81/30

EISCAT
TECHNICAL
NOTE

EISCAT UHF antenna direction calibration

by
Markku Lehtinen
Anna-Liisa Turunen

KIRUNA
Sweden

EISCAT UHF antenna direction calibration

by

Markku Lehtinen
Anna-Liisa Turunen

EISCAT Technical Note 81/30
Printed in Sweden
EISCAT Scientific Association
Kiruna, August 1981
ISSN 0349-2710

EISCAT UHF antenna direction calibration

1. Introduction

In the EISCAT radar system it is necessary to make the beams intersect at an arbitrary point in space, and there are programs to calculate the geometry to do this. However, there are various sources of error in the antenna pointing so that the antenna does not point exactly in the direction commanded which can be read from the antenna control unit (ACU). These errors we call azimuth or elevation offsets, depending on axis. The sign of the offset is defined so that we add the offset figure to the real direction we wish and use the result of that addition when giving the command to the antenna control unit. The following formulas may be used as a definition of the offsets D(a,e) :

$$\begin{aligned} a_{ACU} &= a + D_a(a,e) \\ e_{ACU} &= e + D_e(a,e), \end{aligned} \qquad (1.1)$$

where azimuth is represented by a and elevation by e.

The half power beamwidth of the UHF antennas is about 0.6 degrees, and so a knowledge of the offsets to an accuracy of a few hundreths of a degree would be sufficient.

The offsets were determined by measuring the positions of radio stars in the sky and comparing the measured results with computed ones. However, the radio stars follow fixed orbits in the local sky and it is impossible to get offset measurements at an arbitrary point. That is why the final offset figures are obtained from a model based on a consideration of the possible error sources and fitted to the measured data to describe the measurements as well as possible.

To get the best possible fit, the measured data should be evenly distributed in the local sky, and so the measurements should be made for many sources and for at least one complete 24-hour period. The measurements are made by an RT program ANTCAL, which has the coordinates of six stars preprogrammed and the user may choose the stars by name. This program does the measurements periodically until stopped and it goes through the sequence of stars selected so that it may be left to do the measurements overnight by itself.

The fitting of the models is made by a backround program ANTCAL-CONTROL, which analyzes the data collected by ANTCAL and makes different graphical maps of that data on the line printer or at the terminal. ANTCAL-CONTROL also controls the running of the data collecting program. These programs are described in more detail in the following chapters. In appendix B there are some flow charts describing the data flow in the calibration system and the functioning of the ANTCAL-program.

2. Discussion of the error sources

There are many possible sources of error in the structure of the antenna. The most important of them are:

- simple offsets in the angle encoders
- inclination of the basement
- tilted elevation axis
- non-perpendicularity of the beam with elevation axis
- errors in the joints between the angle encoders and antenna axes
- deformation of the antenna as a function of elevation
- refraction in the atmosphere
- unevenness of the azimuth rail

These effects may be represented by a linear combination of a set of functions each characteristic of one of these errors :

$$D_a = \sum_i k_i^a f_i^a(a,e) \qquad D_e = \sum_i k_i^e f_i^e(a,e) \qquad (2.1)$$

The different cases are studied separately:

2.1. Inclination of the basement

If the basement makes an angle A with the horizontal plane so that the azimuth of the axis perpendicular to the basement is B, it is easy to see that to a first approximation this results in offsets both in elevation and in azimuth represented by

$$D_e(a,e) = A\cos(a-B) = m_1\cos(a) + m_2\sin(a) \qquad (2.2)$$

$$D_a(a,e) = \tan(e)(m_2\cos(a) - m_1\sin(a))$$

where

$$m_1 = A\cos(B), \qquad m_2 = A\sin(B)$$

2.2. Inclination of the elevation axis

If the elevation axis is not parallel to the basement, the additional error is found to be

$$D_a(a,e) = n\tan(e) \qquad D_e(a,e) = 0\ , \tag{2.3}$$

where n is the angle the elevation axis makes with the basement. n is counted positive, when the clockwise end of the axis (the end far from the ladder) is higher than the other end.

2.3. Non-perpendicularity of the beam with elevation axis

This might result for example from a misalignment of the small mirror. The effect in the direction of elevation is a constant offset and is accounted in that. So, the terms resulting from this are

$$D_a(a,e) = p/\cos(e) \qquad D_e(a,e) = 0\ , \tag{2.4}$$

where p is the angle the beam makes with the perpendicular of the elevation axis (counted positive, when the beam deviates counterclockwise from the perpendicular).

2.4. Errors in the mechanical joints connecting the angle encoders to the antenna

There are several joints between different axes joining the antenna and the angle encoders. All the joints resemble the cardan joint found for example in a car connecting the cardan shaft to the gear box in the front and to the planet gear in the rear. The cardan joint may be described as two forked shafts that are joined together through an x-shaped part, one bar of which is attached with bearings to the fork of one axis and the other bar to the fork of the other axis. In our case there is an elastic washer instead of that x-shaped part, but that washer allows movements in exactly the same directions as a rigid cross would. We study the case, when the joint is not perfect, say that the two axes make an angle C with each other, and in addition to that the forks are not perpendicular to the axes, where the errors are denoted by D and E. In this case the output axis does not turn round evenly, even if the input axis is rotated at constant speed. If we denote the angle traversed by the input axis by "in" and the angle traversed by the output axis by "out", it can be proven that in case of small errors there is the following relation between these angles:

$$out = in + (C^2/4)\sin(2in) + C\,(D\sin(in) + E\cos(in)) \tag{2.5}$$

The angles are calculated with respect to the plane defined by the two non-parallel axes.

It was found that the encoders were rather carelessly installed with a several-millimeter mismatch in the elevation encoder and an erroneous joint in the azimuth encoder. The joint connecting the azimuth encoder to the rest of the system was found to have installing errors of the order of 2 degrees in the parameters C, D or E. The general contribution to the offsets may thus be represented as

$$D_a(a,e) = q_1\cos(a)+q_2\sin(a)+q_3\cos(2a)+q_4\sin(2a)$$

(2.6)

$$D_e(a,e) = r_1\cos(e)+r_2\sin(e)+r_3\cos(2e)+r_4\sin(2e)$$

In practice it was found to be impossible to fit all the terms for elevation. There is elevation data only from an interval of 5...80 degrees, and these functions are too near to each other in the sense of linear dependence in this angle range.

2.5. Deformation of the antenna as a function of elevation.

As the moment of the gravitation forces deflecting the antenna is proportional to cos(e), the effect can be described by

$$D_e(a,e) = s\cos(e) \qquad D_a(a,e) = 0 . \tag{2.7}$$

As the deformations of the antenna might be more complicated than a mere deflection of a rigid paraboloid, the following expression might also be used

$$D_e(a,e) = t_1\cos(e) + t_2\sin(e) \qquad D_a(a,e)=0. \tag{2.8}$$

2.6. Refraction in the atmosphere

In a simple model good enough, when e>5 deg, the refraction may be accounted by a formula derived from Lenz's law by supposing that the refractive index is nearly equal to 1 :

$$D_e(a,e) = u\cot(e) \qquad D_a(a,e)=0., \tag{2.9}$$

where 1+u is the refractive index of air.

Here we must suppose that the atmosphere and ionosphere consist of horizontal layers so that the refraction angle only depends on the refractive indices above and below, and does not depend on what is in between. Thus, this formula is applicable only for star measurements, since in the ionosphere the beam could follow rather strange paths due to electron density variations. This may need some further study and a model based on an electron density profile might be applied.

2.7. Unevenness of the azimuth rail

The azimuth rail might be somewhat warped causing some errors in elevation. The rail consists of 8 parts and thus the following model might be tried

$$D_e(a,e) = v_1 \cos(Ba) + v_2 \sin(Ba) \qquad (2.10)$$

$$D_a(a,e) = 0 \ .$$

Of course, a complete harmonic expansion might be tried here, but at least in Sodankyla the coefficients turned out to be zero.

2.8. Hysteresis and other effects

By hysteresis we mean a difference between measurements with sweeps made in opposite directions. Some hysteresis may result from the low pass filter in the data sampling. This is not serious, and the effect will be cancelled out when making the analysis with scans in both directions.

More serious is a hysteresis that results from mechanical tolerances or lack of rigidity in the antenna structure. If the antenna is not rigid, it cannot quite follow the movements its axes make, and the offsets measured for the positive - direction sweeps are systematically greater than those measured for negative - direction sweeps. If there is allowance in the joints between the antenna and the angle encoders, the positive direction offsets tend to be systematically smaller than the negative direction offsets.

In Tromso it was found that the waveguide mechanical system and allowances there caused the antenna to oscillate at an amplitude of about 0.1 degrees in azimuth. This oscillation could not be seen in the software, since the ACU readings remained constant.

No effort is made to include these kinds of errors in the offset model, but rather they should be corrected before any calibration is made; otherwise someone might be hit by loose bolts scattering incoherently from the antenna. Hysteresis can be calculated by the analysis program, but it is considered too unreliable to include it in the final offset model.

In addition to this there are some voltage offsets in the analogue part of the servo system so that it does not turn the antenna exactly to the direction commanded. This can be seen as a constant difference between the instruction given to the ACU and the position read from the ACU after completing that instruction. This amounts to a few hundredths of a degree, and there seems to be no means to adjust this offset in the ACU itself. In the pointing subroutines the discrepancy is corrected by giving all commands to the ACU with a proper offset.

2.9. Summary of the terms in the offset model

As a summary, the following models are suggested to be used to describe the antenna offsets. These models are also used as default in the antenna offset analysis program.

$$
\begin{aligned}
D_a(a,e) = {} & k_1^A + k_2^a/\cos(e) + k_3^a \tan(e) + \\
& k_4^a \tan(e)\cos(a) + k_5^a \tan(e)\sin(a) + \\
& k_6^a \cos(a) + k_7^a \sin(a) + k_8^a \cos(2a) + k_9^a \sin(2a)
\end{aligned}
\qquad (2.11)
$$

$$
\begin{aligned}
D_e(a,e) = {} & k_1^e + k_2^e \cos(a) + k_3^e \sin(a) + \\
& k_4^e \cos(e) + k_5^e \sin(e) + \\
& k_6^e \cot(e) .
\end{aligned}
$$

3. The analysis of measured data

3.1. Method of fitting a model to offset data

If we have a set of measured offset values and the corresponding azimuth and elevation values

$$D_i \, , \quad (a_i , e_i) \, , \quad i=1,\ldots,m \tag{3.1}$$

a model

$$D(a,e) = \sum_{i}^{n} k_i f_i (a,e) \tag{3.2}$$

has to be fitted to them in some way. Here the f functions are some functions that describe different elementary errors of the antenna corresponding for example to different mechanical misalignments in the structure. The fit is made by requiring that the mean square deviation between the model and the measured data attains a minimum:

$$V = \sum_{j=1}^{m} \left(\sum_{i=1}^{n} k_i f_i (a_j , e_j) - D_j \right)^2 = \min \tag{3.3}$$

The minimum is found by requiring that the partial derivatives with respect to the unknown coefficients ki are zero:

$$(d/dk_h) \; V(k_1 , k_2 , \ldots , k_n) = 0 \, , \quad h=1,\ldots,n \tag{3.4}$$

This is found to be equivalent to

$$\sum_{i=1}^{n} \left(\sum_{j=1}^{m} f_h (a_j , e_j) f_i (a_j , e_j) \right) k_i = \sum_{j=1}^{m} f_k (a_j , e_j) D_j \tag{3.5}$$

If we introduce the matrix A and vector $\underline{b}$ by

$$A_{hi} = \sum_{j=1}^{m} f_h(a_j, e_j)\, f_i(a_j, e_j) \quad \text{and}$$

$$b_h = \sum_{j=1}^{m} f_h(a_j, e_j)\, D_j \,, \tag{3.6}$$

The equation (3.5) may be represented in matrix notation as

$$\underline{\underline{A}}\,\underline{k} = \underline{b} \quad \Leftrightarrow \quad \underline{k} = \underline{\underline{A}}^{-1}\underline{b} \,. \tag{3.7}$$

So, the solution is found in terms of the inverse matrix to be

$$k_i = \sum_{h=1}^{n} A^{-1}_{ih}\, b_h \tag{3.8}$$

3.2. Errors of the fitted model

Next we are going to study the effect of errors of the measured offsets on the coefficients ki. We suppose that the model we are fitting is exactly accurate and thus capable of describing the offsets completely, but that the measured values Dj contain some gaussian noise. If we denote the values computed from the model with a superscript 1 and the exact values with a superscript o, we can approximate the noise for each point measured by

$$d_j = D_j^{o} - D_j^{1} = D_j - D_j^{1} \,, \tag{3.9}$$

where we have approximated the exact value for the offset by the model fitted. This causes an error in the error estimates equivalent to discarding the term SQRT(N/(N-1)) that usually appears in these kinds of estimates of simpler type. Since an exact calculation would here present considerable difficulties, we are content with this estimate.

Because the offset is obtained by calculating the linear combination of the coefficients ki (formula (3.1)), we do not calculate the variance of a single coefficient, but rather we need the variance of an arbitrary linear combination of the coefficients. Since the errors in the coefficients are not statistically independent of each other, this is essential, and we should expect the result to depend on the coefficients in that linear combination in a symmetric quadratic form. By (3.8), the linear combination can be represented as

$$\sum_{i=1}^{n} c_i k_i = \sum_{j=1}^{m} D_j \left(\sum_{i,h=1}^{n} A^{-1}_{ih} c_i f(a_j, e_j) \right) \qquad (3.10)$$

If we now suppose that the model we are trying to fit is exact and complete, there should be no systematic error in between the different measured offsets, and the errors of the different Dj:s may be supposed to be linearly independent of each other. Thus, we may approximate the variance of the calculated linear combination of the coefficients by the sum of the squares of the independent term errors :

$$\left\langle \left(\sum_{i} c_i k_i \right)^2 \right\rangle = \sum_{j=1}^{m} d_j^2 \left(\sum_{i,h=1}^{n} A^{-1}_{ih} c_i f(a_j, e_j) \right)^2 \qquad (3.11)$$

$$= \sum_{r,s=1}^{n} c_r c_s E_{rs},$$

where we have defined the so called error matrix E by the formula

$$E_{rs} = \sum_{j=1}^{m} d_j^2 \left(\sum_{p=1}^{n} A^{-1}_{rp} f(a_j, e_j) \right) \left(\sum_{q=1}^{n} A^{-1}_{sq} f(a_j, e_j) \right) \qquad (3.12)$$

If the differences dj are supposed to be of approximately constant magnitude d=sqrt(<dj **2>) for all a and e, this formula may be considerably simplified by using (3.6) :

$$E_{rs} = d^2 A^{-1}_{rs} \qquad (3.13)$$

The estimated error of the model (3.2) may thus be obtained from the formula

$$\langle \Delta (D(a,e))^2 \rangle = \sum_{r,s=1}^{n} f_r(a,e)\, f_s(a,e)\, E_{rs} \qquad (3.14)$$

As a summary we see that the estimated error of a linear combination of the coefficients is given by a quadratic form, the matrix of which is E. An eigenvector expansion of E may give much information about the goodness of the set of functions that was fitted. If E has a particularly large eigenvalue, it most propably means that the functions are not very well linearly independent of each other with respect to the data distribution obtained. The dependence is then found between those functions, whose components are particularly large in the direction corresponding to the large eigenvalue.

The eigenvector expansion is useful in computing error estimates of linear combinations of the coefficients ki. The error estimate may be computed from

$$\langle \Delta(\sum_i c_i K_i)^2 \rangle = \bar{c}^T E c = \bar{c}^T \sum_{i=1}^{n} u_i^2\, \bar{e}_i\, \bar{e}_i^T\, c$$

$$= \sum_{i=1}^{n} u_i^2\, (\bar{e}_i \cdot c)^2 \quad , \qquad (3.15)$$

where the $\bar{e}_n$ are the normalized (column)eigenvectors of E and un**2 are the corresponding eigenvalues. The spectral representation for the error matrix calculated by (3.12) is supplied by the ANTCAL-CONTROL program.

4. Antenna calibration programs

4.1. The control program ANTCAL-CONTROL

A command processor background program ANTCAL-CONTROL was written to control the whole calibration task. By giving different commands it is possible to define the data collecting input parameters, start the data collecting and stop it, and monitor the working of the collecting program. In addition to that it is possible to make charts of the measured data and analyze it by fitting models to it. Charts about these analysis results are also available.

The commands to the program may be abbreviated provided they remain unambiguous. The possible parameters for the commands may be given at the same line as the command, but if not given, the program asks for them. If no answer comes default values are usually assumed. The complete set of commands is explained below.

4.1.1. Commands for controlling the data collecting

Define-antcal-input : this command creates the file containing the necessary information for the data collecting program. It asks for the stars used, data store files, etc. The questions are :

Periodically : answer yes if you want to start a periodical measurement.

Initial offsets from model : answer no if you want to give initial offsets manually for each star, and yes if you want the old model to be used as initial. (No model available means zero offsets).

Number of stars : number of stars for which the measurements are made.

Delay between stars : the number of minutes the program should hold between measurements of sequencial stars, default is zero.

1'st star : 1.star number (1=CAS-A 2=CYG-A 3=3C123 4=TAU-A 5=VIR-A 6=ORINEB -1=1950-coord.)

Initial azoff,eloff : initial estimates for the offsets.

Sweep speed : return for default .15 deg/sec.

Sweep size : return for default 4.2 deg for elev and 4.2/cos(e) for azimuth.

Sweep direction : return for alternating direction.

Check data and antenna : answer yes to activate checks that everything has really happened as it should.

Output file : lists some additional information about the program. If not wished, press return. This is usually only used for testing the program.

Store file : store file name for calibration results.

Display-antcal-input : displays the input file defined by the previous command.

Start-antcal <input file> : copies the file specified to the file (RT)ANTCAL-INPUT:DATA and gives the RT command RT ANTCAL to start the data collecting. (Note: RT-commands are restricted for users RT and SYSTEM.)

Stop-antcal : gives the RT command RT ACSTOP to stop the data collecting program.

Display-status : copies the file (RT)ANTCAL-STATUS:DATA to the terminal so that the user may see what is currently happening in the collecting program.

Store-file-heading <store file> : writes a heading to the store file.

Store-file-comment <store file> : writes a short comment to the data store file.

Comment-about-these-pgms : this command is used to give praising comments about the excellence of this program package. The comments go to the file (RT)ANTCAL-COMMENT:DATA, which may sometimes possibly be read by the authors of this program.

4.1.2. Commands for the analysis of measured data

Input <inp file> <az/el> <start.date> <end.date>
<max number iter> <max last corr> <direction> :
This command is used to append more data to the internal buffers. The input file is scanned throush and the data found that fulfills the criteria specified is appended to the internal data buffer. The default values for the criteria are as follows :
<input file> : (RT)ANTCAL:DATA
<az/el> : no default value
<starting date> : 0000 00 00 00.00
<ending date> : 1999 00 00 00.00
<maximum number of iterations allowed> : 3
<maximum last correction allowed> : .1 deg
<sweeping direction> : +-
Note : The internal data buffer is limited to 1000 points.

Initialize :
This command is used to clear the cumulative matrices used for calculation of the fitting coefficients. This command should always be used before computing a new set of coefficients.

Model-input :

This command is used to choose a set of functions to build the model used. The user may answer "def" to get the default model of formula (2.11). Otherwise he may choose functions from a collection of functions of a and e by specifying a range of each, after which all the cross products between these sets are appended to the model. The set of functions available is

1	1	11	SIN(5*AZ)	1	1	11	TAN(EL)
2	COS(AZ)	12	COS(6*AZ)	2	EL	12	TAN(EL)**2
3	SIN(AZ)	13	SIN(6*AZ)	3	EL**2	13	TAN(EL)**3
4	COS(2*AZ)	14	COS(7*AZ)	4	EL**3	14	SIN(EL)
5	SIN(2*AZ)	15	SIN(7*AZ)	5	EL**4	15	SIN(2*EL)
6	COS(3*AZ)	16	COS(8*AZ)	6	EL**5	16	SIN(3*EL)
7	SIN(3*AZ)	17	SIN(8*AZ)	7	EL**6	17	COS(EL)
8	COS(4*AZ)	18	COS(9*AZ)	8	EL**7	18	COS(2*EL)
9	SIN(4*AZ)	19	SIN(9*AZ)	9	EL**8	19	COS(3*EL)
10	COS(5*AZ)	20	SW.DIR/2	10	COT(EL)	20	1/COS(EL)

Here, the function SW.DIR/2=+0.5, if the sweep is made in positive direction, and SW.DIR/2=-0.5, if the sweep is made in negative direction. So, the coefficient of it corresponds to the magnitude of hysteresis.

Data-map :

Produces a map of the distribution of the data in the internal buffers, as well as a rough map of the measured offset values on the output file.

Compute :

Fits a model to the data in the internal buffers. A model must be defined before compute command is given. This command produces a histogram representing the goodness of the fit together with the fit parameters at the output file.

Model-map :

Produces a map of the offsets calculated in the whole sky at the output file together with a map displaying the differences between the measured offsets and the fitted ones.

Error-map :

Produces a map of the estimated errors of the fitted model calculated from (3.14) and outputs the eigenvector expansion of the error matrix (3.12).

Output-file :

Closes the present output file and opens a new one. Default for the output file is the old one, and it is possible to set the output from eg. the line printer by giving the command "OUTPUT-FILE,,".

Calculate <a> <e> :

Calculates single values of the offsets from the model fitted for given a/e values.

Clean <maximum deviation> :
Throws away all those measured values from the internal buffers, which do not fit to the model last fitted within the maximum deviation value. Default for maximum deviation is 0.1 degrees.

Write <output file> :
Stores the program's internal buffers and fit results to a file, from which they may be again read back by the read command. Eliminates the need for doing everything from the beginning if one must break the the working with this program for some reason. Default for the file name is ANT-OFFSET-ANAL:DATA.

Read <input file> :
Recovers the data written by the write command. Default for the input file is ANT-OFFSET-ANAL:DATA.

Help :
Gives a list of the commands available.

Exit :
Used to return to SINTRAN-III.

4.2. Data collecting programs

4.2.1. General

The data collecting is done by a set of RT-programs, which are controlled by a background program. The user defines the measuring procedure, which is carried out periodically by the main RT-program ANTCAL. The gathered data consists of antenna pointings to stars together with corresponding calculated offset values.

A special iterative sequence of sweeps is employed to find the stars. First, theoretical star positions are calculated and possible initial offset values are added to these. Then these values are used in making two sweeps first one in azimuth and the second one in elevation through the star. By finding the peak in the sampled data in the azimuth sweep it is possible to see how far the calculated azimuth was from the correct one. Similarly, from the elevation sweep data a correction for the elevation offset can be found. These corrections are then used for making a new, hopefully better pair of sweeps through the star. At most three pairs of sweeps are made and the correction is stored into the calibration data file if it at some stage is less than .02 deg in elevation and .02 deg / cos(e) in azimuth.

If a pair of offsets is accepted or three pairs of sweeps are made in vain, the sweeping direction is changed and the measurement is done again in the reverse direction to get rid of some time delay errors. After this, when measurements are made succesfully or unsuccesfully in both directions a new star is chosen and the procedure is repeated. So, this program scans through all the user input stars in sequence.

4.2.2. Star programs

Star programs are the same programs that are used in the antenna control software package. So, there are six stars preprogrammed into the system and if more stars are needed this can be achieved by giving -1 as an answer when the computer asks for the star number. After that the program asks the 1950-coordinates and initiates that star to the internal table. At most 6+10 stars can be used this way. If the same star is used several times, at most 20 stars may be specified.

The six preprogrammed stars are:

STAR NO	NAME	R.ASC H	M	S	DECLIN DEG	M	S	POWER DENS	SIZE M
1	CAS-A	23	21	12	+58	32	45	3300	4
2	CYG-A	19	57	45	+40	35	02	2340	1.2
3	3C123	04	33	56	+29	34	13	70	1
4	TAU-A	05	31	31	+21	59	17	955	5
5	VIR-A	12	28	17	+12	39	49	263	5
6	ORINEB	05	33		-05	24		340	10
	3C390	18	45	52	+79	42	48	17	

UNIT FOR POWER DENSITY: 10^{-26} W M^{-2} HZ^{-1} AT 1000 MHZ.

Star 3C390 is not preprogrammed, but should be used, since it is the only useful star in the high northern sky.

4.2.3. The main program ANTCAL

The main RT-program ANTCAL controls the sweep and star changing sequence, sampling of data and computing the results. It is initiated by the control program command START-ANTCAL or by operating system command RT ANTCAL. ANTCAL takes its input from the file (RT)ANTCAL-INPUT:DATA, where the user has specified a set of stars to be measured and other input parameters, as the sweep definitions etc.

The measuring is started from the first star in the list and is repeated for next stars until the end of the list. If the procedure was defined to be periodical, the whole measuring procedure is repeated until the programs are terminated externally or the program itself finds an error condition, e.g. 10 sequential sampling errors or 10 pointing errors.

The star sweeps can be done either to + direction, i.e. clockwise in azimuth and up in elevation, or to - direction, i.e. counterclockwise in azimuth and down in elevation. Normally both directions are scanned for each star. One sweep over a star is performed by a subroutine SWEEP, which is described in the next section.

During the sweep the data is gathered into a buffer, an array in a common linking segment, and afterwards the data from the azimuth and elevation sweeps are transformed to fixed length data vectors by a subroutine DATATRN. When the data from both azimuth and elevation sweeps are available, the offset values are calculated. The subroutine ANTOFF calculates the differences AZDIF and ELDIF between the antenna pointing values and the real star position. New offset values are got as a result when these differences are added to the previously used offset values together with the correction caused by the error between the calculated and real pointings of the antenna in the middle of the sweep. If the difference is not inside specified limits in either azimuth or elevation, the same sweeps in both azimuth and elevation with new offsets are performed again. This iterative measuring is repeated until the difference is satisfactory, or at most three times. The succesfully measured offset values are stored into the file (RT)ANTCAL-DATA:DATA, together with the information of the star positions, the sweep direction and number of iterations used.

4.2.3.1. The subroutine SWEEP

The subroutine SWEEP performs one sweep over a star, either in azimuth or elevation, either in positive or negative direction and with defined sweep size and speed.

Star positions are calculated at the beginning, middle and end of the sweep and depending on the sweep direction the half-sweep size is either subtracted from or added to the beginning and end star positions. Linear interpolation is used between these positions. So, eg during elevation sweep the azimuth changes also a little to take care of the star's movement in the sky during the sweep. The date and time needed in star calculations are taken from the CPU-clock. The start time of the sweep is the current time plus a delay of 6 seconds and the sweep time is calculated from the sweep size and speed.

During the sweep two tasks must be executed simultaneously and with independent frequencies: to move the antenna and to sample the data. In addition the antenna position in the middle of the sweep must be checked to be able to correct a possible error in the pointing. Three small RT-programs are written for the sweep: ANTSMP to sample the data, ANTCNT to move the antenna continuously and ANTAZEL to check the middle position of the antenna. These RT-programs are started by ABSET-commands: ANTSMP and ANTCNT one second before the start-time of the sweep and ANTAZEL at the middle-time of the sweep. The calling program ANTCAL itself is in RTWT-state during the sweep, but is restarted at the end-time of the sweep and ANTSMP and ANTCNT will then be aborted. To avoid a delaying of antenna move commands due to possible heavy load of computer, the segment where RT-program ANTCNT is loaded is fixed in memory before the sweep and unfixed afterwords.

4.2.3.2. Autocorrelation algorithm ANTOFF

A method is needed to calculate the position of the peak of the beam pattern of the measured data. This should be as immune to noise as possible. That is why the position of the peak is found by calculating the zero of the cross correlation function of the measured pattern during the sweep and the derivative of a theoretical model of the beam-pattern. A simple besselian model for the diffraction of waves from a circular hole is used to approximate the theoretical model. The following formula is used:

$$\text{Power}(r) = J_1^2(x) / x^2, \qquad x = r*0.993/5.1356 \qquad (4.1)$$

The data collected is contained in the common vector IDATA that is transformed to a vector of length 128 by the subroutine DATATRN. This is because the length of IDATA is variable; it depends on elevation during azimuth sweep. Another vector of length 128 is formed from the the theoretical pattern and these two vectors are given to subroutine DISPLAC, which calculates their displacement by finding the zero of the cross correlation function between the measured data and the

derivative of the model. A possible non-uniform background is first eliminated by fitting a linear function to the first and last 20 points of the measured data vector and subtracting that from the data.

4.2.4. Continuous move program ANTCNT

The sweep over a star must be done with as even speed as possible. That is why the move commands to ACU are given by an independent RT-program ANTCNT, which can be executed with a high priority and even be fixed in memory during the sweep.

The start position of the antenna and the increments for azimuth and elevation are given in common data area on the common linking segment. The pointing commands are given periodically in a loop, where the azimuth and elevation are incremented until the pointing command is illegal, i.e. either azimuth or elevation is out of limits. The pointing interval is given externally. The default sweep speed is .15 deg/sec and the sweep is achieved by giving the antenna move commands at a 5Hz frequency. Two herz was tried first, but it was found to be the resonance frequency of the elevation counterweights.

Due to the servo system the antenna lags about one second with respect to the commands given. This has been taken care of by giving the commands in advance. The right amount of this displacement is found by checking the antenna position at the middlepoint of the sweep and comparing it to the command given. The difference of these two values is then used in the next scan to make the scan go through the middle point at the right time. The checking is done by an RT-program ANTAZEL.

4.2.5. Sampling program ANTSMP

The data sampling is made through a CAMAC A/D converter unit, which is sampled at a frequency of 10 Hz by an RT-program ANTSMP. This program is initiated by the program ANTCAL to start at the beginning of the sweep and it stores the data in the common variable idata. The proper crate unit, module position and ADC-mask should be checked from the ANTSMP program listings. At Sodankyla the following values were used : IC =1, IN=12, MASKADC=400B.

The voltage input to the CAMAC should be positive and it should be greater than 1 volt when the star is in the beam. In the receivers 30 MHz filter settings are best for getting a noiseless signal. The data can be outputted from almost any test output of the receivers, but a low pass filter of 15...40 Hz is needed before CAMAC to integrate the signal. In addition a 10 Hz trigger signal has to be inputted to the CAMAC module from the frequency divider.

A controller program command TEST-CAMAC-SMP was made for checking that the data flows correctly out from CAMAC. This program reads the CAMAC with commands similar to the commands in the ANTSMP program and displays the data read at the terminal.

4.2.6. Program to terminate the data collecting

The data collecting by the RT-program ANTCAL goes on for ever if the measurement was defined periodical. To terminate the measurement it is recommended to use the control program command STOP-ANTCAL, which starts a special RT-program ACSTOP, which takes care of aborting all RT-programs involved in the calibration system, closes all files possible opened by ANTCAL, and unfixes the segment of ANTCNT if termination is executed during a sweep. The program ACSTOP can also be started by command RT ACSTOP.

5. Practical use of the calibration system

5.1. Loading the programs

The calibration system consists of several files, from which all program files are owned by user ANTENNA-EISCAT and all data files are owned by user RT. The loading of programs is performed by mode-files.

5.1.1. Loading the background control program

```
MODE (ANTE)ANTCAL-CONTROL:MODE,,
```

Following files must be present:

```
(ANTE)ANTCAL-CONTROL:SYMB
(SYS)SSPMATR:BRF
(KER)TEK-GRAF:BRF
(ANTE)ANTCAL-CONTROL:PROG
```

If the file names or their owners differ, the mode-file must be corrected. Here, (SYS)SSPMATR is the file containing the matrix routines MINV, GPPROD and EIGEN from the scientific subroutine package, and (KER)TEK-GRAF is a file containing the Tektronix draw routines written by Joe Armstrong (version which does not need the assembly declarations).

In the outputs from the ANTCAL-CONTROL-program the site name is defined by the site code IS, which is given a fixed value in the program. If the analysis is performed using data from a remote site this leads to erroneous headings in outputs. Therefore the value given to IS must be corrected before compiling and loading the program (see the listing, line 75). Later maybe we can use a command to change the site code.

5.1.2. Loading the real time programs

```
MODE (ANTE)ANTCAL-RT-LOAD:MODE,,
```

Following files must be present:

```
(ANTE)ANTCAL-MAIN:SYMB          (ANTE)ANTCAL-MAIN:BRF
(ANTE)ANTCAL-OFF-CALC:SYMB      (ANTE)ANTCAL-OFF-CALC:BRF
(ANTE)ANTCAL-SWEEP:SYMB         (ANTE)ANTCAL-SWEEP:BRF
(ANTE)ANTCAL-SAMP:SYMB          (ANTE)ANTCAL-SAMP:BRF
(ANTE)ACU-SUBROUTINES:SYMB      (ANTE)ACU-SUBR-RT:BRF
(ANTE)STAR-PACK:SYMB            (ANTE)STAR-PACK:BRF
(ANTE)LIBRARY:SYMB              (ANTE)LIBRARY-RT:BRF
```

Before loading the segment numbers must be defined for the site involved and the correct segment numbers changed in the file ANTCAL-RT-LOAD:MODE. The segment numbers for programs ANTCNT and ANTSMP are also used in ANTCAL-program and therefore the variables "MOVSEG" and "SMPSEG" must be given the correct values. (See the listing, lines 471-480).

The system also uses a number of files to store and deal different data. They should also be present during calibration. The files are the following :

- for information transfer between RT and background:
  (RT)ANTCAL-INPUT:DATA
  (RT)ANTCAL-STATUS:DATA
  (RT)ANTCAL-IN-STNDRD:DATA
  (RT)ANTCAL-DATA:DATA
- for user comments:
  (RT)ANTCAL-COMMENT:DATA

5.2. CAMAC connections

The data must be fed in to the CAMAC through a multiplexed ADC unit 9087. The position of that unit must correspond to the parameter values in the ANTSMP-program. These values (ICR,IN,MASKADC) may be checked from the file ANTCAL-SAMP:SYMB. Here, ICR is the crate number, IN is the station number and MASKADC is the mask word which determines which channels of the ADC unit are sampled. Only one bit should be set in the mask word MASKADC. This bit determines the pin in the Cannon interface to which the signal must be tied.

The signal itself is radio noise from the stars. It may be taken from any test output, but to get best results the bandwidth should be as large as possible (30 MHz). The detected signal should then go through a low band pass filter. The sampling rate is 10Hz, and so the bandwidth of that filter should be of this order. This filter is meant for integrating the signal over the period between different samples. If the limit frequency is too low, it is seen as an indeterminacy in the timing and it may lead to hysteresis effects. If the limit frequency is too big, the data will be noisier. The signal should be amplified so that for the most powerful star it uses the whole range of the CAMAC ADC unit, but does not get clipped. So, for Cas-A the signal should be near to 10V so that it will not be lost to the limited resolution of the ADC unit for the weakest stars. However, some allowance must be left for a possible change in the background noise.

5.3. Choice of stars

The data gathered should be as evenly distributed in the local sky as possible. That is why all the stars available should be used for calibration over at least one complete 24-hour period. There are six stars which may be specified simply by a number, and in addition to these one star with a very high declination is useful since it fills the hole in the high northern sky where the other stars never go. The stars are :

STAR NO	NAME	R.ASC H	M	S	DECLIN DEG	M	S	POWER DENS	SIZE M
1	CAS-A	23	21	12	+58	32	45	3300	4
2	CYG-A	19	57	45	+40	35	02	2340	1.2
3	3C123	04	33	56	+29	34	13	70	1
4	TAU-A	05	31	31	+21	59	17	955	5
5	VIR-A	12	28	17	+12	39	49	263	5
6	ORINEB	05	33		-05	24		340	10
	3C390	18	45	52	+79	42	48	17	

UNIT FOR POWER DENSITY: 10^{-26} W M HZ AT 1000 MHZ.

When specifying the measuring sequence, the movements of the antenna should be minimized. The distance from one star to another may be rather large, especially in azimuth, and the stars should be chosen in such an order that the distances between consecutive stars are not very long. Most roughly, the positions are determined by the right ascension figures, and there are drawings about the star positions at the appendix A. With help of these it is easy to choose the star sequence. A sequence that is recommended is :

1,1,2,2,5,5,5,6,6,6,4,4,3,3,3C390

The same star appears many times in this sequence because this way the traversing time between different stars is diminished. At some time of the day it may take a rather long time to go to the 3C390, since as contrary to the other stars this never leaves the northern half of the sky. This is why it might be measured alone, in a separate run.

In practice, the star sequence is specified by the control program command DEFINE-ANTCAL-INPUT. It is also possible to specify whether initial offsets are calculated by some old model, or if they are specified by hand for each star separately. If an old model is available in the pointing routines, it saves time to use it (if it is at all correct), and then the sweep size may also be diminished, to 1.2 ... 1.5 deg, from 2.1 deg, which also saves time.

After specifying the input parameters, it is possible to start the collecting program by the command START-ANTCAL. When running, the status of that program may be monitored by the command DISPLAY-STATUS, that tells on the terminal, what the program is doing. The collecting may then be stopped by the STOP-ANTCAL command.

5.4. Notice on timing accuracy

When making calibration measurements it is extremely important that the CPU clock is in correct time. The earth rotates at a velocity of 0.0040166 deg/sec, and so an error of 2 seconds in the CPU clock would lead to a systematic error of the order of 0.01 degrees in the measured offsets.

5.5. Analysis of results

The analysis of results and fitting of models is done by the control program. The whole analysis is accomplished by the commands INPUT-DATA, DATA-MAP, INPUT-MODEL, INITIALIZE, COMPUTE, MODEL-MAP, ERROR-MAP. If the output is wished to some file or line printer, the command OUTPUT-FILE must be given first.

The defaults are recommended to be used as models, possibly enhanced with the hysteresis terms. After inputting the data and model, the COMPUTE command produces the coefficients of that model, together with their errors. An example run of the analysis can be found from the appendix D.

5.6. Use of the results

The analysis produces a model for the antenna offsets. This model should then be applied to the antenna pointing routines. It may be inserted by hand to the subroutine OFFSET in the file (ANTE)ACU-SUBROUTINES:SYMB. (See the listing, lines 246-262). However, the refraction and hysteresis terms should be left at this stage, since the former is only good for stars and not inside the ionosphere, and the latter cannot be used in a reliable and simple way.

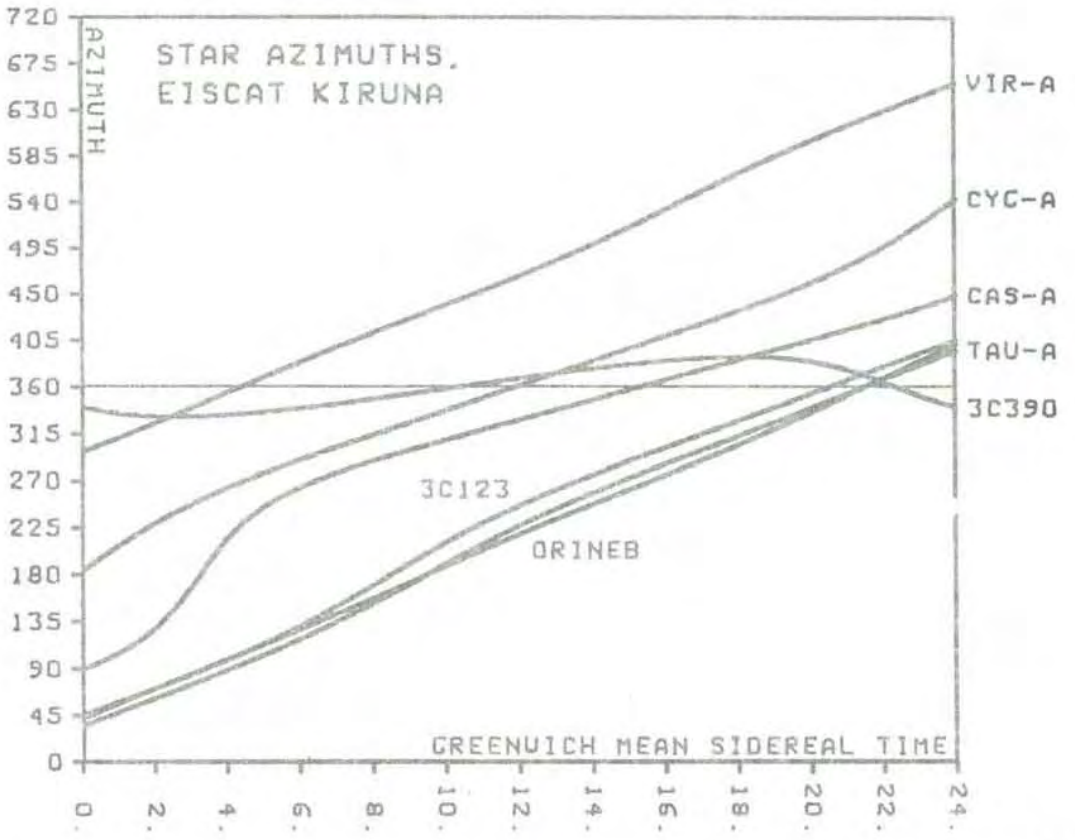


Figure A1. Star azimuths versus Greenwich mean sidereal time in Kiruna.

