      endif

      WRITE(40, 1300)
1300  FORMAT( /,' ~~~ Plot Setup ~~~ ',/)

      IF (PLTTYP .EQ. "A ") WRITE(40, 1420)
1420  FORMAT(' Average plots only ')
      IF (PLTTYP .EQ. "S ") WRITE(40,1430)
1430  FORMAT(' Single records, then average plots ')


      IF (LINDB .EQ. "L ") THEN
            WRITE(40, 1330)
1330        FORMAT(' Scale of Signal Axis : Linear')
            IF(LOHI .EQ. 'H ') WRITE(40,1350)
            IF(LOHI .EQ. 'M ') WRITE(40,1360)
            IF(LOHI .EQ. 'L ') WRITE(40,1370)
      ELSE
            WRITE(40, 1340)
1340        FORMAT(' Scale of Signal Axis : dB')
            IF(LOHI .EQ. 'H ') WRITE(40,1380)
            IF(LOHI .EQ. 'M ') WRITE(40,1390)
            IF(LOHI .EQ. 'L ') WRITE(40,1400)
      ENDIF

1350   FORMAT('                          : 0 - 20,000  (High)')
1360  FORMAT('                           : 0 - 2,000  (Medium)')
1370  FORMAT('                           : 0 - 400  (Low)')
1380   FORMAT('                          : 0 - 40  (High)')
1390   FORMAT('                          : 0 - 20  (Medium)')
1400   FORMAT('                          : 0 - 12  (Low)')


      WRITE(40, 1470)
1470  FORMAT(/, ' ~~~ Outputs ~~~ ',/)

      WRITE(40,1475)
1475  FORMAT(' Plots :  Altitude vs. Relative Signal Power (RSP) ')

      WRITE(40,1480)
1480  FORMAT('          RSP vs. Time,  Altitude vs. Beta')
      IF(MULTFI) THEN
            WRITE(40,1485)
1485  FORMAT(' for individual files plus a composite of all files')
      ENDIF

      IF (PRNDAT .EQ. "N ") THEN
            WRITE(40, 1530)
1530  FORMAT(" Printouts : No raw data printout ")
      ELSE
            WRITE(40, 1540)
1540  FORMAT(" Printouts : Raw data printed out in file  ",
     +                     "BETA.RAWDAT")
      ENDIF

      IF (PRNPRF .EQ. "Y ") THEN
            WRITE(40, 1560)
1560  FORMAT("             BETA profile printed out in file  ",
     +                     "BETA.PRINT")

      ELSE
            WRITE(40, 1550)
1550  FORMAT("             No BETA profile printout ")
      ENDIF

      IF (ARCDAT .EQ. "N ") THEN
            WRITE(40, 1570)
1570  FORMAT(" This profile was not archived.")
      ELSE
            WRITE(40, 1590) ARCNAM
1590  FORMAT(' This profile was archived to file ',A25)
      ENDIF

      WRITE(40, 1600)
1600  FORMAT(/)
```



```
      RETURN
      END


      SUBROUTINE WRITEPROFILE
C-----------------------------------------------------------------------
C     Write the beta profile to a user defined file
C-----------------------------------------------------------------------
C
C     This program was written by personnel from
C     NOAA/ERL/Wave Propagation Laboratory/Atmospheric Lidar Group
C
C     Software Responsibility : A. Weickmann   (303) 497-6595
C     Scientific Responsibility : M.J. Post    (303) 497-6048
C
C     Modified :  10/02/90      by  AMW
C-----------------------------------------------------------------------
C                 INPUTS
C Variable  type,common          Description
C --------  ----------  ----------------------------------------
C     ALT( )        r,signl      Altitude array, km
C     BEGALT        r,calc       Beginning altitude of the noise window, km asl
C     BETACO( ) r,signl Array to store the BETA coefficients
C     BETASD( ) r,signl Array to store BETA std. dev. values
C     BW                  r,vars Bandwidth of signals in MHz
C     ENDALT        r,calc       Ending altitude for the noise window, km asl
C     IFN           i,vars       File number
C     IHEAD( )  i,header         Integer header from tape
C     IREC          i,vars       Start record number
C     NREC          i,vars       # of records to process
C     RLAM          r,vars       Wavelength,  meters
C     SNR( )        r,signl      Array to store the Signal to Noise Ratios
C
C                 OUTPUTS
C Variable  type,common          Description
C --------  ----------  ----------------------------------------
C           none
C-----------------------------------------------------------------------

      INCLUDE "CB.CALC"
      INCLUDE "CB.HEADER"
      INCLUDE "CB.SIGNL"
      INCLUDE "CB.VARS"

      INTECER*2 I, J, JPASS, LREC
      REAL*4 RLAMM, BWW
      character*20 textfile

10      print *
      print *,' Enter the text file name '
      print *,' ? : '
      read(*,'(A20)',err=10) textfile

      open(42,file=textfile,status='NEW',err=20)
      goto 30

20    print *
      print *,' File name must be new, choose again.'
      goto 10

30    continue

      write(42,1010) ifn,(ihead(i),i=1,6),irec,nrec
1010  format(i6,2x,i2,2('/',i2),2x,i2,2(':',i2),2x,2i6)
      RLAMM = RLAM * 1.0E6
      BWW = BW / 1.0E6
      WRITE(42,1040) RLAMM,BWW,BEGALT,ENDALT
1040  FORMAT(F6.3,3(3x,f4.1))

      DO 50 J = 1, ICOUNT
            JPASS = J
            IF (SNR(JPASS) .NE. 0) THEN
                  SNRDB = (10. * ALOG10(SNR(JPASS)))
            ELSE
                  SNRDB = 0.
```

```
            ENDIF
            WRITE (42,1070)  ALT(JPASS), BETACO(JPASS),
     +              SNRDB, BETASD(JPASS)
 50     CONTINUE
1070     FORMAT(F7.3, 7X, 1PE9.2, 2X, 0PF12.3, 2X, 1PE9.2)

        close(42)

        RETURN
        END
```

```
C************ COMPILE THESE ROUTINES  F77/INT=2/OPT **************
C          IC is a fortran package that calculates the system range
Cfunction and absorption attenuation correction factors at various
Cranges for correcting non dB signals.  There are three subroutines
Cin the package.
CApplication of correction factors:
C          Corrected Signal(R) = Signal(R) * ACRCTN(R) * SCRCTN(R)
CSubroutines:
C          1. CRINIT - This routine initializes data arrays and values.
C        This routine must be called before any of the others are
C        called.
C        Input variables:
C                ABSFILE - type character*14 - this variable contains
C                    the name of the absorption correction profile
C                    if ISSN = 0.  The format of file ABSFILE is
C                    free formatted records with each record con-
C                    taining range in meters as the first value and
C                    the corresponding absorption value in km-1 as
C                    the second value.  These files can be created
C                    with program ABSORP_VS_Z.
C                ZOLT - type real*4 - the height of the lidar in
C                    meters :
C                    a. MSL if  1 <= ISSN <= 5
C                    b. AGL if ISSN = 0 (height above absorption
C                          profile z=0)
C                RPT - type real*4 - a reference range generally
C                    equal to 1.5 kilometers.
C                ISSN - type integer*2 - the absorption season model
C                    number.
C                    1 = tropical
C                    2 = midlatitude summer
C                    3 = midlatitude winter
C                    4 = subarctic summer
C                    5 = subarctic winter
C                    0 = the absorption model used is obtained from
C                    a file contained in variable ABSFILE.
C                NWAVE - type integer*2 - wavelength number.
C                    0 = wavelength of 10.59 micrometers
C                    1 = wavelength of 9.25 micrometers
C        Output variables: none
C        Common blocks: ACORR, SCORR
C          2. SYSCRCTN - This routine calculates the system range
C        function correction values for various range values.
C        Input variables:
C                RANGE - type real*4 - 1 dimensional array containing
C                    range values in kilometers.
C                NRPTS - type integer*2 - the number of range values.
C        Output variables:
C                SCRCTN - type real*4 - 1 dimensional array contain-
C                    ing the system range function correction
C                    values.
C        Common blocks: SCORR
C          3. ABSCRCTN - This routine calculates the absorption atten-
C        uation correction factors for various range values.
C        Input variables:
C                RANGE, NRPTS - as described above under SYSCRCTN.
C                THETA - type real*4 - the elevation angle in degrees.
C        Output variables:
C                ACRCTN - type real*4 - 1 dimensional array contain-
C                    ing the absorption attenuation correction
C                    values.
C        Common blocks: ACORR
C-----------------------------------------------------------------
C SI,RI - deviation of system funct. from R**2 & corresponding range(km)
C NSPTS - number of system pts
C KJ,ZJ - volumetric atten. coeff. (/km) & corresponding altitude (m)
C NKPTS - number of attenuation pts
C ZOL - height of lidar above Zj = 0 (m)
C RP - reference range, usually of calibration target (km)
C NWAVE - wavelength number ie. 0 = 10.59 um, 1 = 9.25 um
C ABSFILE - name of file with attenuation values (ZJ KJ)
C THETA, THETAR - elevation angle in degrees and radians
C DELR - range gate length or difference between range values (km)
C SL - system function values corresponding to program ranges
C KL - attenuation values corresponding to program ranges (per km)
C RL, NRPTS - program ranges (km) & number of these pts
C ACRCTN - intensity correction factor for absorption attenuation
C SCRCTN - intensity correction factor for system function
```