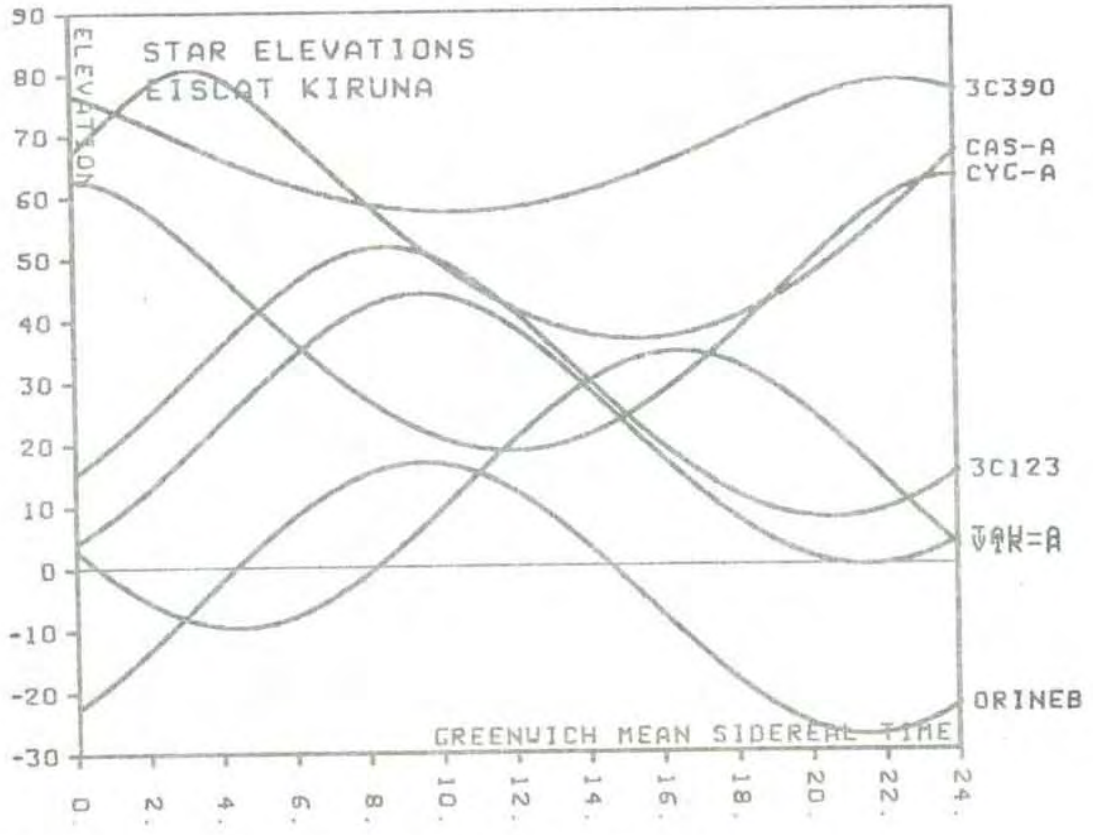


Figure A2. Star elevations in Kiruna.

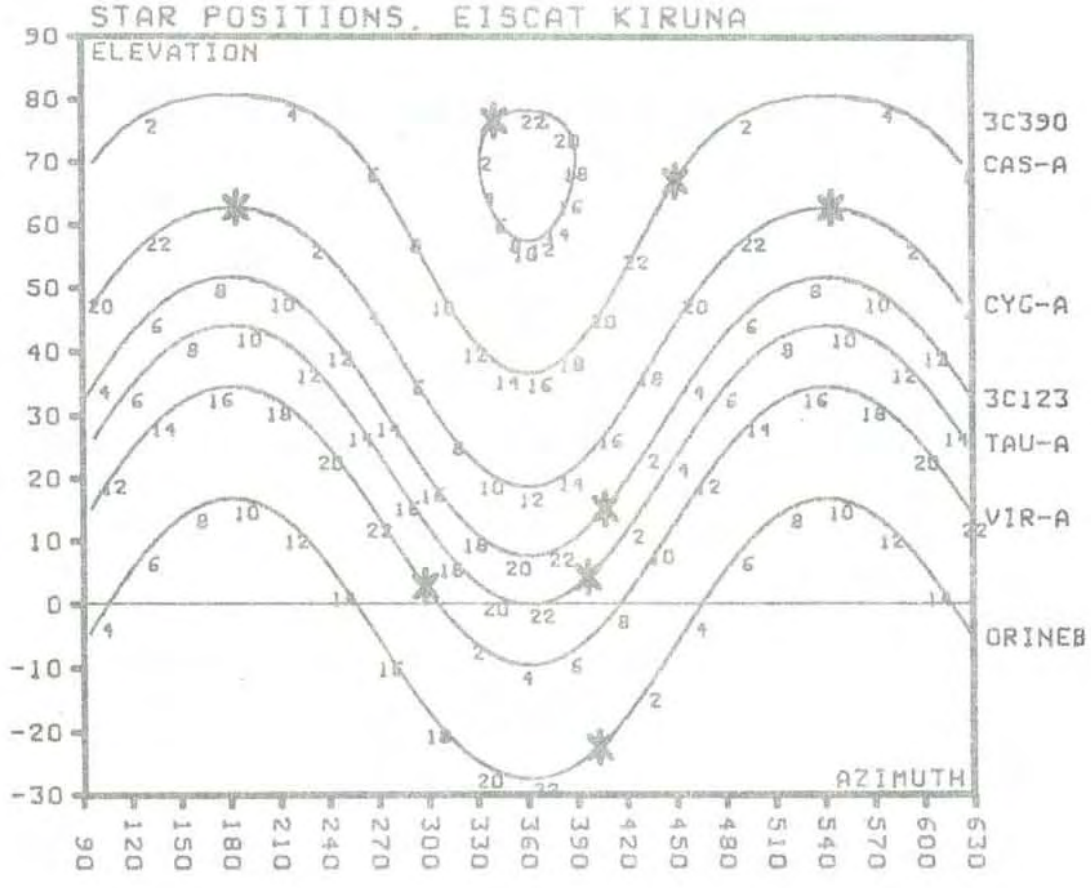
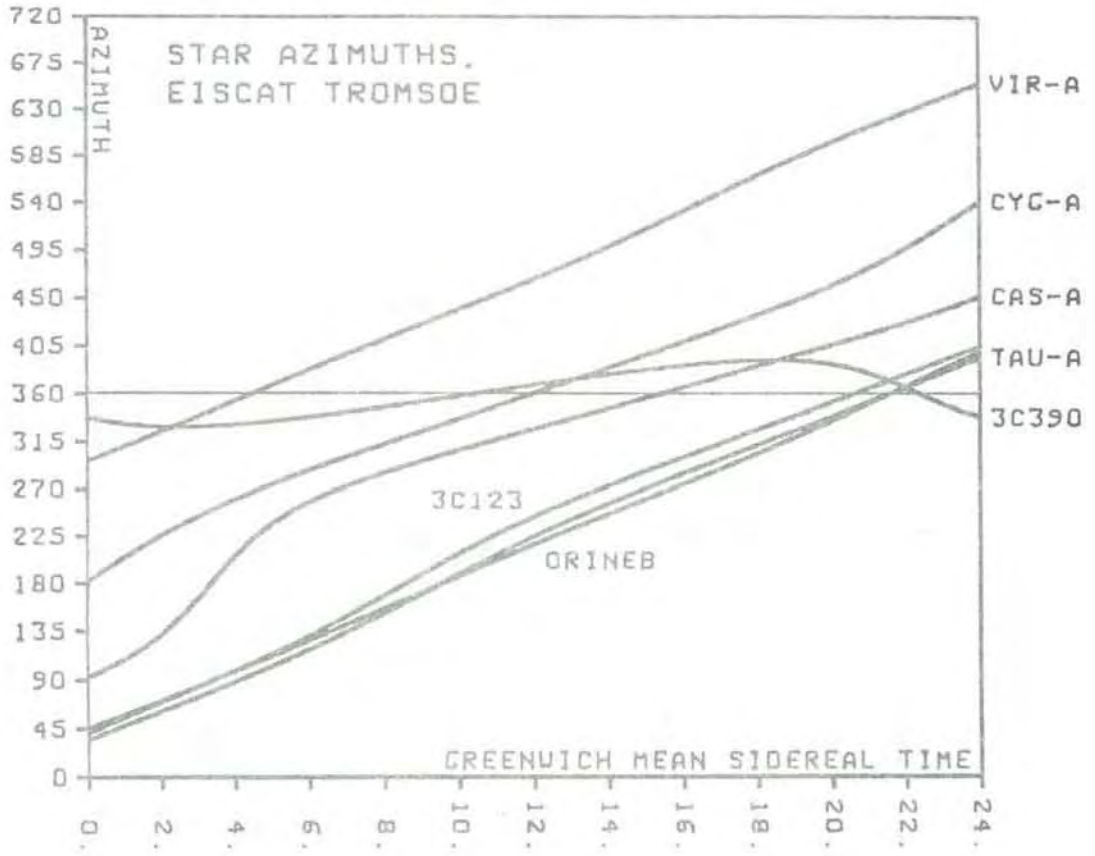
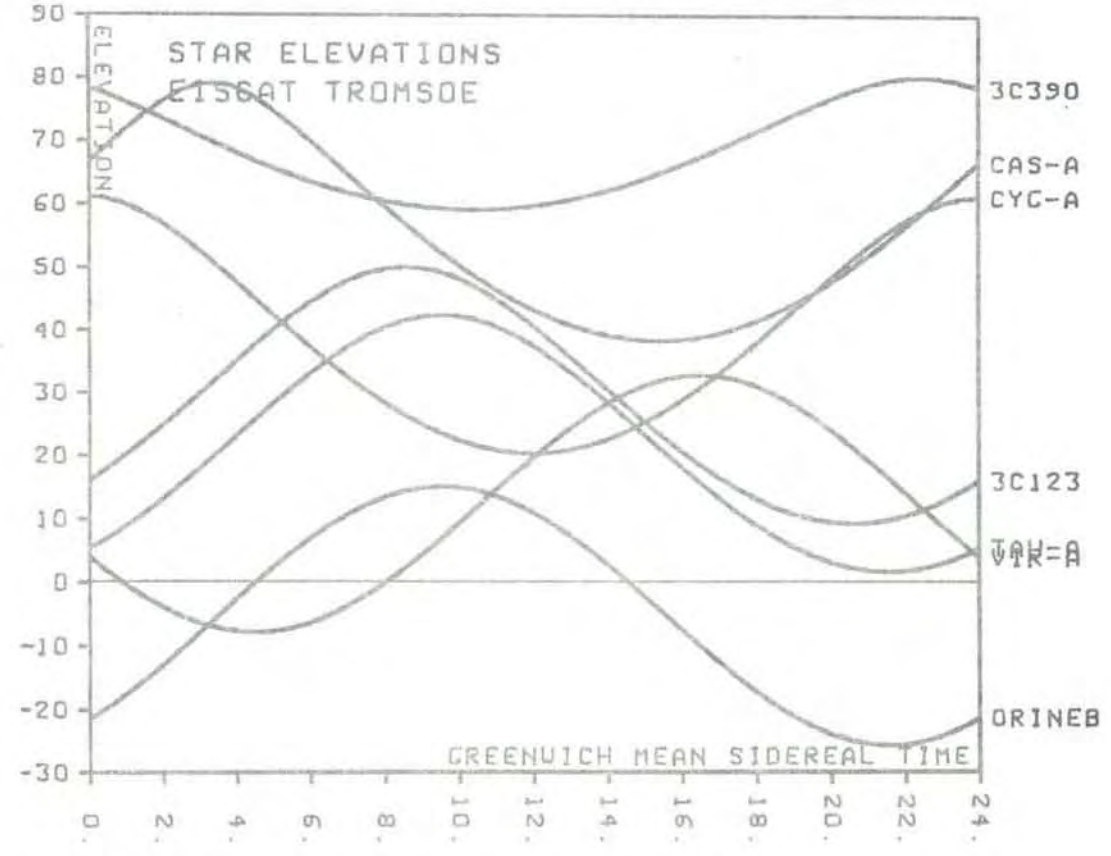


Figure A3. Star positions in Kiruna. Greenwich mean sidereal times at two-hour intervals are plotted.



Fisure A4. Star azimuths versus Greenwich mean sidereal time in Tromsoe.



Fisure A5. Star elevations in Tromsoe.

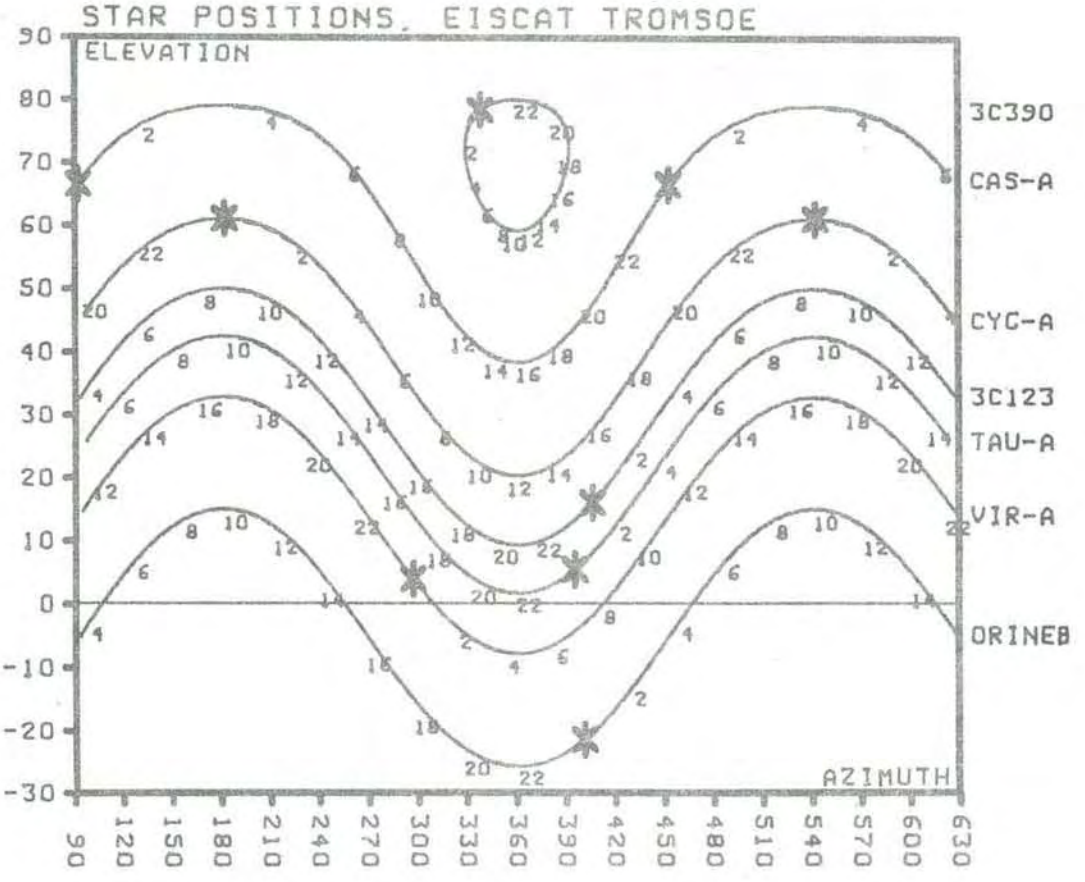


Figure A6. Star positions in Tromsoe. Greenwich mean sidereal times at two-hour intervals are plotted.

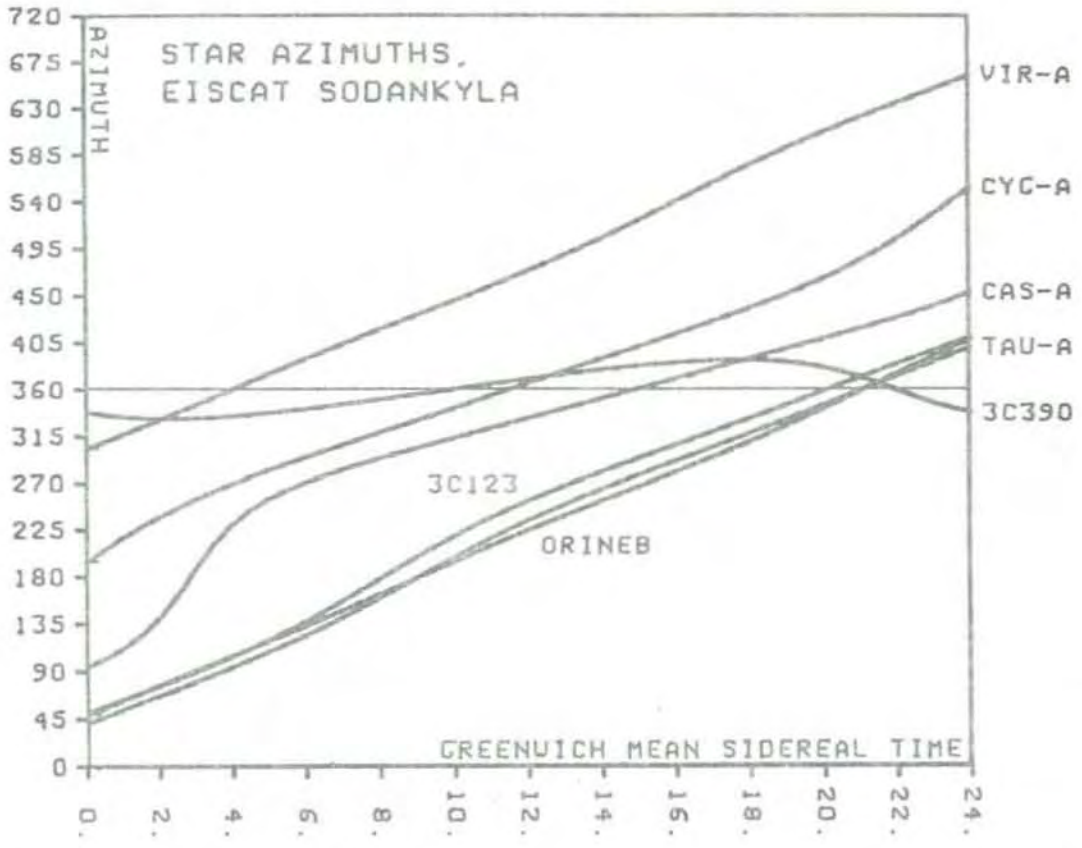


Figure A7. Star azimuths versus Greenwich mean sidereal time in Sodankyla.

Figure A8. Star elevations in Sodankyla.

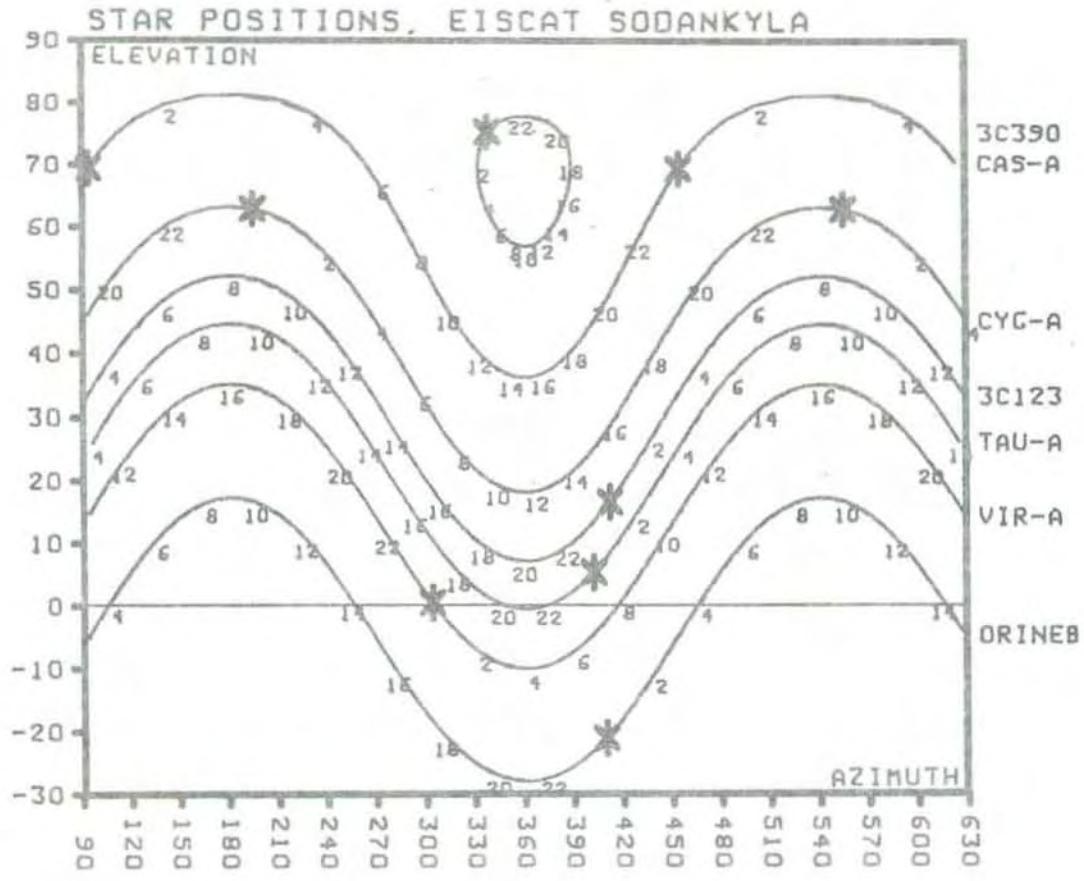
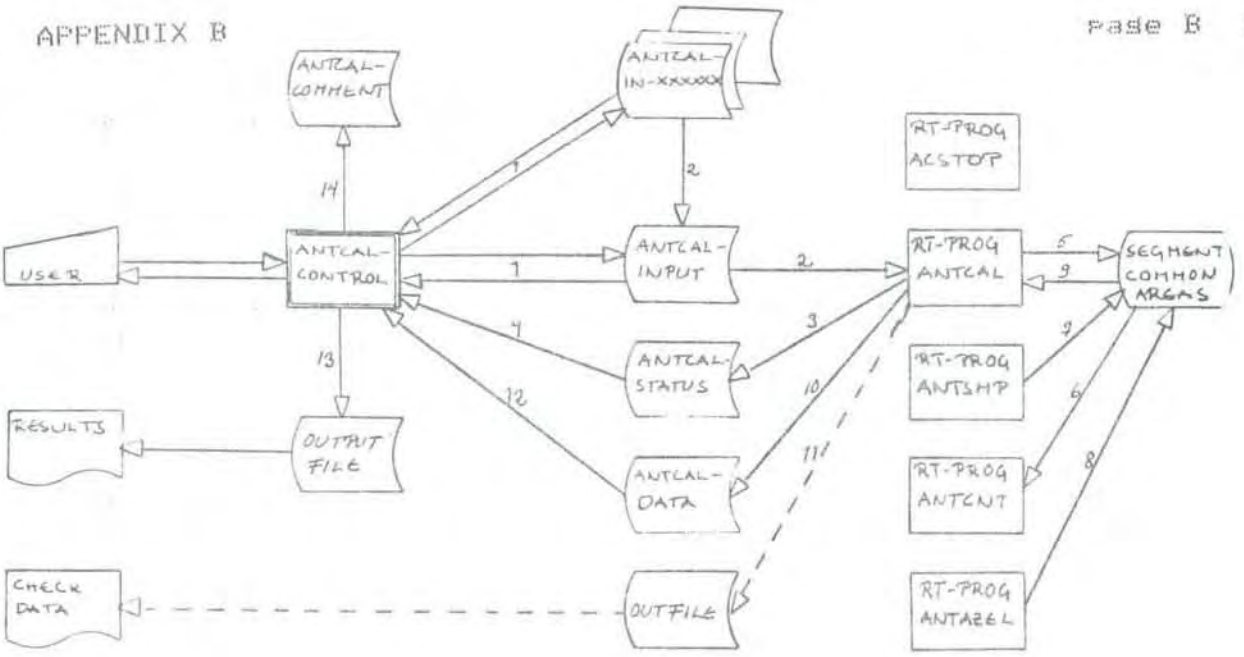


Figure A9. Star positions in Sodankyla. Greenwich mean sidereal times at two-hour intervals are plotted.

GREENWICH MEAN SIDEREAL TIME

MON	DAY	UT 00:00	03:00	06:00	09:00	12:00	15:00	18:00	21:00
JAN	1	06:42	09:43	12:43	15:44	18:44	21:45	00:45	03:46
	11	07:22	10:22	13:23	16:23	19:24	22:24	01:25	04:25
	21	08:01	11:02	14:02	17:03	20:03	23:04	02:04	05:05
FEB	1	08:44	11:45	14:45	17:46	20:46	23:47	02:47	05:48
	11	09:24	12:24	15:25	18:25	21:26	00:26	03:27	06:27
	21	10:03	13:04	16:04	19:05	22:05	01:06	04:06	07:07
MAR	1	10:35	13:35	16:36	19:36	22:37	01:37	04:38	07:38
	11	11:14	14:15	17:15	20:16	23:16	02:17	05:17	08:18
	21	11:54	14:54	17:55	20:55	23:56	02:56	05:57	08:57
APR	1	12:37	15:38	18:38	21:39	00:39	03:40	06:40	09:41
	11	13:17	16:17	19:17	22:18	01:18	04:19	07:19	10:20
	21	13:56	16:56	19:57	22:57	01:58	04:58	07:59	10:59
MAY	1	14:35	17:36	20:36	23:37	02:37	05:38	08:38	11:39
	11	15:15	18:15	21:16	00:16	03:17	06:17	09:18	12:18
	21	15:54	18:55	21:55	00:56	03:56	06:57	09:57	12:58
JUN	1	16:38	19:38	22:39	01:39	04:40	07:40	10:41	13:41
	11	17:17	20:17	23:18	02:18	05:19	08:19	11:20	14:20
	21	17:56	20:57	23:57	02:58	05:58	08:59	11:59	15:00
JUL	1	18:36	21:36	00:37	03:37	06:38	09:38	12:39	15:39
	11	19:15	22:16	01:16	04:17	07:17	10:18	13:18	16:19
	21	19:55	22:55	01:56	04:56	07:57	10:57	13:58	16:58
AUG	1	20:38	23:39	02:39	05:40	08:40	11:41	14:41	17:42
	11	21:17	00:18	03:18	06:19	09:19	12:20	15:20	18:21
	21	21:57	00:57	03:58	06:58	09:59	12:59	16:00	19:00
SEP	1	22:40	01:41	04:41	07:42	10:42	13:43	16:43	19:44
	11	23:20	02:20	05:21	08:21	11:22	14:22	17:23	20:23
	21	23:59	03:00	06:00	09:01	12:01	15:02	18:02	21:03
OCT	1	00:39	03:39	06:40	09:40	12:41	15:41	18:42	21:42
	11	01:18	04:18	07:19	10:19	13:20	16:20	19:21	22:21
	21	01:57	04:58	07:58	10:59	13:59	17:00	20:00	23:01
NOV	1	02:41	05:41	08:42	11:42	14:43	17:43	20:44	23:44
	11	03:20	06:21	09:21	12:22	15:22	18:23	21:23	00:24
	21	04:00	07:00	10:01	13:01	16:02	19:02	22:03	01:03
DEC	1	04:39	07:40	10:40	13:41	16:41	19:42	22:42	01:43
	11	05:18	08:19	11:19	14:20	17:20	20:21	23:21	02:22
	21	05:58	08:58	11:59	14:59	18:00	21:00	00:01	03:01

Table A1. Greenwich mean sidereal time for 1981. For other years the times differ only a few minutes and this table may be used to set rough estimates.



Fisure B1. Data flow in ANTCAL-system.
The explanation of numbered arrays is given below. The commands expressed with quotation marks refer to ANTCAL-CONTROL-program.

1. 'DEFINE-ANTCAL-INPUT <filename>,<input parameters>'
The user gives the input parameters, which are stored in a file, default file name is (RT)ANTCAL-INPUT:DATA. It is possible to create a separate file for different measuring procedures (for standard stars, for high stars, etc.)
'DISPALY-ANTCAL-INPUT <filaname>'
The user checks the content of an input file.

2. 'START-ANTCAL <filename>'
If filename is not (RT)ANTCAL-INPUT:DATA, the content of file filaname is copied into (RT)ANTCAL-INPUT:DATA. RT-program ANTCAL reads input parameters from (RT)ANTCAL-INPUT:DATA.

3. ANTCAL writes its status, i.e. what it is doing, into the file (RT)ANTCAL-STATUS:DATA. The old content of the file is overwritten each time the measurements are started with a new star, but the latest measured offsets are restored into the file.

4. 'DISPLAY-STATUS'
The user checks what is going on.

5. ANTCAL stores the sweep-information for ANTCNT into the common linking segment.

6. ANTCNT uses the sweep-information from the common linking segment.

7. ANTSMP stores the measured data into the common linking segment.

8. ANTAZEL stores the check-information (time and antenna position) into the common linking segment.

9. ANTCAL uses the measured data and the check-information from the common linking segment.

10. ANTCAL writes the calculated offset-values into the store-file (RT)ANTCAL-DATA:DATA.

11. If an output-file for ANTCAL was defined, some check-data is written into the file during the measurements. By conditional compiling '+' some additional information is stored from each sweep. Used for testing only.

12. 'INPUT <filename>'
The user reads the data by ANTCAL-CONTROL-program, the default file name is (RT)ANTCAL-DATA:DATA.

13. 'OUTPUT <filename>'
All analysis results produced after this command are written into the file filename, the default file is terminal.

14. 'COMMENT-ABOUT-THESE-PGMS'
The user writes comment into the comment file.

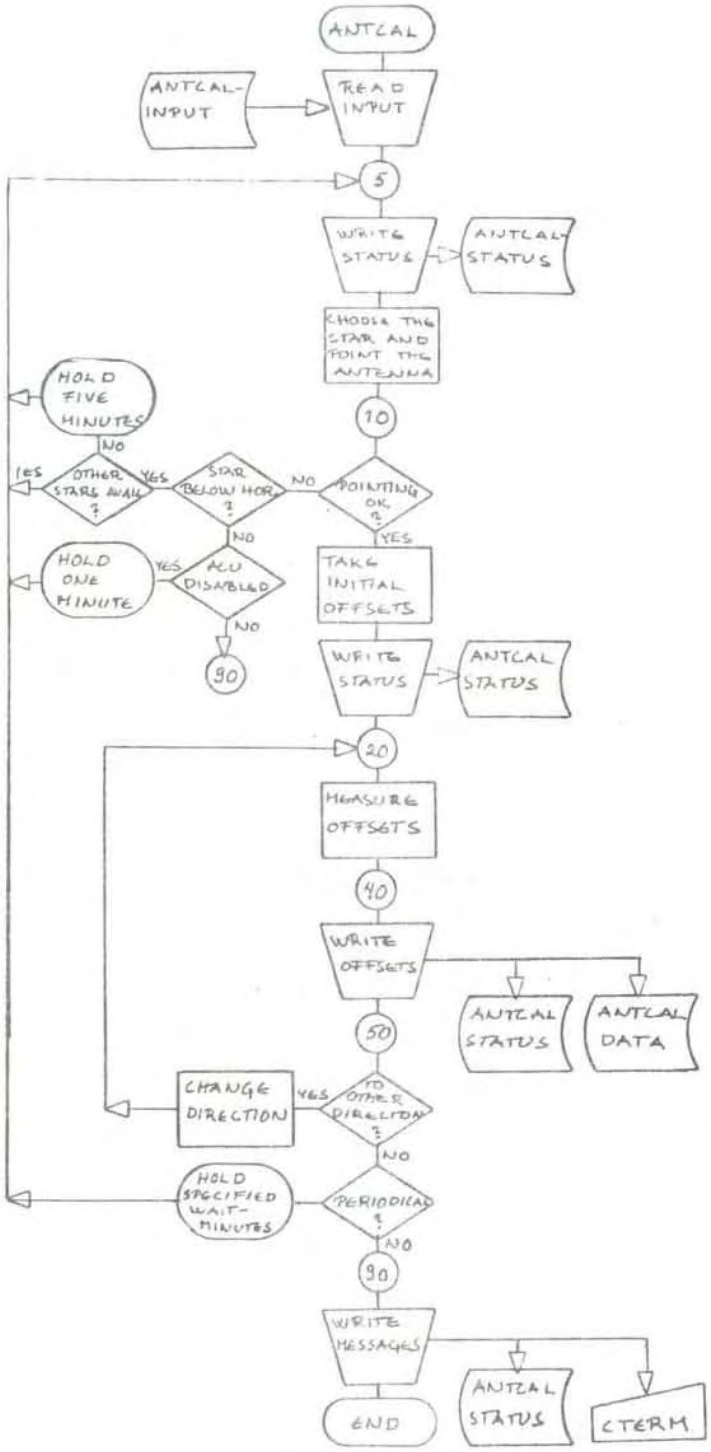


Figure B2. Block diagram of the ANTCAL program.

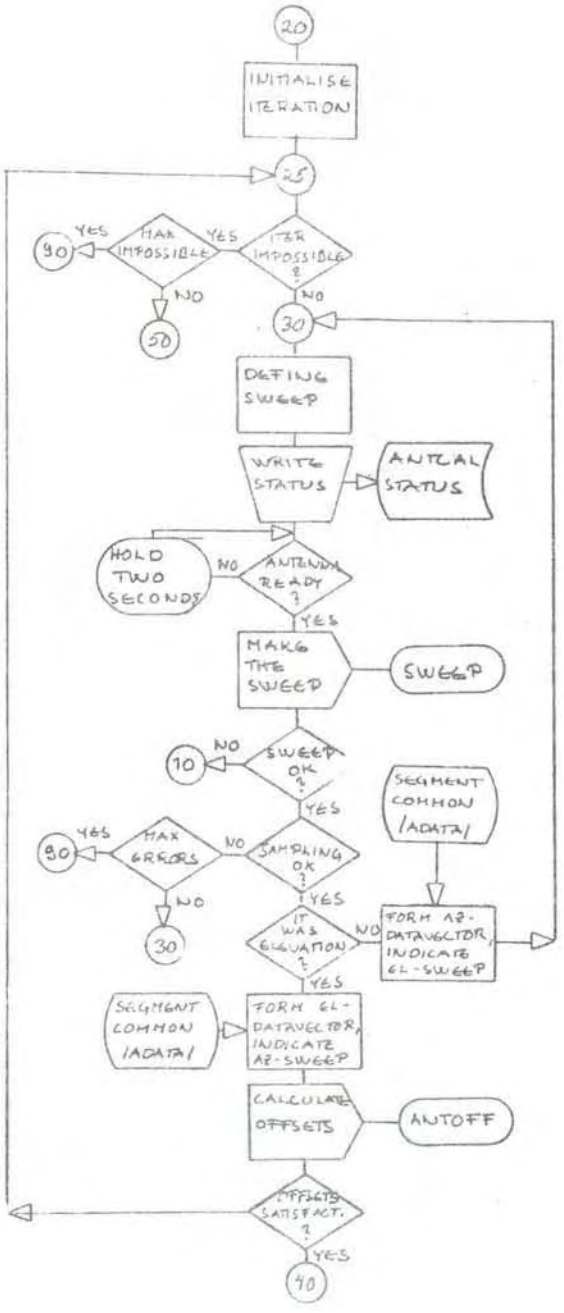


Figure B3. Block diagram of the measuring process. This is the block between line numbers 20 and 40 of the previous diagram expanded.

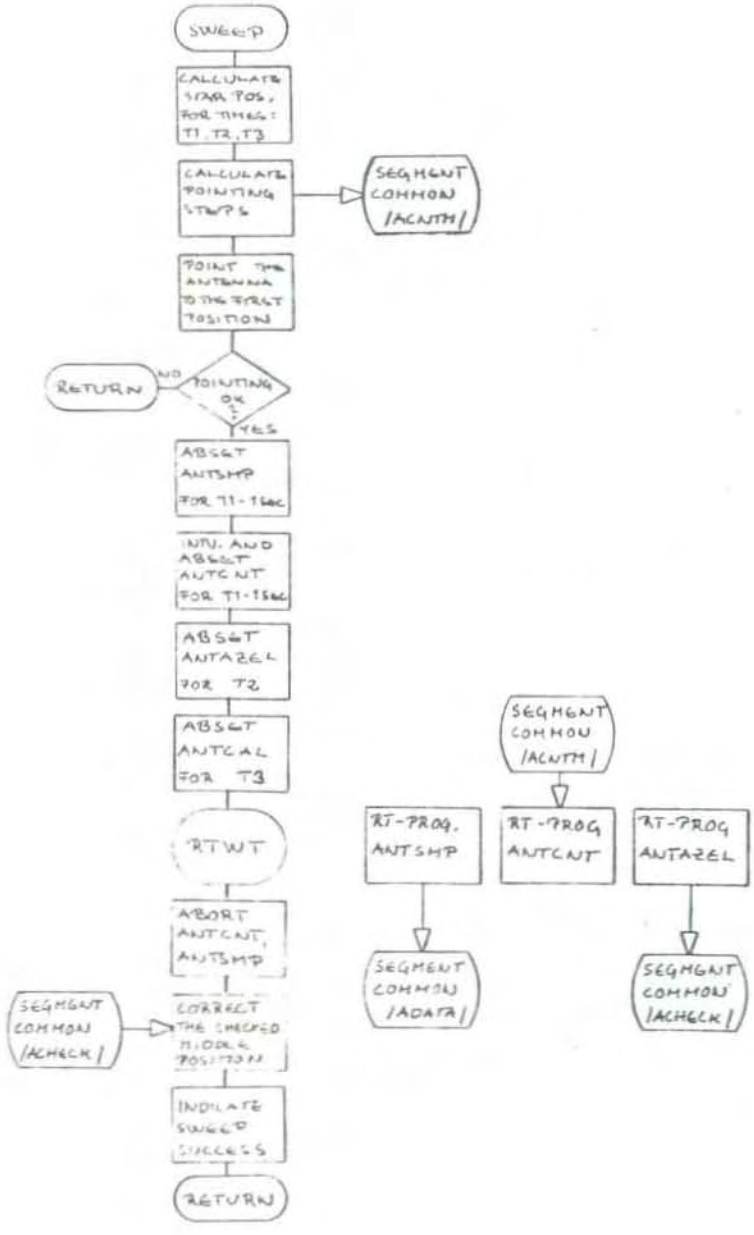


Figure B4. Block diagram of the sweep subroutine.

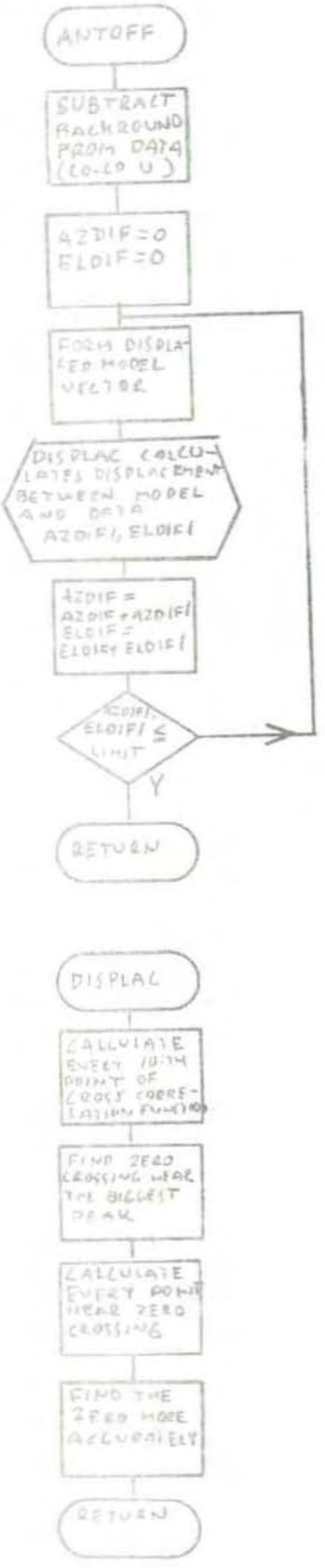


Figure B5. Block diagram of the routine ANTOFF that calculates the displacements of the beam patterns.