```
      SUBROUTINE CRINIT(ABSFILE,ZOLT,RPT,ISSN,NWAVE)

      DIMENSION BEAM9(30), SI(30), RI(30), KJ(0:101), ZJ(0:101)
      DIMENSION XABSRP9(31,5), XABSRP10(31,5), BEAM10(30)
      REAL KJ
      CHARACTER*14 ABSFILE
      COMMON/ACORR/KJ, ZJ, NKPTS, ZOL
      COMMON/SCORR/SI, RI, NSPTS, RP
C     Apodization factors calculated at 9.25 microns accounting
C     for scanner astigmatism and X 10 telescope expansion ratio.
C     The values are computed for 1 to 30 km.
      DATA BEAM9/ .20508E+01, .11555E+01, .72203E+00, .51662E+00,
     & .40785E+00, .34276E+00, .30149E+00, .27383E+00, .25447E+00,
     & .24045E+00, .23000E+00, .22202E+00, .21580E+00, .21087E+00,
     & .20691E+00, .20369E+00, .20103E+00, .19881E+00, .19695E+00,
     & .19538E+00, .19404E+00, .19288E+00, .19189E+00, .19102E+00,
     & .19027E+00, .18960E+00, .18902E+00, .18851E+00, .18805E+00,
     & .18764E+00/
C     Apodization factors calculated at 10.6 microns accounting
C     for scanner astigmatism and X 10 telescope expansion ratio.
C     Used graded reflectivity output coupler.  Values are for
C     ranges 1 to 30 km
      DATA BEAM10 / .22863E+01, .12136E+01, .75318E+00,
     & .53808E+00, .42421E+00, .35618E+00, .31327E+00, .28442E+00,
     & .26426E+00, .24963E+00, .23874E+00, .23044E+00, .22403E+00,
     & .21894E+00, .21480E+00, .21144E+00, .20867E+00, .20634E+00,
     & .20446E+00, .20282E+00, .20137E+00, .20017E+00, .19917E+00,
     & .19823E+00, .19754E+00, .19676E+00, .19620E+00, .19572E+00,
     & .19522E+00, .19479E+00/

C Absorption profile values are from 0 km to 30 km units of km**-1
C for wavelength of 10.59
      DATA XABSRP10/     !TROPICAL :
     & .45342, .27221, .18187, .09768, .06382,
     & .05077, .04089, .03253, .02588, .02124, .01664, .01290,
     & .01026, .00774, .00568, .00433, .00307, .00276, .00338,
     & .00412, .00497, .00596, .00709, .00769, .00833, .00902,
     & .01355, .01892, .02279, .02010, .01026,
C     MIDLATITUDE SUMMER :
     & .30402, .18345, .11644, .07643, .05706, .04473, .03679,
     & .03049, .02430, .01989, .01554, .01243, .00945, .00739,
     & .00739, .00742, .00742, .00741, .00737, .00767, .00799,
     & .00831, .00866, .00939, .00979, .01016, .00993, .01501,
     & .02343, .03396, .04050,
C     MIDLATITUDE WINTER :
     & .07326, .06014, .04967, .04139, .03266, .02641, .02140,
     & .01720, .01380, .01099, .01077, .00847, .00830, .00812,
     & .00795, .00778, .00762, .00746, .00731, .00715, .00714,
     & .00715, .00715, .00715, .00713, .00716, .00782, .01174,
     & .02011, .03212, .03922,
C     SUBARCTIC SUMMER :
     & .18186, .11825, .08542, .06263, .04862, .03800, .02957,
     & .02319, .01775, .01395, .01068, .01066, .01065, .01064,
     & .01064, .01069, .01064, .01064, .01064, .01064, .01065,
     & .01063, .01063, .01063, .01102, .01190, .01030, .01614,
     & .02667, .03839, .04164,
C     SUBARCTIC WINTER :
     & .03914, .04008, .03512, .03060, .02522, .01964, .01530,
     & .01184, .00900, .00781, .00780, .00779, .00779, .00778,
     & .00778, .00778, .00758, .00740, .00721, .00702, .00681,
     & .00667, .00650, .00633, .00616, .00599, .00735, .00945,
     & .01508, .02270, .03280/
C Absorption profile values are from 0 km to 30 km units of km**-1
C for wavelength of 9.25
      DATA XABSRP9/      !TROPICAL :
     & .39113, .25256, .18051, .11453, .08300,
     & .06754, .05585, .04507, .03617, .02983, .02344, .01822,
     & .01448, .01091, .00799, .00605, .00426, .00384, .00434,
     & .00493, .00532, .00549, .00541, .00491, .00430, .00365,
     & .00133, .00040, .00013, .00004, .00001,
C     MIDLATITUDE SUMMER :
     & .27635, .18407, .12960, .09446, .07492, .06109, .05107,
     & .04281, .03445, .02844, .02241, .01823, .01410, .01141,
     & .01138, .01109, .01074, .01033, .00982, .00946, .00880,
     & .00800, .00703, .00615, .00512, .00421, .00112, .00036,
     & .00012, .00004, .00001,
C     MIDLATITUDE WINTER :
     & .08740, .07461, .06404, .05533, .04503, .03712, .03052,
```

```
     &  .02497, .02036, .01667, .01668, .01374, .01372, .01353,
     &  .01307, .01250, .01182, .01108, .01014, .00912, .00815,
     &  .00692, .00586, .00472, .00377, .00297, .00078, .00022,
     &  .00007, .00002, .00001,
C       SUBARCTIC SUMMER :
     &  .18022, .12873, .09870, .07770, .06355, .05153, .04117,
     &  .03284, .02575, .02063, .01622, .01646, .01644, .01652,
     &  .01621, .01597, .01531, .01475, .01376, .01247, .01096,
     &  .00939, .00782, .00647, .00539, .00455, .00108, .00039,
     &  .00014, .03839, .04164,
C       SUBARCTIC WINTER :
     &  .05123, .05297, .04718, .04189, .03523, .02796, .02208,
     &  .01748, .01371, .01265, .01325, .01379, .01453, .01450,
     &  .01416, .01411, .01355, .01238, .01110, .00965, .00808,
     &  .00661, .00532, .00420, .00316, .00240, .00061, .00017,
     &  .00005, .00001, .00000/

      RP = RPT
      ZOL = ZOLT

      NSPTS=30
      DO 40 I=1,30
        RI(I) = I
        IF (NWAVE .EQ. 0) SI(I) = BEAM10(I)
        IF (NWAVE .EQ. 1) SI(I) = BEAM9(I)
40          CONTINUE

      IF (ISSN .NE. 0) THEN
        DO 50 I=1,31
            IF (NWAVE .EQ. 0) KJ(I) = XABSRP10(I,ISSN)
            IF (NWAVE .EQ. 1) KJ(I) = XABSRP9(I,ISSN)
            ZJ(I) = (I - 1) * 1000.                                  !meters
50    CONTINUE
        NKPTS = 31
      ELSE IF (ISSN .EQ. 0) THEN
        OPEN(20,FILE=ABSFILE)
        NKPTS = 0
        DO 60 I=1,100
            READ(20,*,END= 100) ZJ(I), KJ(I)
            NKPTS = NKPTS + 1
60    CONTINUE
100   CLOSE(20)
      END IF

      RETURN
      END
C*****************************************************************
      SUBROUTINE SYSCRCTN(RL,NRPTS,SCRCTN)

      DIMENSION SI(30), RI(30), RL(NRPTS), SL(2064), SCRCTN(NRPTS)
      COMMON/SCORR/SI, RI, NSPTS, RP

C  Find correction for system function

      DO 50 L=1,NRPTS
C  find first RI greater than RL
        DO 20 I=1,NSPTS
            II= I - 1
            IF (RL(L) .LT. RI(I)) GO TO 30
20    CONTINUE

        SL(L) = SI(NSPTS)                          !RL(L) .gt. RI(nspts)
        GO TO 40

30    IF (II .EQ. 0) THEN                  !RL(L) .lt. RI(1)
            SL(L) = SI(1)
        ELSE
            SL(L) = ((RL(L)-RI(II))/(RI(II+1)-RI(II)))*(SI(II+1)-SI(II))
     &                + SI(II)
        END IF
40    SCRCTN(L) = SL(L) * (RL(L)/RP)**2
50          CONTINUE

      RETURN
      END
C*****************************************************************
      SUBROUTINE ABSCRCTN(RL,NRPTS,THETA,ACRCTN)

      DIMENSION KJ(0:101), ZJ(0:101), RL(NRPTS)
      DIMENSION KL(2064), ZL(0:2065), ACRCTN(NRPTS),DELR(2064)
      REAL KJ, KL
      COMMON/ACORR/KJ, ZJ, NKPTS, ZOL

C  set do loop variables

      IF (THETA.GE.0.) THEN
        LB = 1
        LE = NRPTS
        LD = +1
      ELSE
        LB = NRPTS
        LE = 1
        LD = -1
      END IF