CONTENTS OF FILE : (ANTENNA-EISCAT)ANTCAL-CONTROL:SYMB;1

```
  16 >        PROGRAM ANTOFF
 235 >        SUBROUTINE STORE(AZ,EL,IDIR,OFF)
 252 >        SUBROUTINE COMPUTE
 335 >        SUBROUTINE INITIALIZE
 361 >        SUBROUTINE OFFMODEL(AZ,EL,IDIR,OFF)
 378 >        FUNCTION FUNCMOD(I,AZ,EL,IDIR)
 433 >        SUBROUTINE INPUT
 527 >        SUBROUTINE CLEAN
 563 >        SUBROUTINE MODELINP
 681 >CG     SUBROUTINE PLOTDATA
 716 >        SUBROUTINE MAP1(LP)
 750 >        SUBROUTINE MAP2(LP)
 775 >        SUBROUTINE MAP3(LP)
 796 >        SUBROUTINE MAP4(LP)
 835 >        SUBROUTINE MAP5(LP)
 862 >        SUBROUTINE PRINTOFF(LP,OFFMAP,TITLE)
 894 >        SUBROUTINE SYMBOL(A,C)
 914 >        SUBROUTINE ERRANAL
 953 >        SUBROUTINE ERROR(AZ,EL,IDIR,ERR)
 978 >        SUBROUTINE ERRINTRI.
1014 >        SUBROUTINE ERRINDIAG
1047 >        SUBROUTINE GETS(PRM,S,LS,DEF)
1092 >        SUBROUTINE GETI(PRM,I,IDEF)
1102 >        SUBROUTINE GETR(PRM,R,RDEF)
1111 >        SUBROUTINE SETTABS(LINE,LEN)
1142 >        LOGICAL FUNCTION TABSTOP(J)
1178 >        SUBROUTINE FITCMND(COMMAND,CNO,IERR)
1219 >        LOGICAL FUNCTION LEGAL(SHORT,CMND)
1259 >        SUBROUTINE INPDEFINE
1342 >        SUBROUTINE OPENS(IF,FILENAME,ACCESS,STATUS,IER)
1353 >        SUBROUTINE TITLOFF
1370 >        SUBROUTINE COMMOFF
1398 >        SUBROUTINE CAMACTEST
```

CONTENTS OF FILE : (ANTENNA-EISCAT)ANTCAL-MAIN:SYMB;1

```
  50 >        PROGRAM ACSTOP,50
  79 >        PROGRAM ANTCAL,50
 432 >        SUBROUTINE ATOSTAR(NSITE,NSTAR,AZOFF,ELOFF,SWSIZE,STAZ,STEL,
 433 >     *       AZWRAP,IFL)
 453 >        SUBROUTINE SWEEPINIT(IFL)
 484 >        SUBROUTINE SWEEP(NSITE,NSTAR,IAE,IDIR,SCANSIZE,SPEED,AZOFF,ELOFF,
 485 >     * DAX,DEX,AZWRAP,YEAR,MON,DAY,UT2,STAZ2,STEL2,IFL)
 546 >        SUBROUTINE STOROFF(ISF,AZAZ,AZEL,AZOFF,ELAZ,ELEL,ELOFF,
 547 >     * STAR,IDIR,ITEH,ADIF,EDIF)
```

CONTENTS OF FILE : (ANTENNA-EISCAT)ANTCAL-OFF-CALC:SYMB;1

```
  38 >CS     PROGRAM ANTOFFTEST
  76 >CS     SUBROUTINE DATASIM(AZDIF,ELDIF,STELO,SCALE,NOISE,
  77 >CS    *                   NAZ,MAZ,IDATAAZ,
  78 >CS    *                   NEL,MEL,IDATAEL)
 116 >        SUBROUTINE ANTOFF(AZDATA,ELDATA,STELO,EPS,MAXITER,HSCANSZ,CALCODE
 117 >     *                   ,AZDIF,ELDIF,IER,IOUT)
 198 >        FUNCTION AMODEL(X,Y)
 205 >        FUNCTION BESJ1(R)
 226 >        SUBROUTINE DISPLAC(T,M,D)
 281 >        SUBROUTINE PIIRRA(C,I,J)
 306 >        SUBROUTINE DATATRN(N,MCENT,IDATA,DATALEN,OUTLEN,DATA)
```

CONTENTS OF FILE : (ANTENNA-EISCAT)ANTCAL-SWEEP:SYMB;1

```
  11 >        PROGRAM ANTCNT,65
  25 >        PROGRAM ANTAZEL,64
```

CONTENTS OF FILE : (ANTENNA-EISCAT)ANTCAL-SAMP:SYMB;1

```
   1 >        PROGRAM ANTSMP,63
```

CONTENTS OF FILE : (ANTENNA-EISCAT)ACU-SUBROUTINES;SYMB;1

```
  20 >        SUBROUTINE AZTURN(INEG,AZ,IPOS,AZPOS)
  43 >        SUBROUTINE ANTSWEEP(AZ,EL,AZINCR,ELINCR,AZMAX,ELMAX,HTIME)
  78 >        SUBROUTINE ACUTEST
 145 >        SUBROUTINE ANTPNT(AZ,EL,IFL)
 202 >        SUBROUTINE ANTDIR(AZDIR,ELDIR,AFL,IFL)
 226 >        SUBROUTINE OFFSET(IOFF,AZ,EL,AOFF,EOFF)
 287 >        SUBROUTINE ANTHNK
 296 >        SUBROUTINE ANTMES(MTYPE,IFL)
 333 >        SUBROUTINE ACUAZ(AZI,IFL)
 362 >        SUBROUTINE ACUEL(ELEV,IFL)
 400 >        SUBROUTINE ACUPOS(AZI,ELEV,SETFL,IFL)
 437 >        SUBROUTINE BACUAZ(AZI,IFL)
 471 >        SUBROUTINE BACUEL(ELEV,IFL)
 515 >        SUBROUTINE BAZPOS
 536 >        SUBROUTINE BELPOS
 557 >        SUBROUTINE ACUBITS(IRW,MSW,LSW,IAE)
```

CONTENTS OF FILE : (ANTENNA-EISCAT)STAR-PACK;SYMB;1

```
  18 >        SUBROUTINE STARCLEAR
  31 >        SUBROUTINE STARCALL(SNAME,RA1950,DE1950,INYEAR,MON,DAY,UT,IOTRM,
  32 >       * NSTAR,IFL)
 110 >        SUBROUTINE PLNTCALL(PNAME,SRA,RADIFF,SDE,DEDIFF,INYEAR,MON,DAY,
 111 >       * UT,MUT,NSTAR,IFL)
 169 >        SUBROUTINE STARINIT(NSTAR,INYEAR,MON,DAY,UT,IOTRM,IFL)
 262 >        SUBROUTINE STARAZEL(NSITE,NSTAR,INYEAR,MON,DAY,UT,A,E,IOTRM,IFL)
 339 >        SUBROUTINE STARRADE(NSITE,INYEAR,MON,DAY,UT,A,E,RA,DE,IOTRM,IFL)
 405 >        SUBROUTINE STARPRM(RA1950,DE1950,NEWSTAR,IOTRM,IFL)
 431 >       SUBROUTINE PLNTPRM(INYEAR,MON,DAY,UT,NEWSTAR,SRA,SDE,STI,JD,
 432 >      *                   IOTRM,IFL)
 478 >        SUBROUTINE SITEPRM(GEOLAT,GEOLONG,IOTRM,IFL)
 500 >        SUBROUTINE STARNAM(NSTAR,SNAME)
 513 >        SUBROUTINE STARSITR(RA1950,DE1950,YEAR,MON,DAY,UT,RA,DE,GAST,JD)
 571 >       SUBROUTINE PRECES(JD,UT,ZETA,Z,THETA,EPS)
 596 >       SUBROUTINE ERTVEL(JD,UT,VE)
 630 >       SUBROUTINE SIDRAL(JD,UT,GMST)
 649 >        SUBROUTINE ROTATIONS
 674 >       SUBROUTINE COORD(AO,BO,AP,BP,A1,B1,A2,B2)
 699 >        SUBROUTINE REFR(ELT,ELA)
 713 >        SUBROUTINE HOURS(RA,IRAH,IRAM,RAS,IRAS)
 723 >        FUNCTION HMSTOH(H)
 732 >        FUNCTION HTOHMS(H)
 741 >        INTEGER FUNCTION LENGTH(A)
```

CONTENTS OF FILE : (ANTENNA-EISCAT)LIBRARY;SYMB;1

```
   1 >        SUBROUTINE WOPEN(IF,FILENAME,ACCESS,STATUS,ISEC,MAX,IER)
  21 >        SUBROUTINE OUTSTR(A)
  28 >        SUBROUTINE OUTMES(LUNIT,MESSAGE)
```

CONTENTS OF FILE : (ANTENNA-EISCAT)ANTCAL-CONTROL;MODE;1

```
  1 >@FTN
  2 >CO-CO G
  3 >COM (ANTE)ANTCAL-CONTROL,,100
  4 >EX
  5 >@NRL
  6 >L 100,SSPMATR,TEK-GRAF,FTNL
  7 >DUMP (ANTE)ANTCAL-CONTROL
  8 >EX
```

CONTENTS OF FILE : (ANTENNA-EISCAT)ANTCAL-CONTROL;SYMB;1

```
  1 >****   ANTENNA OFFSET ANALYSIS PROGRAMS / MARKKU LEHTINEN 13.12.1979
  2 >****
  3 >****   BACKROUND PROGRAM TO FIT A MODEL TO ANTENNA CALIBRATION DATA
  4 >****   THE MODEL IS A LINEAR COMBINATION OF A SET OF USER INPUT FUNCTIONS.
  5 >****   THE COEFFICIENTS ARE FITTED BY A MEAN SQUARE METHOD, AND DIFFERENT
  6 >****   KINDS OF MAPS CONCERNING THE FIT AND DATA MAY BE MADE WITH LINE
  7 >****   PRINTER.  USE 'HELP' TO GET MORE INFORMATION.
  8 >****
  9 >****   COMPILING:
 10 >****         USE CO-CO G IF YOU WANT A PLOT OF MEASURED DATA ON TEKTRONIX
 11 >****   LOADING:
 12 >****   SUBROUTINE FILES NEEDED:
 13 >****         SSPMATR          (MINV,GMPRD,EIGEN)
 14 >****         TEK-GRAF         (IF CO-CO G)
 15 >****
 16 >      PROGRAM ANTOFF
 17 >C
 18 >100   FORMAT (/,X,
 19 >     * 'INPUT            TO READ DATA FROM CALIBRATION DATA FILE'/X,
 20 >     * 'DATA-MAP         TO GET A MAP OF THE INPUT DATA ON LINE PRINTER'/X,
 21 >     * 'PLOT-DATA        TO GET A MAP OF THE DATA ON TEKTRONIX'/X,
 22 >     * 'MODEL-INPUT      TO GIVE THE MODEL TO FIT'/X,
 23 >     * 'COMPUTE          TO MAKE THE FIT TO THE INPUT DATA'/X,
 24 >     * 'MODEL-MAP        TO GET A MAP OF THE FIT RESULTS'/X,
 25 >     * 'ERROR-MAP        TO GET ERRORS OF THE CALCULATED OFFSETS'/X,
 26 >     * 'OUTPUT-FILE      OPENS A NEW OUTPUT FILE'/X,
 27 >     * 'CALCULATE        TO CALCULATE SINGLE OFFSET VALUES'/X,
 28 >     * 'CLEAN            TO DELETE TOO BAD DATA'/X,
 29 >     * 'WRITE            TO STORE THE STATE OF THIS PROGRAM TO A FILE' X,
 30 >     * 'READ             TO RECOVER THE STATE WRITTEN BY WRITE'/X,
 31 >     * 'INITIALIZE       TO NULL INTERNAL DATA BUFFERS AND MATRICES'/X
 32 >     * 'DEFINE-ANTCAL-INPUT   TO DEFINE INPUT TO DATA GATHERING PGM'/ ,
 33 >     * 'DISPLAY-ANTCAL-INPUT TO DISPLAY THE GATHERING PGM INPUT FILE /X,
 34 >     * 'START-ANTCAL          TO START MEASURING THE DATA'/X,
 35 >     * 'STOP-ANTCAL           TO STOP THE MEASURING PROGRAM'/X,
 36 >     * 'DISPLAY-STATUS        TO SEE, WHAT THE GATHERING PGM IS DOING /X,
 37 >     * 'STORE-FILE-COMMENT    TO WRITE A COMMENT TO STORE FILE'/X,
 38 >     * 'STORE-FILE-HEADING    TO WRITE A HEADING TO STORE FILE'/X,
 39 >     * 'TEST-CAMAC-SAMPLING   TO TEST IF DATA IS SAMPLED CORRECTLY'/X
 40 >     * 'HELP             FOR HELP'/X,
 41 >     * 'EXIT             TO FINISH'/X,
 42 >     * 'COMMENT-ABOUT-THESE-PROGRAMS   TO GIVE A COMMENT TO THE AUTHOR')
 43 >C
 44 >C
 45 >      COMMON/FUNCMOD/IFUNCEL(40),IFUNCAZ(40),NFUNC,CFUNCAZ(20),
 46 >     *                CFUNCEL(20)
 47 >      CHARACTER CFUNCAZ*10,CFUNCEL*10
 48 >      COMMON/FITDATA/AMATR(1600),AINHOMOG(40),COEFF(40)
 49 >      COMMON/OFFSDATA/AZVECT(1000),ELVECT(1000),IDIRVECT(1000),
 50 >     *                OFFVECT(1000),NPOINT,AZOREL
 51 >      CHARACTER AZOREL*9
 52 >      COMMON/ERRORS/ERRMATR(820),ERRVECT(40),EIGVAL(40),EIGVECT(1600)
 53 >      COMMON/SCRATCH/OFFMAT(108,20), OFFNUM(108,20),OFFMAP(108,20)
 54 >      INTEGER OFFMAT,OFFNUM; CHARACTER OFFMAP*1
 55 >      COMMON/TITLE/IS,STAT,MON; CHARACTER STAT(3)*10,MON(12)*10
 56 >      CHARACTER CMND*32,FILE*32,OUTFILE*32
 57 >      INTEGER ITIM(7)
 58 >      COMMON/COMMAND/NCMND,CCMND; CHARACTER CCMND(30)*32
 59 >      COMMON/BUFFER/CBUF,PBUF; CHARACTER CBUF*132; INTEGER PBUF
 60 >      CHARACTER LINE*80
 61 >      DATA NCMND/24/,CCMND/
 62 >     * 'INPUT','MODEL-INPUT','DATA-MAP','PLOT-DATA','COMPUTE',
 63 >     * 'MODEL-MAP','ERROR-MAP','OUTPUT-FILE','CALCULATE',
```

```
64 >      * 'CLEAN',' WRITE',' READ',' INITIALIZE',' HELP',' EXIT',
65 >      * 'DEFINE-ANTCAL-INPUT',' DISPLAY-ANTCAL-INPUT',
66 >      * 'START-ANTCAL',' STOP-ANTCAL',' DISPLAY-STATUS',
67 >      * 'TEST-CAMAC-SAMPLING',
68 >      * 'COMMENT-ABOUT-THESE-PROGRAMS',
69 >      * 'STORE-FILE-HEADING',' STORE-FILE-COMMENT',
70 >      * ' ',' ',' ',' ',' '/
71 >        DATA STAT/' KIRUNA    ',' TROMSOE   ',' SODANKYLA'/
72 >        DATA MON /' JANUARY   ',' FEBRUARY  ',' MARCH     ',' APRIL     ',
73 >      *           ' MAY       ',' JUNE      ',' JULY      ',' AUGUST    ',
74 >      *           ' SEPTEMBER',' OCTOBER   ',' NOVEMBER  ',' DECEMBER  '/
75 >        IS=3
76 >C
77 >101    FORMAT(X,A,10X,12,A,I5,5X,'EISCAT=',A)
78 >C
79 >        CALL CLOCK(ITIM); M=ITIM(6)
80 >        WRITE(1,101) 'ANTENNA OFFSET ANALYSIS',ITIM(5),MON(M),
81 >      *                          ITIM(7),STAT(IS)
82 >        WRITE(1,'(* WRITE HELP IF YOU NEED IT !*)')
83 >        LP=11
84 >        CBUF=' '
85 >        OUTFILE='TERM'
86 >        OPEN(11,FILE=OUTFILE,ACCESS='W')
87 >        CALL INITIALIZE
88 >999     DOWHILE(.TRUE.)
89 >        CBUF=' '
90 >        CALL GETS('COMND : ',CMND,LC,' ')
91 >        CALL FITCMND(CMND,ICMND,IERR)
92 >        IF (IERR.NE.0) THEN
93 >           IF (IERR.EQ.1) THEN
94 >              OUTPUT(1)'NO SUCH COMMAND'
95 >           ELSE
96 >              OUTPUT(1) 'AMBIGUOUS COMMAND'
97 >           ENDIF
98 >        ENDIF
99 >        IF (CMND.EQ.'INPUT') THEN
100 >          CALL INPUT
101 >       ELSE IF (CMND.EQ.'ERROR-MAP') THEN
102 >          CALL MAP5(LP)
103 >          WRITE(11,'(///)')
104 >          CALL ERRANAL
105 >          WRITE(11,'(1H1)')
106 >       ELSE IF (CMND.EQ.'DATA-MAP') THEN
107 >          CALL MAP1(11)
108 >          WRITE(11,'(//)')
109 >          CALL MAP2(11)
110 >          WRITE(11,'(1H1)')
111 >       ELSE IF (CMND.EQ.'MODEL-MAP') THEN
112 >          CALL MAP3(11)
113 >          WRITE(11,'(///)')
114 >          CALL MAP4(11)
115 >          WRITE(11,'(1H1)')
116 >       ELSE IF(CMND.EQ.'CLEAN') THEN
117 >          CALL CLEAN
118 >       ELSE IF(CMND.EQ.'TESTFUN') THEN
119 >          WRITE(1,'(/*$GIVE I,AZ,EL : *)')
120 >          INPUT(1) I,AZ,EL
121 >          X=FUNCMOD(I,AZ,EL,IDIR)
122 >          WRITE(1,'(* FUNCMOD = *,E13.4)') X
123 >       ELSE IF (CMND.EQ.'CALCULATE') THEN
124 >          AZ=1;EL=1
125 >          DOWHILE (AZ.GT.0)
126 >             CALL GETR('AZ : ',AZ,0.)
127 >             CALL GETR('EL : ',EL,0.)
128 >             CALL OFFMODEL(AZ,EL,0,OFF)
129 >             WRITE(1,'(1H+,T30,*OFF = *,E11.3)') OFF
130 >          ENDDO
131 >       ELSE IF (CMND.EQ.'WRITE') THEN
132 >          CALL GETS('FILE NAME : ',FILE,I,'ANT-OFFSET-ANAL:DATA')
133 >          OPEN(12,FILE=FILE,STATUS='UNKNOWN',ACCESS='W')
134 >          WRITE(12) IFUNCEL,IFUNCAZ,NFUNC,CFUNCAZ,CFUNCEL
135 >          WRITE(12) AMATR,AINHOMOG,COEFF
136 >          WRITE(12) AZVECT,ELVECT,IDIRVECT,OFFVECT,NPOINT,AZOREL
137 >          WRITE(12) ERRMATR
138 >          CLOSE(12)
139 >       ELSE IF (CMND.EQ.'READ') THEN
140 >          CALL GETS('FILE NAME : ',FILE,I,'ANT-OFFSET-ANAL:DATA')
141 >          OPEN(12,FILE=FILE,ACCESS='R')
142 >          READ(12) IFUNCEL,IFUNCAZ,NFUNC,CFUNCAZ,CFUNCEL
143 >          READ(12) AMATR,AINHOMOG,COEFF
144 >          READ(12) AZVECT,ELVECT,IDIRVECT,OFFVECT,NPOINT,AZOREL
145 >          READ(12) ERRMATR
```

```
146 >          CLOSE(12)
147 >CG     ELSE IF (CMND.EQ.'PLOT-DATA') THEN
148 >CG        CALL PLOTDATA
149 >        ELSE IF (CMND.EQ.'MODEL-INPUT') THEN
150 >           CALL MODELINP
151 >        ELSE IF (CMND.EQ.'COMPUTE') THEN
152 >           CALL COMPUTE
153 >        ELSE IF (CMND.EQ.'INITIALIZE') THEN
154 >           CALL INITIALIZE
155 >        ELSE IF (CMND.EQ.'OUTPUT-FILE') THEN
156 >           CLOSE(11)
157 >           CALL GETS('OUTPUT FILE : ',OUTFILE,LS,OUTFILE)
158 >           OPEN(11,FILE=OUTFILE,ACCESS='W')
159 >        ELSE IF (CMND.EQ.'DEFINE-ANTCAL-INPUT') THEN
160 >           CALL INPDEFINE
161 >        ELSE IF (CMND.EQ.'DISPLAY-ANTCAL-INPUT') THEN
162 >           CALL GETS('CALIBRATION INPUT FILE : (RT)ANTCAL-IN-',FILE,LS,
163 >     *                              '(RT)ANTCAL-INPUT:DATA')
164 >           IF (FILE.NE.'(RT)ANTCAL-INPUT:DATA') THEN
165 >              FILE=FILE(1:MIN0(LS,6))
166 >              FILE='(RT)ANTCAL-IN-'//FILE(1:-1)//':DATA'
167 >           ENDIF
168 >           WRITE(1,'(= FILE NAME= *,A)') FILE(1:-1)
169 >           OPEN(12,FILE=FILE,ACCESS='R',STATUS='OLD')
170 >           DOWHILE (.TRUE.)
171 >              READ(12,'(A)',ERR=500) LINE
172 >              I=80; DOWHILE(LINE(I:I).EQ.' '.AND.I.GT.1); I=I-1; ENDDO
173 >              OUTPUT(1) LINE(1:I)
174 >           ENDDO
175 >500        CLOSE(12)
176 >        ELSE IF (CMND.EQ.'START-ANTCAL') THEN
177 >           CALL GETS('CALIBRATION INPUT FILE : (RT)ANTCAL-IN-',FILE,LS
178 >     *                          '(RT)ANTCAL-INPUT:DATA')
179 >           IF (FILE.NE.'(RT)ANTCAL-INPUT:DATA') THEN
180 >              FILE=FILE(1:MIN0(LS,6))
181 >              FILE='(RT)ANTCAL-IN-'//FILE(1:-1)//':DATA'
182 >              CALL COMND('COPY '//'(RT)ANTCAL-INPUT:DATA '//FILE//'''')
183 >           ENDIF
184 >           WRITE(1,'(= FILE NAME= *,A)') FILE(1:-1)
185 >           CALL COMND("RT ANTCAL'")
186 >        ELSE IF (CMND.EQ.'STOP-ANTCAL') THEN
187 >           CALL COMND("RT ACSTOP'")
188 >        ELSE IF (CMND.EQ.'DISPLAY-STATUS') THEN
189 >           CALL OPEN5(12,'(RT)ANTCAL-STATUS:DATA','R','OLD',IER)
190 >           IF (IER.NE.0) GOTO 999
191 >           DOWHILE (.TRUE.)
192 >              READ(12,'(A)',ERR=501) LINE
193 >              I=80; DOWHILE(LINE(I:I).EQ.' '.AND.I.GT.1); I=I-1; ENDDO
194 >              OUTPUT(1) LINE(1:I)
195 >           ENDDO
196 >501        CLOSE(12)
197 >        ELSE IF (CMND.EQ.'TEST-CAMAC-SAMPLING') THEN
198 >           CALL CAMACTEST
199 >        ELSE IF (CMND.EQ.'COMMENT-ABOUT-THESE-PROGRAMS') THEN
200 >           OPEN(12,FILE='(RT)ANTCAL-COMMENT:DATA',ACCESS='WA',
201 >     *                          STATUS='UNKNOWN')
202 >           CALL GETS('PLEASE,GIVE YOUR NAME : ',LINE,LS,'NN')
203 >           OUTPUT(1)
204 >           OUTPUT(1) 'WRITE YOUR COMMENT AND TERMINATE WITH CNTRL-L :'
205 >           OUTPUT(1)
206 >           OUTPUT(12)
207 >           WRITE(12,101) 'COMMENT FROM  '//LINE(1:LS)//',     ',
208 >     *                             ITIM(5),MON(M),ITIM(7),STAT(IS)
209 >           I=0
210 >           DOWHILE (I.EQ.0)
211 >              READ(0,'(A)') LINE
212 >              DOFOR J=1,80; IF(ICHAR(LINE(J:J)).EQ.148) I=J; ENDDO; J= -1
213 >              IF (I.NE.0) J=I-1
214 >              IF (J.NE.0) THEN
215 >                 DOWHILE(LINE(J:J).EQ.' '); J=J-1; ENDDO
216 >                 OUTPUT(12) LINE(1:J)
217 >              ENDIF
218 >           ENDDO
219 >           CBUF=''
220 >           OUTPUT(12); CLOSE(12)
221 >        ELSE IF (CMND.EQ.'STORE-FILE-COMMENT') THEN
222 >           CALL COMMOFF
223 >        ELSE IF (CMND.EQ.'STORE-FILE-HEADING') THEN
224 >           CALL TITLOFF
225 >        ELSE IF (CMND.EQ.'EXIT') THEN
226 >           CLOSE(11); STOP
227 >        ELSE IF (CMND.EQ.'HELP') THEN
```

```
228 >           WRITE(1,100)
229 >         ENDIF
230 >         ENDDO
231 >         CLOSE(11)
232 >         END
233 >
234 >
235 >         SUBROUTINE STORE(AZ,EL,IDIR,OFF)
236 >C        ================================
237 >         COMMON/FUNCMOD/IFUNCEL(40),IFUNCAZ(40),NFUNC,CFUNCAZ(20),
238 >     *                              CFUNCEL(20)
239 >         CHARACTER CFUNCAZ*10,CFUNCEL*10
240 >         COMMON/FITDATA/AMATR(1600),AINHOMOG(40),COEFF(40)
241 >         COMMON/OFFSDATA/AZVECT(1000),ELVECT(1000),IDIRVECT(1000),
242 >     *                              OFFVECT(1000),NPOINT,AZOREL
243 >         CHARACTER AZOREL*9
244 >         AZVECT(NPOINT)=AZ
245 >         ELVECT(NPOINT)=EL
246 >         IDIRVECT(NPOINT)=IDIR
247 >         OFFVECT(NPOINT)=OFF
248 >         RETURN
249 >         END
250 >
251 >
252 >         SUBROUTINE COMPUTE
253 >C        ==================
254 >         COMMON/FUNCMOD/IFUNCEL(40),IFUNCAZ(40),NFUNC,CFUNCAZ(20),
255 >     *                              CFUNCEL(20)
256 >         CHARACTER CFUNCAZ*10,CFUNCEL*10
257 >         COMMON/FITDATA/AMATR(1600),AINHOMOG(40),COEFF(40)
258 >         COMMON/OFFSDATA/AZVECT(1000),ELVECT(1000),IDIRVECT(1000),
259 >     *                              OFFVECT(1000),NPOINT,AZOREL
260 >         CHARACTER AZOREL*9
261 >         COMMON/ERRORS/ERRMATR(820),ERRVECT(40)
262 >         CHARACTER LINE*132,FILE*32
263 >         DIMENSION IHIST(101)
264 >         DIMENSION IHELP1(40),IHELP2(40)
265 >C        WRITE(1,'(/'' OUTPUT FILE : '')')
266 >C        INPUT(1) FILE
267 >C        OPEN(11,FILE=FILE,ACCESS='W')
268 >         DOFOR I=1,NPOINT
269 >           AZ=AZVECT(I)
270 >           EL=ELVECT(I)
271 >           IDIR=IDIRVECT(I)
272 >           OFF=OFFVECT(I)
273 >           DOFOR N=1,NFUNC
274 >             AINHOMOG(N)=AINHOMOG(N)+FUNCMOD(N,AZ,EL,IDIR)*OFF
275 >             DOFOR M=N,NFUNC
276 >               AMATR(N+(M-1)*NFUNC)=FUNCMOD(N,AZ,EL,IDIR)*
277 >     *                 FUNCMOD(M,AZ,EL,IDIR)+AMATR(N+(M-1)*NFUNC)
278 >             ENDDO
279 >           ENDDO
280 >         ENDDO
281 >         DOFOR N=1,NFUNC; DOFOR M=1,N
282 >           AMATR(N+(M-1)*NFUNC)=AMATR(M+(N-1)*NFUNC)
283 >         ENDDO; ENDDO
284 >         DOFOR I=1,NFUNC; COEFF(I)=AINHOMOG(I); ENDDO
285 >         CALL MINV(AMATR,NFUNC,DET,IHELP1,IHELP2)
286 >         CALL GMPRD(AMATR,AINHOMOG,COEFF,NFUNC,NFUNC,1)
287 >         WRITE(1,'(' DET=',E9.2)') DET
288 >         CALL ERRINDIAG
289 >         DOFOR I=1,NFUNC
290 >             ERRVECT(I)=SQRT(ERRMATR(I*(I-1)/2+I))
291 >         ENDDO
292 >         AMSQR=0; ERRMAX=0; DOFOR I=1,101; IHIST(I)=0;ENDDO
293 >         DOFOR I=1,NPOINT
294 >               CALL OFFMODEL (AZVECT(I),ELVECT(I),IDIRVECT(I),OFF)
295 >               ERR=OFFVECT(I)-OFF
296 >               AMSQR=ERR**2+AMSQR
297 >               ERRMAX=AMAX1(ABS(ERR),ERRMAX)
298 >               J=100*ERR+.5+51
299 >               IF(J.GT.101) J=101; IF(J.LT.1) J=1
300 >               IHIST(J)=IHIST(J)+1
301 >         ENDDO
302 >         MAXHIST=0; DOFOR I=1,101;MAXHIST=MAXO(IHIST(I),MAXHIST);ENDDO
303 >         AMSQR=SQRT(AMSQR/NPOINT)
304 >         WRITE(11,'(X,A,A,I4,A,F6.3,A,F6.3)')
305 >     * AZOREL,' OFFSET MODEL FIT, ',NPOINT,' POINTS   MSQR ERR =',
306 >     * AMSQR,'  MAX ERR =',ERRMAX
307 >         WRITE(11,'(/99(/X,I2,2X,A10,1H=,A10,2H =,F8.3,=  +-=,F5.3))')
308 >     * ((I,CFUNCAZ(IFUNCAZ(I)),CFUNCEL(IFUNCEL(I)),COEFF(I),
309 >     *                              ERRVECT(I)),I=1,NFUNC)
```

```
310 >        IF (NFUNC.GT.30) WRITE(11,'(1H1)')
311 >        WRITE(11,'(6X,* ERROR DISTRIBUTION*)')
312 >        MULT=MAXHIST/30+1
313 >        DOFOR II=1,30; I=(31-II)*MULT
314 >           WRITE(LINE,'(X,5X,I3,1H:)') I
315 >           DOFOR J=1,101
316 >              IF (IHIST(J).GE.I) THEN
317 >                 LINE(10+J:10+J)='*'
318 >              ELSE IF (IHIST(J).GT.I-MULT) THEN
319 >                 LINE(J+10:J+10)=CHAR(60B+IHIST(J)-(I-MULT))
320 >              ELSE
321 >                 LINE(J+10:J+10)=' '
322 >              ENDIF
323 >           ENDDO
324 >           WRITE(11,'(A111)') LINE
325 >        ENDDO
326 >        WRITE(11,'(X,T11,10(*I----+----*),1HI,*  MEAS.OFFST-*)')
327 >        WRITE(11,'(T9,10(F4.1,6X),F4.1,*  MODEL OFFST*)')
328 >     *                          (((I-6)/10.),I=1,11)
329 >        WRITE(11,'(1H1)')
330 >C       CLOSE(11)
331 >        RETURN
332 >        END
333 >
334 >
335 >        SUBROUTINE INITIALIZE
336 >C       ====================
337 >        COMMON/FUNCMOD/IFUNCEL(40),IFUNCAZ(40),NFUNC,CFUNCAZ(20),
338 >     *                          CFUNCEL(20)
339 >        CHARACTER CFUNCAZ*10,CFUNCEL*10
340 >        COMMON/FITDATA/AMATR(1600),AINHOMOG(40),COEFF(40)
341 >        COMMON/OFFSDATA/AZVECT(1000),ELVECT(1000),IDIRVECT(1000),
342 >     *                          OFFVECT(1000),NPOINT,AZOREL
343 >        CHARACTER AZOREL*9
344 >        COMMON/ERRORS/ERRMATR(820)
345 >        COMMON/TITLE/IS,STAT,MON; CHARACTER STAT(3)*10,MON(12)*10
346 >        DIMENSION ITIM(7)
347 >C
348 >101     FORMAT(X,A,10X,I2,A,I5,5X,*EISCAT*,A)
349 >C
350 >        CALL CLOCK(ITIM); M=ITIM(6)
351 >        WRITE(11,101) 'ANTENNA OFFSET ANALYSIS',ITIM(5),MON(M),
352 >     *                          ITIM(7),STAT(IS)
353 >        WRITE(11,'(1H ,30(1H*),*   INITIALIZED  *,30(1H*)/)')
354 >        DOFOR I=1,1600; AMATR(I)=0; ENDDO
355 >        DOFOR I=1,40; AINHOMOG(I)=0; ENDDO
356 >        DOFOR I=1,820; ERRMATR(I)=0; ENDDO
357 >        RETURN
358 >        END
359 >
360 >
361 >        SUBROUTINE OFFMODEL(AZ,EL,IDIR,OFF)
362 >C       ==========================
363 >        COMMON/FUNCMOD/IFUNCEL(40),IFUNCAZ(40),NFUNC,CFUNCAZ(20),
364 >     *                          CFUNCEL(20)
365 >        CHARACTER CFUNCAZ*10,CFUNCEL*10
366 >        COMMON/FITDATA/AMATR(1600),AINHOMOG(40),COEFF(40)
367 >        COMMON/OFFSDATA/AZVECT(1000),ELVECT(1000),IDIRVECT(1000),
368 >     *                          OFFVECT(1000),NPOINT,AZOREL
369 >        CHARACTER AZOREL*9
370 >        OFF=0
371 >        DOFOR I=1,NFUNC
372 >           OFF=OFF+COEFF(I)*FUNCMOD(I,AZ,EL,IDIR)
373 >        ENDDO
374 >        RETURN
375 >        END
376 >
377 >
378 >        FUNCTION FUNCMOD(I,AZ,EL,IDIR)
379 >C       =========================
380 >        COMMON/FUNCMOD/IFUNCEL(40),IFUNCAZ(40),NFUNC,CFUNCAZ(20),
381 >     *                          CFUNCEL(20)
382 >        CHARACTER CFUNCAZ*10,CFUNCEL*10
383 >        AZ1=AZ*3.14159265/180.; EL1=EL*3.14159265/180.
384 >        GOTO (1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20),
385 >     *                          IFUNCEL(I)
386 >1       FUNCEL=1; GOTO 98
387 >2       FUNCEL=EL1; GOTO 98
388 >3       FUNCEL=EL1**2; GOTO 98
389 >4       FUNCEL=EL1**3; GOTO 98
390 >5       FUNCEL=EL1**4; GOTO 98
391 >6       FUNCEL=EL1**5; GOTO 98
```

```
392 >7      FUNCEL=EL1**6; GOTO 98
393 >8      FUNCEL=EL1**7; GOTO 98
394 >9      FUNCEL=EL1**8; GOTO 98
395 >10     FUNCEL=COS(EL1)/SIN(EL1); GOTO 98
396 >11     FUNCEL=SIN(EL1)/COS(EL1); GOTO 98
397 >12     FUNCEL=(SIN(EL1)/COS(EL1))**2; GOTO 98
398 >13     FUNCEL=(SIN(EL1)/COS(EL1))**3; GOTO 98
399 >14     FUNCEL=SIN(EL1); GOTO 98
400 >15     FUNCEL=SIN(EL1*2); GOTO 98
401 >16     FUNCEL=SIN(EL1*3); GOTO 98
402 >17     FUNCEL=COS(EL1); GOTO 98
403 >18     FUNCEL=COS(EL1*2); GOTO 98
404 >19     FUNCEL=COS(EL1*3); GOTO 98
405 >20     FUNCEL=1/COS(EL1); GOTO 98
406 >98     GOTO(21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,
407 >    *       40), IFUNCAZ(I)
408 >21     FUNCAZ=1; GOTO 99
409 >22     FUNCAZ=COS(AZ1); GOTO 99
410 >23     FUNCAZ=SIN(AZ1); GOTO 99
411 >24     FUNCAZ=COS(2*AZ1); GOTO 99
412 >25     FUNCAZ=SIN(2*AZ1); GOTO 99
413 >26     FUNCAZ=COS(3*AZ1); GOTO 99
414 >27     FUNCAZ=SIN(3*AZ1); GOTO 99
415 >28     FUNCAZ=COS(4*AZ1); GOTO 99
416 >29     FUNCAZ=SIN(4*AZ1); GOTO 99
417 >30     FUNCAZ=COS(5*AZ1); GOTO 99
418 >31     FUNCAZ=SIN(5*AZ1); GOTO 99
419 >32     FUNCAZ=COS(6*AZ1); GOTO 99
420 >33     FUNCAZ=SIN(6*AZ1); GOTO 99
421 >34     FUNCAZ=COS(7*AZ1); GOTO 99
422 >35     FUNCAZ=SIN(7*AZ1); GOTO 99
423 >36     FUNCAZ=COS(8*AZ1); GOTO 99
424 >37     FUNCAZ=SIN(8*AZ1); GOTO 99
425 >38     FUNCAZ=COS(9*AZ1); GOTO 99
426 >39     FUNCAZ=SIN(9*AZ1); GOTO 99
427 >40     FUNCAZ=IDIR/2.; GOTO 99
428 >99     FUNCMOD=FUNCAZ*FUNCEL
429 >      RETURN
430 >      END
431 >
432 >
433 >      SUBROUTINE INPUT
434 >C     ================
435 >      COMMON/OFFSDATA/AZVECT(1000),ELVECT(1000),IDIRVECT(1000),
436 >    *                 OFFVECT(1000),NPOINT,AZOREL
437 >      CHARACTER AZOREL*9
438 >      CHARACTER CMND*32,FILE*32,OUTFILE*32
439 >      CHARACTER DIR*1,DIR1*2,FIT*2,CODE*1,LINE*132
440 >      INTEGER Y,M,D,Y1,M1,D1,Y2,M2,D2
441 >      CALL GETS('INPUT FILE : ',FILE,I,'(RT)ANTCAL-DATA:DATA')
442 >      OPEN(10,FILE=FILE,ACCESS='R',ERR=400)
443 >      GOTO 401
444 >400   CALL HOLD(10,2)
445 >      OPEN(10,FILE=FILE,ACCESS='R')
446 >401   FIT=''
447 >      DOWHILE (FIT.NE.'AZ'.AND.FIT.NE.'EL')
448 >         CALL GETS('AZ OR EL OFFST FIT ? (AZ/EL) : ',FIT,I,AZOREL(1:2))
449 >      ENDDO
450 >      IF (FIT.EQ.'AZ') THEN
451 >         AZOREL='AZIMUTH'
452 >      ELSE
453 >         AZOREL='ELEVATION'
454 >      ENDIF
455 >      CALL GETI('STARTING DATE (Y,M,D,HH.MM) : ',Y1,0)
456 >      CALL GETI('STARTING MONTH : ',M1,0)
457 >      CALL GETI('STARTING DAY : ',D1,0)
458 >      CALL GETR('STARTING TIME (UT  HH.MM) : ',UT1,0.)
459 >      CALL GETI('ENDING DATE (Y,M,D,HH.MM) : ',Y2,1999)
460 >      CALL GETI('ENDING MONTH : ',M2,0)
461 >      CALL GETI('ENDING DAY : ',D2,0)
462 >      CALL GETR('ENDING TIME (UT  HH.MM) : ',UT2,0.)
463 >      DATE1=1000000*(Y1-1900)+10000*M1+100*D1+UT1
464 >      DATE2=1000000*(Y2-1900)+10000*M2+100*D2+UT2
465 >      CALL GETI('MAXIMUM NO ITERATIONS : ',MAXITER,3)
466 >      CALL GETR('MAXIMUM LAST CORRECTION : ',CORRMAX,.1)
467 >      DIR1=''
468 >      DOWHILE (DIR1.NE.'+'.AND.DIR1.NE.'-'.AND.DIR1.NE.'+-')
469 >         CALL GETS('SWEEP DIRECTION (+ / - / +-) : ',DIR1,I,'+-')
470 >      ENDDO
471 >      WRITE(11,101) FIT,FILE,Y1,M1,D1,UT1,Y2,M2,D2,UT2,MAXITER,CORRMAX,
472 >    *                 DIR1
473 >101   FORMAT(X,'INPUT ',A2,'DATA FROM FILE : ',A32,/,
```

```
474 >      *       * STARTING DATE *,I4,X,J2,X,J2,X,F5.2,
475 >      *       *    ENDING DATE *,I4,X,J2,X,J2,X,F5.2,
476 >      *       *    MAXITER *,I1,*   MAX LAST CORR *,F6.3,*  DIR*,A2/)
477 >         NPOINT=0
478 >         DOWHILE(.TRUE.)
479 >            NERR=0
480 >            ERRCODE=1
481 >            DOWHILE(ERRCODE.NE.0)
482 >               READ(10,'(A)',ERR=888) LINE
483 >888            NERR=NERR+1
484 >               IF(NERR.GT.5)GOTO 999
485 >            ENDDO
486 >            CODE=LINE(2:2)
487 >            IF (CODE.EQ.' ') THEN
488 >               READ(LINE,100) Y,M,D,UT,AZAZ,ELAZ,AZOFF,AZEL,ELEL,ELOFF,
489 >      *                       STAR,DIR,ITER,AZLAST,ELLAST
490 >   100        FORMAT(2X,I5,2(1X,I2),F7.2,T24,F6.2,T32,F5.2,
491 >      *       T39,F6.3,T48,F6.2,T56,F5.2,T63,F6.3,T72,A6,T80,A1,
492 >      *       T94,I1,T88,F6.3,T95,F6.3)
493 >CT            WRITE(1,200) Y,M,D,UT,AZAZ,ELAZ,AZOFF,AZEL,ELEL,ELOFF,
494 >CT     *                   STAR,DIR,ITER
495 >CT200         FORMAT(2X,I5,2(1X,I2),F7.2,4X,F6.2,2X,F5.2,
496 >CT     *       2X,F6.3,2X,F6.2,2X,F5.2,2X,F6.3,2X,A6,2X,A1,2X,A1)
497 >               DATE=1000000*(Y-1900)+10000*M+100*D+UT
498 >               IF (DATE.LE.DATE2.AND.DATE.GE.DATE1.AND.
499 >      *           ITER.LE.MAXITER.AND.
500 >      *           ABS(AZLAST).LE.CORRMAX.AND.ABS(ELLAST).LE.CORRMAX.
501 >      *           AND.(DIR.EQ.DIR1(1:1).OR.DIR.EQ.DIR1(2:2)))THEN
502 >                  NPOINT=NPOINT+1
503 >                  IF (NPOINT.GT.1000) THEN
504 >                     OUTPUT(1) 'DATA BUFFER FULL'; NPOINT=NPOINT-1
505 >                     CLOSE(10)
506 >                     RETURN
507 >                  ENDIF
508 >                  IF (FIT.EQ.'AZ')THEN
509 >                           AZ=AZAZ; EL=ELAZ; OFF=AZOFF
510 >                  ELSE IF (FIT.EQ.'EL') THEN
511 >                           AZ=AZEL;EL=ELEL; OFF=ELOFF
512 >                  ELSE; STOP; ENDIF
513 >                  IF(DIR.EQ.'+')THEN;IDIR=1;ELSE;IDIR=-1;ENDIF
514 >                  CALL STORE(AZ,EL,IDIR,OFF)
515 >               ENDIF
516 >            ENDIF
517 >         ENDDO
518 >999      CONTINUE
519 >CT       WRITE(1,'(99(X,F7.3,2X,F6.3,4X,F6.3,/)')
520 >CT     *       ((AZVECT(I),ELVECT(I),OFFVECT(I)),I=1,NPOINT)
521 >         WRITE(1,'(X,I4,*  POINTS READ*)') NPOINT
522 >         CLOSE(10)
523 >         RETURN
524 >         END
525 >
526 >
527 >         SUBROUTINE CLEAN
528 >C        ================
529 >         COMMON/FUNCMOD/IFUNCEL(40),IFUNCAZ(40),NFUNC,CFUNCAZ(20),
530 >      *                           CFUNCEL(20)
531 >         CHARACTER CFUNCAZ*10,CFUNCEL*10
532 >         COMMON/FITDATA/AMATR(1600),AINHOMOG(40),COEFF(40)
533 >         COMMON/OFFSDATA/AZVECT(1000),ELVECT(1000),IDIRVECT(1000),
534 >      *                           OFFVECT(1000),NPOINT,AZOREL
535 >         CHARACTER AZOREL*9
536 >         CALL GETR('MAXIMUM DEVIATION FROM MODEL LAST FITTED : ',
537 >      *                           ERRMAX,1)
538 >         WRITE(11,'(X,*DATA CLEAN, ERRMAX=*,F6.3,*,  POINTS DELETED :=)'
539 >      *       ERRMAX
540 >         I=0; LIMUP=NPOINT
541 >         DOWHILE(I.LT.LIMUP)
542 >            I=I+1
543 >            CALL OFFMODEL(AZVECT(I),ELVECT(I),IDIRVECT(I),OFF)
544 >            ERR=ABS(OFFVECT(I)-OFF)
545 >            IF (ERR.GT.ERRMAX) THEN
546 >               WRITE(11,'(X,*AZ=*,F6.2,A,F5.2,3(2X,A,F6.3))')
547 >      *             AZVECT(I),'  EL=',ELVECT(I),'OFF.MOD=',OFF,
548 >      *             'OFF.MEAS=',OFFVECT(I),'DIFF=',OFF-OFFVECT(I)
549 >               AZVECT(I)=AZVECT(LIMUP)
550 >               ELVECT(I)=ELVECT(LIMUP)
551 >               IDIRVECT(I)=IDIRVECT(LIMUP)
552 >               OFFVECT(I)=OFFVECT(LIMUP)
553 >               I=I-1; LIMUP=LIMUP-1
554 >            ENDIF
555 >         ENDDO
```

```
556 >        IF (NPOINT-LIMUP.GT.5) WRITE(11,'(1H1)')
557 >        NPOINT=LIMUP
558 >        RETURN
559 >        END
560 >
561 >
562 >
563 >        SUBROUTINE MODELINP
564 >C       ======================
565 >        COMMON/FUNCMOD/IFUNCEL(40),IFUNCAZ(40),NFUNC,CFUNCAZ(20),
566 >     *                          CFUNCEL(20)
567 >        COMMON/OFFSDATA/AZVECT(1000),ELVECT(1000),IDIRVECT(1000),
568 >     *                          OFFVECT(1000),NPOINT,AZOREL
569 >        CHARACTER AZOREL*9
570 >        CHARACTER CHELP*10
571 >        CHARACTER CFUNCAZ*10,CFUNCEL*10
572 >        DATA CFUNCEL/
573 >     *    '1',
574 >     *    'EL',
575 >     *    'EL**2',
576 >     *    'EL**3',
577 >     *    'EL**4',
578 >     *    'EL**5',
579 >     *    'EL**6',
580 >     *    'EL**7',
581 >     *    'EL**8',
582 >     *    'COT(EL)',
583 >     *    'TAN(EL)',
584 >     *    'TAN(EL)**2',
585 >     *    'TAN(EL)**3',
586 >     *    'SIN(EL)',
587 >     *    'SIN(2*EL)',
588 >     *    'SIN(3*EL)',
589 >     *    'COS(EL)',
590 >     *    'COS(2*EL)',
591 >     *    'COS(3*EL)',
592 >     *    '1/COS(EL)'/
593 >        DATA CFUNCAZ/
594 >     *    '1',
595 >     *    'COS(AZ)',
596 >     *    'SIN(AZ)',
597 >     *    'COS(2*AZ)',
598 >     *    'SIN(2*AZ)',
599 >     *    'COS(3*AZ)',
600 >     *    'SIN(3*AZ)',
601 >     *    'COS(4*AZ)',
602 >     *    'SIN(4*AZ)',
603 >     *    'COS(5*AZ)',
604 >     *    'SIN(5*AZ)',
605 >     *    'COS(6*AZ)',
606 >     *    'SIN(6*AZ)',
607 >     *    'COS(7*AZ)',
608 >     *    'SIN(7*AZ)',
609 >     *    'COS(8*AZ)',
610 >     *    'SIN(8*AZ)',
611 >     *    'COS(9*AZ)',
612 >     *    'SIN(9*AZ)',
613 >     *    'SW,DIR/2'/
614 >        WRITE(1,'(10(X,I2,X,A10,3X,I2,X,A10,6X,I2,X,A10,3X,I2,X,A10/))')
615 >     * ((I,CFUNCAZ(I),I+10,CFUNCAZ(I+10),I,CFUNCEL(I),I+10,
616 >     *       CFUNCEL(I+10)),I=1,10)
617 >        NFUNC=0
618 >101     DOWHILE(.TRUE.)
619 >           CALL GETS('GIVE IAZ1,IAZ2,IEL1,IEL2 (OR "DEF"): ',CHELP,I,'0')
620 >           IF (CHELP(1:1).EQ.'D'.AND.AZOREL(1:2).EQ.'AZ') THEN
621 >              IFUNCAZ(1)=1 ;IFUNCEL(1)=1
622 >              IFUNCAZ(2)=1 ;IFUNCEL(2)=20
623 >              IFUNCAZ(3)=1 ;IFUNCEL(3)=11
624 >              IFUNCAZ(4)=2 ;IFUNCEL(4)=11
625 >              IFUNCAZ(5)=3 ;IFUNCEL(5)=11
626 >              IFUNCAZ(6)=2 ;IFUNCEL(6)=1
627 >              IFUNCAZ(7)=3 ;IFUNCEL(7)=1
628 >              IFUNCAZ(8)=4 ;IFUNCEL(8)=1
629 >              IFUNCAZ(9)=5 ;IFUNCEL(9)=1
630 >              NFUNC=9
631 >              WRITE(1,'(999(X,A,I2,*) = *,A10,1H*,A10,/))')
632 >     *        ((I,'NFUNC(',I,CFUNCAZ(IFUNCAZ(I)),CFUNCEL(IFUNCEL(I))),
633 >     *                          I=1,NFUNC)
634 >              GOTO 101
635 >           ELSE IF (CHELP(1:1).EQ.'D'.AND.AZOREL(1:2).EQ.'EL') THEN
636 >              IFUNCAZ(1)=1 ;IFUNCEL(1)=1
637 >              IFUNCAZ(2)=2 ;IFUNCEL(2)=1
```

```
638 >              IFUNCAZ(3)=3 ;IFUNCEL(3)=1
639 >              IFUNCAZ(4)=1 ;IFUNCEL(4)=17
640 >              IFUNCAZ(5)=1 ;IFUNCEL(5)=14
641 >              IFUNCAZ(6)=1 ;IFUNCEL(6)=10
642 >              NFUNC=6
643 >              WRITE(1,'(999(X,A,I2,*) = *,A10,1H*,A10,/))')
644 >     *        (('NFUNC(',I,CFUNCAZ(IFUNCAZ(I)),CFUNCEL(IFUNCEL(I))),
645 >     *                          I=1,NFUNC)
646 >              GOTO 101
647 >           ELSE
648 >              IF (CHELP.EQ.'') THEN; I=1; CHELP='0'; ENDIF
649 >              CHELP(I+1:I+1)=','; INPUT(CHELP(1:I+1)) IAZ1
650 >           ENDIF
651 >           IF (IAZ1.EQ.0.AND.NFUNC.EQ.0) THEN
652 >              OUTPUT(1)'THE IDEA IS TO CHOOSE A COMBINATION OF CROSS '
653 >              OUTPUT(1)'PRODUCTS BETWEEN THE AZ AND EL FUNCTIONS LISTED.'
654 >              OUTPUT(1)'YOU MUST SPECIFY TWO INTERVALS IAZ1...IAZ2 AND '
655 >             OUTPUT(1)'IEL1...IEL2 AND THEN ALL THE PRODUCTS BETWEEN THE'
656 >              OUTPUT(1)'CORRESPONDING AZ AND EL FUNCTIONS ARE ADDED TO'
657 >              OUTPUT(1)'THE LIST OF FUNCTIONS TO BE FITTED.'
658 >              OUTPUT(1)'   TO GET DEFAULT CHOISE OF FUNCTIONS ANSWER '
659 >              OUTPUT(1)'"DEF" TO THE QUESTION.'
660 >              OUTPUT(1)'   EXIT BY ENTERING A RETURN'
661 >              GOTO 101
662 >           ELSE IF (IAZ1.EQ.0) THEN
663 >              GOTO 100
664 >           ENDIF
665 >           CALL GETI('IAZ2 :',IAZ2,0)
666 >           CALL GETI('IEL1 :',IEL1,0)
667 >           CALL GETI('IEL2 :',IEL2,0)
668 >           IF (IAZ1*IAZ2*IEL1*IEL2.EQ.0) GOTO 100
669 >           DOFOR IAZ=IAZ1,IAZ2; DOFOR IEL=IEL1,IEL2
670 >              NFUNC=NFUNC+1
671 >              IFUNCAZ(NFUNC)=IAZ
672 >              IFUNCEL(NFUNC)=IEL
673 >           ENDDO; ENDDO
674 >        ENDDO
675 >100     WRITE(1,'(999(X,A,I2,*) = *,A10,1H*,A10,/))')
676 >     * (('NFUNC(',I,CFUNCAZ(IFUNCAZ(I)),CFUNCEL(IFUNCEL(I))),I=1,NFUNC)
677 >        RETURN
678 >        END
679 >
680 >
681 >CG     SUBROUTINE PLOTDATA
682 >C      ===================
683 >CG     COMMON/OFFSDATA/AZVECT(1000),ELVECT(1000),IDIRVECT(1000),
684 >CG     *                          OFFVECT(1000),NPOINT,AZOREL
685 >CG     CHARACTER AZOREL*9
686 >CG     CHARACTER DUMMY*1
687 >CG     IX1=30; IX2=1000; IY1=500;IY2=750
688 >CG     CALL WIPE
689 >CG     CALL DRAW(IX1,IY1,IX2,IY1)
690 >CG     CALL DRAW(IX1,IY2,IX2,IY2)
691 >CG     CALL DRAW(IX1,IY1,IX1,IY2)
692 >CG     CALL DRAW(IX2,IY1,IX2,IY2)
693 >CG     CALL MOVEC(IX1,IY1-15)
694 >CG     WRITE(1,'(* 90      AZIMUTH *)')
695 >CG     CALL MOVEC(IX2-50,IY1-15)
696 >CG     WRITE(1,'(* 630*)')
697 >CG     CALL MOVEC((IX1+IX2)/2-15,IY1-15)
698 >CG     WRITE(1,'(* 360*)')
699 >CG     CALL MOVEC(IX1-30,IY1)
700 >CG     WRITE(1,'(* 0*)')
701 >CG     CALL MOVEC(IX1-30,IY2)
702 >CG     WRITE(1,'(* 90*)')
703 >CG     CALL MOVEC(IX1-30,IY2+25)
704 >CG     WRITE(1,'(* EL*)')
705 >CG     DOFOR I=1,NPOINT
706 >CG        IX=(AZVECT(I)-90)/540.*(IX2-IX1)+IX1
707 >CG        IY=(ELVECT(I)-0)/90.*(IY2-IY1)+IY1
708 >CG        CALL POINT(IX,IY)
709 >CG     ENDDO
710 >CG     CALL MOVEC(0,IY1-50)
711 >CG     WRITE(1,'(X,A9,A,I4,A)') AZOREL,' OFFSET DATAPOINT DISTRIBUTION,'
712 >CG     *                          ,NPOINT,' POINTS'
713 >CG     RETURN
714 >CG     END
715 >
716 >        SUBROUTINE MAP1(LP)
717 >C       ===================
718 >        CHARACTER AZOREL*9,TITLE*50,C*1
719 >        COMMON/OFFSDATA/AZVECT(1000),ELVECT(1000),IDIRVECT(1000),
```

```
720 >      *                          OFFVECT(1000),NPOINT,AZOREL
721 >        COMMON/SCRATCH/OFFMAT(108,20), OFFNUM(108,20),OFFMAP(108,20)
722 >        INTEGER OFFMAT,OFFNUM; CHARACTER OFFMAP*1
723 >        DO FOR I=1,108
724 >           DO FOR J=1,20
725 >              OFFMAT(I,J)=0
726 >              OFFNUM(I,J)=0
727 >              OFFMAP(I,J)=' '
728 >           ENDDO
729 >        ENDDO
730 >        TITLE=''
731 >        DO FOR I=1,NPOINT
732 >           IAZ=INT((AZVECT(I)-90.)/5.)+1
733 >           IEL=INT(ELVECT(I)/5.)+1
734 >           OFFMAT(IAZ,IEL)=OFFMAT(IAZ,IEL)+OFFVECT(I)*1000.
735 >           OFFNUM(IAZ,IEL)=OFFNUM(IAZ,IEL)+1
736 >        ENDDO
737 >        DO FOR I=1,108
738 >           DO FOR J=1,20
739 >              IF (OFFNUM(I,J).NE.0) THEN
740 >                 CALL SYMBOL(OFFMAT(I,J)/OFFNUM(I,J)/100.,C)
741 >              ELSE; C=' ';ENDIF
742 >              OFFMAP(I,J)=C
743 >           ENDDO
744 >        ENDDO
745 >        TITLE=AZOREL//' OFFSETS, MEASURED*10'
746 >        CALL PRINTOFF(LP,OFFMAP,TITLE)
747 >        RETURN
748 >        END
749 >C
750 >        SUBROUTINE MAP2(LP)
751 >C       ===================
752 >        CHARACTER AZOREL*9,TITLE*50,C*1
753 >        COMMON/OFFSDATA/AZVECT(1000),ELVECT(1000),IDIRVECT(1000),
754 >      *                          OFFVECT(1000),NPOINT,AZOREL
755 >        COMMON/SCRATCH/OFFMAT(108,20), OFFNUM(108,20),OFFMAP(108,20)
756 >        INTEGER OFFMAT,OFFNUM; CHARACTER OFFMAP*1
757 >        DO FOR I=1,108
758 >           DO FOR J=1,20
759 >              OFFNUM(I,J)=0
760 >              OFFMAP(I,J)=' '
761 >           ENDDO
762 >        ENDDO
763 >        DO FOR I=1,NPOINT
764 >           IAZ=INT((AZVECT(I)-90.)/5.)+1
765 >           IEL=INT(ELVECT(I)/5.)+1
766 >           OFFNUM(IAZ,IEL)=OFFNUM(IAZ,IEL)+1
767 >           CALL SYMBOL(OFFNUM(IAZ,IEL)*1.,C)
768 >           OFFMAP(IAZ,IEL)=C
769 >        ENDDO
770 >        WRITE(TITLE,'(*NUMBER OF OFFSETS MEASURED, TOTAL=*,I4)') NPOINT
771 >        CALL PRINTOFF(LP,OFFMAP,TITLE)
772 >        RETURN
773 >        END
774 >C
775 >        SUBROUTINE MAP3(LP)
776 >C       ===================
777 >        CHARACTER AZOREL*9,TITLE*50,C*1
778 >        COMMON/OFFSDATA/AZVECT(1000),ELVECT(1000),IDIRVECT(1000),
779 >      *                          OFFVECT(1000),NPOINT,AZOREL
780 >        COMMON/SCRATCH/OFFMAT(108,20), OFFNUM(108,20),OFFMAP(108,20)
781 >        INTEGER OFFMAT,OFFNUM; CHARACTER OFFMAP*1
782 >        DO FOR I=1,108
783 >           DO FOR J=1,20
784 >              AZ=90.+(I-1)*5.+2.5
785 >              EL=(J-1)*5.+2.5
786 >              CALL OFFMODEL(AZ,EL,0,OFF)
787 >              CALL SYMBOL(OFF*10.,C)
788 >              OFFMAP(I,J)=C
789 >           ENDDO
790 >        ENDDO
791 >        TITLE=AZOREL//' OFFSETS, MODEL*10'
792 >        CALL PRINTOFF(LP,OFFMAP,TITLE)
793 >        RETURN
794 >        END
795 >C
796 >        SUBROUTINE MAP4(LP)
797 >C       ===================
798 >        CHARACTER AZOREL*9,TITLE*50,C*1
799 >        COMMON/OFFSDATA/AZVECT(1000),ELVECT(1000),IDIRVECT(1000),
800 >      *                          OFFVECT(1000),NPOINT,AZOREL
801 >        COMMON/SCRATCH/OFFMAT(108,20), OFFNUM(108,20),OFFMAP(108,20)
```

```
      INTEGER OFFMAT,OFFNUM; CHARACTER OFFMAP*1
      DO FOR I=1,108
         DO FOR J=1,20
            OFFMAT(I,J)=0
            OFFNUM(I,J)=0
            OFFMAP(I,J)=' '
         ENDDO
      ENDDO
      DO FOR I=1,NPOINT
         IAZ=INT((AZVECT(I)-90.)/5.)+1
         IEL=INT(ELVECT(I)/5.)+1
         OFFMAT(IAZ,IEL)=OFFMAT(IAZ,IEL)+OFFVECT(I)*1000.
         OFFNUM(IAZ,IEL)=OFFNUM(IAZ,IEL)+1
      ENDDO
      DO FOR I=1,108
         DO FOR J=1,20
            IF(OFFNUM(I,J).NE.0) THEN
               AZ=92.5+(I-1)*5.
               EL=2.5+(J-1)*5.
               CALL OFFMODEL(AZ,EL,0,OFF)
               CALL SYMBOL((OFFMAT(I,J)/OFFNUM(I,J))/50.-OFF*20.,C)
               OFFMAP(I,J)=C
            ELSE
               OFFMAP(I,J)=' '
            ENDIF
         ENDDO
      ENDDO
      TITLE=AZOREL//' OFFSETS, (MEASURED-MODEL)*20'
      CALL PRINTOFF(LP,OFFMAP,TITLE)
      RETURN
      END
C
C
      SUBROUTINE MAP5(LP)
C     ===================
      CHARACTER AZOREL*9,TITLE*50,C*1
      COMMON/OFFSDATA/AZVECT(1000),ELVECT(1000),IDIRVECT(1000),
     *                OFFVECT(1000),NPOINT,AZOREL
      COMMON/SCRATCH/OFFMAT(108,20), OFFNUM(108,20),ERRMAP(108,20)
      INTEGER OFFMAT,OFFNUM; CHARACTER ERRMAP*1
      CALL ERRINTRI
      DO FOR I=1,108
         DO FOR J=1,20
            AZ=90.+(I-1)*5.+2.5
            EL=(J-1)*5.+2.5
            CALL ERROR(AZ,EL,0,ERR)
            IF (AZOREL(1:1).EQ.'A') ERR=ERR*COS(EL*3.14159265/180.)
            CALL SYMBOL(ERR*100.,C)
            ERRMAP(I,J)=C
         ENDDO
      ENDDO
      IF (AZOREL(1:1).EQ.'A') THEN
         TITLE='ESTIMATED AZIMUTH OFFSET ERRORS*100*COS(EL)'
      ELSE
         TITLE='ESTIMATED ELEVATION OFFSET ERRORS*100'
      ENDIF
      CALL PRINTOFF(LP,ERRMAP,TITLE)
      RETURN
      END
C
      SUBROUTINE PRINTOFF(LP,OFFMAP,TITLE)
C     ====================================
      CHARACTER*1 OFFMAP(108,20),TITLE
      INTEGER ITIM(7)
      CHARACTER LINE*132
C
100   FORMAT(//,10X,A,5X,I2,A,I5,5X,'EISCAT',A,//)
110   FORMAT(8X,'AZ ',19(I3,3X))
120   FORMAT(5X,'EL',6X,109('*'))
C
      COMMON/TITLE/IS,STAT,MON; CHARACTER STAT(3)*10,MON(12)*10
      DO FOR I=1,132; LINE(I:I)=' '; ENDDO
      CALL CLOCK(ITIM); M=ITIM(6)
      WRITE(LP,100) TITLE,ITIM(5),MON(M),ITIM(7),STAT(IS)
      IA=90
      WRITE(LP,110) (IA+I*10,I=0,54,3)
      WRITE(LP,120)
      LINE(14:14)='!'
      LINE(123:123)='!'
      DO FOR K=1,20
         WRITE(LINE(6:12),'(I2,X,1H-,I3)') 100-K*5,105-K*5
         J=21-K
```

```
884 >            DO FOR I=1,108
885 >               LINE(14+I:14+I)=OFFMAP(I,J)
886 >            ENDDO
887 >            WRITE(LP,'(A)') LINE
888 >            DO FOR I=15,122; LINE(I:I)=' '; ENDDO
889 >         ENDDO
890 >         WRITE(LP,120)
891 >         RETURN
892 >         END
893 >C
894 >         SUBROUTINE SYMBOL(A,C)
895 >C        ======================
896 >         CHARACTER C*1
897 >         IA=NINT(A)
898 >         IF(IA.GE.0) THEN
899 >            IF(IA.LE.9) THEN
900 >               C=CHAR(IA+48)
901 >            ELSE
902 >               C=CHAR(IA/10+64)
903 >            ENDIF
904 >         ELSE
905 >            IF(IA.LT.-10) THEN
906 >               C=CHAR(IA/10+91)
907 >            ELSE
908 >               C=CHAR(IA+58)
909 >            ENDIF
910 >         ENDIF
911 >         RETURN
912 >         END
913 >
914 >         SUBROUTINE ERRANAL
915 >C        ==================
916 >         COMMON/FUNCMOD/IFUNCEL(40),IFUNCAZ(40),NFUNC,CFUNCAZ(20),
917 >     *                              CFUNCEL(20)
918 >         CHARACTER CFUNCAZ*10,CFUNCEL*10
919 >         COMMON/FITDATA/AMATR(1600),AINHOMOG(40),COEFF(40)
920 >         COMMON/OFFSDATA/AZVECT(1000),ELVECT(1000),IDIRVECT(1000),
921 >     *                              OFFVECT(1000),NPOINT,AZOREL
922 >         CHARACTER AZOREL*9
923 >         COMMON/ERRORS/ERRMATR(820),ERRVECT(40),EIGVAL(40),EIGVECT(1600)
924 >         DIMENSION FUNCVECT(40)
925 >         DIMENSION HELP(820)
926 >         DOFOR I=1,NFUNC*(NFUNC+1)/2
927 >               HELP(I)=ERRMATR(I)
928 >         ENDDO
929 >         CALL EIGEN(HELP,EIGVECT,NFUNC,0)
930 >         DOFOR I=1,NFUNC
931 >               EIGVAL(I)=SQRT(HELP(I*(I-1)/2+I))
932 >         ENDDO
933 >         WRITE(11,'(X,A/)')
934 >     * 'EIGENVALUES AND EIGENVECTORS=1000 OF THE ERROR MATRIX :'
935 >         WRITE(11,'(* SQRT EV=,T10,*I *,99(I5))')(I,I=1,NFUNC)
936 >         WRITE(11,'(X,T10,*I=*,99(A))') ('-----',I=1,NFUNC)
937 >         DOFOR J=1,NFUNC
938 >            WRITE(11,'(X,F7.3,T10,*I *,99(I5),/)')
939 >     *        EIGVAL(J),(1000.*EIGVECT(I+(J-1)*NFUNC),I=1,NFUNC)
940 >         ENDDO
941 >         RETURN
942 >         END
943 >C
944 >C
945 >****    SUBROUTINE THAT CALCULATES THE MEAN SQUARE ERROR <DF**2> OF
946 >****             NFUNC
947 >****              0
948 >****    F(AZ,EL)= >  FUNCMOD(N,AZ,EL,IDIR)*COEFF(N)
949 >****              /
950 >****              --
951 >****              N=1
952 >****
953 >         SUBROUTINE ERROR(AZ,EL,IDIR,ERR)
954 >C        ================================
955 >         COMMON/FUNCMOD/IFUNCEL(40),IFUNCAZ(40),NFUNC,CFUNCAZ(20),
956 >     *                              CFUNCEL(20)
957 >         CHARACTER CFUNCAZ*10,CFUNCEL*10
958 >         COMMON/FITDATA/AMATR(1600),AINHOMOG(40),COEFF(40)
959 >         COMMON/OFFSDATA/AZVECT(1000),ELVECT(1000),IDIRVECT(1000),
960 >     *                              OFFVECT(1000),NPOINT,AZOREL
961 >         CHARACTER AZOREL*9
962 >         COMMON/ERRORS/ERRMATR(820)
963 >         DIMENSION FUNCVEC(40)
964 >         DOFOR I=1,NFUNC; FUNCVEC(I)=FUNCMOD(I,AZ,EL,IDIR); ENDDO
965 >         ERR=0
```

```
 966 >        DOFOR I=1,NFUNC; DOFOR J=I,NFUNC
 967 >           IR=J*(J-1)/2+I
 968 >           IF(J.NE.I) THEN
 969 >              ERR=ERR+2*FUNCVEC(I)*ERRMATR(IR)*FUNCVEC(J)
 970 >           ELSE
 971 >              ERR=ERR+FUNCVEC(I)*ERRMATR(IR)*FUNCVEC(J)
 972 >           ENDIF
 973 >        ENDDO;ENDDO
 974 >        ERR=SQRT(ERR)
 975 >        RETURN
 976 >        END
 977 >C
 978 >        SUBROUTINE ERRINTRI
 979 >C       ===================================================================
 980 >C       SUBROUTINE TO CALCULATE THE OFF-DIAGONAL ELEMENTS OF THE ERROR MATRIX
 981 >C       -------------------------------------------------------------------
 982 >        COMMON/FUNCMOD/IFUNCEL(40),IFUNCAZ(40),NFUNC,CFUNCAZ(20),
 983 >     *                          CFUNCEL(20)
 984 >        CHARACTER CFUNCAZ*10,CFUNCEL*10
 985 >        COMMON/FITDATA/AMATR(1600),AINHOMOG(40),COEFF(40)
 986 >        COMMON/OFFSDATA/AZVECT(1000),ELVECT(1000),IDIRVECT(1000),
 987 >     *                          OFFVECT(1000),NPOINT,AZOREL
 988 >        CHARACTER AZOREL*9
 989 >        COMMON/ERRORS/ERRMATR(820)
 990 >        COMMON/SCRATCH/SUM1(820),SUM2(820)
 991 >        DOFOR K=1,NPOINT
 992 >           AZ=AZVECT(K); EL=ELVECT(K); IDIR=IDIRVECT(K)
 993 >           ITRIANGLE=NFUNC*(NFUNC-1)/2+NFUNC
 994 >           DOFOR I=1,ITRIANGLE; SUM1(I)=0; SUM2(I)=0; ENDDO; OFF=0
 995 >           DOFOR N=1,NFUNC
 996 >              FMOD=FUNCMOD(N,AZ,EL,IDIR)
 997 >              OFF=OFF+COEFF(N)*FMOD
 998 >              DOFOR I=1,NFUNC-1; DOFOR J=I+1,NFUNC
 999 >                 IR=J*(J-1)/2+I
1000 >                 SUM1(IR)=SUM1(IR)+FMOD*AMATR(I+(N-1)*NFUNC)
1001 >                 SUM2(IR)=SUM2(IR)+FMOD*AMATR(J+(N-1)*NFUNC)
1002 >              ENDDO; ENDDO
1003 >           ENDDO
1004 >           DFI=OFF-OFFVECT(K)
1005 >           DOFOR I=1,NFUNC-1; DOFOR J=I+1,NFUNC
1006 >              IR=J*(J-1)/2+I
1007 >              ERRMATR(IR)=ERRMATR(IR)+DFI*DFI*SUM1(IR)*SUM2(IR)
1008 >           ENDDO; ENDDO
1009 >        ENDDO
1010 >CT      WRITE(1,'(X,*I,J=*,I2,I2,*ERR =*,E11.2)') I,J,ERR
1011 >        RETURN
1012 >        END
1013 >
1014 >        SUBROUTINE ERRINDIAG
1015 >C       ===================================================================
1016 >C       SUBROUTINE TO CALCULATE THE DIAGONAL ELEMENTS OF THE ERROR MATRIX
1017 >C       -------------------------------------------------------------------
1018 >        COMMON/FUNCMOD/IFUNCEL(40),IFUNCAZ(40),NFUNC,CFUNCAZ(20),
1019 >     *                          CFUNCEL(20)
1020 >        CHARACTER CFUNCAZ*10,CFUNCEL*10
1021 >        COMMON/FITDATA/AMATR(1600),AINHOMOG(40),COEFF(40)
1022 >        COMMON/OFFSDATA/AZVECT(1000),ELVECT(1000),IDIRVECT(1000),
1023 >     *                          OFFVECT(1000),NPOINT,AZOREL
1024 >        CHARACTER AZOREL*9
1025 >        COMMON/ERRORS/ERRMATR(820)
1026 >        COMMON/SCRATCH/SUM1(820),SUM2(820)
1027 >        DOFOR K=1,NPOINT
1028 >           AZ=AZVECT(K); EL=ELVECT(K); IDIR=IDIRVECT(K)
1029 >           DOFOR I=1,NFUNC; SUM1(I)=0; ENDDO; OFF=0
1030 >           DOFOR N=1,NFUNC
1031 >              FMOD=FUNCMOD(N,AZ,EL,IDIR)
1032 >              OFF=OFF+COEFF(N)*FMOD
1033 >              DOFOR I=1,NFUNC
1034 >                 SUM1(I)=SUM1(I)+FMOD*AMATR(I+(N-1)*NFUNC)
1035 >              ENDDO
1036 >           ENDDO
1037 >           DFI=OFF-OFFVECT(K)
1038 >           DOFOR I=1,NFUNC
1039 >              IR=I*(I-1)/2+I
1040 >              ERRMATR(IR)=ERRMATR(IR)+DFI*DFI*SUM1(I)*SUM1(I)
1041 >           ENDDO
1042 >        ENDDO
1043 >CT      WRITE(1,'(X,*I,J=*,I2,I2,*ERR =*,E11.2)') I,J,ERR
1044 >        RETURN
1045 >        END
1046 >
1047 >        SUBROUTINE GETS(PRM,S,LS,DEF)
```

```
1048 >C      ============================
1049 >       IMPLICIT INTEGER(A-Z)
1050 >       COMMON/BUFFER/CBUF,PBUF; CHARACTER CBUF*132; INTEGER PBUF
1051 >       CHARACTER PRM*80,S*32,DEF*32,NUL*1
1052 >       DATA PBUF,CBUF/1,''/
1053 >CC     OUTPUT(1) 'JUST CAME TO GETS, PBUF=',PBUF,'CBUF='
1054 >CC     OUTPUT(1) CBUF
1055 >       NUL=CHAR(0)
1056 >       LPRM=LEN(PRM); LDEF=LEN(DEF)
1057 >100    DOWHILE (CBUF(PBUF:PBUF).EQ.' '.AND.PBUF.LE.132
1058 >     *         .AND.CBUF(PBUF:PBUF).NE.NUL)
1059 >          PBUF=PBUF+1
1060 >       ENDDO
1061 >       IF (CBUF(PBUF:PBUF).EQ.',') THEN
1062 >          S=DEF; PBUF=PBUF+1
1063 >          LS=LEN(DEF)
1064 >          GOTO 222
1065 >       ENDIF
1066 >       IF (PBUF.LT.133.AND.CBUF(PBUF:PBUF).NE.NUL) THEN
1067 >          P=PBUF
1068 >          DOWHILE (CBUF(P:P).NE.' '.AND.CBUF(P:P).NE.NUL.AND.P.LT.132
1069 >     *                        .AND.CBUF(P:P).NE.',')
1070 >             P=P+1
1071 >          ENDDO
1072 >          IF (P.EQ.133) P=132
1073 >          S=CBUF(PBUF:P-1)
1074 >          LS=P-PBUF
1075 >          PBUF=P+1
1076 >       ELSE
1077 >             WRITE(1,'(/1H$,A)') PRM(1:LPRM)
1078 >             READ(1,'(A)') CBUF
1079 >             IF (CBUF.EQ.'') CBUF=DEF
1080 >             IF (CBUF.EQ.'') CBUF=','
1081 >             PBUF=1
1082 >             GOTO 100
1083 >       ENDIF
1084 >222    CONTINUE
1085 >CC     OUTPUT(1) 'GOING OUT FROM GETS, PBUF=',PBUF,' CBUF='
1086 >CC     OUTPUT(1) CBUF
1087 >CC     OUTPUT(1) 'STRING OUTPUTTED ='//'|'//S(1:LS)//'|'
1088 >CC     OUTPUT(1) 'LS='//LS
1089 >       RETURN
1090 >       END
1091 >
1092 >       SUBROUTINE GETI(PRM,I,IDEF)
1093 >C      ============================
1094 >       CHARACTER PRM*32,DEF*6,S*10
1095 >       WRITE(DEF,'(I6)') IDEF
1096 >       CALL GETS(PRM,S,LS,DEF)
1097 >       S(LS+1:10)=',          '
1098 >       INPUT(S) I
1099 >       RETURN
1100 >       END
1101 >
1102 >       SUBROUTINE GETR(PRM,R,RDEF)
1103 >C      ============================
1104 >       CHARACTER PRM*32,DEF*32,S*32
1105 >       WRITE(DEF,'(E15.8,15X)') RDEF
1106 >       CALL GETS(PRM,S,LS,DEF)
1107 >       S(LS+1:32)=','
1108 >       INPUT(S) R
1109 >       RETURN
1110 >       END
1111 >       SUBROUTINE SETTABS(LINE,LEN)
1112 >C      ------------------------------------------------------------------
1113 >C      SUBROUTINE TO TO SET STANDARD QED TABS TO A LINE WITH ONLY
1114 >C      CONTROL-I ASCII MARKS
1115 >C      ==================================================================
1116 >       DIMENSION LINE(132),LINE2(200)
1117 >       LOGICAL TABSTOP
1118 >       NSP=40B
1119 >       I=0;J=1
1120 >       DOWHILE(J.LE.132)
1121 >          I=I+1
1122 >          IF(LINE(I).NE.11B) THEN
1123 >             LINE2(J)=LINE(I)
1124 >             J=J+1
1125 >          ELSE
1126 >             IF (TABSTOP(J))  THEN
1127 >                LINE2(J)=NSP
1128 >                J=J+1
1129 >             ENDIF
```

```
1130 >              DOWHILE(.NOT.TABSTOP(J))
1131 >                 LINE2(J)=NSP
1132 >                 J=J+1
1133 >              ENDDO
1134 >           ENDIF
1135 >        ENDDO
1136 >        K=132;DOWHILE(LINE2(K).EQ.NSP);K=K-1;ENDDO
1137 >        DOFOR I=1,K; LINE(I)=LINE2(I); ENDDO
1138 >        DOFOR I=K+1,132; LINE(I)=0;ENDDO
1139 >        LEN=K
1140 >        RETURN
1141 >        END
1142 >        LOGICAL FUNCTION TABSTOP(J)
1143 >        INTEGER ITABS(8),NTABS
1144 >        DATA ITABS/8,14,30,40,50,60,70,80/,NTABS/8/
1145 >        TABSTOP=.FALSE.
1146 >        DOFOR I=1,NTABS
1147 >             IF(J.EQ.ITABS(I)) TABSTOP=.TRUE.
1148 >        ENDDO
1149 >        RETURN
1150 >        END
1151 >
1152 >*************************************************************************
1153 >*
1154 >*       SUBROUTINE TO FIND A COMMAND FROM A COMMAND SET IF AN ABBREVIATION
1155 >*       IS GIVEN.  COMMANDS AND ABBREVIATIONS FOLLOW THE SAME RULES
1156 >*       AS EG IN SINTRAN III.  (FOR EXAMPLE  HEL--OUT  IS A LEGAL ABBRE-
1157 >*       VIATION OF  HELP-ME-OUT-OF-THIS)  THE COMMAND SET USED IS CONTAINED
1158 >*       IN THE COMMON STRING ARRAY CCMND:
1159 >*
1160 >*             COMMON/COMMAND/NCMND,CCMND; CHARACTER CCMND(30)*32
1161 >*
1162 >*       NCMND IS EQUAL TO THE NUMBER OF COMMANDS USED AND THE DIMENSION OF
1163 >*       CCMND SHOULD BE AT LEAST SO MUCH.
1164 >*
1165 >*       CALLING SEQUENCE:
1166 >*
1167 >*             CALL CMNDFIT(COMMAND,ICMND,IERR)
1168 >*
1169 >*             COMMAND=        THE ABBREVIATION INPUT TO THE PROGRAM.
1170 >*                             RETURNS AS COMPLETE VERSION OF THE COMMAND
1171 >*             ICMND=          THE NUMBER OF THE COMMAND RETURNED
1172 >*             IERR=           1, IF NO FITTING COMMAND WAS FOUND
1173 >*                             2, IF AMBIGUOUS ABBREVIATION
1174 >*                             0, OTHERWISE
1175 >*
1176 >*       CONDITIONAL COMPILATION H, IF AUTOMATIC HELP IS WISHED.
1177 >*************************************************************************
1178 >        SUBROUTINE FITCMND(COMMAND,CNO,IERR)
1179 >        IMPLICIT INTEGER(A-Z)
1180 >        CHARACTER COMMAND*32
1181 >        COMMON/COMMAND/NCMND,CCMND; CHARACTER CCMND(30)*32
1182 >        LOGICAL LEGAL
1183 >        I=1
1184 >        DOWHILE(I.LE.NCMND.AND..NOT.LEGAL(COMMAND,CCMND(I))); I=I+1;ENDDO
1185 >        IF (I.GT.NCMND) THEN; IERR=1; GOTO 1; ENDIF
1186 >        IF (COMMAND.EQ.CCMND(I)) THEN; IERR=0; CNO=I; GOTO 1; ENDIF
1187 >        J=I+1
1188 >        DOWHILE(J.LE.NCMND.AND..NOT.LEGAL(COMMAND,CCMND(J))); J=J+1;ENDDO
1189 >        IF(J.GT.NCMND)THEN; COMMAND=CCMND(I); CNO=I; IERR=0; GOTO 1;ENDIF
1190 >        DOWHILE (J.LE.NCMND.AND.COMMAND.NE.CCMND(J)); J=J+1; ENDDO
1191 >        IF (J.GT.NCMND) THEN
1192 >           IERR=2; GOTO 1
1193 >        ELSE
1194 >           COMMAND=CCMND(J); CNO=J; IERR=0; GOTO 1
1195 >        ENDIF
1196 >1       CONTINUE
1197 >CH      IF (COMMAND.EQ.'HELP'.AND.IERR.EQ.0) THEN
1198 >CH         CALL GETS('WHICH COMMAND : ',COMMAND,LS,'')
1199 >CH         DOFOR I=1,NCMND
1200 >CH            IF (LEGAL(COMMAND(1:LS),CCMND(I)))
1201 >CH     *         WRITE(1,'(X,A)') CCMND(I)
1202 >CH         ENDDO
1203 >CH      COMMAND='HELP'
1204 >CH      ENDIF
1205 >        RETURN
1206 >        END
1207 >*************************************************************************
1208 >*
1209 >*       LOGICAL FUNCTION TO TEST IF A WORD IS A LEGAL ABBREVIATION OF
1210 >*       ANOTHER ONE.  SAME RULES AS IN SINTRAN COMMANDS ARE USED.
1211 >*       CALLING SEQUENCE:
```

```
1212 >*
1213 >*            IF (LEGAL(ABBREV,COMMAND)) THEN ....
1214 >*
1215 >*            ABBREV=          A STRING CONTAINING THE ABBREVIATION
1216 >*            COMMAND=         A STRING CONTAINING THE COMPLETE VERSION
1217 >*
!218 >*************************************************************************
1219 >       LOGICAL FUNCTION LEGAL(SHORT,CMND)
1220 >       CHARACTER SHORT*32,CMND*32
1221 >       DIMENSION DASHS(10),DASHC(10)
1222 >       IF (SHORT.EQ.' ') THEN; LEGAL=.TRUE.; RETURN; ENDIF
1223 >       IF (SHORT(1:1).NE.'-'.AND.SHORT(1:1).NE.CMND(1:1)) THEN
1224 >          LEGAL=.FALSE.; RETURN
1225 >       ENDIF
1226 >       IDASHS=0; IDASHC=0
1227 >       DOFOR I=1,32
1228 >          IF (SHORT(I:I).EQ.'-') THEN
1229 >             IDASHS=IDASHS+1; DASHS(IDASHS)=I
1230 >          ENDIF
1231 >          IF (CMND(I:I).EQ.'-') THEN
1232 >             IDASHC=IDASHC+1; DASHC(IDASHC)=I
1233 >          ENDIF
1234 >       ENDDO
1235 >C      DASH(IDASH+1) DENOTES THE END OF STRING:
1236 >       I=1
1237 >       DOWHILE (SHORT(I:I).NE.' '.AND.I.LE.32); I=I+1; ENDDO
1238 >       DASHS(IDASHS+1)=I
1239 >       I=1
1240 >       DOWHILE (CMND(I:I).NE.' '.AND.I.LE.32); I=I+1; ENDDO
1241 >       DASHC(IDASHC+1)=I
1242 >       IF (IDASHS.GT.IDASHC) THEN; LEGAL=.FALSE.; RETURN;ENDIF
1243 >       LEGAL=.TRUE.
1244 >       DOFOR I=1,IDASHS+1
1245 >          IF (I.EQ.1) THEN
1246 >             PBEGS=1; PBEGC=1
1247 >          ELSE
1248 >             PBEGS=DASHS(I-1)+1; PBEGC=DASHC(I-1)+1
1249 >          ENDIF
1250 >          PENDS=DASHS(I)-1; PENDC=DASHC(I)-1
1251 >          IF (PBEGS.LE.PENDS) THEN
1252 >             IF(SHORT(PBEGS:PENDS).NE.CMND(PBEGC:PBEGC+PENDS-PBEGS))THEN
1253 >                LEGAL=.FALSE.; RETURN
1254 >             ENDIF
1255 >          ENDIF
1256 >       ENDDO
1257 >       RETURN
1258 >       END
1259 >       SUBROUTINE INPDEFINE
1260 >C      ------------------------------------------------------------------
1261 >C      SUBROUTINE TO GENERATE AN INPUT FILE FOR THE ANTENNA CALIBRATION
1262 >C      RT PROGRAM "ANTCAL".  WHEN INITIATED, ANTCAL READS IT'S INPUT
1263 >C      PARAMETERS FROM THIS FILE "(RT)ANTCAL-INPUT:DATA".
1264 >C      ==================================================================
1265 >       CHARACTER OK*1, STOREFI*20, OUTFILE*20, COMMENT*80, HELP*2
1266 >       CHARACTER SNAME*6,FILE*32
1267 >       LOGICAL PERIOD
1268 >       MAX=1
1269 >C
1270 >       OUTPUT(1) 'INPUT FILE GENERATION FOR THE DATA GATHERING PROGRAM'
1271 >       OUTPUT(1)
1272 >       CALL GETS('CALIBRATION INPUT FILE : (RT)ANTCAL-IN-',FILE,LS,
1273 >     *                          '(RT)ANTCAL-INPUT:DATA')
1274 >       IF (FILE.NE.'(RT)ANTCAL-INPUT:DATA') THEN
1275 >          FILE=FILE(1:MIND(LS,6))
1276 >          FILE='(RT)ANTCAL-IN-'//FILE(1:-1)//':DATA'
1277 >       ENDIF
1278 >       WRITE(1,'(* FILE NAME= *,A)') FILE(1:-1)
1279 >       OPEN(9,FILE=FILE,ACCESS='W',STATUS='UNKNOWN')
1280 >       CALL GETS('PERIODICALLY ? (Y/N) : ',OK,LS,'Y')
1281 >       WRITE(9,'(* PERIODICALLY ? (Y/N) : *,A1)') OK
1282 >       PERIOD=OK.EQ.'Y'
1283 >       CALL GETS('INITIAL OFFSETS FROM MODEL? (Y/N) : ',OK,LS,'Y')
1284 >       WRITE(9,'(* INITIAL OFFSETS FROM MODEL? (Y/N) : *,A1)') OK
1285 >       IF(PERIOD) THEN
1286 >          CALL GETI('NUMBER OF STARS (MAX 20, DIFFERENT 10) : ',MAX,1)
1287 >          WRITE(9,'(* NUMBER OF STARS (MAX 10) : *,I2)') MAX
1288 >          CALL GETI('DELAY BETWEEN STARS (MINUTES) : ',NWM,0)
1289 >          WRITE(9,'(* DELAY BETWEEN STARS (MINUTES) : *,I4)') NWM
1290 >       ENDIF
1291 >       DO FOR IND=1,MAX
1292 >10        WRITE(HELP,'(I2)') IND
1293 >          CALL GETI(HELP//'. SOURCE INDEX : ',NST,0)
```

```
1294 >           IF(NST.LE.0.AND.NST.NE.-1.OR.NST.GT.6) THEN
1295 >              WRITE(1,'(/,* AVAILABLE STARS ARE:*,/,X,A)')
1296 >     *        '1=CAS-A, 2=CYG-A, 3=3C123, 4=TAU-A, 5=VIR-A, 6=ORINEB'//
1297 >     *        ' ,-1 =1950-COORD.'
1298 >              GOTO 10
1299 >           ELSE IF (NST.GT.0) THEN
1300 >              WRITE(9,'(1H ,12,*, SOURCE INDEX :,*,I2,*,,,,,*)')IND,NST
1301 >           ELSE IF (NST.EQ.-1) THEN
1302 >              CALL GETS('STAR NAME : ',SNAME,LS,'      ')
1303 >              CALL GETR('RA 1950 (H.MMSS) : ',RA,0.)
1304 >              CALL GETR('DE 1950 (D.MMSS) : ',DE,0.)
1305 >              WRITE(9,'(1H ,A,A,2(*,*,F9.5))')
1306 >     *                 'SOURCE INDEX NAME RA DE :,-1,',SNAME(1:-1),RA,DE
1307 >           ENDIF
1308 >           IF (OK.NE.'Y') THEN
1309 >              CALL GETR('INITIAL AZOFF,ELOFF : ',AOF,0)
1310 >              CALL GETR('INITIAL ELOFF : ',EOF,0)
1311 >              WRITE(9,'(* INITIAL AZOFF ELOFF :,*,F9.3,*,*,F9.3)')AOF,EOF
1312 >           ENDIF
1313 >        ENDDO
1314 >        CALL GETR('SWEEP SIZE (IF NOT 2.1 DEGR.) : ',SWSIZE,2.1)
1315 >        WRITE(9,'(* SWEEP SIZE (IF NOT 2.1 DEGR.) :,*F9.3)')SWSIZE
1316 >        CALL GETR('SWEEP SPEED (IF NOT 0.15 DEG/SEC) : ',SPEED,0.15)
1317 >        WRITE(9,'(* SWEEP SPEED (IF NOT 0.15 DEG/SEC) :,*,F9.3)')SPEED
1318 >        CALL GETI('SWEEP DIRECTION (+1=UP, -1=DOWN, 0=BOTH) : ',ISDIR,0)
1319 >        WRITE(9,'(* SWEEP DIRECTION (+1=UP 1=DOWN 0=BOTH) :,*,I2)')ISDIR
1320 >        CALL GETS('CHECK DATA AND ANTENNA (Y/N) : ',OK,LS,'Y')
1321 >        WRITE(9,'(* CHECK DATA AND ANTENNA (Y/N):,*,A)') OK
1322 >        CALL GETS('OUTPUT FILE (NOT TERM!) : ',OUTFILE,LS,'')
1323 >        IF (OUTFILE.EQ.'N'.OR.OUTFILE.EQ.'NO') OUTFILE=''
1324 >        WRITE(9,'(* OUTPUT FILE (NOT TERM!) :,*,A)')OUTFILE
1325 >101     CALL GETS('STORE FILE (OR "NO") :: ',STOREFI,LS,
1326 >     *                        '(RT)ANTCAL-DATA:DATA')
1327 >        IF(STOREFI.NE.'NO'.AND.STOREFI.NE.'N') THEN
1328 >           CALL OPEN5(12,STOREFI,'WA','UNKNOWN',IER)
1329 >           IF (IER.NE.0) GOTO 101
1330 >           CLOSE(12)
1331 >           WRITE(9,'(* STORE FILE :,*,A)')STOREFI
1332 >        ELSE
1333 >           WRITE(9,'(* STORE FILE :,,*)')
1334 >        ENDIF
1335 >        CLOSE(9)
1336 >        END
1337 >C       ------------------------------------------------------------
1338 >C       SUBROUTINE TO OPEN FILES. IT TRIES FIRST ONCE, AND IF THE OPENING
1339 >C       DOES NOT SUCCEED AFTER 5 TRIALS AT 2 SECONDS INTERVALS, ERROR
1340 >C       MESSAGE IS GIVEN.
1341 >C       ============================================================
1342 >        SUBROUTINE OPEN5(IF,FILENAME,ACCESS,STATUS,IER)
1343 >        CHARACTER FILENAME*1, ACCESS*1,STATUS*1
1344 >        DOFOR I=1,5
1345 >           OPEN(IF,FILE=FILENAME,ACCESS=ACCESS,STATUS=STATUS,ERR=100)
1346 >100        IER=ERRCODE
1347 >           IF (IER.EQ.0) RETURN
1348 >           CALL HOLD(2,2)
1349 >        ENDDO
1350 >        WRITE(1,'(* OPEN ERROR : *,I4,*   FILE : *,A)') IER,FILENAME
1351 >        RETURN
1352 >        END
1353 >        SUBROUTINE TITLOFF
1354 >        COMMON/BUFFER/CBUF,PBUF; CHARACTER CBUF*132; INTEGER PBUF
1355 >        COMMON/TITLE/IS,STAT,MON; CHARACTER STAT(3)*10,MON(12)*10
1356 >        CHARACTER FILE*32
1357 >        CALL GETS('STORE FILE : ',FILE,LS,'(RT)ANTCAL-DATA:DATA')
1358 >        CALL OPEN5(12,FILE,'WA','UNKNOWN',IER)
1359 >        IF (IER.NE.0) RETURN
1360 >        WRITE(12,100) STAT(IS)
1361 >100     FORMAT(2H H,/,2H H,20X,*E I S C A T   U H F   A N T E N N A   *,
1362 >     * *C A L I B R A T I O N   *,A10,/,2H H,/,2H H,24X,*A Z I M U T H*,
1363 >     * 11X,*E L E V A T I O N*,11X,*ANT NO   LAST CORRECT.*,8X,*W I N*,
1364 >     * * D*,/,2H H,4X,*DATE      UTIME*,2(*      AZ      EL     OFF *),
1365 >     * 4X,*STAR  DIR ITE    AZ      EL      TEM  VEL DIR*,/,2H H)
1366 >        CLOSE(12)
1367 >        RETURN
1368 >        END
1369 >C
1370 >        SUBROUTINE COMMOFF
1371 >        COMMON/BUFFER/CBUF,PBUF; CHARACTER CBUF*132; INTEGER PBUF
1372 >        COMMON/TITLE/IS,STAT,MON; CHARACTER STAT(3)*10,MON(12)*10
1373 >        DIMENSION ITIM(7)
1374 >        CHARACTER FILE*32
1375 >        CALL GETS('STORE FILE : ',FILE,LS,'(RT)ANTCAL-DATA:DATA')
```

```
1376 >        CALL OPEN5(12,FILE,'WA','OLD',IER)
1377 >        IF (IER.NE.0) RETURN
1378 >        IF (CBUF(PBUF:132).EQ.'') THEN
1379 >           WRITE(1,'(/,1HS,"COMMENT : *)')
1380 >           READ(1,'(A80)') CBUF
1381 >           PBUF=1
1382 >        ENDIF
1383 >        I=132; DOWHILE(CBUF(I:I).EQ.' '); I=I-1; ENDDO
1384 >        CALL CLOCK(ITIM)
1385 >        WRITE(12,100) (ITIM(8-J),J=1,3),ITIM(4),ITIM(5),CBUF(PBUF:I)
1386 >100     FORMAT(2H C,/,2H C,15,2(*,*,J2),2X,J2,*,*,J2,4X,A,/,2H C)
1387 >        CBUF=''; CLOSE(12)
1388 >        RETURN
1389 >        END
1390 >
1391 >C       ***************************************************************
1392 >C       SUBROUTINE TO SIMULATE THE DATA SAMPLING THROUGH CAMAC
1393 >C       THIS PROGRAM USES THE SAME CAMAC CALLS AS THE ANTSMP PROGRAM.
1394 >C       IT IS POSSIBLE TO TEST WITH THIS THAT THE DATA FLOWS CORRECTLY
1395 >C       FROM CAMAC TO COMPUTER. THE SAMPLED DATA IS SHOWN AT THE
1396 >C       TERMINAL
1397 >C       ---------------------------------------------------------------
1398 >        SUBROUTINE CAMACTEST
1399 >        CHARACTER MASK*6,HELP*6,A*60
1400 >        A=' '; A(10:10)=':'; A(50:50)=':'
1401 >        CALL GETI('CRATENO : ',ICR,0)
1402 >        CALL GETI('STATIONNO : ',IN,12)
1403 >        CALL GETS('MASK WORD (IN OCTAL) : ',MASK,LS,'400')
1404 >        HELP='000000';HELP(6-LS+1:6)=MASK(1:LS)
1405 >        READ(HELP,'(O6)') IMASK
1406 >        CALL GETR('TIME INTERVAL (IN SECONDS) : ',SEC,2.)
1407 >        IHOLD=SEC*50.
1408 >        OUTPUT(1)
1409 >        N=0
1410 >C       WRITE MASK AND ENABLE TRIGGER
1411 >        CALL CAMAC(0,IST,ICR,IN,0,9)
1412 >        CALL CAMAC(IMASK,IST,ICR,IN,0,17)
1413 >        CALL CAMAC(0,IST,ICR,IN,0,26)
1414 >        BMAX=10; BMIN=0; N=71
1415 >        DOWHILE(.TRUE.)
1416 >           IF (N.EQ.71) THEN
1417 >              N=1
1418 >              AMAX=BMAX; AMIN=BMIN
1419 >              BMIN=9999; BMAX=-9999
1420 >              WRITE(1,'(*   VOLTS*,T17,F6.2,T57,F6.2)') AMIN,AMAX
1421 >           ENDIF
1422 >           CALL HOLD(IHOLD,1)
1423 >           CALL CAMAC(0,IST,ICR,IN,0,25)
1424 >           CALL CAMAC(IDATA,IST,ICR,IN,0,2)
1425 >           IDATA=IAND(IDATA,7777B)
1426 >           VOLT=IDATA*10./7777B
1427 >           BMAX=AMAX1(VOLT,BMAX); BMIN=AMIN1(VOLT,BMIN)
1428 >           I=(VOLT-AMIN)/(AMAX-AMIN)*40+10; IF (I.LT.1) I=1
1429 >           IF (I.GT.59) I=59
1430 >           WRITE(1,'(X,F6.2,T11,A,1H*,A)') VOLT,A(1:I-1),A(I+1:-1)
1431 >           N=N+1
1432 >           IF (ISIZE(1).GT.0) THEN
1433 >              N=INCH(1); RETURN
1434 >           ENDIF
1435 >        ENDDO
1436 >        END
```