C find difference between range gate values
      DELR(1) = RL(1) - 0.
      DO 5 L = 2,NRPTS
        DELR(L) = RL(L) - RL(L-1)
5     CONTINUE

C  Find absorption correction

      THETAR = THETA * 3.14159 / 180.
      ZL(0) = ZOL                                  !zero range
      ZL(NRPTS+1) = ((RL(NRPTS)+DELR(NRPTS))*SIN(THETAR)) * 1000. + ZOL
      KJ(0) = KJ(1)
      ZJ(0) = -10000.
      KJ(NKPTS+1) = 0.0
      ZJ(NKPTS+1) = ZJ(NKPTS) + 10000.          ! + 10 KM

      DO 100 L=LB,LE,LD
        LW1 = L-1
        IF (THETA.LT.0.) LW1 = L+1
        ZL(L) = (RL(L) * SIN(THETAR)) * 1000. + ZOL
        DO 60 J=1,NKPTS
            IF (ZL(LW1) .LE. ZJ(J)) THEN
                JM = J
                GO TO 70
            END IF
60    CONTINUE

        KL(L) = 0.0                              ! no ZL(LW1) .LE. ZJ(J)
        GO TO 100

70    DO 80 JPMP1=JM,NKPTS
          IF (ZL(L) .LT. ZJ(JPMP1)) THEN
              IF (JPMP1 .EQ. JM) THEN
          IF (JPMP1 .EQ. 1) THEN
            KL(L) = KJ(1)
            GO TO 100
          ELSE                    !both between same two pts
            J1 = JPMP1 -1
            KL(L) = ((ZL(LW1)+ZL(L))/2. - ZJ(J1)) *
     &                ((KJ(J1+1)-KJ(J1))/(ZJ(J1+1)-ZJ(J1))) + KJ(J1)
            GO TO 100
          END IF
              ELSE IF (JPMP1 .GT. JM) THEN
          J1 = JM
          M = JPMP1 - J1 - 1
          IF (THETA .EQ. 0) THEN
            KL(L) = ((ZL(LW1)+ZL(L))/2. - ZJ(J1)) *
     &                ((KJ(J1+1)-KJ(J1))/(ZJ(J1+1)-ZJ(J1))) + KJ(J1)
            GO TO 100
          END IF
          GO TO 90
              END IF
          END IF
80    CONTINUE

        IF (JM .EQ. NKPTS) THEN  ! no ZL(L) .lt. ZJ(J)
            KL(L) = 0.0
            GO TO 100
        ELSE IF (JM .NE. NKPTS) THEN
            J1 = JM
            M = NKPTS - J1
```

```
            GO TO 90
          END IF

90      KL(L) = (((ZJ(J1)-ZL(LW1))*(KJ(J1)-KJ(J1-1))
          &          /(ZJ(J1)-ZJ(J1-1))+KJ(J1-1)) * (ZJ(J1)-ZL(LW1)))
          &          + (((ZL(L)-ZJ(J1+M))*(KJ(J1+1+M)-KJ(J1+M))
          &          /(ZJ(J1+1+M)-ZJ(J1+M))+KJ(J1+M)) * (ZL(L)-ZJ(J1+M)))
          SUMKL = 0.0
          DO 92 N=0,M-1
               SUMKL = SUMKL + (((KJ(J1+1+N)+KJ(J1+N))/2.)
          &                 *(ZJ(J1+1+N)-ZJ(J1+N)))
92      CONTINUE
          KL(L) = (KL(L) + SUMKL)/(ZL(L)-ZL(LW1))
100       CONTINUE

        TAO = 0.0
        DO 110 L = 1,NRPTS
          DTAO = KL(L) * DELR(L)
          TAO = TAO + DTAO
          ACRCTN(L) = EXP(2.*TAO)
110       CONTINUE

        RETURN
        END


        SUBROUTINE TAPOS(IPACK,IFILE,IREC,ICURF,ICURR,IHEAD)
C-----------------------------------------------------------------------
C  COMPILE THIS F77/INT=2
C  ALL INTEGERS ARE INTEGER*2 UNLESS DECLARED OTHERWISE
C  This version was completed by K. R. Healy on Jan. 30 1990.
C  This routine positions the tape at the requested file, reads
C  the tape header, then positions tape at requested record.
C  When using this in a program, initialize ICURR to 1 and ICURF
C  to 0 at the very start of the program before calling TAPOS.
C  Be sure to update ICURR when reading records outside this routine.
C-----------------------------------------------------------------------
        INTEGER*2 IHEAD(200)
        INTEGER*4 IPACK(6),ICMD,NUMBR,IER

C If at beginning of tape, skip file to file requested

5       IF (ICURF .NE. IFILE) THEN
          ICMD=IFILE-ICURF
          IF (ICMD.LE.0) ICMD=ICMD-1
          CALL MTSKPF(IPACK,ICMD,NUMBR,IER)

          IF (IER .EQ. 6) THEN
               CALL MTREW(IPACK)
               IER = 0
          END IF

          IF (IER.NE.0) THEN
               WRITE(6,*) 'TAPOS: tape error = ',IER
               STOP
          END IF

          ICURR=1

          IF (IFILE .EQ. 0)  GO TO 10
C This is only true on AOS/VS so i removed the code but kept the comment.
C When backspacing, tape is positioned just before file mark.
C Must skip file over that file mark to file header.
        END IF

C Read 200 word tape header

10            IF (ICURR .EQ. 1) THEN
          ICMD=200*2
          CALL MTREAD(IPACK,IHEAD,ICMD,NUMBR,IER)
          IF (IER.NE.0) THEN
               WRITE(6,*) 'TAPOS: tape error = ',IER
               STOP
          END IF
        END IF

        ICURF = IHEAD(7)
        IF (ICURF .NE. IFILE) GO TO  5

C Skip records to requested record

        IF (IREC.NE.ICURR) THEN
          ICMD=IREC -ICURR
          CALL MTSPCE(IPACK,ICMD,NUMBR,IER)
          IF (IER.NE.0) THEN
               WRITE(6,*) 'TAPOS: tape error = ',IER
               STOP
          END IF
          ICURR=IREC
          WRITE(6,*)' ICURR = ',ICURR
        ENDIF

        RETURN
        END
```

Appendix B

Common Block Listings

Table of Contents

CB.CALC B-1
CB.CHARAC B-1
CB.HEADER B-1
CB.MULTFL B-1
CB.PANEL B-1
CB.PLOT B-1
CB.SETUP B-2
CB.SIGNL B-2
CB.VARS B-2

```
      COMMON /CALC/
     * ARCDAT,          !Option to archive the beta profile data Y or N
     * ATTEN,           !Attenuation factor, if present
     * AVNOIS,          !Average noise in noise window
     * AVNOSA,          !Attenuated average noise in noise window
     * BEGALT,          !Beginning altitude of the noise window, km asl
     * BETAFL,          !Flag to plot the BETA profile Y or N
     * CAL,             !Calib factor for adjusting beta retrieval
     * CALDAT,          !Calibration date MMDDYY
     * DALT,            !Requested altitude increment for averaging
     *                  !processed data (as opposed to ZINC)
     * DBCAL,           !Continuous wave calibration SNR, dB
     * ENDALT,          !Ending altitude for the noise window, km asl
     * ICOUNT,          !Number of altitude increments in the profile
     * NNOIS,           !Number of noise samples in the noise window
     * NQC,             !Quality control for calibration and beta
     *                  !     1 = good      0 = bad
     * PRF,             !Pulse Repetition Frequency in Hz
     * P1,              !P1, P2, and P3 are used in the BETA
     * P2,              !   equation, please see BETA.F77 for
     * P3,              !   more details
     * PRNDAT,          !Option to print out the raw data, Y or N
     * PRNPRF,          !Option to print out the BETA profile Y or N
     * PWRCAL,          !Continuous Wave Power at time of calib, Watts
     * RANCAL,          !Range for  c.w. calibration, meters
     * SDNOIS,          !Standard deviation of noise in noise window
     * SHTCAL,          !Shot noise, (old lidar calibrations)
     * SHTDAT,          !Shot noise for the data run, (new lidar)
     * SHTREF,          !Shot noise for the calibration run, (new)
     * ZINC             !Actual altitude increment for averaging
     *                  ! processed data (as opposed to DALT)
C ------------------------------------------------------------------
      INTEGER * 2  ARCDAT, BETAFL, ICOUNT, PRNDAT, PRNPRF, NNOIS
      INTEGER * 4  CALDAT, NQC
      REAL * 4 AVNOIS, BEGALT, CAL, DALT, DBCAL, ENDALT, PRF
      REAL * 4 P1, P2, P3, PWRCAL, RANCAL, SDNOIS, SHTCAL, SHTDAT
      REAL * 4 SHTREF, ZINC, ATTEN, AVNOSA
C ------------------------------------------------------------------
```

```
      COMMON/CHARAC/
     * ARCNAM,          !Archive disk filename
     * RSDATA,          !Name of radiosonde data file
     * SITNAM           !Name of lidar site
C------------------------------------------------------------------
      character * 25 RSDATA
      CHARACTER * 25 ARCNAM
      CHARACTER * 30 SITNAM
C------------------------------------------------------------------
```