CONTENTS OF FILE : (ANTENNA-EISCAT)ANTCAL-RT-LOAD;MODE;1

```
 1 >@CC CREATION OF ANTENNA CALIBRATION SEGMENTS
 2 >@CC    SEGMENT 72 CONTAINS COMMONS /ADATA/, /ACNTM/ AND /ACHECK/
 3 >@FTN
 4 >CO-CO PD
 5 >COM (ANTE)ANTCAL-MAIN,,(ANTE)ANTCAL-MAIN
 6 >COM (ANTE)ANTCAL-SAMP,,(ANTE)ANTCAL-SAMP
 7 >COM (ANTE)ANTCAL-SWEEP,,(ANTE)ANTCAL-SWEEP
 8 >EX
 9 >@FTN
10 >LIBR
11 >COM (ANTE)STAR-PACK,,(ANTE)STAR-PACK
12 >CO-CO R
13 >COM (ANTE)ACU-SUB,,(ANTE)ACU-SUB-RT
14 >EX
15 >@FTN
16 >CO-CO U
17 >COM (ANTE)ANTCAL-OFF-CALC,,(ANTE)ANTCAL-OFF-CALC
18 >EX
19 >@RT-LOADER
20 >PART-CL-RTF 72
21 >CLEAR-SEG 72
22 >Y
23 >PART-CL-RTF 74
24 >CLEAR-SEG 74
25 >Y
26 >NEW-SEG 74,1,DM,,,
27 >NEW-SEG 72,1,ND,,,
28 >SET-L-A 72,150000
29 >SET-SEG-COMMON ADATA
30 >SET-SEG-COMMON ACNTM
31 >SET-SEG-COMMON ACHECK
32 >NREE-L (ANTE)ANTCAL-MAIN,72,,
33 >LOAD (ANTE)ANTCAL-OFF-CALC,74,72
34 >LOAD (ANTE)ACU-SUBR-RT,74,72
35 >LOAD (ANTE)STAR-PACK,74,72
36 >LOAD (ANTE)LIBRARY-RT,74,72
37 >LOAD FTNLIB,74,72
38 >END-LOAD
39 >NO
40 >WRITE-REFERENCES,,
41 >EXIT
42 >@CC    DATA SAMPLING PROGRAM
43 >@RT-LOADER
44 >PART-CL-RTF 75
45 >CLEAR-SEG 75
46 >Y
47 >NEW-SEG 75,1,ND,,,
48 >NREE-L (ANTE)ANTCAL-SAMP,72,,
49 >LOAD FTNLIB,75,72
50 >END-LOAD
51 >NO
52 >WRITE-REFERENCES,,
53 >EXIT
54 >@CC    ANTENNA CONTINUOUS MOVE -PROGRAM
55 >@RT-LOADER
56 >PART-CL-RTF 76
57 >CLEAR-SEG 76
58 >Y
59 >NEW-SEG 76,1,ND,,,
60 >NREE-L (ANTE)ANTCAL-SWEEP,72,,
61 >LOAD (ANTE)ACU-SUBR-RT,76,72
62 >LOAD FTNLIB,76,72,
63 >END-LOAD
64 >NO
65 >WRITE-REFERENCES,,
66 >WR-RTF 72,,,
67 >EXIT
```

CONTENTS OF FILE : (PACK-TWO:ANTENNA-EISCAT)ANTCAL-MAIN:SYMB;1

```
  1 >C ***************************************************************
  2 >C      ANTENNA CALIBRATION PROGRAM      VERSION 1980.0827
  3 >C                              ANNA-LIISA TURUNEN   NOVEMBER 1979
  4 >C           LAST CHANGED AUGUST 1980
  5 >C
  6 >C      NOTE! - CPU CLOCK MUST BE IN UT !
  7 >C            - SET CORRECT VALUES FOR SMPSEG AND MOVSEG
  8 >C            - SAMPLING INTERVAL IS SET TO 10 HZ.
  9 >C            - ANTENNA MOVING INTERVAL IS   5 HZ.
 10 >C
 11 >C      CONDITIONAL COMPILING:
 12 >C           D FOR DATA SAMPLING     (RT-PROGRAM ANTSMP MUST BE AVAILABLE)
 13 >C           P FOR ANTENNA POINTING (RT-PROGRAMS ANTCNT AND ANTAZEL " )
 14 >C           + FOR ADDITIONAL OUTPUT ON OUTPUT-FILE
 15 >C
 16 >C      RT-LOADING:              @MODE (ANTE)ANTCAL-RT:MODE,,
 17 >C                               (IF PROGRAM FILES ARE ON USER ANTENNA)
 18 >C           OR                  @MODE (RT)ANTCAL-RT:MODE,,
 19 >C                               (IF PROGRAM FILES ARE ON USER RT)
 20 >C      START PROGRAM:           @RT ANTCAL
 21 >C           OR                  BY START-COMMAND IN (ANTE)ANTCAL-CONTROL:PROG
 22 >C      STOP PROGRAM:            @RT ACSTOP
 23 >C           OR                  BY STOP-COMMAND IN (ANTE)ANTCAL-CONTROL:PROG
 24 >C
 25 >C           NOTE! RT PROGRAM ACSTOP IS NEEDED, BECAUSE
 26 >C           SEGMENTS ARE FIXED AND UNFIXED BY PROGRAM AND ALSO
 27 >C           BECAUSE FILE UNITS REMAIN OCCUPIED IF NOT CLOSED
 28 >C           BY PROGRAM ITSELF.
 29 >C
 30 >C      IN SOME ERROR CONDITIONS PROGRAM WRITES A MESSAGE ON COMMAND
 31 >C      TERMINAL (BY SUBROUTINE OUTSTR(A)). IF COMMAND TERMINAL IS
 32 >C      NOT FREE, PROGRAM WAITS FOR IT!
 33 >C
 34 >C      SUBROUTINE-FILES NEEDED:
 35 >C           (ANTE)ANTCAL-OFF-CALC
 36 >C           (ANTE)ACU-SUBR-RT      (ANTPNT,ACUPOS,ANTMES)
 37 >C           (ANTE)STAR-PACK
 38 >C           (ANTE)LIBRARY-RT             (WOPEN, OUTSTR, OUTMES)
 39 >C
 40 >C      RT-COMMON AREAS:
 41 >C           /SITE/
 42 >C           /POINT/
 43 >C      COMMON AREAS IN LINKING SEGMENT:
 44 >C           /ADATA/LIMIT,NDATA,IERSMP,IDATA(2000)
 45 >C           /ACNTM/AZ,EL,DAZ,DEL,IFLAG
 46 >C           /ACHECK/IMIN,ISEC,IBAS,A0,E0
 47 >C
 48 >C ***************************************************************
 49 >C
 50 >      PROGRAM ACSTOP,50
 51 >C     ********************
 52 >C     TO STOP ANTENNA CALIBRATION PROGRAMS,
 53 >C     TO UNFIX SEGMENTS AND TO CLOSE FILES
 54 >C     ************************************
 55 >      INTEGER ANTCAL,ANTCNT,ANTAZEL,ANTSMP,SMPSEG
 56 >      COMMON /SWPROG/ANTCAL,ANTCNT,ANTAZEL,ANTSMP
 57 >      COMMON /SWSEGM/MOVSEG,SMPSEG,LINSEG
 58 >      COMMON /STOPINFO/MOVE,IF1,IF2,IF3; LOGICAL MOVE
 59 >C
 60 >      IF(ANTCAL.GT.0) CALL ABORT(ANTCAL)
 61 >      IF(ANTCNT.GT.0) CALL ABORT(ANTCNT)
 62 >      IF(ANTAZEL.GT.0) CALL ABORT(ANTAZEL)
 63 >      IF(ANTSMP.GT.0) CALL ABORT(ANTSMP)
 64 >      IF(MOVE) THEN
 65 >CP       CALL UNFIX(MOVSEG)
 66 >CD       CALL UNFIX(SMPSEG)
 67 >      ENDIF
 68 >      IF(IF1.GT.0) CLOSE(IF1)
 69 >      IF(IF2.GT.0) CLOSE(IF2)
 70 >      IF(IF3.GT.0) CLOSE(IF3)
 71 >      CALL WOPEN(10,'(RT)ANTCAL-STATUS:DATA','WA','OLD',10,10,IFER)
 72 >      IF(IFER.EQ.0) THEN
 73 >         WRITE(10,'(//,* ANTCAL ABORTED BY ACSTOP*)')
 74 >         CLOSE(10)
 75 >      ENDIF
 76 >      CALL OUTSTR('ANTCAL ABORTED BY ACSTOP!)
```

```
 77 >        END
 78 >C
 79 >        PROGRAM ANTCAL,50
 80 >C       **********************
 81 >        PARAMETER MAXDIM=20
 82 >        INTEGER ITIM(7),YEAR,MON,DAY,STARLIST(MAXDIM)
 83 >        DIMENSION AOFIN(10),EOFIN(10)
 84 >        LOGICAL PERIOD,CHD,IMPOSS,MODEL,SAMPERR
 85 >        CHARACTER OK*1,DUMMY*40,FILE*32,STOREFI*32,OUTFILE*32
 86 >        CHARACTER SCAN*6,DIR*2,PTNAME*6,SNAME*6,OSNAME*6
 87 >        CHARACTER MES*60
 88 >        DIMENSION AZDATA(128),ELDATA(128)
 89 >        LOGICAL SETFL
 90 >        COMMON /POINT/IACOND,SETFL,NC,PTNAME,P1,P2,P3,P4,P5,
 91 >       * IOFF,NSREF,AZREF,ELREF,RNREF,AZLOC,ELLOC,RNLOC,
 92 >       * AZACU,ELACU,AZCOM,ELCOM,PPDLOC,PPDOFF
 93 >        COMMON /SITE/IMAT,CTERM,ERRDEV;INTEGER CTERM,ERRDEV
 94 >        COMMON /STOPINFO/MOVE,IOF,ISF,IF; LOGICAL MOVE
 95 >        COMMON /ADATA/LIMIT,NDATA,IERSMP,IDATA(2000)
 96 >        COMMON /ACNTM/AZ,EL,DAZ,DEL,IFLAG
 97 >        COMMON /ACHECK/IMIN,ISEC,IBAS,A0,E0
 98 >        DATA LIMIT,NDATA,IERSMP,IDATA/2003*0/
 99 >        DATA AZ,EL,DAZ,DEL,A0,E0/6=0./
100 >        DATA IMIN,ISEC,IBAS/3=0/
101 >C
102 >        IOF=0; ISF=0; IF=10; MES=''
103 >        IF(CTERM.LE.0.OR.IMAT.LE.0) THEN
104 >           MES='RT-COMMON NOT CORRECTLY INITIALIZED!'; GOTO 90
105 >        ENDIF
106 >        NSITE=IMAT; IF(NSITE.EQ.4) NSITE=3
107 >        PERIOD=.FALSE.; MODEL=.TRUE.; CHD=.FALSE.; SETFL=.TRUE.
108 >        SAMPERR=.FALSE.
109 >        OK=''; STOREFI=''; OUTFILE=''; MAX=1; IND=0
110 >C
111 >        CALL SWEEPINIT(IFL)
112 >        IF(IFL.NE.0) THEN
113 >           MES='ERROR IN SWEEPINIT'; GOTO 90
114 >        ENDIF
115 >C
116 >CP      IFL=-1
117 >CP      DOWHILE(IFL.LT.0)
118 >CP         IFL=0; CALL ACUPOS(AZACU,ELACU,SETFL,IFL)
119 >CP         IF(IFL.LT.0) CALL ANTMES(0,IFL)
120 >CP         CALL HOLD(5,2)
121 >CP      ENDDO
122 >C
123 >        CALL STARCLEAR
124 >C
125 >        OPEN(IF,FILE='(RT)ANTCAL-INPUT:DATA',ACCESS='R')
126 >        INPUT(IF) DUMMY,OK; IF(OK.EQ.'Y') PERIOD=.TRUE.
127 >        INPUT(IF) DUMMY,OK; IF(OK.NE.'Y') MODEL=.FALSE.
128 >        IF(PERIOD) THEN; INPUT(IF) DUMMY,MAX
129 >                         INPUT(IF) DUMMY,IWM
130 >                         IF(MAX.GT.MAXDIM) MAX=MAXDIM
131 >        ENDIF
132 >        CALL CLOCK(ITIM)
133 >        YEAR=ITIM(7); MON=ITIM(6); DAY=ITIM(5)
134 >        UT=ITIM(4)+ITIM(3)/60.+ITIM(2)/3600.
135 >        CALL STARINIT(0,YEAR,MON,DAY,UT,0,IFL)
136 >        IF(IFL.NE.0) THEN
137 >           MES='ERROR IN STARINIT'; GOTO 90
138 >        ENDIF
139 >        DO FOR IND=1,MAX
140 >           INPUT(IF) DUMMY,NST,SNAME,RA1950,DE1950
141 >           IF(NST.EQ.-1) THEN
142 >              RA=HMSTOH(RA1950)
143 >              DE=HMSTOH(DE1950)
144 >              CALL STARCALL(SNAME,RA,DE,YEAR,MON,DAY,UT,0,NST,IFL)
145 >              IF(IFL.NE.0) THEN
146 >                 MES='ERROR IN STARCALL'; GOTO 90
147 >              ENDIF
148 >           ENDIF
149 >           STARLIST(IND)=NST
150 >           IF(.NOT.MODEL) INPUT(IF) DUMMY,AOFIN(IND),EOFIN(IND)
151 >        ENDDO
152 >        INPUT(IF) DUMMY,SWSIZE; IF(SWSIZE.EQ.0.) SWSIZE=2.1
153 >        INPUT(IF) DUMMY,SPEED; IF(SPEED.EQ.0..OR.SPEED.GT.2.) SPEED=0.15
154 >        INPUT(IF) DUMMY,ISDIR; IF(ISDIR.GT.0)ISDIR=1
155 >                               IF(ISDIR.LT.0)ISDIR=-1
156 >        INPUT(IF) DUMMY,OK; CHD=OK.EQ.'Y'
157 >        INPUT(IF) DUMMY,OUTFILE
158 >        IOF=0
```

```
159 >         IF(OUTFILE.NE.'') THEN
160 >            IOF=8; FILE=OUTFILE
161 >            CALL WOPEN(IOF,FILE,'WA',,'UNKNOWN',10,2,IFER)
162 >            IF(IFER.NE.0) GOTO 90
163 >         ENDIF
164 >         INPUT(IF) DUMMY,STOREFI
165 >         CLOSE(IF)
166 >         ISF=0
167 >         IF(STOREFI.NE.'') THEN
168 >            ISF=9; FILE=STOREFI
169 >            CALL WOPEN(ISF,FILE,'WA',,'UNKNOWN',10,2,IFER)
170 >            IF(IFER.NE.0) GOTO 90
171 >            CLOSE(ISF)
172 >         ENDIF
173 >         CALL OUTSTR('READY TO START OFFSET MEASUREMENTS!')
174 >         CALL OUTSTR('WHEN YOU WANT TO STOP GIVE: BRT ACSTOP !')
175 >         GOTO 41
176 >C
177 >C        INITIALIZE ERROR COUNTERS;
178 >   41    ILLCNT=0; ILLMAX=10
179 >         IMPCNT=0; IMPMAX=10; IMPOSS=.TRUE.
180 >         OSNAME=''; AZOFF=0.; ELOFF=0.; AZAZ=0.; AZEL=0.; ELAZ=0.; ELEL=0.
181 >C
182 >   5     DO WHILE(.TRUE.)
183 >            CALL CLOCK(ITIM)
184 >C           THE STATUS OF ANTCAL IS WRITTEN INTO FILE ANTCAL-STATUS:DATA
185 >            FILE='(RT)ANTCAL-STATUS:DATA'
186 >            CALL WOPEN(IF,FILE,'W',,'UNKNOWN',10,10,IFER)
187 >            IF(IFER.NE.0) GOTO 90
188 >            WRITE(IF,100) (ITIM(8-I),I=1,6),IMAT
189 >   100      FORMAT(//,' EISCAT ANTENNA CALIBRATION  ',
190 >       *    2(I5,2(',',J2)),'  SITE',I2,//,' LAST MEASURED OFFSETS:')
191 >            WRITE(IF,101) OSNAME,'AZOFF:',AZOFF,AZAZ,AZEL,
192 >       *                          'ELOFF:',ELOFF,ELAZ,ELEL
193 >   101      FORMAT(1X,A6,2(X,A6,F6.3,' (',2F6.2,')'))
194 >            WRITE(IF,'(' IFL=',I3,' ILLCNT=',I2,' IMPCNT=',I2,/X,A)')IFL,
195 >       *       ILLCNT,IMPCNT,MES(1:-1)
196 >            CLOSE(IF); MES=''
197 >C
198 >C           TAKE NEW STAR WITH INITIAL OFFSETS;
199 >            IND=IND+1; IF(IND.GT.MAX) IND=1; NSTAR=STARLIST(IND)
200 >            IDIR=ISDIR; IF(ISDIR.EQ.0) IDIR=1
201 >            IOFF=0
202 >C
203 >C           POINT ANTENNA TO THE STAR;
204 >   6        CALL ATOSTAR(NSITE,NSTAR,AZOFF,ELOFF,SWSIZE,STAZO,STELO,
205 >       *       AZWRAP,IFL)
206 >            CALL STARNAME(NSTAR,SNAME)
207 >C
208 >   10       IF(IFL.EQ.20) THEN
209 >C              SOURCE BELOW HORIZON
210 >C              TRY ANOTHER STAR,IF MORE THAN 1 DEFINED;
211 >               MES=SNAME//' BELOW HORIZON'
212 >               IF(MAX.GT.1) GOTO 5
213 >C              WAIT 5 MINUTES AND TRY AGAIN;
214 >               CALL HOLD(5,3); GOTO 5
215 >            ELSEIF(IFL.EQ.-3) THEN
216 >C              ACU DISABLED, WAIT 1 MINUTE AND TRY AGAIN;
217 >               CALL HOLD(1,1)
218 >               MES='ACU DISABLED'; GOTO 5
219 >            ELSEIF(IFL.NE.0) THEN
220 >C              ERROR SITUATION;
221 >               MES=''; WRITE(MES,'(' ANTENNA ERROR :',I5)')IFL
222 >               ILLCNT=ILLCNT+1
223 >               IF(ILLCNT.GE.ILLMAX) GOTO 90
224 >               GOTO 5
225 >            ENDIF
226 >C
227 >C           POINTING SUCCESFUL;
228 >            IF(MODEL) THEN
229 >               CALL OFFSET(1,STAZO+AZWRAP,STELO,AZOFF,ELOFF)
230 >            ELSE
231 >               AZOFF=AOFIN(IND); ELOFF=EOFIN(IND)
232 >            ENDIF
233 >C
234 >            IF(IOF.NE.0) THEN
235 >               WRITE(IOF,'(//' EISCAT ANTENNA CALIBRATION')')
236 >               WRITE(IOF,'(2(I5,2(',',J2)),'  SITE',I2,3X,A6,':',2F7.3)')
237 >       *          (ITIM(8-I),I=1,6),IMAT,SNAME,STAZO,STELO
238 >            ENDIF
239 >            FILE='(RT)ANTCAL-STATUS:DATA'
240 >            CALL WOPEN(IF,FILE,'WA',,'OLD',10,10,IFER)
```

```
241 >           IF(IFER.NE.0) GOTO 90
242 >           WRITE(IF,'(* NOW MEASURING:*,A,2F8.3)') SNAME,STAZO,STELO
243 >           CLOSE(IF)
244 >C
245 >C          HERE WE RETURN WITH NEW SWEEP DIRECTION:
246 >  20       AZDIF2=.1/COS(STELO*.01745); ELDIF2=.1
247 >           ITER=0; IAE=1; DAX=0.; DEX=0.
248 >           IF(IDIR.EQ.1) DIR=' +'; IF(IDIR.EQ.-1) DIR=' -'
249 >C
250 >C          SWEEPS TO ONE DIRECTION ARE REPEATED UNTIL ITERATION SUCCESFUL
251 >C          OR MAXIMUM 3 TIMES:
252 >           DO WHILE(ABS(AZDIF2).GT.0.02/COS(STELO*.0174533)
253 >     *        .OR.ABS(ELDIF2).GT.0.02)
254 >              ITER=ITER+1
255 >              IF(ITER.GT.3.OR.ABS(AZDIF2).GT.0.7/COS(STELO*0.0174533)
256 >     *          .OR.ABS(ELDIF2).GT.0.7) THEN
257 >                 IF(IMPOSS) THEN
258 >                    IMPCNT=IMPCNT+1
259 >                    IF(IMPCNT.GE.IMPMAX) THEN
260 >                       MES='ANTCAL IMPOSSIBLE!'; GOTO 90
261 >                    ENDIF
262 >                 ENDIF
263 >                 IMPOSS=.TRUE.
264 >                 GO TO 50
265 >              ENDIF
266 >C
267 >C                MAKE A SWEEP ACROSS THE STAR:
268 >C
269 >  30            SCANSIZE=SWSIZE
270 >              IF(IAE.EQ.1) SCANSIZE=SCANSIZE/COS(STELO*3.14159/180.)
271 >              HSCANT=NINT(SCANSIZE/SPEED); SCANSIZE=HSCANT*SPEED
272 >              NMEAN=10*HSCANT+1; LIMIT=2*(NMEAN-1)+1
273 >              IF(HSCANT.GE.100) THEN
274 >                 MES=SNAME//' SCAN TIME TOO LONG'; GOTO 5
275 >              ENDIF
276 >              IF(IAE.EQ.1) SCAN='AZSCAN'; IF(IAE.EQ.2) SCAN='ELSCAN'
277 >C
278 >C             WRITE SWEEP-INFORMATION TO STATUS-FILE:
279 >              FILE='(RT)ANTCAL-STATUS;DATA'
280 >              CALL WOPEN(IF,FILE,'WA','OLD',10,9,IFER)
281 >              IF(IFER.NE.0) GOTO 90
282 >              IF(MES.NE.'')WRITE(IF,'(X,A)')MES(1:-1); MES=''
283 >              WRITE(IF,130) SNAME,SCAN,ITER,DIR,2*SCANSIZE,2*HSCANT,
284 >     *        AZOFF,ELOFF
285 >              CLOSE(IF)
286 >C
287 >  130         FORMAT(X,A6,X,A6,I2,A2,F6.3,* DEGR*,I5,* SEC  OFFSETS:*,
288 >     *        2F6.2)
289 >              IF(IOF.NE.0) THEN
290 >                 WRITE(IOF,130)SNAME,SCAN,ITER,DIR,2*SCANSIZE,2*HSCANT,
291 >     *           AZOFF,ELOFF
292 >                 IF(OUTFILE.NE.'L-P') CLOSE(IOF)
293 >              ENDIF
294 >C             WAIT UNTIL ANTENNA READY:
295 >CP            DO WHILE(.NOT.SETFL)
296 >CP               CALL ACUPOS(AZACU,ELACU,SETFL,IFL)
297 >CP               IF(.NOT.SETFL) THEN
298 >CP                  CALL HOLD(2,2)
299 >CP               ENDIF
300 >CP            ENDDO
301 >C
302 >              CALL SWEEP(NSITE,NSTAR,IAE,IDIR,SCANSIZE,SPEED,AZOFF,ELOFF,
303 >     *        DAX,DEX,AZWRAP,YEAR,MON,DAY,UT2,STAZ2,STEL2,IFL)
304 >C
305 >              IF(IOF.NE.0.AND.OUTFILE.NE.'L-P') THEN
306 >                 FILE=OUTFILE
307 >                 CALL WOPEN(IOF,FILE,'WA','OLD',10,20,IFER)
308 >                 IF(IFER.NE.0) IOF=0
309 >              ENDIF
310 >C
311 >C             STAR BELOW HORIZON OR OTHER POINTING ERROR ? :
312 >              IF(IFL.NE.0) GO TO 10
313 >C
314 >CD            IF(CHD) THEN
315 >C+               IF(IOF.NE.0) THEN
316 >C+                  WRITE(IOF,'(* LIMIT,NDATA :*,2I5,5X,*IERSMP :*,I5)')
317 >C+     *                       LIMIT,NDATA,IERSMP
318 >C+                  WRITE(IOF,'(* AO,EO : *,2F8.3)')AO,EO
319 >C+               ENDIF
320 >CD               IF(NDATA.LT.LIMIT.OR.IERSMP.NE.0) THEN
321 >CD                  IMPCNT=IMPCNT+1
322 >CD                  MES='!; WRITE(MES,'(*SAMPLING ERROR*,I3)')IMPCNT
```

```
323 >C+                IF(IOF.NE.0) WRITE(IOF,'(X,A)')MES(1:-1)
324 >CD                IF(IMPCNT.GE.10) GOTO 90
325 >CD                GO TO 30
326 >CD             ENDIF
327 >CD          ENDIF
328 >C
329 >            IF(IAE.EQ.1) THEN
330 >CD             AZAO=AO-AZOFF; DAX=DAX+AZAO-STAZ2; AZDAZ=DAZ
331 >CD             CALL DATATRN(NDATA,NMEAN,IDATA,SCANSIZE=COS(STELO*
332 >CD    *                     .0174533),SWSIZE/1.05,AZDATA)
333 >               AZUT=UT2; AZAZ=STAZ2; AZEL=STEL2
334 >               IAE=2; GOTO 30
335 >            ELSE
336 >CD             ELEO=EO-ELOFF; DEX=DEX+ELEO-STEL2; ELDEL=DEL
337 >CD             CALL DATATRN(NDATA,NMEAN,IDATA,SCANSIZE,SWSIZE/1.05,
338 >CD    *                     ELDATA)
339 >               ELUT=UT2; ELAZ=STAZ2; ELEL=STEL2
340 >               IAE=1
341 >            ENDIF
342 >C
343 >C           CALCULATE OFFSETS:
344 >            EPS=0.01; MAXITER=2*ITER
345 >CD          CALL ANTOFF(AZDATA,ELDATA,STEL2,EPS,MAXITER,SWSIZE/1.05,
346 >CD    *     'U',AZDIF2,ELDIF2,IER,IOF)
347 >            AZDIF2=IDIR*AZDIF2; ELDIF2=IDIR*ELDIF2
348 >C
349 >            UT=AZUT+AZDIF2/SPEED/3600.; AZUTDIF=UT-AZUT
350 >            CALL STARAZEL(NSITE,NSTAR,YEAR,MON,DAY,UT,AZSTAZO,AZSTELO,
351 >    *        0,IFL)
352 >            IF(AZSTAZO.LT.90.) THEN; AZSTAZO=AZSTAZO+360.
353 >            ELSE;                    AZSTAZO=AZSTAZO+AZWRAP
354 >            ENDIF
355 >            AZOFF=AZOFF+AZDIF2-(AZSTAZO-AZAO)
356 >C
357 >            UT=ELUT+ELDIF2/SPEED/3600.; ELUTDIF=UT-ELUT
358 >            CALL STARAZEL(NSITE,NSTAR,YEAR,MON,DAY,UT,ELSTAZO,ELSTELO,
359 >    *       0,IFL)
360 >            ELOFF=ELOFF+ELDIF2-(ELSTELO-ELEO)
361 >C
362 >            IF(IOF.EQ.0) GOTO 40
363 >C
364 >C+         WRITE(IOF,'(/,'* AZDIF2,ELDIF2:',2F8.3)') AZDIF2,ELDIF2
365 >C+         WRITE(IOF,'(' IER FROM ANTOFF:',I3)') IER
366 >C+         WRITE(IOF,'(' AZUT, AZUTDIF:',2F8.3)') AZUT,AZUTDIF
367 >C+         WRITE(IOF,'(' ELUT, ELUTDIF:',2F8.3)') ELUT,ELUTDIF
368 >C+         WRITE(IOF,'(' AZDAZ,ELDEL  :',2F8.3)') AZDAZ,ELDEL
369 >C+         WRITE(IOF,'(' DAX, DEX     :',2F8.3)') DAX,DEX
370 >C+         WRITE(IOF,'(' AZAO,AZAZ,DIF:',3F8.3)')AZAO,AZAZ,AZAO-AZAZ
371 >C+         WRITE(IOF,'(' ELEO,ELEL,DIF:',3F8.3)')ELEO,ELEL,ELEO-ELEL
372 >C+         WRITE(IOF,'(' AZSTAZO-AZAO :',F8.3)')AZSTAZO-AZAO
373 >C+         WRITE(IOF,'(' ELSTELO-ELEO :',F8.3)')ELSTELO-ELEO
374 >C+         WRITE(IOF,'(/,' AZDATA:',/,(8F10.2))') (AZDATA(I),I=1,128)
375 >C+         WRITE(IOF,'(/,' ELDATA:',/,(8F10.2))') (ELDATA(I),I=1,128)
376 >C
377 >  40       ENDDO
378 >C
379 >C          WRITE RESULTS:
380 >            OSNAME=SNAME
381 >            IF(IOF.NE.0) THEN
382 >               WRITE(IOF,101) SNAME,'AZOFF:',AZOFF,AZAZ,AZEL,
383 >    *                                 'ELOFF:',ELOFF,ELAZ,ELEL
384 >            ENDIF
385 >            IF(ISF.NE.0) THEN
386 >               FILE=STOREFI
387 >               CALL WOPEN(ISF,FILE,'WA','OLD',10,10,IFER)
388 >               IF(IFER.NE.0) GOTO 90
389 >               CALL STOROFF(ISF,AZAZ,AZEL,AZOFF,ELAZ,ELEL,ELOFF,
390 >    *          SNAME,IDIR,ITER,AZDIF2,ELDIF2)
391 >               CLOSE(ISF)
392 >            ENDIF
393 >            FILE='(RT)ANTCAL-STATUS:DATA'
394 >            CALL WOPEN(IF,FILE,'WA','OLD',10,10,IFER)
395 >            IF(IFER.EQ.0) THEN
396 >               WRITE(IF,101) SNAME,'AZOFF:',AZOFF,AZAZ,AZEL,
397 >    *                                 'ELOFF:',ELOFF,ELAZ,ELEL
398 >               CLOSE(IF)
399 >            ENDIF
400 >            IMPOSS=.FALSE.; IMPCNT=0
401 >C
402 >C          CHANGE DIRECTION, IF BOTH DIRECTIONS SPECIFIED:
403 >  50       IF(ISDIR.EQ.0.AND.IDIR.EQ.1) THEN
404 >               IDIR=-1; GOTO 20
```

```
405 >           ENDIF
406 >C
407 >           IF(.NOT.PERIOD) GOTO 90
408 >C          WAIT IWM MINUTES, BEFORE POINTING NEXT STAR:
409 >           CALL HOLD(IWM,3)
410 >         ENDDO
411 >C
412 >   90    IF(IFER.NE.0) THEN
413 >           WRITE(MES,'(* FILE ERROR :*,I5)') IFER
414 >         ENDIF
415 >         IF(IOF.GT.0) CLOSE(IOF)
416 >         IF(ISF.GT.0) CLOSE(ISF)
417 >         CLOSE(IF)
418 >         FILE='(RT)ANTCAL-STATUS:DATA'
419 >         CALL WOPEN(IF,FILE,'WA','OLD',10,10,IFER)
420 >         IF(IFER.EQ.0) THEN
421 >           WRITE(IF,'(1X,A)')MES(1:I-1)
422 >           CALL CLOCK(ITIM)
423 >           WRITE(IF,'(/,* ANTCAL TERMINATED  *,2(I5,2(*.*,I2)))')
424 >     *       (ITIM(8-I),I=1,6)
425 >           WRITE(IF,'(* IFL=*,I3,* ILLCNT=*,I2,* IMPCNT=*,I2)')IFL,
426 >     *       ILLCNT,IMPCNT
427 >           CLOSE(IF)
428 >         ENDIF
429 >         CALL OUTSTR(MES(1:I-1)//'  ANTCAL TERMINATED')
430 >         END
431 >C
432 >         SUBROUTINE ATOSTAR(NSITE,NSTAR,AZOFF,ELOFF,SWSIZE,STAZ,STEL,
433 >     *       AZWRAP,IFL)
434 >C        ************************************************************
435 >         INTEGER ITIM(7),YEAR,MON,DAY
436 >         LOGICAL SETFL
437 >         CALL CLOCK(ITIM); YEAR=ITIM(7); MON=ITIM(6); DAY=ITIM(5)
438 >         UT=ITIM(4)+ITIM(3)/60.+ITIM(2)/3600.
439 >         CALL STARAZEL(NSITE,NSTAR,YEAR,MON,DAY,UT,STAZ,STEL,0,IFL)
440 >         CALL ACUPOS(AZACU,ELACU,SETFL,IFL)
441 >         AZ=STAZ; EL=STEL-SWSIZE-0.1
442 >         IF(AZ.LT.100.) AZ=AZ+360.
443 >         IF(AZACU.GT.270.) THEN
444 >           IF(AZACU-AZ.GT.180.) AZ=AZ+360.
445 >           IF(AZ.GT.610.) AZ=AZ-360.
446 >         ENDIF
447 >         AZWRAP=AZ-STAZ
448 >         AZ=AZ+AZOFF; EL=EL+ELOFF
449 >CP       IFL=0; CALL ANTPNT(AZ,EL,IFL)
450 >         RETURN
451 >         END
452 >C
453 >         SUBROUTINE SWEEPINIT(IFL)
454 >C        ************************
455 >C        TO GET PROGRAM ADDRESSES AND SEGMENT NUMBERS
456 >C        WHICH ARE USED BY SUBROUTINE SWEEP AND RT-PROGRAM ACSTOP
457 >C        ************************************************************
458 >         COMMON /SITE/IMAT
459 >         INTEGER GRTDA,ANTCAL,ANTCNT,ANTAZEL,ANTSMP,SMPSEG
460 >         COMMON /SWPROG/ANTCAL,ANTCNT,ANTAZEL,ANTSMP
461 >         COMMON /SWSEGM/MOVSEG,SMPSEG,LINSEG
462 >C
463 >         ANTCAL=GRTDA(7HANTCAL')
464 >         IF(ANTCAL.LE.0) THEN; IFL=-5; RETURN; ENDIF
465 >CP       ANTCNT=GRTDA(7HANTCNT')
466 >CP       IF(ANTCNT.LE.0) THEN; IFL=-5; RETURN; ENDIF
467 >CP       ANTAZEL=GRTDA(8HANTAZEL')
468 >CP       IF(ANTAZEL.LE.0) THEN; IFL=-5; RETURN; ENDIF
469 >CD       ANTSMP=GRTDA(7HANTSMP')
470 >CD       IF(ANTSMP.LE.0) THEN; IFL=-5; RETURN; ENDIF
471 >         IF(IMAT.EQ.1) THEN
472 >CP         MOVSEG=155B
473 >CD         SMPSEG=154B
474 >         ELSEIF(IMAT.EQ.2) THEN
475 >CP         MOVSEG=  0B
476 >CD         SMPSEG=  0B
477 >         ELSEIF(IMAT.EQ.4) THEN
478 >CP         MOVSEG= 76B
479 >CD         SMPSEG= 75B
480 >         ENDIF
481 >         RETURN
482 >         END
483 >C
484 >         SUBROUTINE SWEEP(NSITE,NSTAR,IAE,IDIR,SCANSIZE,SPEED,AZOFF,ELOFF,
485 >     *   DAX,DEX,AZWRAP,YEAR,MON,DAY,UT2,STAZ2,STEL2,IFL)
486 >C        ************************************************************
```

```
487 >        INTEGER ITIM(7),YEAR,MON,DAY,HSCANT,SCANT
488 >        DIMENSION STAZ(3),STEL(3),IH(3),IM(3),IS(3)
489 >        COMMON /ACNTM/AZ,EL,DAZ,DEL,IFLAG
490 >        COMMON /ACHECK/IMIN,ISEC,IBAS,AO,EO
491 >        INTEGER GATDA,ANTCAL,ANTCNT,ANTAZEL,ANTSMP,SMPSEG
492 >        COMMON /SWPROG/ANTCAL,ANTCNT,ANTAZEL,ANTSMP
493 >        COMMON /SWSEGM/MOVSEG,SMPSEG,LINSEG
494 >        COMMON /STOPINFO/MOVE; LOGICAL MOVE
495 >C
496 >        IFL=0
497 >        IF(SPEED.LE.0.) THEN; IFL=-4; RETURN; ENDIF
498 >        CALL CLOCK(ITIM); YEAR=ITIM(7); MON=ITIM(6); DAY=ITIM(5)
499 >        UT1=ITIM(4)+ITIM(3)/60.+ITIM(2)/3600.+6.01/3600.
500 >C
501 >        HSCANT=NINT(SCANSIZE/SPEED); SCANT=2*HSCANT
502 >        DO FOR I=1,3
503 >           UT=UT1+(I-1)*HSCANT/3600.
504 >           CALL STARAZEL(NSITE,NSTAR,YEAR,MON,DAY,UT,STAZ(I),
505 >       *      STEL(I),0,IFL)
506 >           CALL HOURS(UT,IH(I),IM(I),S,IS(I))
507 >           IS(I)=S; IF(IH(I).GE.24) IH(I)=IH(I)-24
508 >        ENDDO
509 >        IF(AZWRAP.EQ.0..AND.STAZ(3).LT.100.) AZWRAP=360.
510 >        DO FOR I=1,3; STAZ(I)=STAZ(I)+AZWRAP; ENDDO
511 >        UT2=UT1+HSCANT/3600.
512 >        STAZ2=STAZ(2);          STEL2=STEL(2)
513 >        IF(IAE.EQ.1) THEN
514 >           A1=STAZ(1)-IDIR*SCANSIZE+AZOFF; E1=STEL(1)+ELOFF
515 >           A2=STAZ(3)+IDIR*SCANSIZE+AZOFF; E2=STEL(3)+ELOFF
516 >           DAZ=(A2-A1)/(5*SCANT);          DEL=(E2-E1)/(5*SCANT)
517 >           AZ=A1-DAZ;                      EL=E1
518 >        ELSEIF(IAE.EQ.2) THEN
519 >           E1=STEL(1)-IDIR*SCANSIZE+ELOFF; A1=STAZ(1)+AZOFF
520 >           E2=STEL(3)+IDIR*SCANSIZE+ELOFF; A2=STAZ(3)+AZOFF
521 >           DAZ=(A2-A1)/(5*SCANT);          DEL=(E2-E1)/(5*SCANT)
522 >           AZ=A1;                          EL=E1-DEL
523 >        ENDIF
524 >C
525 >CP      CALL ANTPNT(AZ,EL,IFL); IF(IFL.NE.0) RETURN
526 >        MOVE=.TRUE.
527 >CP      UT=UT1-1./3600.
528 >CP      CALL HOURS(UT,IH1,IM1,S,IS1); IS1=S; IF(IH1.GE.24) IH1=IH1-24
529 >CD      CALL ABSET(ANTSMP,IS1,IM1,IH1)
530 >CP      CALL INTV(ANTCNT,10,1); CALL FIX(MOVSEG)
531 >CP      CALL ABSET(ANTCNT,IS1,IM1,IH1)
532 >CP      CALL ABSET(ANTAZEL,IS(2),IM(2),IH(2))
533 >        CALL ABSET(0,IS(3),IM(3),IH(3))
534 >        CALL RTWT
535 >CP      CALL ABORT(ANTCNT); CALL UNFIX(MOVSEG)
536 >CD      CALL HOLD(60,1)
537 >CD      CALL ABORT(ANTSMP)
538 >        MOVE=.FALSE.
539 >        TDIF=ISEC+IBAS/50.-IS(2); IF(TDIF.LT.0) TDIF=TDIF+60.
540 >        AO=AO-TDIF*SPEED*IDIR
541 >        EO=EO-TDIF*SPEED*IDIR
542 >        IFL=IFLAG
543 >        RETURN
544 >        END
545 >C
546 >        SUBROUTINE STOROFF(ISF,AZAZ,AZEL,AZOFF,ELAZ,ELEL,ELOFF,
547 >       * STAR,IDIR,ITER,ADIF,EDIF)
548 >        CHARACTER STAR*6,DIR*5
549 >        DIMENSION ITIM(7)
550 >        CALL CLOCK(ITIM)
551 >        DIR=' '; IF(IDIR.GT.0) DIR='  +  '; IF(IDIR.LT.0) DIR='  -  '
552 >        WRITE(ISF,100) (ITIM(8-I),I=1,3),ITIM(4),ITIM(3),
553 >       * AZAZ,AZEL,AZOFF,ELAZ,ELEL,ELOFF,STAR,DIR,ITER,ADIF,EDIF
554 >100     FORMAT(2X,I5,2(',',J2),2X,J2,'.',J2,2(2X,2F7.2,F8.3),3X,A,A,
555 >       * I2,2X,2F7.3,I5X)
556 >C      * I2,2X,2F7.3,2X,2I4,I5)
557 >        RETURN
558 >        END
```