```
      COMMON/HEADER/
     * DELAY,           !Corrected delay to 1st sample (data point) us
     * DIGTIM,          !Time between digitizer samples
     * DIGALT,          !Altitude increment per digitizer sample
     * DL1SAM,          !Delay to the first sample, us (microseconds)
     * DGTZRT,          !Digitizer rate, MHz
     * IHEAD(200),      !Integer header from tape
     * ISTART,          !Starting sample index for data within a record
     * ISTOP,           !Stopping sample index for data within a record
     * NPTS,            !Number of points (or samples) to be processed
     * NPTSPP,          !Number of points (or samples) per pulse
     * NRECDG,          !Number of records digitized per file in raw
     *                  !   data mode
     * R1,              !Begin range, adjusted for lidar altitude and
     *                  !   elevation angle
     * R2,              !End range, adjusted for lidar altitude and
     *                  !   elevation angle
     * TRGMOD           !Trigger mode
C------------------------------------------------------------------
      INTEGER * 2 IHEAD, NPTS
      INTEGER * 4 ISTART, ISTOP
      REAL * 4  NPTSPP, NRECDG, DELAY, DIGTIM, DIGALT, DL1SAM
      REAL * 4  DGTZRT, R1, R2, TRGMOD
      REAL*4 PARM(50)

      EQUIVALENCE (IHEAD(51), PARM(1))
C------------------------------------------------------------------
```

```
      COMMON /MULTFL/
     * MFAVPW (2048),   !Power avgd over multfiles
     * MFDATE (10,3),   !Date for each file
     * MFDGTZ,          !Digitizing rate for multifiles
     * MFDL1S,          !Delay to 1st sample for multifiles
     * MFERRR,          !True if some file won't work in multifile
                        !   processing.
     * MFIFN (10),      !Multi file numbers.
     * MFIREC (10),     !Multi file beg rec number.
     * MFNPTS,          !Value of npts for multifiles
     * MFNREC (10),     !Multi file number of recs.
     * MFTIME (10,3),   !Time for each file
     * MULTFI,          !TRUE if multifiles averaging in effect
     * MF1PAS,          !True if this is 1st file in multifile process
     * MFNMFI,          !Number of files averaged.
     * MFNMPL           !Number of pulses in MFAVGPWR array
C------------------------------------------------------------------
      LOGICAL   MULTFI, MF1PAS, MFERRR
      INTEGER*2 MFNMFI, MFIFN, MFIREC, MFNREC, MFNMPL, MFNPTS
      INTEGER*2 MFDATE, MFTIME
      REAL*4    MFAVPW, MFDL1S, MFDGTZ
C------------------------------------------------------------------
```

```
      COMMON/PANEL/
     * GATEIN,          !Range increment for processor data, km
     * IATTEN,          !Attenuation shown on processor panel, dB
     * IPULSE,          !Number of pulses averaged in processor
     * ILAG,            !Lag as it appears on processor panel
     * IWIDTH,          !Range gate width as it appears on panel, us
     * RDELAY,          !Delay as it appears on panel (us = microsec)
     * TINC             !Time increment between pulse (s)
C------------------------------------------------------------------
      INTEGER*2 IATTEN, IPULSE, ILAG, IWIDTH
      REAL * 4 GATEIN, RDELAY, TINC
C------------------------------------------------------------------
```

```
      COMMON /PLOT/
     * BFLAG,           !Flag for labeling the graphs
     * LINDB,           !Linear or decibel scaling
     * LOHI,            !Low to high scaling flag
     * PLTTYP,          !Flag for single or average plots
     * XMAX,            !Maximum x defining the graphic data window
```

```
     *  XMIN,              !Minimum x defining the graphic data window
     *  YMAX,              !Maximum y defining the graphic data window
     *  YMIN               !Minimum y defining the graphic data window

C ------------------------------------------------------------
          INTEGER * 2  LINDB, LOHI, PLTTYP
          INTEGER * 4  BFLAG
          REAL * 4 XMAX, XMIN, YMAX, YMIN
C ------------------------------------------------------------
```

```
          COMMON/SETUP/
     *  ICURF,             !Current file number
     *  ICURR,             !Current record number
     *  IPACK(6),          !MTAPE array     * DO NOT CHANGE *
     *  REINIT             !Re-initialization flag

C-----------------------------------------------------------------
          INTEGER*4 IPACK
          integer*2 icurf, icurr
          LOGICAL REINIT
C-----------------------------------------------------------------
```

```
          COMMON/SIGNL/
C  8/31/90 RPM - changed array sizes of 350 to 2048 for running
C  average case.
     *  ABSRP (30),        !Absorption array, in units of km-1
     *  ALT (2048),        !Altitude array, km
     *  AVGPWR (2048),     !Average power array
     *  AVSGNL (2048),     !Average power at each altitude
     *  BETACO (2048),     !Array to store the BETA coefficients
     *  BETAMN (2048),     !Array to store minimum BETA values
     *  BETASD (2048),     !Array to store BETA std. dev. values
     *  IDAT (2064),       !Array to store data from tape
     *  PLBTSD(2048),      !Array for plotting beta std. dev.
     *  PWR (5000),        !Signal power array
     *  QFAC (2048),       !Q-factor array
     *  QUAL,              !The quality factor threshold value.
     *  SNR (2048),        !Array to store the Signal to Noise Ratios
     *  SNRSDD (2048)      !Array to store SNR std.dev. in DB

C-----------------------------------------------------------------
          REAL * 4 ABSRP, ALT,  AVGPWR, BETACO, BETAMN, BETASD
          REAL * 4 PLBTSD, PWR, QFAC, QUAL, SNR, SNRSDD, AVSGNL
          INTEGER*2  IDAT
C-----------------------------------------------------------------
```

```
          COMMON/VARS/
C AVEENG added by TG in August 1990
C BTYPE added by RPM in September 1990
     *  ALTLID,            !Lidar altitude in km ASL
     *  AZIMTH,            !Azimuth angle of scanner, deg w.r.t. north
     *  AVEENG,            !A/D Average energy per pulse
     *  BTYPE,             !Beta calc type: A=archive, B=block, R=running avg.
     *  BW,                !Bandwidth of signals in MHz
     *  CLRAIR,            !Altitude of the last clear air (km)
     *  COSRAD,            !Cosine of zenith angle of scanner (radians)
     *  EJ,                !Energy/pulse (J)
     *  ELEVA,             !Elevation angle of the scanner, deg above horiz
     *  IFN,               !File number
     *  IPLTA,             !Plot to screen =0 or plotter =1 or both =2
     *  IPLTN,             !Consecutive plot number
     *  IREC,              !Start record number
     *  IREND,             !End record number
     *  ISITE,             !Site number from lidarsite file
     *  MODEL,             !Atmospheric absorption model
     *  NREC,              !# of records to process
     *  OP,                !Operator's initials
     *  RANGE1,            !Initial range, km
     *  RANGE2,            !End range, km
     *  RLAM,              !Wavelength,  meters
     *  SATCOR,            !Digitizer saturation correction flag
     *  SONDHT             !Altitude, m, asl of highest radiosonde data

C-----------------------------------------------------------------
          INTEGER * 2 IFN, IREC, IREND,  MODEL, NREC, OP, IPLTA, BTYPE
          INTEGER * 4 ISITE, IPLTN, SATCOR
          REAL * 4 AZIMTH, COSRAD, BW, ALTLID, EJ, ELEVA, RANGE1
          REAL * 4 AVEENG, RANGE2, RLAM, CLRAIR, SONDHT
C-----------------------------------------------------------------
```

Appendix C

Recent Software Changes

Hardcopies of the software found in Appendix A do not reflect some recent changes in the software. The modifications to the affected subroutines and common blocks are described.

calbeta.f	eliminated the use of BETAMN( ), BETASD( ), and SNRSDD( ) and are now using BNOIS( ) and BRATIO( ). BNOIS replaces BETASD and BRATIO replaces SNRSDD. The computations were made more obvious and the variables were named more appropriately. BETAMN( ) was eliminated because it was the same as BETASD( ).
initco.f	removed SONDHT check against and override fill of ABSRP( ) with 1.0. Sondeabsp.f correctly handles values above the maximum radiosonde height by setting them to 0.0 and integrating.
newcalib.f	changed message to specify radiosonde file input as pressure, height, temperature, and dewpoint. Added existance check to radiosonde filename and limited filename to 14 characters in length.
pltprofile.f	use BNOIS instead of BETASD.
printprofile.f	changed column headers and column information to reflect changes noted in calbeta.f
sondeabsp.f	reads data file containing pressure, height, temperature, and dewpoint. Changed equation to compute volumetric mixing ratio; now uses dewpoint in a modified Clausius-Clapeyron vapor pressure equation. Changed 1000 word arrays to nlev=100.
txgrfr.f	added program name to lower left corner of plot.
txgrft.f	added program name to lower left corner of plot.
writeprofile.f	use BNOIS instead of BETASD.
CB.CALC	remove SHTCAL, no longer used.
CB.CHARAC	radiosonde datafile, RSDATA, now character*14.
CB.SIGNL	eliminate BETASD( ), BETAMN( ), and SNRSDD( ); add BNOIS( ) and BRATIO( ).

☆U.S. GOVERNMENT PRINTING OFFICE: 1991_573_00721066