CONTENTS OF FILE : (ANTENNA-EISCAT)ANTCAL-OFF-CALC:SYMB;1

```
1 >C*****************************************************************
2 >C*                                                               *
3 >C*      ANTENNA OFFSET CALCULATIONS FOR ANTENNA CALIBRATION       *
4 >C*      MARKKU LEHTINEN 12.8.1980                                *
5 >C*                                                               *
6 >C*      THE ANTENNA MUST BE SWEPT PAST A RADIOSTAR IN BOTH AZIMUTH *
7 >C*      AND ELEVATION.  INCOMING POWER DENSITIES ARE SAMPLED AND  *
8 >C*      STORED AS INPUT DATA TO ANTOFF PROGRAM TO THE TWO VECTORS *
9 >C*      AZDATA(128) AND ELDATA(128). HERE, ELDATA(1-126) MUST CONTAIN *
10 >C*     THE SAMPLES CORRESPONDING TO ELEVATION SWEEP FROM -2.0 DEG TO *
11 >C*     +2.0 DEG,  AND AZDATA(1-128) MUST CONTAIN THE SAMPLES FROM *
12 >C*     THE AZIMUTH SWEEP OF +-2.0/COS(ELEVATION). SINCE IT IS IM- *
13 >C*     POSSIBLE TO SAMPLE EXACTLY ACCORDING TO THESE SPECIFICATIONS *
14 >C*     THERE IS A SUBROUTINE DATATRN TO CONVERT THE SAMPLED DATA  *
15 >C*     VALUES IDATA TO AZDATA AND ELDATA.                        *
16 >C*                                                               *
17 >C*     THE PROGRAM ANTOFF GIVES ESTIMATES TO THE ERRORS MADE IN THE *
18 >C*     SWEEPS.  IF THESE ERRORS ARE GREAT ,  IT IS ADVISABLE TO MAKE *
19 >C*     THE SWEEPS AGAIN BY TAKING INTO ACCOUNT THESE ERRORS.  THIS *
20 >C*     WAY IT IS POSSIBLE TO OBTAIN MORE RELIABLE ERROR ESTIMATES, *
21 >C*     SINCE THE SECOND SWEEPS WILL MOST CERTAINLY GO THROUGH    *
22 >C*     THE MAIN BEAM, S/N RATIO WILL BE BETTER, AND THE ERRORS IN *
23 >C*     THE BEAM PATTERN MODEL WILL HAVE LESS EFFECT,TOO.         *
24 >C*                                                               *
25 >C*     CONDITIONAL COMPILATIONS:  S FOR SIMULATION AND TESTING   *
26 >C*                                D FOR GRAPHIC DISPLAY          *
27 >C*                                U FOR COMPENSATION OF NON-UNIFORM *
28 >C*                                  BACKROUND                    *
29 >C*                                R FOR REAL TIME DISPLAY        *
30 >C*                                + FOR OUTPUT ON TERMINAL       *
31 >C*                                                               *
32 >C*     SUBROUTINES NEEDED :                                      *
33 >C*                          START, GAUSS FROM SSPSTAT3           *
34 >C*                           (FOR SIMULATION ONLY)               *
35 >C*                          DRAWING ROUTINES FROM TEK-GRAF       *
36 >C*                           (FOR GRAPHIC OUTPUT OPTION ONLY)    *
37 >C*****************************************************************
38 >CS     PROGRAM ANTOFFTEST
39 >CS     DIMENSION AZDATA(128),ELDATA(128)
40 >CS     DIMENSION IDATAAZ(2000),IDATAEL(2000)
41 >CS     REAL NOISE
42 >CS     WRITE(1,'(/,1H$,'GIVE STELO,NAZ,NEL : ')')
43 >CS     INPUT(1) STELO,NAZ,NEL
44 >CS     MAZ=NAZ/2; MEL=NEL/2
45 >CS     DOWHILE (.TRUE.)
46 >CS           WRITE(1,'(/,1H$,'GIVE AZDIF,ELDIF,SCALE,NOISE : ')')
47 >CS           INPUT(1) AZDIF,ELDIF,SCALE,NOISE
48 >CS           CALL DATASIM(AZDIF,ELDIF,STELO,SCALE,NOISE,
49 >CS    *                        NAZ,MAZ,IDATAAZ,
50 >CS    *                        NEL,MEL,IDATAEL)
51 >CS           CALL DATATRN(NAZ,MAZ,IDATAAZ,AZDATA)
52 >CS           CALL DATATRN(NEL,MEL,IDATAEL,ELDATA)
53 >CS           CALL ANTOFF(AZDATA,ELDATA,STELO,.01,3,AZDIF,ELDIF,IER,1)
54 >CS           OUTPUT(1) 'CALCULATED OFFSETS : ',AZDIF,ELDIF
55 >CS           OUTPUT(1) 'ERROR CODE : ',IER
56 >CS     ENDDO
57 >CS     END
58 >C*****************************************************************
59 >C*                                                               *
60 >C*     SUBROUTINE TO SIMULATE THE ANTENNA CALIBRATION DATA        *
61 >C*     INPUT PARAMETERS :                                        *
62 >C*           AZDIF,ELDIF : SIMULATED ERRORS IN AZ AND EL         *
63 >C*           STELO : ELEVATION DURING AZIMUTH SCAN               *
64 >C*           SCALE : A FACTOR TO SIMULATE AN ERROR IN THE THEORETICAL *
65 >C*                   BEAM PATTERN (SCALE=1 FOR NO ERROR)         *
66 >C*           NOISE : NOISE POWER / SIGNAL AT MAXIMUM OF BEAM PATTERN *
67 >C*           NAZ,NEL : NUMBER OF SIMULATED DATAPOINTS            *
68 >C*           MAZ,MEL : DATAPOINT CORRESPONDING TO THE SUPPOSED   *
69 >C*                     STAR POSITION                             *
70 >C*                                                               *
71 >C*     OUTPUT PARAMETERS :                                       *
72 >C*           IDATAAZ : SIMULATED AZ DATA                         *
73 >C*           IDATAEL : SIMULATED EL DATA                         *
74 >C*                                                               *
75 >C*****************************************************************
76 >CS     SUBROUTINE DATASIM(AZDIF,ELDIF,STELO,SCALE,NOISE,
77 >CS    *                        NAZ,MAZ,IDATAAZ,
```

```
 78 >CS    *                          NEL,MEL,IDATAEL)
 79 >CS      DIMENSION IDATAAZ(NAZ),IDATAEL(NEL)
 80 >CS      REAL NOISE
 81 >CS      EL=STELO*3.14159265/180.
 82 >CS      CALL START(3.14159265+AZDIF+ELDIF+NOISE**SCALE)
 83 >CS      DELX=2.1/(MAZ-1) ; DELY=2.1/(MEL-1)
 84 >CS      DOFOR I=1,MAZ
 85 >CS        CALL GAUSS(F);F=F*NOISE*.25
 86 >CS        IDATAAZ(I)=10000*(AMODEL(SCALE*((I-MAZ)*DELX+AZDIF*COS(EL)),
 87 >CS    *                            SCALE*ELDIF)+F)
 88 >CS      ENDDO
 89 >CS      DOFOR I=1,NEL
 90 >CS        CALL GAUSS(F);F=F*NOISE*.25
 91 >CS        IDATAEL(I)=10000*(AMODEL(SCALE*AZDIF*COS(EL),
 92 >CS    *                          SCALE*((I-MEL)*DELY+ELDIF))+F)
 93 >CS      ENDDO
 94 >CS      RETURN
 95 >CS      END
 96 >*************************************************************************
 97 >*                                                                       *
 98 >*      ANTENNA OFFSET CALCULATIONS / MARKKU LEHTINEN 31.5.1979           *
 99 >*                                                                       *
100 >*      INPUT PARAMETERS: AZDATA(128),ELDATA(128) SAMPLED VALUES OF      *
101 >*      POWER DURING AZIMUTH AND ELEVATION SCANS THROUGH THE SUPPOSED    *
102 >*      STAR POSITION, DATA(1) <= -HSCANSIZE/1.05,                       *
103 >*      DATA(128) <= HSCANSIZE/1.05.                                     *
104 >*      MAXITER= MAXIMUM NUMBER OF ITERATIONS TO BE DONE                 *
105 >*      STELO=ELEVATION DURING AZIMUTH SCAN                              *
106 >*      EPS=CONVERGENCE LIMIT IN DEGREES (AZ LIMIT = EPS*COS(STELO)      *
107 >*      HSCANSZ=SIZE OF THE SWEEP/2                                      *
108 >*      CALCODE=A STRING CONTAINING A COMBINATION OF CHARACTERS          *
109 >*              WHICH DETERMINE THE METHOD OF FITTING THE PEAK           *
110 >*                                                                       *
111 >*      OUTPUT PARAMETERS: AZDIF,ELDIF = CALCULATED DIFFERENCES          *
112 >*      OF AZIMUTH AND ELEVATION,  IER = 0 IF ITERATION DID CONVERGE     *
113 >*      IER=1 IF ITERATION DID NOT CONVERGE.                             *
114 >*                                                                       *
115 >*************************************************************************
116 >       SUBROUTINE ANTOFF(AZDATA,ELDATA,STELO,EPS,MAXITER,HSCANSZ,CALCODE
117 >     *                          ,AZDIF,ELDIF,IER,IOUT)
118 >       DIMENSION AZDATA(128),ELDATA(128),T(128)
119 >       REAL AZDATA,ELDATA,D
120 >       CHARACTER CALCODE*4; LOGICAL UNIFORM,INITIAL,CLIP
121 >CD     COMMON/NPICT/NPICT
122 >CD     NPICT=0
123 >       UNIFORM=.FALSE.; INITIAL=.FALSE.; CLIP=.FALSE.
124 >       DOFOR I=1,4
125 >          IF (CALCODE(I:I).EQ.'U') UNIFORM=.TRUE.
126 >          IF (CALCODE(I:I).EQ.'I') INITIAL=.TRUE.
127 >          IF (CALCODE(I:I).EQ.'C') CLIP=.TRUE.
128 >       ENDDO
129 >       ITRM=1
130 >CD     ITRM=34
131 >CR     CALL RESRV(ITRM,1,0)
132 >       EL=STELO*3.14159265/180.
133 >       DELEL=2*HSCANSZ/127.; DELAZ=DELEL
134 >       DELAZ=2*HSCANSZ/1.05/127.*COS(STELO*3.141592/180.)
135 >       AZDIF=0; ELDIF=0
136 >C      A COMPENSATION OF NON-UNIFORM BACKROUND BY A LINEAR
137 >C      RMS FIT TO THE FIRST AND LAST 20 DATAPOINTS, WHICH ARE
138 >C      SUPPOSED TO CONSIST ONLY OF BACKROUND NOISE
139 >CU     SI=0; SAZ=0; SEL=0; SII=0; SIAZ=0; SIEL=0
140 >CU     DOFOR I=1,20
141 >CU           SI=SI+I
142 >CU           SAZ=SAZ+AZDATA(I); SEL=SEL+ELDATA(I)
143 >CU           SII=SII+I*I
144 >CU           SIAZ=SIAZ+I*AZDATA(I); SIEL=SIEL+I*ELDATA(I)
145 >CU     ENDDO
146 >CU     DOFOR I=109,128
147 >CU           SI=SI+I
148 >CU           SAZ=SAZ+AZDATA(I); SEL=SEL+ELDATA(I)
149 >CU           SII=SII+I*I
150 >CU           SIAZ=SIAZ+I*AZDATA(I); SIEL=SIEL+I*ELDATA(I)
151 >CU     ENDDO
152 >CU     A=(SAZ*SII-SIAZ*SI)/(40*SII-SI*SI)
153 >CU     B=(SI*SAZ-40*SIAZ)/(SI*SI-40*SII)
154 >CU     DOFOR I=1,128; AZDATA(I)=AZDATA(I)-(A+I*B); ENDDO
155 >CU     A=(SEL*SII-SIEL*SI)/(40*SII-SI*SI)
156 >CU     B=(SI*SEL-40*SIEL)/(SI*SI-40*SII)
157 >CU     DOFOR I=1,128; ELDATA(I)=ELDATA(I)-(A+I*B); ENDDO
158 >C      INITIAL VALUES FOR AZDIF,ELDIF BY CALCULATION OF MOMENTS OF
159 >C      AZDATA AND ELDATA.  GOOD FOR SMALL NOISE
```

```
160 >CT      AZNORM=0; ELNORM=0
161 >CT      DOFOR I=1,128
162 >CT           AZDIF=AZDIF+(I-64.5)*AZDATA(I)
163 >CT           AZNORM=AZNORM+AZDATA(I)
164 >CT           ELDIF=ELDIF+(I-64.5)*ELDATA(I)
165 >CT           ELNORM=ELNORM+ELDATA(I)
166 >CT      ENDDO
167 >CT      AZDIF=-AZDIF/AZNORM*DELAZ; ELDIF=-ELDIF/ELNORM*DELEL
168 >CT      OUTPUT(ITRM) 'INITIAL AZDIF,ELDIF = ',-AZDIF/COS(EL),-ELDIF
169 >        IER=0
170 >        DO FOR ITER=0,MAXITER
171 >           ELDIF1=ELDIF;AZDIF1=AZDIF
172 >           DOFOR I=1,128
173 >              T(I)=AMODEL(AZDIF,(I-64.5)*DELEL+ELDIF)
174 >           ENDDO
175 >           CALL DISPLAC(T,ELDATA,D)
176 >           D1=D
177 >           ELDIF=D*DELEL+ELDIF
178 >           DOFOR I=1,128
179 >              T(I)=AMODEL((I-64.5)*DELAZ+AZDIF,ELDIF)
180 >           ENDDO
181 >           CALL DISPLAC(T,AZDATA,D)
182 >           AZDIF=D*DELAZ+AZDIF
183 >C+        WRITE(ITRM,'(* AZDIF,ELDIF : *,2F8.3)')-AZDIF/COS(EL),-ELDIF
184 >C?        WRITE(IOUT,'(* AZDIF,ELDIF : *,2F8.3)')-AZDIF/COS(EL),-ELDIF
185 >C         WRITE(IOUT,'(* AZD, ELD    : *,2F8.3)') D,D1
186 >           IF(ABS(AZDIF-AZDIF1).LE.EPS.AND.ABS(ELDIF-ELDIF1).LE.EPS) THEN
187 >              AZDIF=-AZDIF/COS(EL); ELDIF=-ELDIF
188 >CR            CALL RELES(ITRM,1)
189 >              RETURN
190 >           ENDIF
191 >        ENDDO
192 >        IER=1; AZDIF=-AZDIF/COS(EL); ELDIF=-ELDIF
193 >CR      CALL RELES(ITRM,1)
194 >        RETURN; END
195 >*****************************************************************
196 >*       MODEL FOR ANTENNA BEAM PATTERN / MARKKU LEHTINEN 31.5.1979    *
197 >*****************************************************************
198 >        FUNCTION AMODEL(X,Y)
199 >        R=SQRT(X*X+Y*Y)/.993*5.1356
200 >        IF (R.LT.1E-30) R=1.E-30
201 >        AMODEL=(BESJ1(R)/R)**2
202 >        RETURN
203 >        END
204 >C       BESSEL FUNCTION FOR INDEX 1.
205 >        FUNCTION BESJ1(R)
206 >        IF (ABS(R).GT.10) THEN; BESJ1=0; RETURN; ENDIF
207 >        SUM=1; TERM=1 ;N=0
208 >        X=R*R/4
209 >        DOWHILE (ABS(TERM).GT..003)
210 >           N=N+1
211 >           TERM=TERM*(-X)/N/(N+1)
212 >           SUM=SUM+TERM
213 >        ENDDO
214 >        BESJ1=(R/2)*SUM
215 >        RETURN
216 >        END
217 >
218 >*****************************************************************
219 >*
220 >*       SUBROUTINE TO CALCULATE THE DISPLACEMENT OF A DATAVECTOR
221 >*       COMPARED TO A THEORITICAL MODEL T(I) BY CALCULATING THE
222 >*       ZERO CROSSING OF THE CROSS CORRELATION FUNCTION
223 >*       BETWEEN THE MEASURED DATA M(I) AND THE DERIVATIVE OF T(I).
224 >*       IF THERE ARE MORE THAN ONE ZERO CROSSINGS, THE ZERO AFTER
225 >*       THE HIGHEST PEAK OF THE CROSS CORRELATION A(I) IS SEARCHED FOR.
226 >*
227 >*****************************************************************
228 >        SUBROUTINE DISPLAC(T,M,D)
229 >        DIMENSION M(128),T(128),A(128),DERT(128)
230 >        REAL D,NORM; REAL M
231 >CD      COMMON/NPICT/NPICT
232 >CD      CALL PIIRRA(M,NPICT*170+1,110)
233 >CD      CALL PIIRRA(T,NPICT*170+1,220)
234 >C       CALCULATION OF THE DERIVATIVE OF T
235 >        DOFOR I=1,128
236 >           IF (I.EQ.1) THEN
237 >              DERT(I)=T(2)-T(1)
238 >           ELSE IF (I.EQ.128) THEN
239 >              DERT(I)=T(128)-T(127)
240 >           ELSE
241 >              DERT(I)=(T(I+1)-T(I-1))/2
```

```
242 >           ENDIF
243 >           A(I)=0
244 >         ENDDO
245 >C        CALCULATION OF CROSS CORRELATION BETWEEN M AND DERIVATIVE OF T
246 >C        AT INTERVALS OF 10 POINTS
247 >         DOFOR I=-60,60,10
248 >               SUM=0
249 >               DOFOR J=MAXO(1,1-I),MINO(128,128-I)
250 >                 SUM=SUM+M(J)*DERT(I+J)
251 >               ENDDO
252 >               SUM=SUM/(MINO(128,128-I)-MAXO(1,1-I)+1.)
253 >               A(I+64)=SUM
254 >         ENDDO
255 >C        FINDING THE BIGGEST POSITIVE PEAK OF A :
256 >         AMAX=0
257 >         DOFOR I=-60,60,10
258 >            IF (A(I+64).GE.AMAX) THEN; AMAX=A(I+64); IMAX=I; ENDIF
259 >         ENDDO
260 >C        FINDING THE FIRST NEGATIVE POINT OF THE CALCULATED A(I):S :
261 >         DOWHILE (A(IMAX+64).GT.0); IMAX=IMAX+10; ENDDO
262 >C        CALCULATION OF THE CROSS CORRELATION FUNCTION AROUND THE ZERO
263 >C        CROSSING :
264 >         DOFOR I=MAXO(IMAX-10,-63),MINO(IMAX+10,63)
265 >               SUM=0
266 >               DOFOR J=MAXO(1,1-I),MINO(128,128-I)
267 >                 SUM=SUM+M(J)*DERT(I+J)
268 >               ENDDO
269 >               SUM=SUM/(MINO(128,128-I)-MAXO(1,1-I)+1.)
270 >               A(I+64)=SUM
271 >         ENDDO
272 >CD       CALL PIIRRA(A,NPICT=170+1,440)
273 >CD       IF (NPICT.LT.5) THEN; NPICT=NPICT+1;ELSE;NPICT=0;ENDIF
274 >C        FINDING THE ZERO CROSSING :
275 >         DOWHILE (A(I+64).LE.0); I=I-1; ENDDO
276 >C        CALCULATION OF THE ZERO OF A
277 >         D=64+I+(0-A(64+I))/(A(64+I+1)-A(64+I))-64
278 >         RETURN
279 >         END
280 >*****   SUBROUTINE TO PLOT THE VECTOR C TO POSITION I,J
281 >         SUBROUTINE PIIRRA(C,I,J)
282 >         REAL MAX
283 >CD       ASSEMBLY POINT,DREL,MOVEC,WIPE,GRDEV
284 >         DIMENSION C(128)
285 >CR       CALL GRDEV(34)
286 >         MAX=0
287 >         DOFOR K=1,128,1
288 >               MAX=AMAX1(MAX,ABS(C(K)))
289 >         ENDDO
290 >CD       CALL POINT(I,J); CALL DREL(I+128,J)
291 >CD       CALL POINT(I+1,J+NINT(100.*C(1)/MAX))
292 >CD       DOFOR K=2,128,1
293 >CD          IF (C(K).NE.0) CALL DREL(I+K,J+NINT(100.*C(K)/MAX))
294 >CD       ENDDO
295 >CD       CALL MOVEC(10,760-I/16)
296 >         RETURN
297 >         END
298 >*************************************************************
299 >*                                                           *
300 >*       A MOVING AVERAGE SUBROUTINE, WHICH TRANSFORMS A SET OF DATA-  *
301 >*       VALUES IDATA(N) TO ANOTHER SET DATA(128).  HERE IDATA(1)      *
302 >*       CORRESPONDS TO DEG -2.1 AND IDATA(MCENT) TO DEG +2.1.         *
303 >*       DATA(1) CORRESPONDS TO DEG -2.0 AND DATA(128) TO DEG +2.0     *
304 >*                                                           *
305 >*************************************************************
306 >         SUBROUTINE DATATRN(N,MCENT,IDATA,DATALEN,OUTLEN,DATA)
307 >         DIMENSION IDATA(N),DATA(128)
308 >C        IDATA(1)=-2.1 DEGREES, IDATA(N)=+2.1 DEG
309 >         A=(N-1)*OUTLEN/DATALEN/127.
310 >C#       DMIN=99999.
311 >         DOFOR I=1,128
312 >               FN=MCENT+(I-64.5)*A
313 >               LOWLIM=INT(FN-A/2.+1.5)
314 >               LIMUP=INT(FN+A/2-.5)
315 >               SUM=(FN+A/2.-.5-LIMUP)*IDATA(LIMUP+1)+
316 >     *            (LOWLIM-FN-.5+A/2.)*IDATA(LOWLIM-1)
317 >               IF (LOWLIM.LE.LIMUP) THEN
318 >                  DOFOR J=LOWLIM,LIMUP
319 >                    SUM=SUM+IDATA(J)
320 >                  ENDDO
321 >               ENDIF
322 >               DATA(I)=SUM/A
323 >C#             DMIN=AMIN1(DMIN,DATA(I))
324 >         ENDDO
325 >C#       DO FOR I=1,128; DATA(I)=DATA(I)-DMIN; ENDDO
326 >         RETURN
327 >         END
```

CONTENTS OF FILE : (ANTENNA-EISCAT)ANTCAL-SWEEP;SYMB;1

```
 1 >*****************************************************************
 2 >*      ANTENNA CONTINUOUS MOVE -PROGRAM
 3 >*                              ANNA-LIISA TURUNEN   1979.06.20
 4 >*      - MOVE COMMANDS TO ACU ARE ISSUED WITH INTERVAL OF 0.5 SECONDS
 5 >*      - START POSITION OF ANTENNA (AZ,EL) AND INCREMENTS TO AZIMUTH
 6 >*        AND ELEVATION (DAZ,DEL) IN RT COMMON AREA /ACNTM/ GIVEN BY
 7 >*        THE CALLING PROGRAM
 8 >*      - IF ACU IS DISABLED PROGRAM ABORTS ITSELF
 9 >*                  15.01.1980 ( COMMON /ADATA/)
10 >*****************************************************************
11 >       PROGRAM ANTCNT,65
12 >       COMMON /ACNTM/AZ,EL,DAZ,DEL,IFLAG
13 >       IFLAG=0; CALL ANTPNT(AZ,EL,IFLAG)
14 >       IF(IFLAG.NE.0) THEN
15 >C#        CALL RESRV(1,1,0)
16 >C#        WRITE(1,'(* IFLAG, AZ,EL :*,I5,2F16.4)') IFLAG,AZ,EL
17 >C#        WRITE(1,'(* ACU ERROR, ANTCNT ABORTED*)')
18 >C#        CALL RELES(1,1)
19 >          CALL ABORT(0)
20 >       ENDIF
21 >       AZ=AZ+DAZ
22 >       EL=EL+DEL
23 >       END
24 >C
25 >       PROGRAM ANTAZEL,64
26 >       INTEGER ITIM(7); LOGICAL SETFL
27 >       COMMON /ACHECK/IMIN,ISEC,IBAS,AO,EO
28 >       CALL CLOCK(ITIM)
29 >       CALL ACUPOS(AO,EO,SETFL,IFL)
30 >       IBAS=ITIM(1); ISEC=ITIM(2); IMIN=ITIM(3)
31 >       END
32 >EOF
```

CONTENTS OF FILE : (ANTENNA-EISCAT)ANTCAL-SAMP;SYMB;1

```
 1 >        PROGRAM ANTSMP,63
 2 >C       ********************************
 3 >C       PROGRAM IS STARTED ONE SECOND BEFORE SAMPLING IS STARTED AND
 4 >C       ITS EXECUTION IS SET PERIODICALLY WITH 1 SECOND INTERVAL.
 5 >C       DATA IS TRIGGED INTO CAMAC WITH 100 MS INTERVALS (5 BASIC TU).
 6 >C       ADC-MODULE IS LOCATED IN CAMAC CRATE ICR=0, POSITION 12.
 7 >C       ADC MASK IS 400B.
 8 >C
 9 >C       PROGRAM FIRST WAITS 46 BASIC TIME UNITS AND THEN CLEARS ADC.
10 >C       THEN IT SKIPS ONE TIME INTERVAL (=1 SECOND) AND WAITS 200 MS
11 >C       (=10 BASIC TU) BEFORE IT STARTS TO READ ADC.
12 >C       IN EACH LOOP PROGRAM READS 10 DATA VALUES.
13 >C       DATA IS STORED INTO DATA VECTOR IN COMMON /ADATA/
14 >C       PROGRAM IS TERMINATED WHEN DATA VECTOR IS FULL OR
15 >C       WHEN WANTED NUMBER OF DATA VALUES IS READ (=L IN COMMON ADATA)
16 >C       ************************************************************
17 >        PARAMETER ICR=0,IN=12,MASKADC=400B
18 >        COMMON /ADATA/LIMIT,NDATA,IER,IDATA(2000)
19 >C$      DOUBLE INTEGER T0,T1,TIME,TARRAY(50)
20 >C$      DIMENSION NARRAY(50),KARRAY(50),IDATA(2000),ITIM(7),JTIM(7)
21 >C
22 >C$      T0=TIME(DUMMY); I=0; LIMIT=50; CALL CLOCK(ITIM)
23 >C$$     DO FOR J=1,50; IDATA(J)=9; ENDDO
24 >        CALL INTV(0,1,2)
25 >        NDATA=0; IER=0; K=0
26 >        CALL HOLD(46,1)
27 >C       SET MASK,CLEAR FIFO AND ENABLE TRIGGER:
28 >        CALL CAMAC(MASKADC,IST,ICR,IN,0,17)
29 >        CALL CAMAC(0,IST,ICR,IN,0,9)
30 >        CALL CAMAC(0,IST,ICR,IN,0,26)
31 >        CALL RTWT
32 >        CALL HOLD(10,1)
33 >        DO WHILE(.TRUE.)
34 >              CALL CAMAC(IVAL,ISTAT,ICR,IN,0,2)
35 >              IF(IAND(ISTAT,1000008).NE.0) THEN
36 >                 NDATA=NDATA+1
37 >                 IDATA(NDATA)=IAND(IVAL,7777B)
38 >                 IF(NDATA.EQ.LIMIT.OR.NDATA.EQ.2000) GOTO 90
39 >                 K=K+1; IF(K.GT.30) THEN; IER=-1; GOTO 90; ENDIF
40 >              ELSE
41 >C$               I=I+1; TARRAY(I)=TIME(0); KARRAY(I)=K; NARRAY(I)=NDATA
42 >C$               IF(I.EQ.50) GOTO 90
43 >                 K=0; CALL RTWT
44 >              ENDIF
45 >        ENDDO
46 >  90    CONTINUE
47 >C$      CALL RESRV(9,1,0)
48 >C$      WRITE(9,'(X,10(X,06))')(IDATA(I),I=1,50)
49 >C$      WRITE(9,'(/,3(X,16))')((TARRAY(I)-T0,NARRAY(I),KARRAY(I)),I=1,10)
50 >C$      WRITE(9,'(/," IER=",I4," IVAL=",06," STAT=",06)')IER,IVAL,ISTAT
51 >C$      CALL RELES(9,1)
52 >        CALL ABORT(0)
53 >        END
```

CONTENTS OF FILE : (ANTENNA-EISCAT)ACU-SUBROUTINES:SYMB;1

```
 1 >=====================================================
 2 >*      ACU-SUBROUTINES          VERSION 1980.0827
 3 >*                               ANNA-LIISA TURUNEN
 4 >*      CONDITIONAL COMPILING:
 5 >*            R  FOR REAL TIME
 6 >*            +  FOR BACKGROUND
 7 >*            #  FOR TESTING IN BACKGROUND
 8 >*
 9 >*      CORRECT OFFSET-MODEL FOR EACH STATION SHOULD BE INSERTED
10 >*      IN OFFSET-SUBROUTINE IN THE "IF(IMAT.EQ.?)...ELSEIF..."-STATEMENT
11 >*      STATION CODE IMAT IS TAKEN FROM RT-COMMON /SITE/
12 >*      RT-COMMON /POINT/ IS USED IN SUBROUTINES
13 >*            ANTPNT AND ANTDIR
14 >*      LOGICAL SETFL
15 >*      COMMON /POINT/IACOND,SETFL,NC,PTNAME,P1,P2,P3,P4,P5,
16 >*     * IOFF,NSREF,AZREF,ELREF,RNREF,AZLOC,ELLOC,RNLOC,
17 >*     * AZACU,ELACU,AZCOM,ELCOM,PPOLOC,PPOOFF
18 >=====================================================
19 >*
20 >       SUBROUTINE AZTURN(INEG,AZ,IPOS,AZPOS)
21 >C      ************************************
22 >C      TO CHANGE INPUT AZ (EITHER AZ=AZ OR AZ=AZ+360) :
23 >C         90. < AZ-INEG < AZ < AZ+IPOS < 630.
24 >C      DEFAULT VALUES: INEG=IPOS=15
25 >C      ************************************
26 >       IF(ABS(AZ-1080.).LT.0.1) RETURN
27 >       IF(AZ.LT.90.) AZ=AZ+360.
28 >       IF(AZPOS.GT.270.) THEN
29 >          IF((AZPOS-AZ).GT.180.) AZ=AZ+360.
30 >          IF(AZ.GT.630.) AZ=AZ-360.
31 >       ENDIF
32 >       IN=INEG; IP=IPOS
33 >       IF(IN.GT.180) IN=180; IF(IP.GT.180) IP=180
34 >       IF(IN.LT.0) IN=0; IF(IP.LT.0) IP=0
35 >       IF(IN.EQ.0.AND.IP.EQ.0) THEN
36 >          IN=15; IP=15
37 >       ENDIF
38 >       IF((AZ-IN).LT.90.) AZ=AZ+360.
39 >       IF((AZ+IP).GT.630.) AZ=AZ-360.
40 >       RETURN
41 >       END
42 >C
43 >       SUBROUTINE ANTSWEEP(AZ,EL,AZINCR,ELINCR,AZMAX,ELMAX,HTIME)
44 >C      ********************************************************
45 >C      TO CHANGE ANTENNA POINTING FROM AZ,EL BY INCREMENTS
46 >C      AZINCR,ELINCR UNTIL TOTAL INCREMENTS ARE AZMAX,ELMAX.
47 >C      HTIME SPECIFIES THE TIME DELAY BETWEEN POINTING COMMANDS
48 >C      (RECOMMENDED ANTENNA SWEEP IS 0.15 DEGREES/SECOND)
49 >C      (ABS(AZINCR)=HTIME*0.15 OR ABS(ELINCR)=HTIME*0.15)
50 >C      ********************************************************
51 >       DOUBLE INTEGER TM
52 >       LOGICAL AMOVE,EMOVE
53 >       AMOVE=.FALSE.; EMOVE=.FALSE.
54 >       AZI=AZ; ELEV=EL; TM=50.*HTIME; ITIME=TM
55 >       IF(AZMAX.NE.0..AND.AZINCR.NE.0.) THEN
56 >          AMOVE=ABS(AZINCR).LE.ABS(AZMAX)
57 >       ENDIF
58 >       IF(ELMAX.NE.0..AND.ELINCR.NE.0.) THEN
59 >          EMOVE=ABS(ELINCR).LE.ABS(ELMAX)
60 >       ENDIF
61 >       IF(.NOT.AMOVE.AND..NOT.EMOVE) RETURN
62 >CR     CALL DSET(0,TM)
63 >       CALL ANTPNT(AZI,ELEV,IFL)
64 >       DO WHILE(AMOVE.OR.EMOVE)
65 >             IF(AMOVE) AZI=AZI+AZINCR
66 >             IF(EMOVE) ELEV=ELEV+ELINCR
67 >CR           CALL RTWT
68 >CR           CALL DSET(0,TM)
69 >           CALL ANTPNT(AZI,ELEV,IFL)
70 >           AMOVE=ABS(AZI+AZINCR-AZ).LE.ABS(AZMAX)
71 >           EMOVE=ABS(ELEV+ELINCR-EL).LE.ABS(ELMAX)
72 >C#        WRITE(1,'(* SWEEP AZ,EL :*,2F9.3)')AZI,ELEV
73 >C+        CALL HOLD(ITIME,1)
74 >       ENDDO
75 >       RETURN
76 >       END
```

```
 77 >C
 78 >      SUBROUTINE ACUTEST
 79 >C     ******************
 80 >C     TO MAKE ACU BIT TEST
 81 >C     ONLY IN BACKGROUND!
 82 >C     ******************
 83 >C+    CHARACTER A*1
 84 >C+    LOGICAL SETFL,TEST
 85 >C+    DOUBLE INTEGER DD,DAZ(6),DEL(6)
 86 >C+    DATA DAZ/200000B,177777B,177000B,200777B,212525B,165252B/,
 87 >C+   *     DEL/240000B,237777B,220000B,217777B,225252B,212525B/
 88 >C+    TEST=.FALSE.
 89 >C+1   WRITE(1,'(//,* ACU BIT TEST*,/)')
 90 >C+    CALL ACUPOS(AZ,EL,SETFL,IFL)
 91 >C+    IF(IFL.LT.0) THEN
 92 >C+       CALL ANTMES(0,IFL)
 93 >C+       WRITE(1,'(/,*$TEST RUN (Y/N) :*)'); INPUT(1) A
 94 >C+       TEST=(A.EQ.'Y')
 95 >C+    ENDIF
 96 >C+    DO FOR I=1,6
 97 >C+       DD=DAZ(I)
 98 >C+       WRITE(1,'(/,I3,*. AZ TEST, DD= *,O6)') I,DD
 99 >C+       CALL BAZ(DD,IFL)
100 >C+       IF(.NOT.TEST.AND.IFL.NE.0) THEN
101 >C+          CALL ANTMES(0,IFL)
102 >C+          WRITE(1,'(/,*$CONTINUE ? (Y/N) : *)')
103 >C+          INPUT(1) A; IF(A.NE.'Y') GOTO 99
104 >C+       ELSE
105 >C+          SETFL=.FALSE.
106 >C+       ENDIF
107 >C+       DO WHILE(.NOT.SETFL)
108 >C+          CALL ACUPOS(AZ,EL,SETFL,IFL)
109 >C+          CALL HOLD(1,2)
110 >C+       ENDDO
111 >C+       DO FOR J=1,5
112 >C+          CALL HOLD(1,2)
113 >C+          CALL BAZPOS
114 >C+       ENDDO
115 >C+       CALL ACUPOS(AZ,EL,SETFL,IFL)
116 >C+       DIFF=0.00549316*DAZ(I)-AZ
117 >C+       WRITE(1,'(39X,*DIFF. *,F10.4)') DIFF
118 >C+    ENDDO
119 >C+    DO FOR I=1,6
120 >C+       DD=DEL(I)
121 >C+       WRITE(1,'(/,I3,*. EL TEST, DD= *,O6)') I,DD
122 >C+       CALL BEL(DD,IFL)
123 >C+       IF(.NOT.TEST.AND.IFL.NE.0) THEN
124 >C+          CALL ANTMES(0,IFL)
125 >C+          WRITE(1,'(/,*$CONTINUE ? (Y/N) : *)')
126 >C+          INPUT(1) A; IF(A.NE.'Y') GOTO 99
127 >C+       ELSE
128 >C+          SETFL=.FALSE.
129 >C+       ENDIF
130 >C+       DO WHILE(.NOT.SETFL)
131 >C+          CALL ACUPOS(AZ,EL,SETFL,IFL)
132 >C+          CALL HOLD(1,2)
133 >C+       ENDDO
134 >C+       DO FOR J=1,5
135 >C+          CALL HOLD(1,2)
136 >C+          CALL BELPOS
137 >C+       ENDDO
138 >C+       CALL ACUPOS(AZ,EL,SETFL,IFL)
139 >C+       DIFF=0.00549316*DEL(I)-360.-EL
140 >C+       WRITE(1,'(39X,*DIFF. *,F10.4)') DIFF
141 >C+    ENDDO
142 >99    RETURN
143 >      END
144 >C
145 >      SUBROUTINE ANTPNT(AZ,EL,IFL)
146 >C     ****************************************
147 >C     TO POINT ANTENNA TO AZ,EL (PHYSICAL DIRECTION)
148 >C     IFL    = 0         = NORMAL RETURN
149 >C            = POSITIVE  = WRONG VALUE OF AZ OR EL
150 >C            = NEGATIVE  = ACU DISABLED
151 >C     IF COMMAND IS ACCEPTED, SETFL IS SET TO .FALSE.
152 >C     ****************************************
153 >      LOGICAL SETFL
154 >      COMMON /POINT/IACOND,SETFL,NC,PTNAME,P1,P2,P3,P4,P5,
155 >     * IOFF,NSREF,AZREF,ELREF,RNREF,AZLOC,ELLOC,RNLOC,
156 >     * AZACU,ELACU,AZCOM,ELCOM,PPOLOC,PPOOFF
157 >      COMMON /SITE/IMAT
158 >      PARAMETER AZMIN=90., AZMAX=630.
```

```
159 >C
160 >       IF(ABS(AZ-1080.).LT.0.1) AZ=AZLOC
161 >       IF(IOFF.GT.0) THEN
162 >          CALL OFFSET(IOFF,AZ,EL,AOFF,EOFF)
163 >C#        WRITE(1,'(/,'OFFS :=',2(F8.3,'* + ',F8.3))')AZ,AOFF,EL,EOFF
164 >          AZI=AZ+AOFF; ELEV=EL+EOFF
165 >       ELSE
166 >          AZI=AZ;       ELEV=EL
167 >       ENDIF
168 >       IF(IMAT.EQ.1) THEN
169 >          AZI=AZI;        ELEV=ELEV+0.01
170 >          ELMIN=1.8;      ELMAX=95.3
171 >       ELSEIF(IMAT.EQ.2) THEN
172 >          AZI=AZI-0.02;   ELEV=ELEV+0.03
173 >          ELMIN=3.0;      ELMAX=98.0
174 >       ELSEIF(IMAT.EQ.4) THEN
175 >          AZI=AZI+0.015; ELEV=ELEV+0.010
176 >          ELMIN=1.8;      ELMAX=98.0
177 >       ELSE
178 >          ELMIN=1.8; ELMAX=90.
179 >       ENDIF
180 >C
181 >       IFL=0
182 >       IF(AZI.LT.AZMIN.OR.AZI.GT.AZMAX) IFL=IFL+10
183 >       IF(ELEV.LT.ELMIN.OR.ELEV.GT.ELMAX) IFL=IFL+20
184 >       IF(IFL.NE.0) THEN
185 >          IACOND=IFL
186 >          RETURN
187 >       ENDIF
188 >       CALL ACUAZ(AZI,IFLA)
189 >       IF(IFLA.EQ.0) THEN
190 >             SETFL=.FALSE.
191 >             AZLOC=AZ; AZCOM=AZI
192 >       ENDIF
193 >       CALL ACUEL(ELEV,IFLE)
194 >       IF(IFLE.EQ.0) THEN
195 >             SETFL=.FALSE.
196 >             ELLOC=EL; ELCOM=ELEV
197 >       ENDIF
198 >       IFL=IFLA+IFLE; IACOND=IFL
199 >       RETURN
200 >       END
201 >C
202 >       SUBROUTINE ANTDIR(AZDIR,ELDIR,AFL,IFL)
203 >C      ************************************
204 >C      TO READ ANTENNA POINTING DIRECTION
205 >C      VALUES OF IACOND,SETFL,AZLOC,ELLOC,AZACU,ELACU
206 >C      IN COMMON /POINT/ ARE UPDATED.
207 >C      ************************************
208 >       LOGICAL SETFL,AFL
209 >       COMMON /POINT/IACOND,SETFL,NC,PTNAME,P1,P2,P3,P4,P5,
210 >     * IOFF,NSREF,AZREF,ELREF,RNREF,AZLOC,ELLOC,RNLOC,
211 >     * AZACU,ELACU,AZCOM,ELCOM,PPDLOC,PPDOFF
212 >C
213 >       CALL ACUPOS(AZACU,ELACU,AFL,IFL)
214 >       IF(IOFF.NE.0) THEN
215 >          CALL OFFSET(IOFF,AZACU,ELACU,AOFF,EOFF)
216 >          AZ=AZACU-AOFF; EL=ELACU-EOFF
217 >          CALL OFFSET(IOFF,AZ,EL,AOFF,EOFF)
218 >          AZDIR=AZACU-AOFF; ELDIR=ELACU-EOFF
219 >       ELSE
220 >          AZDIR=AZACU; ELDIR=ELACU
221 >       ENDIF
222 >       IACOND=IFL; SETFL=AFL; AZLOC=AZDIR; ELLOC=ELDIR
223 >       RETURN
224 >       END
225 >C
226 >       SUBROUTINE OFFSET(IOFF,AZ,EL,AOFF,EOFF)
227 >C      ************************************
228 >C      NEW OFFSET MODEL 30.06.1980 FOR ELEVATION
229 >C      NEW OFFSET MODEL 07.07.1980 FOR AZIMUTH
230 >C      DIMENSION OFFSMAT(2,54,10)
231 >       COMMON /SITE/IMAT
232 >C
233 >       IF(IOFF.EQ.0) THEN
234 >          AOFF=0.; EOFF=0.
235 >       ELSEIF(IOFF.EQ.1) THEN
236 >          A=AZ; IF(A.GT.360.) A=A-360.
237 >          A=A*0.0174533; E=EL*0.0174533
238 >          SINAZ=SIN(A); COSAZ=SQRT(1.-SINAZ*SINAZ)
239 >          IF(AZ.GT.90..AND.AZ.LT.270..OR.AZ.GT.450.) COSAZ=-COSAZ
240 >          SINEL=SIN(E); COSEL=SQRT(1.-SINEL*SINEL)
```

```
241 >           SIN2AZ=2*SINAZ*COSAZ
242 >           SIN3AZ=3*SINAZ-4*SINAZ*SINAZ*SINAZ
243 >           COS2AZ=1.-2.*SINAZ*SINAZ
244 >           COS3AZ=-3*COSAZ+4*COSAZ*COSAZ*COSAZ
245 >           TANEL=SINEL/COSEL
246 >           IF(IMAT.EQ.1) THEN
247 >              AOFF=0.
248 >              EOFF=0.
249 >           ELSEIF(IMAT.EQ.2) THEN
250 >              AOFF=0.
251 >              EOFF=0.
252 >           ELSEIF(IMAT.EQ.4) THEN
253 >              AOFF=-0.035 + 0.066*COSAZ + 0.089*SINAZ +
254 >     *                      0.014*COS2AZ + 0.003*SIN2AZ -
255 >     *                      0.001*COS3AZ - 0.007*SIN3AZ +
256 >     *                0.007*COSAZ*TANEL - 0.001*SINAZ*TANEL -
257 >     *                      0.006*TANEL + 0.027/COSEL
258 >              EOFF= 0.487 + 0.005*COSAZ - 0.005*SINAZ +
259 >     *                      0.173*SINEL - 0.132*COSEL + 0.012/TANEL
260 >           ELSE
261 >              AOFF=0.; EOFF=0.
262 >           ENDIF
263 >        ELSE
264 >           AOFF=0.; EOFF=0.
265 >        ENDIF
266 >        RETURN
267 >C
268 >C       IAZ=AINT((AZ+5.)/10.)-9.
269 >C       IEL=AINT((EL+5.)/10.)
270 >C
271 >C       INTERPOLATING USING THREE POINTS
272 >C
273 >C       PAZ=AZ/10.-8-IAZ
274 >C       PEL=AL/10.-IEL+1
275 >C       Q=1-PAZ-PEL
276 >C
277 >C       AZI=AZ+Q*OFFSMAT(1,IAZ,IEL)
278 >C      *     +PAZ*OFFSMAT(1,IAZ+1,IEL)
279 >C      *     +PEL*OFFSMAT(1,IAZ,IEL+1)
280 >C
281 >C       ELEV=EL+Q*OFFSMAT(2,IAZ,IEL)
282 >C      *      +PAZ*OFFSMAT(2,IAZ+1,IEL)
283 >C      *      +PEL*OFFSMAT(2,IAZ,IEL+1)
284 >C       RETURN
285 >        END
286 >C
287 >        SUBROUTINE ANTHNK
288 >C       ********************************
289 >C       TO HOOT THE ALARM HORN IN ANTENNA
290 >        ICR=0; IN1=19; IN0=18
291 >        CALL CAMAC(IDATA,IS,ICR,IN0,0,9)
292 >        IDATA=177611B
293 >        CALL CAMAC(IDATA,IS,ICR,IN0,1,16)
294 >        RETURN; END
295 >C
296 >        SUBROUTINE ANTMES(MTYPE,IFL)
297 >C       ********************************
298 >        CHARACTER TEXT*50
299 >        COMMON /SITE/IMAT,CTERM,ERRDEV; INTEGER CTERM,ERRDEV
300 >        TEXT=' '
301 >        IF(IFL.LT.0) THEN
302 >           IF(IFL.EQ.-1) THEN
303 >              TEXT='ANTENNA AZIMUTH DISABLED!'
304 >           ELSEIF(IFL.EQ.-2) THEN
305 >              TEXT='ANTENNA ELEVATION DISABLED!'
306 >           ELSEIF(IFL.EQ.-3) THEN
307 >              TEXT='ANTENNA CONTROL UNIT DISABLED!'
308 >           ELSE
309 >              WRITE(TEXT,'(* ANTENNA ERROR :*,I4)') IFL
310 >           ENDIF
311 >        ELSEIF(IFL.GT.1) THEN
312 >           IF(IFL.EQ.10) THEN
313 >              TEXT='ILLEGAL AZIMUTH'
314 >           ELSEIF(IFL.EQ.20) THEN
315 >              TEXT='ILLEGAL ELEVATION'
316 >           ELSEIF(IFL.EQ.30) THEN
317 >              TEXT='ILLEGAL COMMAND'
318 >           ELSEIF(IFL.EQ.8) THEN
319 >              TEXT='ILLEGAL AZIMUTH, ELEVATION DISABLED!'
320 >           ELSEIF(IFL.EQ.19) THEN
321 >              TEXT='ILLEGAL ELEVATION, AZIMUTH DISABLED!'
322 >           ELSE
```

```
323 >             WRITE(TEXT,'(* ANTENNA ERROR :*,I4)') IFL
324 >          ENDIF
325 >       ELSE
326 >          RETURN
327 >       ENDIF
328 >C+     CTERM=1
329 >       CALL OUTMES(CTERM,TEXT)
330 >       IF(MTYPE.NE.0) CALL OUTMES(ERRDEV,TEXT)
331 >       RETURN; END
332 >C
333 >       SUBROUTINE ACUAZ(AZI,IFL)
334 >C      ************************************
335 >C      TO GIVE MOVE AZIMUTH COMMAND TO ACU
336 >C      LEGAL AZIMUTHS: 90. <= AZ <= 630.
337 >C      IFL    =  0  NORMAL RETURN
338 >C             = 10  WRONG VALUE OF AZIMUTH
339 >C             = -1  AZIMUTH DISABLED
340 >C      ************************************
341 >       PARAMETER ICR=0, INI=19, INO=18
342 >       PARAMETER BINCR=0.00549316
343 >       PARAMETER AZMIN=90., AZMAX=630.
344 >       DOUBLE INTEGER DD
345 >C
346 >       IFL=0
347 >       IF(AZI.LT.AZMIN.OR.AZI.GT.AZMAX) THEN; IFL=10; RETURN; ENDIF
348 >       DD=AZI/BINCR;  DD=IOR(DD,17400000B)
349 >       CALL CLBIT(DD,22)
350 >       N=0; DO FOR J=0,21; N=N+IBIT(DD,J); ENDDO
351 >       IF(MOD(N,2).EQ.0) CALL STBIT(DD,22)
352 >       LSW=IAND(DD,177777B); DD=ISHFT(DD,-16); MSW=IAND(DD,177B)
353 >       LSW=NOT(LSW); MSW=NOT(MSW)
354 >       CALL CAMAC(LSW,IS,ICR,INO,0,16)
355 >       CALL CAMAC(MSW,IS,ICR,INO,1,16)
356 >       CALL CAMAC(LSW,IS,ICR,INI,0,0)
357 >       CALL CAMAC(MSW,IS,ICR,INI,1,0)
358 >       IF(IAND(MSW,36B).EQ.0) IFL=-1
359 >       RETURN
360 >       END
361 >C
362 >       SUBROUTINE ACUEL(ELEV,IFL)
363 >C      ****************************
364 >C      TO GIVE MOVE ELEVATION COMMAND TO ACU
365 >C      LEGAL ELEVATIONS: ELMIN <= EL <= ELMAX
366 >C      WHERE ELMIN AND ELMAX DEFINED FOR EACH SITE
367 >C      IFL    =  0  NORMAL RETURN
368 >C             = 20  WRONG VALUE OF ELEVATION
369 >C             = -2  ELEVATION DISABLED
370 >C      ****************************************
371 >       PARAMETER ICR=0, INI=19, INO=18
372 >       PARAMETER BINCR=0.00549316
373 >       DOUBLE INTEGER DD
374 >C
375 >       IFL=0
376 >       IF(IMAT.EQ.1) THEN
377 >          ELMIN=1.8;      ELMAX=95.3
378 >       ELSEIF(IMAT.EQ.2) THEN
379 >          ELMIN=3.0;      ELMAX=98.0
380 >       ELSEIF(IMAT.EQ.4) THEN
381 >          ELMIN=1.8;      ELMAX=98.0
382 >       ELSE
383 >          ELMIN=1.8; ELMAX=90.
384 >       ENDIF
385 >       IF(ELEV.LT.ELMIN.OR.ELEV.GT.ELMAX) THEN; IFL=20; RETURN; ENDIF
386 >       DD=ELEV/BINCR; DD=IOR(DD,136000000B)
387 >       CALL CLBIT(DD,22)
388 >       N=0; DO FOR J=0,21; N=N+IBIT(DD,J); ENDDO
389 >       IF(MOD(N,2).EQ.0) CALL STBIT(DD,22)
390 >       LSW=IAND(DD,177777B); DD=ISHFT(DD,-16); MSW=IAND(DD,177B)
391 >       LSW=NOT(LSW); MSW=NOT(MSW)
392 >       CALL CAMAC(LSW,IS,ICR,INO,0,16)
393 >       CALL CAMAC(MSW,IS,ICR,INO,1,16)
394 >       CALL CAMAC(LSW,IS,ICR,INI,0,0)
395 >       CALL CAMAC(MSW,IS,ICR,INI,1,0)
396 >       IF(IAND(MSW,36B).EQ.0) IFL=-2
397 >       RETURN
398 >       END
399 >C
400 >       SUBROUTINE ACUPOS(AZI,ELEV,SETFL,IFL)
401 >C      ************************************
402 >C      TO READ ACU DISPLAYS
403 >C      IFL    = 0 IF ACU READY (COMPUTER MODE)
404 >C             = NEGATIVE IF ACU NOT IN COMPUTER MODE
```

```
405 >C      ********************************************
406 >       PARAMETER ICR=0, INI=19, INO=18
407 >       PARAMETER BINCR=0.00549316
408 >       DOUBLE INTEGER DD
409 >       LOGICAL SETFL
410 >       IFL=0
411 >C ...  CLEAR INPUT REGISTERS:
412 >       CALL CAMAC(LSW,IS,ICR,INI,0,2)
413 >       CALL CAMAC(MSW,IS,ICR,INI,1,2)
414 >C ...  READ AZIMUTH:
415 >       LSWA=0; MSWA=136B; MSWA=NOT(MSWA)
416 >       CALL CAMAC(LSWA,IS,ICR,INO,0,9)
417 >       CALL CAMAC(MSWA,IS,ICR,INO,1,16)
418 >       CALL CAMAC(LSWA,IS,ICR,INI,0,0)
419 >       CALL CAMAC(MSWA,IS,ICR,INI,1,0)
420 >       IF(IAND(MSWA,36B).EQ.0) IFL=IFL-1
421 >       DD=LSWA; DD=IAND(DD,0177777B)
422 >       AZI=DD*BINCR-IBIT(MSWA,0)*360.
423 >C ...  READ ELEVATION:
424 >       LSWE=0; MSWE=16B; MSWE=NOT(MSWE)
425 >       CALL CAMAC(LSWE,IS,ICR,INO,0,9)
426 >       CALL CAMAC(MSWE,IS,ICR,INO,1,16)
427 >       CALL CAMAC(LSWE,IS,ICR,INI,0,0)
428 >       CALL CAMAC(MSWE,IS,ICR,INI,1,0)
429 >       IF(IAND(MSWE,36B).EQ.0) IFL=IFL-2
430 >       DD=LSWE; DD=IAND(DD,0177777B)
431 >       ELEV= DD*BINCR
432 >C ...  SET COMPLETE FLAG:
433 >       SETFL=(IBIT(MSWA,5).EQ.-1).AND.(IBIT(MSWE,5).EQ.-1)
434 >       RETURN
435 >       END
436 >C
437 >       SUBROUTINE BACUAZ(AZI,IFL)
438 >C      ************************************
439 >C      TO GIVE MOVE AZIMUTH COMMAND TO ACU AND
440 >C      TO PRINT THE BIT PATTERNS OF THE ACU WORDS
441 >C      LEGAL AZIMUTHS: 90. <= AZ <= 630.
442 >C      IFL    =  0  NORMAL RETURN
443 >C             = 10  WRONG VALUE OF AZIMUTH
444 >C             = -1  AZIMUTH DISABLED
445 >C      ************************************
446 >       PARAMETER ICR=0, INI=19, INO=18
447 >       PARAMETER BINCR=0.00549316
448 >       DOUBLE INTEGER DD
449 >C
450 >       IF(AZI.LT.90..OR.AZI.GT.630.) IFL=10
451 >       IF(IFL.NE.0) RETURN
452 >       DD=AZI/BINCR
453 >C
454 >       ENTRY BAZ(DD,IFL)
455 >       DD=IOR(DD,17400000B)
456 >       CALL CLBIT(DD,22)
457 >       N=0; DO FOR J=0,21; N=N+IBIT(DD,J); ENDDO
458 >       IF(MOD(N,2).EQ.0) CALL STBIT(DD,22)
459 >       LSW=IAND(DD,177777B); DD=ISHFT(DD,-16); MSW=IAND(DD,177B)
460 >       CALL ACUBITS(1,MSW,LSW,1)
461 >       LSW=NOT(LSW); MSW=NOT(MSW)
462 >       CALL CAMAC(LSW,IS,ICR,INO,0,16)
463 >       CALL CAMAC(MSW,IS,ICR,INO,1,16)
464 >       CALL CAMAC(LSW,IS,ICR,INI,0,0)
465 >       CALL CAMAC(MSW,IS,ICR,INI,1,0)
466 >       CALL ACUBITS(0,MSW,LSW,1)
467 >       IF(IAND(MSW,36B).EQ.0) IFL=-1
468 >       RETURN
469 >       END
470 >C
471 >       SUBROUTINE BACUEL(ELEV,IFL)
472 >C      ***************************
473 >C      TO GIVE MOVE ELEVATION COMMAND TO ACU AND
474 >C      TO PRINT THE BIT PATTERNS OF THE ACU WORDS
475 >C      LEGAL ELEVATIONS; ELMIN <= EL <= ELMAX
476 >C      WHERE ELMIN AND ELMAX DEFINED FOR EACH SITE
477 >C      IFL    =  0  NORMAL RETURN
478 >C             = 20  WRONG VALUE OF ELEVATION
479 >C             = -2  ELEVATION DISABLED
480 >C      ************************************
481 >       PARAMETER ICR=0, INI=19, INO=18
482 >       PARAMETER BINCR=0.00549316
483 >       DOUBLE INTEGER DD
484 >C
485 >       IF(IMAT.EQ.1) THEN
486 >          ELMIN=1.8;      ELMAX=95.3
```

```
487 >         ELSEIF(IMAT.EQ.2) THEN
488 >            ELMIN=3.0;      ELMAX=98.0
489 >         ELSEIF(IMAT.EQ.4) THEN
490 >            ELMIN=1.8;      ELMAX=98.0
491 >         ELSE
492 >            ELMIN=1.8; ELMAX=90.
493 >         ENDIF
494 >         IF(ELEV.LT.ELMIN.OR.ELEV.GT.ELMAX) IFL=20
495 >         IF(IFL.NE.0) RETURN
496 >         DD=ELEV/BINCR
497 >C
498 >         ENTRY BEL(DD,IFL)
499 >         DD=IOR(DD,13600000B)
500 >         CALL CLBIT(DD,22)
501 >         N=0; DO FOR J=0,21; N=N+IBIT(DD,J); ENDDO
502 >         IF(MOD(N,2).EQ.0) CALL STBIT(DD,22)
503 >         LSW=IAND(DD,177777B); DD=ISHFT(DD,-16); MSW=IAND(DD,177B)
504 >         CALL ACUBITS(1,MSW,LSW,2)
505 >         LSW=NOT(LSW); MSW=NOT(MSW)
506 >         CALL CAMAC(LSW,IS,ICR,IN0,0,16)
507 >         CALL CAMAC(MSW,IS,ICR,IN0,1,16)
508 >         CALL CAMAC(LSW,IS,ICR,IN1,0,0)
509 >         CALL CAMAC(MSW,IS,ICR,IN1,1,0)
510 >         CALL ACUBITS(0,MSW,LSW,2)
511 >         IF(IAND(MSW,36B).EQ.0) IFL=-2
512 >         RETURN
513 >         END
514 >C
515 >         SUBROUTINE BAZPOS
516 >C        ************************************
517 >C        TO GIVE READ AZIMUTH COMMAND TO ACU AND
518 >C        TO PRINT THE BIT PATTERNS OF THE ACU WORDS
519 >C        ****************************************
520 >         PARAMETER ICR=0, IN1=19, IN0=18
521 >C ...    CLEAR INPUT REGISTERS:
522 >         CALL CAMAC(LSW,IS,ICR,IN1,0,2)
523 >         CALL CAMAC(MSW,IS,ICR,IN1,1,2)
524 >C ...    READ AZIMUTH:
525 >         LSWA=0; MSWA=136B
526 >C        CALL ACUBITS(1,MSWA,LSWA,1)
527 >         MSWA=NOT(MSWA)
528 >         CALL CAMAC(LSWA,IS,ICR,IN0,0,9)
529 >         CALL CAMAC(MSWA,IS,ICR,IN0,1,16)
530 >         CALL CAMAC(LSWA,IS,ICR,IN1,0,0)
531 >         CALL CAMAC(MSWA,IS,ICR,IN1,1,0)
532 >         CALL ACUBITS(0,MSWA,LSWA,1)
533 >         RETURN
534 >         END
535 >C
536 >         SUBROUTINE BELPOS
537 >C        ************************************
538 >C        TO GIVE READ ELEVATION COMMAND TO ACU AND
539 >C        TO PRINT THE BIT PATTERNS OF THE ACU WORDS
540 >C        ****************************************
541 >         PARAMETER ICR=0, IN1=19, IN0=18
542 >C ...    CLEAR INPUT REGISTERS:
543 >         CALL CAMAC(LSW,IS,ICR,IN1,0,2)
544 >         CALL CAMAC(MSW,IS,ICR,IN1,1,2)
545 >C ...    READ ELEVATION:
546 >         LSWE=0; MSWE=16B
547 >C        CALL ACUBITS(1,MSWE,LSWE,2)
548 >         MSWE=NOT(MSWE)
549 >         CALL CAMAC(LSWE,IS,ICR,IN0,0,9)
550 >         CALL CAMAC(MSWE,IS,ICR,IN0,1,16)
551 >         CALL CAMAC(LSWE,IS,ICR,IN1,0,0)
552 >         CALL CAMAC(MSWE,IS,ICR,IN1,1,0)
553 >         CALL ACUBITS(0,MSWE,LSWE,2)
554 >         RETURN
555 >         END
556 >C
557 >         SUBROUTINE ACUBITS(IRW,MSW,LSW,IAE)
558 >C        **********************************
559 >C        IRW = 0 IF READ ACU
560 >C              1 IF WRITE ACU
561 >C        IAE = 1 IF AZIMUTH COMMAND
562 >C              2 IF ELEVATION COMMAND
563 >C
564 >         CHARACTER*10 TEXT1,TEXT2
565 >         INTEGER X(23)
566 >         DOUBLE INTEGER DD
567 >C
568 >         DO FOR I=1,23; X(I)=0; ENDDO
```

```
569 >       IF(.NOT.(IAE.EQ.1.OR.IAE.EQ.2)) IAE=1
570 >       DO FOR I=1,16
571 >          X(I+7)=IABS(IBIT(LSW,16-I))
572 >       ENDDO
573 >       DO FOR I=1,7
574 >          X(I)=IABS(IBIT(MSW,7-I))
575 >       ENDDO
576 >       TEXT1='';  TEXT2='';  IADDR=0
577 >       DO FOR I=1,4
578 >          IADDR=IADDR+X(2+I)*2**(I-1)
579 >       ENDDO
580 >       IF(IRW.EQ.0) THEN
581 >          TEXT1=' FROM ACU:'
582 >          IF(X(2).EQ.1) TEXT2(4:6)='SET'
583 >       ELSEIF(IRW.EQ.1) THEN
584 >          TEXT1=' TO ACU  :'
585 >          IF(X(2).EQ.0) TEXT2(5:6)='R'
586 >          IF(X(2).EQ.1) TEXT2(5:6)='W'
587 >       ENDIF
588 >       IF(IADDR.EQ.0) THEN;       TEXT2(8:10)='DIS'
589 >       ELSEIF(IADDR.EQ.17B) THEN; TEXT2(9:10)='AZ'
590 >       ELSEIF(IADDR.EQ.16B) THEN; TEXT2(9:10)='EL'
591 >       ELSEIF(IADDR.EQ.15B) THEN; TEXT2(9:10)='HA'
592 >       ENDIF
593 >       DD=LSW;  DD=IAND(DD,0177777B)
594 >       V=DD*0.00549316+(IAE-2)*IBIT(MSW,0)*360.
595 >       WRITE(1,'(A10,I1,1X,I1,1X,4I1,1X,17I1,A10,F9.4)')TEXT1,X,TEXT2,V
596 >       RETURN
597 >       END
598 >C
```

CONTENTS OF FILE : (ANTENNA-EISCAT)STAR-PACK;SYMB;1

```
>**************************************************************
>*      STAR-PACKAGE                        NOVEMBER 1979      *
>*                                ANNA-LIISA TURUNEN, SODANKYLA*
>*      LAST CHANGED: 10.07.1980  ALSO 14.08.1980              *
>*      THIS FILE CONTAINS SUBROUTINES FOR STAR AND PLANET     *
>*      CALCULATIONS                                           *
>*      NOTE!                                                  *
>*          - COMPILE IN LIBRARY MODE                          *
>*          - IF STAR OR SITE IS UNKNOWN IN SYSTEM (NSTAR=-1,OR*
>*            NSITE=-1), TERMINAL IOTRM IS USED IN CONVERSATION*
>*            AND MUST BE RESERVED FOR INPUT AND OUTPUT IN THE *
>*            CALLING PROGRAM, IF THIS IS AN RT PROGRAM.       *
>*          - IF TIME IS NOT SPECIFIED BY USER, THE CPU-TIME IS*
>*            USED. BE SURE THAT IT IS UT!                     *
>*      FOR BACKGROUND CHECK COMPILE IN CO-CO +                *
>**************************************************************
>*
>       SUBROUTINE STARCLEAR
>C      ------------------------------------------------------
>C      TO CLEAR TABLES FOR STAR PARAMETERS IN COMMON /STAR/
>C      ******************************************************
>       PARAMETER KNOWN=6,MAX=16
>       REAL JDO; CHARACTER NAME*6
>       COMMON /STAR/JDO(MAX),TO(MAX),GAST(MAX),RA(MAX),DE(MAX),NAME(MAX)
>       DO FOR I=1,MAX
>          JDO(I)=0.;TO(I)=0.;GAST(I)=0.;RA(I)=0.;DE(I)=0.;NAME(I)=' '
>       ENDDO
>       RETURN
>       END
>C
>       SUBROUTINE STARCALL(SNAME,RA1950,DE1950,INYEAR,MON,DAY,UT,IOTRM,
>     * NSTAR,IFL)
>C      ------------------------------------------------------------
>C      TO CHECK THAT THE STAR IS CORRECTLY INITIALIZED, I.E. ITS NAME
>C      IS KNOWN TO THE SYSTEM.
>C      IF STAR IS NOT YET INITIALIZED (JDO=0), IT IS NOW INITIALIZED IF
>C      1950-COORDINATES ARE GIVEN IN PARAMETERS.
>C      INPUT:
>C            SNAME             THE NAME OF THE STAR
>C            RA1950            1950-RIGHT ASCENSION OF THE STAR (HOURS)
>C            DE1950            1950-DECLINATION OF THE STAR (DEGREES)
>C            YEAR,MON,DAY, UT (HOURS)
>C                              IF DATE AND TIME IS NOT GIVEN (YEAR=0),
>C                              THE TIME IS READ FROM CPU-CLOCK AND IS
>C                              RETURNED IN YEAR,MON,DAY AND UT
>C            IOTRM             I/O-TERMINAL FOR CONVERSATIONS
>C                              (IN RT-PROGRAMS IOTRM MUST BE RESERVED
>C                              IN CALLING PROGRAM)
>C      OUTPUT:
>C            NSTAR             INDEX OF THE STAR (TO BE USED IN STARAZEL)
>C            IFL = 0           IF NORMAL RETURN
>C                 99           IF USER STOP WANTED
>C                100           STAR NOT KNOWN (NAME OR COORDINATES MISSING)
>C      ******************************************************************
>       PARAMETER KNOWN=6, MAX=16
>       INTEGER YEAR,MON,DAY,ITIM(7)
>       CHARACTER SNAME*6,NAME=6,SNLST(KNOWN)*6
>       REAL JDO,JD
>       COMMON /STAR/JDO(MAX),TO(MAX),GAST(MAX),RA(MAX),DE(MAX),NAME(MAX)
>       DATA SNLST/'CAS-A ','CYG-A ','3C123','TAU-A ','VIR-A ','ORINEB'/
>C
>       IFL=0; NSTAR=0; YEAR=INYEAR
>       IF(YEAR.EQ.0) THEN
>          CALL CLOCK(ITIM); YEAR=ITIM(7); MON=ITIM(6); DAY=ITIM(5)
>          UT=ITIM(4)+ITIM(3)/60.; INYEAR=YEAR
>       ENDIF
>       IF(YEAR.LT.100) YEAR=YEAR+1900
>       IFREE=0; L=LENGTH(SNAME); IF(L.EQ.0) GOTO 10
>       DO FOR I=1,MAX
>          IF(SNAME(1:L).EQ.NAME(I)(1:L)) THEN
>             JD=-693946.5+AINT(AINT(1461.0*(YEAR+(MON-14)/12))/4)+
>     +       367*(MON-2-12*((MON-14)/12))/12+
>     +       (2400-12*YEAR-MON)/1200+DAY
>             IF(ABS(JD-JDO(I)).LT.8) THEN
>                NSTAR=I
>C+              WRITE(1,'(= READY, NSTAR=',I3)') NSTAR
```

```
 77 >                RETURN
 78 >             ELSEIF(I.GE.1.AND.I.LE.KNOWN) THEN
 79 >                CALL STARINIT(I,YEAR,MON,DAY,UT,0,IFL)
 80 >                IF(IFL.EQ.0) NSTAR=I
 81 >C+              WRITE(1,'(* RE-INIT, NSTAR=*,I3)') NSTAR
 82 >                RETURN
 83 >             ENDIF
 84 >          ENDIF
 85 >          IF(IFREE.EQ.0.AND.JDO(I).EQ.0.) IFREE=I
 86 >       ENDDO
 87 >C
 88 >C      STAR WAS NOT FOUND IN TABLE; IT MUST BE INITIALIZED.
 89 >C      FIRST CHECK IF STAR IS KNOWN BY NAME:
 90 >       DO FOR NSTAR=1,KNOWN
 91 >          IF(SNAME.EQ.SNLST(NSTAR)) THEN
 92 >             CALL STARINIT(0,YEAR,MON,DAY,UT,0,IFL)
 93 >C+           WRITE(1,'(* KNOWN INIT, NSTAR=*,I3)') NSTAR
 94 >             RETURN
 95 >          ENDIF
 96 >       ENDDO
 97 >C      IT WAS NOT KNOWN:
 98 >  10   IF(RA1950.EQ.0..OR.DE1950.EQ.0.) THEN
 99 >C+          WRITE(1,'(* UNKNOWN STAR, 1950-COORDINATES EXPECTED=)')
100 >           IFL=100; RETURN
101 >       ENDIF
102 >       NSTAR=IFREE; IF(NSTAR.EQ.0) NSTAR=10
103 >       CALL STARSITR(RA1950,DE1950,YEAR,MON,DAY,UT,SRA,SDE,STI,JD)
104 >       JDO(NSTAR)=JD; TO(NSTAR)=UT; GAST(NSTAR)=STI; NAME(NSTAR)=SNAME
105 >       RA(NSTAR)=SRA; DE(NSTAR)=SDE
106 >C+     WRITE(1,'(/,* SBR STARCALL : *,A,* INIT, NSTAR=*)') SNAME,NSTAR
107 >       RETURN
108 >       END
109 >C
110 >       SUBROUTINE PLNTCALL(PNAME,SRA,RADIFF,SDE,DEDIFF,INYEAR,MON,DAY,
111 >      * UT,MUT,NSTAR,IFL)
112 >C     ------------------------------------------------------------
113 >C     TO INITIALIZE PLANET CALCULATIONS  (ALSO MOON IF SUTO IS GIVEN)
114 >C     INPUT:
115 >C           PNAME             THE NAME OF THE PLANET
116 >C           RA,RADIFF         RIGHT ASCENSION + DIFFERENCE (AT SUTO)
117 >C           DE,DEDIFF         DECLINATION + DIFFERENCE (AT SUTO)
118 >C           YEAR,MON,DAY,UT (HOURS)
119 >C                             IF NOT GIVEN THE CPU-TIME IS USED
120 >C           MUT               MOON UT; CATALOGUE-TIME
121 >C     OUTPUT:
122 >C           NSTAR             INDEX OF THE PLANET IN STAR-TABLE
123 >C           IFL =   0         IF NORMAL RETURN
124 >C                 100         STAR NOT KNOWN (ILLEGAL PARAMETERS)
125 >C     ************************************************************
126 >
127 >       PARAMETER KNOWN=6, MAX=16
128 >       PARAMETER DTR=1.7453293E-2, HTR=2.61799388E-1, RTD=57.295780
129 >       INTEGER YEAR,MON,DAY,ITIM(7)
130 >       CHARACTER PNAME*6, NAME*6
131 >       REAL JD,JDO
132 >       COMMON /STAR/JDO(MAX),TO(MAX),GAST(MAX),RA(MAX),DE(MAX),NAME(MAX)
133 >C
134 >       IFL=0; NSTAR=0; YEAR=INYEAR
135 >       L=LENGTH(PNAME)
136 >       IF(L.EQ.0) THEN; IFL=100; RETURN; ENDIF
137 >       IF(SRA.EQ.0.0.AND.SDE.EQ.0.0) THEN; IFL=100; RETURN; ENDIF
138 >       IF(YEAR.EQ.0) THEN
139 >          CALL CLOCK(ITIM); YEAR=ITIM(7); MON=ITIM(6); DAY=ITIM(5)
140 >          UT=ITIM(4)+ITIM(3)/60.; INYEAR=YEAR
141 >       ENDIF
142 >       IF(YEAR.LT.100) YEAR=YEAR+1900
143 >       JD=-693946.5+AINT(AINT(1461.0*(YEAR+(MON-14)/12))/4)+
144 >      + 367*(MON-2-12*((MON-14)/12))/12+(24002-12*YEAR-MON)/1200+DAY
145 >       IFREE=0
146 >       DO FOR I=KNOWN+1,10
147 >          IF(PNAME(1:L).EQ.NAME(I)(1:L)) THEN
148 >             IF(ABS(JD-JDO(I)).LE.1) THEN
149 >                OLD=24.; IF(PNAME(1:4).EQ.'MOON') OLD=1.
150 >                IF(ABS(UT-TO(I)).LT.OLD) THEN
151 >                   NSTAR=I; RETURN
152 >                ENDIF
153 >             ENDIF
154 >          ENDIF
155 >          IF(IFREE.EQ.0.AND.JDO(I).EQ.0.) IFREE=I
156 >       ENDDO
157 >C
158 >       CALL SIDRAL(JD,0.0,STI)
```

```
159 >        STI=STI+UT*0.2625161706
160 >        SRA=(HMSTOH(SRA)+(UT-SUTO)/24.*(RADIFF/3600.))*HTR
161 >        SDE=(HMSTOH(SDE)+(UT-SUTO)/24.*(DEDIFF/3600.))*DTR
162 >C
163 >        NSTAR=IFREE; IF(NSTAR.EQ.0) NSTAR=10
164 >        JDO(NSTAR)=JD; TO(NSTAR)=UT; GAST(NSTAR)=STI; NAME(NSTAR)=PNAME
165 >        RA(NSTAR)=SRA; DE(NSTAR)=SDE
166 >        RETURN
167 >        END
168 >C
169 >        SUBROUTINE STARINIT(NSTAR,INYEAR,MON,DAY,UT,IOTRM,IFL)
170 >C       ------------------------------------------------------
171 >C       TO INITIALIZE STAR CALCULATIONS
172 >C       INPUT: NSTAR: IF 1,2,...,6 = INDEX OF THE STAR TO BE INITIALIZED
173 >C                     (1=CAS A, 2=CYG A, 3=3C123, 4=TAU A, 5=VIR A, 6=ORINEB)
174 >C                       IF 0         = TO INITIALIZE ALL AVAILABLE STARS
175 >C                       IF -1        = TO INITIALIZE A NEW STAR.
176 >C              YEAR,MON,DAY,UT (IN HOURS)
177 >C                  IF DATE AND TIME IS NOT GIVEN (YEAR=0), THE TIME IS
178 >C                  READ FROM CPU-CLOCK AND IS RETURNED IN YEAR,MON,DAY
179 >C                  AND UT.
180 >C              IOTRM: I/O-TERMINAL    (=1 IN BACKGROUND)
181 >C                       MUST BE RESERVED FOR INPUT AND OUTPUT IN RT.
182 >C       DATA:  SRA1950(I), I=1,2,...,6
183 >C              SDE1950(I), I=1,2,...,6
184 >C              SNAME(I),   I=1,2,...,6
185 >C       OUTPUT :
186 >C       COMMON /STAR/JDO(MAX),TO(MAX),GAST(MAX),RA(MAX),DE(MAX),NAME(MAX)
187 >C             IFL =  0 IF NORMAL RETURN
188 >C                   99 IF USER STOP WANTED
189 >C                  100 STAR NOT KNOWN
190 >C                  101 NO SPACE IN STAR TABLE
191 >C       ************************************************************
192 >        PARAMETER KNOWN=6,MAX=16
193 >        PARAMETER  PI=3.1415927, PI2=6.283185307, PIPER2=1.57079633
194 >        PARAMETER DTR=1.7453293E-2, HTR=2.61799388E-1, RTD=57.295780
195 >        INTEGER YEAR,MON,DAY,ITIM(7)
196 >        REAL JDO,JD
197 >        CHARACTER*6 NEWSTAR,SNAME,NAME,OK*1
198 >        DIMENSION SRA1950(KNOWN),SDE1950(KNOWN),SNAME(KNOWN)
199 >        COMMON /STAR/JDO(MAX),TO(MAX),GAST(MAX),RA(MAX),DE(MAX),NAME(MAX)
200 >        DATA SRA1950/23.35358,19.962556, 4.565444, 5.525333,12.47136,
201 >     *                5.5463889/
202 >        DATA SDE1950/58.545833,40.583889,29.570278,21.988056,12.663611,
203 >     *               -5.4058333/
204 >        DATA SNAME/'CAS-A ','CYG-A ','3C123 ','TAU-A ','VIR-A ','ORINEB'/
205 >C
206 >  1     IFL=0; YEAR=INYEAR
207 >        IF(YEAR.EQ.0) THEN
208 >           CALL CLOCK(ITIM); YEAR=ITIM(7); MON=ITIM(6); DAY=ITIM(5)
209 >           UT=ITIM(4)+ITIM(3)/60.; INYEAR=YEAR
210 >        ENDIF
211 >        IF(YEAR.LT.100) YEAR=YEAR+1900
212 >C
213 >C       CHECK IF STAR COORDINATES ALREADY KNOWN;
214 >  2     IF(NSTAR.LT.0.OR.NSTAR.GT.KNOWN) THEN
215 >C          UNKNOWN STAR:
216 >           IF(IOTRM.LE.0) THEN; IFL=100; RETURN; ENDIF
217 >10         WRITE(IOTRM,'(/,"$1950-COORDINATES ? ")'); INPUT(IOTRM) OK
218 >           IF(OK.EQ.'S') THEN; IFL=99; RETURN; ENDIF
219 >           IF(OK.EQ.'Y') THEN;
220 >              CALL STARPRM(RA1950,DE1950,NEWSTAR,IOTRM,IFL)
221 >              IF(IFL.NE.0) RETURN
222 >              CALL STARSITR(RA1950,DE1950,YEAR,MON,DAY,UT,SRA,SDE,STI,JD)
223 >           ELSE
224 >              WRITE(IOTRM,'(/,"$EPHEMERIS DATA ? ")'); INPUT(IOTRM)OK
225 >              IF(OK.NE.'Y') THEN; IFL=99; RETURN; ENDIF
226 >              CALL PLNTPRM(YEAR,MON,DAY,UT,NEWSTAR,SRA,SDE,STI,JD,
227 >     *                     IOTRM,IFL)
228 >              IF(IFL.NE.0) RETURN
229 >           ENDIF
230 >           IFREE=KNOWN+1
231 >  3        DO WHILE(JDO(IFREE).NE.0)
232 >              IFREE=IFREE+1
233 >              IF(IFREE.GT.MAX) THEN
234 >                 WRITE(IOTRM,'(/," STAR TABLE FULL(*)')
235 >                 WRITE(IOTRM,'(13,"*=",A)')(I,NAME(I),I=1,MAX)
236 >                 WRITE(IOTRM,'(/,"$REPLACING STAR NUMBER : ")')
237 >                 INPUT(IOTRM) IFREE
238 >                 IF(IFREE.EQ.0.OR.IFREE.GT.MAX) THEN
239 >                    IFL=101; RETURN
240 >                 ENDIF
```

```
241 >                JDO(IFREE)=0.0
242 >             ENDIF
243 >          ENDDO
244 >          NSTAR=IFREE; JDO(NSTAR)=JD; TO(NSTAR)=UT;  GAST(NSTAR)=STI
245 >          RA(NSTAR)=SRA; DE(NSTAR)=SDE; NAME(NSTAR)=NEWSTAR
246 >       ELSE
247 >C        KNOWN STARS:
248 >    4     IS=1; ISL=KNOWN
249 >          IF(NSTAR.GT.0) THEN
250 >             IS=NSTAR; ISL=NSTAR
251 >          ENDIF
252 >          DO FOR I=IS,ISL
253 >             RA1950=SRA1950(I); DE1950=SDE1950(I)
254 >             CALL STARSITR(RA1950,DE1950,YEAR,MON,DAY,UT,SRA,SDE,STI,JD)
255 >             JDO(I)=JD; TO(I)=UT; GAST(I)=STI; NAME(I)=SNAME(I)
256 >             RA(I)=SRA; DE(I)=SDE
257 >          ENDDO
258 >       ENDIF
259 >       RETURN
260 >       END
261 >C
262 >       SUBROUTINE STARAZEL(NSITE,NSTAR,INYEAR,MON,DAY,UT,A,E,IOTRM,IFL)
263 >C      ------------------------------------------------------------
264 >C      TO CALCULATE AZIMUTH AND ELEVATION OF A STAR AT GIVEN SITE
265 >C      INPUT: NSITE = SITE NUMBER (USED AS INDEX TO GET COORDINATES)
266 >C                     (1=KIRUNA, 2=TROMSO, 3=SODANKYLA)
267 >C                   = -1 IF NEW SITE COORDINATES WILL BE GIVEN
268 >C                     (ON RETURN NSITE=4, IF NEW SITE IS DEFINED)
269 >C             NSTAR = STAR NUMBER (AS DEFINED IN STARINIT)
270 >C             YEAR,MON,DAY,UT (IN HOURS)
271 >C                     IF DATE IS NOT GIVEN (YEAR=0), IT IS READ FROM
272 >C                     CPU-CLOCK AND IS RETURNED IN YEAR,MON AND DAY
273 >C                     (UT IS NOT CHANGED!)
274 >C             IOTRM = I/O-TERMINAL    (=1 IN BACKGROUND)
275 >C                     MUST BE RESERVED FOR INPUT AND OUTPUT IN RT.
276 >C      DATA: SLAT(I)= LATITUDES OF THE SITES
277 >C            SLNG(I)= LONGITUDES OF THE SITES
278 >C      COMMON /STAR/JDO(MAX),TO(MAX),GAST(MAX),RA(MAX),DE(MAX),NAME(MAX)
279 >C      OUTPUT: AZ    = STAR AZIMUTH AT YEAR,MON,DAY,UT
280 >C              EL    = STAR ELEVATION AT YEAR,MON,DAY,UT
281 >C              IFL   =  0 IF NORMAL RETURN
282 >C                      99 IF USER STOP WANTED
283 >C                     102 ILLEGAL STAR NUMBER
284 >C                     103 STAR NOT INITIALIZED
285 >C                     104 ILLEGAL SITE NUMBER
286 >C      ************************************************************
287 >       PARAMETER KNOWN=6,MAX=16
288 >       PARAMETER  P1=3.1415927, P12=6.283185307, PIPER2=1.57079633
289 >       PARAMETER DTR=1.7453293E-2, HTR=2.61799388E-1, RTD=57.295780
290 >       INTEGER YEAR,MON,DAY,ITIM(7)
291 >       REAL LATGD,LONG,JD,JDO
292 >       DIMENSION SLAT(3),SLNG(3)
293 >       CHARACTER NAME*6
294 >       COMMON /STAR/JDO(MAX),TO(MAX),GAST(MAX),RA(MAX),DE(MAX),NAME(MAX)
295 >C      DATA SLAT/ 67.863000, 69.583000, 67.363889/
296 >C      DATA SLNG/-20.440000,-19.210000,-26.635278/
297 >       DATA SLAT/ 1.184432790    , 1.214452453    , 1.175721660    /
298 >       DATA SLNG/-0.356745299    ,-0.335277749    ,-0.464873298    /
299 >C
300 >    1  IFL=0; YEAR=INYEAR
301 >       IF(NSTAR.LE.0.OR.NSTAR.GT.MAX) THEN; IFL=102; RETURN; ENDIF
302 >       IF(JDO(NSTAR).EQ.0.) THEN; IFL=103; RETURN; ENDIF
303 >       IF(YEAR.EQ.0) THEN
304 >          CALL CLOCK(ITIM); YEAR=ITIM(7); MON=ITIM(6); DAY=ITIM(5)
305 >          INYEAR=YEAR
306 >       ENDIF
307 >       IF(YEAR.LT.100) YEAR=YEAR+1900
308 >       IF(NSITE.GT.0.AND.NSITE.LE.3) THEN
309 >          LATGD=SLAT(NSITE)
310 >          LONG=SLNG(NSITE)
311 >       ELSEIF(NSITE.EQ.-1) THEN
312 >          IF(IOTRM.GT.0) THEN
313 >             CALL SITEPRM(GEOLAT,GEOLONG,IOTRM,IFL)
314 >             IF(IFL.NE.0) RETURN
315 >             LATGD=GEOLAT*DTR
316 >             LONG=GEOLONG*DTR
317 >             NSITE=4
318 >          ELSE
319 >             IFL=104; RETURN
320 >          ENDIF
321 >       ENDIF
322 >    2  JD=-693946.5+AINT(AINT(1461.0*(YEAR+(MON-14)/12))/4)+
```

```
323 >      + 367*(MON-2-12*((MON-14)/12))/12+
324 >      + (24002-12*YEAR-MON)/1200 +DAY
325 >        UTDIF=(JD-JD0(NSTAR))*24.+UT-T0(NSTAR)
326 >        H=GAST(NSTAR)-RA(NSTAR)-LONG
327 >  3     H=H+UTDIF*0.2625161706
328 >        DEC=DE(NSTAR)
329 >  4     CALL COORD(PI,PIPER2-LATGD,0.0,LATGD,H,DEC,AZ,EL)
330 >C       REFRACTION:            REMOVED 1980.06.16
331 >C       ELT=EL
332 >C       CALL REFR(ELT,EL)
333 >C       REFRACTION IS INCLUDED IN OFFSET-VALUES MEASURED AFTER 1980.06.16
334 >        A=AZ*RTD; E=EL*RTD
335 >        IF(A.LT.0) A=A+360.
336 >        RETURN
337 >        END
338 >C
339 >        SUBROUTINE STARRADE(NSITE,INYEAR,MON,DAY,UT,A,E,RA,DE,IOTRM,IFL)
340 >C       ------------------------------------------------------------
341 >C       TO CALCULATE RIGHT ASCENSION AND DECLINATION ON EPOCH 1950
342 >C       WHICH CORRESPOND GIVEN AZIMUTH AND ELEVATION OF ANTENNA.
343 >C       THIS SUBROUTINE IS USED IN ANTENNA BACKGROUND CHECK.
344 >C       INPUT: NSITE = SITE NUMBER (USED AS INDEX TO GET COORDINATES)
345 >C                      (1=KIRUNA, 2=TROMSO, 3=SODANKYLA)
346 >C                    = -1 IF NEW SITE COORDINATES WILL BE GIVEN
347 >C                      (ON RETURN NSITE=4, IF NEW SITE IS DEFINED)
348 >C              YEAR,MON,DAY,UT (IN HOURS)
349 >C                      IF DATE IS NOT GIVEN (YEAR=0), IT IS READ FROM
350 >C                      CPU-CLOCK AND IS RETURNED IN YEAR,MON AND DAY
351 >C                      (UT IS NOT CHANGED!)
352 >C              AZ    = AZIMUTH AT YEAR,MON,DAY,UT
353 >C              EL    = ELEVATION AT YEAR,MON,DAY,UT
354 >C              IOTRM = I/O-TERMINAL   (=1 IN BACKGROUND)
355 >C                      MUST BE RESERVED FOR INPUT AND OUTPUT IN RT.
356 >C       DATA: SLAT(I)= LATITUDES OF THE SITES
357 >C             SLNG(I)= LONGITUDES OF THE SITES
358 >C       OUTPUT: RA    = 1950-RIGHT ASCENSION
359 >C               EL    = 1950-DECLINATION
360 >C               IFL   =  0 IF NORMAL RETURN
361 >C                        99 IF USER STOP WANTED
362 >C                       104 ILLEGAL SITE NUMBER
363 >C       ************************************************************
364 >        PARAMETER  PI=3.1415927, PI2=6.283185307, PIPER2=1.57079633
365 >        PARAMETER DTR=1.7453293E-2, HTR=2.61799388E-1, RTD=57.295780
366 >        INTEGER YEAR,MON,DAY,ITIM(7)
367 >        REAL LATGD,LONG,JD
368 >        DIMENSION SLAT(3),SLNG(3)
369 >        DATA SLAT/ 67.863000, 69.583000, 67.363889/
370 >        DATA SLNG/-20.440000,-19.210000,-26.635278/
371 >C
372 >        IFL=0; YEAR=INYEAR
373 >        IF(YEAR.EQ.0) THEN
374 >           CALL CLOCK(ITIM); YEAR=ITIM(7); MON=ITIM(6); DAY=ITIM(5)
375 >           INYEAR=YEAR
376 >        ENDIF
377 >        IF(YEAR.LT.100) YEAR=YEAR+1900
378 >        IF(NSITE.GT.0.AND.NSITE.LE.3) THEN
379 >           LATGD=SLAT(NSITE)*DTR
380 >           LONG=SLNG(NSITE)*DTR
381 >        ELSEIF(IOTRM.GT.0) THEN
382 >           CALL SITEPRM(GEOLAT,GEOLONG,IOTRM,IFL)
383 >           IF(IFL.NE.0) RETURN
384 >           LATGD=GEOLAT*DTR
385 >           LONG=GEOLONG*DTR
386 >           NSITE=4
387 >        ELSE
388 >           IFL=104; RETURN
389 >        ENDIF
390 >        AZ=A*DTR; EL=E*DTR
391 >        JD=-693946.5+AINT(AINT(1461.0*(YEAR+(MON-14)/12))/4)+
392 >      + 367*(MON-2-12*((MON-14)/12))/12+
393 >      + (24002-12*YEAR-MON)/1200 +DAY
394 >        CALL SIDRAL(JD,UT,GMST)
395 >        CALL COORD(PI,PIPER2-LATGD,0.0,LATGD,AZ,EL,H,DE)
396 >        RA=GMST-H-LONG
397 >        IF(RA.LT.0.0)RA=RA+PI2
398 >C       RA AND DE FOR 1950:
399 >        RA=RA/HTR; DE=DE*RTD
400 >        CALL STARREW(RA,DE,YEAR,MON,DAY,UT,RA1950,DE1950)
401 >        RA=RA1950/HTR; DE=DE1950*RTD
402 >        RETURN
403 >        END
404 >C
```

```
405 >       SUBROUTINE STARPRM(RA1950,DE1950,NEWSTAR,IOTRM,IFL)
406 >C      ------------------------------------------------------
407 >C      TO GET 1950 COORDINATES FOR A NEW STAR
408 >C      INPUT:  IOTRM   =INPUT/OUTPUT TERMINAL
409 >C              NEWSTAR= NAME OF THE STAR (6 CHARACTERS)
410 >C      OUTPUT: RA1950 = RIGHT ASCENSION OF THE STAR
411 >C              DE1950 = DECLINATION OF THE STAR
412 >C              IFL    =  0 IF NORMAL RETURN
413 >C                        99 IF USER STOP WANTED
414 >C      ******************************************************
415 >       CHARACTER NEWSTAR*6, NAME*6, OK*1
416 >       OK='!'; IFL=0
417 >10     IF(OK.EQ.'S') THEN; IFL=99; RETURN; ENDIF
418 >       WRITE(IOTRM,'(/,$,$ NAME OF THE SOURCE : *)')
419 >       INPUT(IOTRM) NEWSTAR
420 >       WRITE(IOTRM,'(/,$,$RIGHT ASCENSION (HOUR,MINSEC) : *)')
421 >       INPUT(IOTRM) RA1950
422 >       WRITE(IOTRM,'(/,$,$     DECLINATION (DEGR,MINSEC) : *)')
423 >       INPUT(IOTRM) DE1950
424 >       WRITE(IOTRM,'(/,$1H$,A,2F13.7,$,"  OK ? *)') NEWSTAR,RA1950,DE1950
425 >       INPUT(IOTRM) OK; IF(.NOT.(OK.EQ.'Y'.OR.OK.EQ.'0')) GOTO 10
426 >       RA1950=HMSTOH(RA1950)
427 >       DE1950=HMSTOH(DE1950)
428 >       RETURN
429 >       END
430 >C
431 >      SUBROUTINE PLNTPRM(INYEAR,MON,DAY,UT,NEWSTAR,SRA,SDE,STI,JD,
432 >     *                   IOTRM,IFL)
433 >C      ------------------------------------------------------------
434 >C      TO GET COORDINATES FOR PLANET AT GIVEN DAY AND TIME
435 >C      TAKEN FROM THE EPHEMERIS.
436 >C      IF TIME IS NOT SPECIFIED, IT IS READ FROM CPU CLOCK
437 >C            SRA   RIGHT ASCENSION ON YEAR,MON,DAY,UT
438 >C            SDE   DECLINATION            - " -
439 >C            STI   GREENWICH APPARENT SIDEREAL TIME   - " -
440 >C            JD    JULIAN DAY NUMBER - 2415020 (= JULIAN DAY NUMBER
441 >C                  AT DATE 1900 JAN 0 12H)
442 >C            IFL = 0 IF NORMAL RETURN
443 >C                  99 IF USER STOP WANTED
444 >C      ************************************************************
445 >       PARAMETER DTR=1.7453293E-2, HTR=2.61799388E-1, RTD=57.295780
446 >       INTEGER YEAR,MON,DAY,ITIM(7)
447 >       CHARACTER NEWSTAR*6,OK*1; REAL JD
448 >       OK='!'; IFL=0; YEAR=INYEAR
449 >       IF(YEAR.EQ.0) THEN
450 >          CALL CLOCK(ITIM); YEAR=ITIM(7); MON=ITIM(6); DAY=ITIM(5)
451 >          UT=ITIM(4)+(ITIM(3)+1)/60.; INYEAR=YEAR
452 >       ENDIF
453 >       IF(YEAR.LT.100) YEAR=YEAR+1900
454 >       JD=-693946.50+AINT(AINT(1461.0*(YEAR+(MON-14)/12))/4)+
455 >     + 367*(MON-2-12*((MON-14)/12))/12+
456 >     + (24002-12*YEAR-MON)/1200 +DAY
457 >       CALL SIDRAL(JD,0.0,STI)
458 >       STH=STI/HTR; STH=HTOHMS(STH)
459 >10     IF(OK.EQ.'S') THEN; IFL=99; RETURN; ENDIF
460 >       WRITE(IOTRM,'(/,$,$ NAME OF THE SOURCE : *)')
461 >       INPUT(IOTRM) NEWSTAR
462 >       MUT=0; IF(NEWSTAR.EQ.'MOON') MUT=UT
463 >       WRITE(IOTRM,'(/,$16,2(*,*,J2),I4,$," UT :*)') YEAR,MON,DAY,MUT
464 >       WRITE(IOTRM,'(/,$1H$,$," GAST (AT 0 UT) : ",F12.7)') STH
465 >       WRITE(IOTRM,'(/,$1H$,$,"RA (HOUR,MINSEC),DIFF (SEC) : *)')
466 >       INPUT(IOTRM) SRA,RADIFF
467 >       WRITE(IOTRM,'(/,$1H$,$,"DE (DEGR,MINSEC),DIFF (SEC) : *)')
468 >       INPUT(IOTRM) SDE,DEDIFF
469 >       WRITE(IOTRM,'(/,$1H$,F13.7,2(F13.7,F10.3),$,"  OK? *)')STH,SRA,
470 >     * RADIFF,SDE,DEDIFF
471 >       INPUT(IOTRM) OK; IF(.NOT.(OK.EQ.'Y'.OR.OK.EQ.'0')) GOTO 10
472 >       STI=STI+UT*0.2625161706
473 >       SRA=(HMSTOH(SRA)+(UT-MUT)*(RADIFF/3600.))*HTR
474 >       SDE=(HMSTOH(SDE)+(UT-MUT)*(DEDIFF/3600.))*DTR
475 >       RETURN
476 >       END
477 >C
478 >       SUBROUTINE SITEPRM(GEOLAT,GEOLONG,IOTRM,IFL)
479 >C      ------------------------------------------
480 >C      TO GET GEOGRAPHIC COORDINATES FOR A SITE
481 >C            IFL = 0 IF NORMAL RETURN
482 >C                  99 IF USER STOP WANTED
483 >C      ******************************************
484 >       CHARACTER OK*1
485 >       OK='!'; IFL=0
486 >       WRITE(IOTRM,'(/,$,$UNKNOWN SITE, DO YOU SPECIFY IT ? *)')
```

```
      INPUT(IOTRM) OK; IF(OK.NE.'Y') THEN; IFL=99; RETURN; ENDIF
10    IF(OK.EQ.'S') THEN; IFL=99; RETURN; ENDIF
      WRITE(IOTRM,'(/,$LATITUDE  (DEGR,MIN,SEC) : *)')
      INPUT(IOTRM) GEOLAT,MIN,SEC
      IF(AMOD(GEOLAT,1.).EQ.0) GEOLAT=GEOLAT+MIN/60.+SEC/3600.
      WRITE(IOTRM,'(/,$LONGITUDE (DEGR,MIN,SEC) : *)')
      INPUT(IOTRM) GEOLONG,MIN,SEC
      IF(AMOD(GEOLONG,1.).EQ.0) GEOLONG=GEOLONG+MIN/60.+SEC/3600.
      WRITE(IOTRM,'(/,1H$,2F13.7,*  OK? *)') GEOLAT,GEOLONG
      INPUT(IOTRM) OK; IF(.NOT.(OK.EQ.'Y'.OR.OK.EQ.'O')) GOTO 10
      RETURN
      END
C
      SUBROUTINE STARNAM(NSTAR,SNAME)
C     ------------------------------
C     TO RETURN NAME OF STAR NSTAR
C     ****************************
      PARAMETER MAX=16
      CHARACTER SNAME*6,NAME*6
      REAL JDO
      COMMON /STAR/JDO(MAX),TO(MAX),GAST(MAX),RA(MAX),DE(MAX),NAME(MAX)
      SNAME=''
      IF(NSTAR.LE.0.OR.NSTAR.GT.MAX) RETURN
      SNAME=NAME(NSTAR)
      RETURN; END
C
      SUBROUTINE STARSITR(RA1950,DE1950,YEAR,MON,DAY,UT,RA,DE,GAST,JD)
C     ----------------------------------------------------------------
C     TO CALCULATE THE RIGHT ASCENSION AND THE DECLINATION OF A STAR AND
C     GREENWICH APPARENT SIDEREAL TIME AT SPECIFIED TIME
C     INPUT: RA1950,DE1950 = CATALOGUED COORDINATES OF THE STAR
C            YEAR,MON,DAY,UT
C     OUTPUT: RA,DE   = STAR COORDINATES AT SPECIFIED TIME
C             GAST    = GREENWICH APPARENT SIDEREAL TIME
C             JD      = JULIAN DAY NUMBER - 2415020 (= JULIAN DAY NUMBER
C                       AT DATE 1900 JAN 0 12H)
C     ****************************************************************
      PARAMETER  PI=3.1415927, PI2=6.283185307, PIPER2=1.57079633
      PARAMETER DTR=1.7453293E-2, HTR=2.61799388E-1, RTD=57.295780
      INTEGER YEAR,DAY
      REAL JD,NORM
      DIMENSION S1950(3),S1(3),S2(3),VE(3)
      DIMENSION SMQBD(3),SMCBD(3),SMCED(3),STQED(3)
C
      IM=1;GOTO 11
C
      ENTRY STARREW(RA1950,DE1950,YEAR,MON,DAY,UT,RA,DE)
C     --------------------------------------------------
      IM=-1
C
11    S1950(1)=COS(DTR*DE1950)*COS(HTR*RA1950)
      S1950(2)=COS(DTR*DE1950)*SIN(HTR*RA1950)
      S1950(3)=SIN(DTR*DE1950)
      JD=-693946.5+AINT(AINT(1461.0*(YEAR+(MON-14)/12))/4)+
     + 367*(MON-2-12*((MON-14)/12))/12+
     +(24002-12*YEAR-MON)/1200 +DAY
C     PRECESSION
      CALL PRECES(JD,UT,ZETA,Z,THETA,EPS)
      CALL RZ(S1950,-ZETA*IM,S1)
      CALL RY(S1,THETA*IM,S2)
      CALL RZ(S2,-Z*IM,SMQBD)
C     ROTATION TO THE ECLIPTIC PLANE
      CALL RX(SMQBD,EPS*IM,SMCBD)
C     ABERRATION
      CALL ERTVEL(JD,UT,VE)
      NORM=0.0
      DO 1 I=1,3
      SMCED(I)=SMCBD(I)+VE(I)
      NORM=NORM+SMCED(I)*SMCED(I)
1     CONTINUE
      NORM=SQRT(NORM)
      DO 2 I=1,3
      SMCED(I)=SMCED(I)/NORM
2     CONTINUE
C     ROTATION TO THE EQUATORIAL PLANE
      CALL RX(SMCED,-EPS*IM,STQED)
C     SIDEREAL TIME
3     CALL SIDRAL(JD,UT,GMST)
      GAST=GMST
      RA=ATAN2(STQED(2),STQED(1)); RA=AMOD(RA+2*PI2,PI2)
      DE=ATAN(STQED(3)/SQRT(STQED(1)*STQED(1)+STQED(2)*STQED(2)))
      RETURN
```

```
      END
C
      SUBROUTINE PRECES(JD,UT,ZETA,Z,THETA,EPS)
C     ------------------------------------------------------------
C     PROGRAM TO CALCULATE ASTRONOMICAL CONSTANTS THAT ARE CHANGING
C     DUE TO PRECESSION. TC IS TROPICAL CENTURIES (36524.21988 DAYS)
C     ELAPSED SINCE THE BEGINNING OF THE BESSELIAN YEAR DENOTED 1950.0
C     (=JULIAN DAY NUMBER 2433282.423). EXPRESSIONS ARE TAKEN FROM TECH.
C     REP. 32-1306, CONST. AND REL. INF. FOR ASTR. CAL. 1968,
C     JPL PAGE 9.                          H.NES, NTH, 1977.11.01
C      3600*180/PI=206264.8062
C     ************************************************************
      REAL JD
      TC=JD*27.379093E-6+UT*1.1407955E-6-66.621065
      TC=JD/36524.22+UT/876581.28-.5
      TC2=TC*TC; TC3=TC*TC2
      ZETA =2304.952*TC+0.3022*TC2+0.0180*TC3
      ZETA =ZETA/206264.8062
      Z    =2304.952*TC+1.0951*TC2+0.0183*TC3
      Z    =Z/206264.8062
      THETA=2004.257*TC-0.4268*TC2-0.0418*TC3
      THETA=THETA/206264.8062
      EPS  =-46.850*TC-0.0034*TC2+0.0018*TC3
      EPS  =EPS/206264.8062+.4092062119
      RETURN
      END
C
      SUBROUTINE ERTVEL(JD,UT,VE)
C     ------------------------------------------------------------
C     SUBROUTINE TO CALCULATE THE VELOCITY OF THE EARTH IN THE MEAN
C     ECLIPTIC COORDINATE SYSTEM OF DATE WITH ORIGIN AT THE SOLAR
C     SYSTEM BARY CENTER. NEGLECTS 'ELLIPTIC ABERRATION' COMPONENT OF
C     VELOCITY. ECCENTRICITY: ACCURACY APPROX. 0.05 DEGREES WHICH
C     GIVES APPROX. 0.02 ARCSEC. ABERRATION CORRECTION ERROR. THE
C     NUMBER OF JULIAN CENTURIES (=36525 DAYS) SINCE THE GREGORIAN
C     DATE 1900 JAN 0 12H (=JULIAN DAY NUMBER 2415020) IS FIRST
C     COMPUTED. THE VELOCITY VECTOR IS THEN COMPUTED FROM EXPRESSIONS
C     GIVEN IN EXPLANATORY SUPPLEMENT, PAGE 98.
C                                          H.NES, NTH, 1977.11.01
C     ************************************************************
      PARAMETER  PI=3.1415927, PI2=6.283185307, PIPER2=1.57079633
      REAL JD,JC,JC2
      DIMENSION VE(3)
      JC=JD*27.378508E-6+UT*1.1407712E-6-66.119644;JC2=JC*JC
      JC=(JD+UT/24.)/36525.
      SUNLNG=4.881627935+628.331951*JC+5.279621E-6*JC2
      SUNLNG=AMOD(SUNLNG,PI2)
      ERTLNG=SUNLNG+PI
      ECC    =0.01675104-0.00004180*JC-0.000000126*JC2
      PERLNG=4.908229662+.0300052642*JC+7.902463004E-6*JC2+
     * 58.17764175E-9*JC*JC2
      PERLNG=AMOD(PERLNG,PI2)
      PERLNG=PERLNG+PI
      DELLNG=ECC*SIN(ERTLNG-PERLNG)*2.0
      ERTLNG=ERTLNG+DELLNG
      VE(1)=-SIN(ERTLNG)*20.4955/206264.8062
      VE(2)= COS(ERTLNG)*20.4955/206264.8062
      VE(3)= 0.0
      RETURN
      END
C
      SUBROUTINE SIDRAL(JD,UT,GMST)
C     ------------------------------------------------------------
C     SUBROUTINE TO CALCULATE GREENWICH MEAN SIDEREAL TIME FROM THE
C     JULIAN DAY NUMBER. THE NUMBER OF JULIAN CENTURIES (=36525 DAYS)
C     SINCE THE GREGORIAN DATE 1900 JAN 0 12H (=JULIAN DAY NUMBER 2415020)
C     IS FIRST CALCULATED. THE EXPRESSION GIVEN IN EXPLANATORY
C     SUPPLEMENT, PAGE 75 IS USED.     H.NES, NTH, 1977.11.01
C     ************************************************************
      PARAMETER  PI=3.1415927, PI2=6.283185307, PIPER2=1.57079633
      REAL JD,JC
      JC=JD*27.378508E-6+UT*1.1407712E-6-66.119644
      JC=(JD+UT/24.)/36525.
      GMST=23925.836+8640184.542*JC+0.0929*JC*JC
      GMST=GMST/3600./24.
      GMST=AMOD(GMST,1.)+AMOD(UT/24.,1.)
      GMST=AMOD(GMST,1.)*PI2
      RETURN
      END
C
       SUBROUTINE ROTATIONS
C     S1 IS THE CARTESIAN COORDINATES OF A VECTOR, S2 IS THE
```

```
651 >C      COORDINATES OF THE SAME VECTOR WHEN THE COORDINATE SYSTEM IS
652 >C      ROTATED A POSITIVE ANGLE 'FI' AROUND THE X-, Y- OR Z-AXIS.
653 >C                                          H.NES, NTH, 1977.11.01
654 >        DIMENSION S1(3),S2(3)
655 >        ENTRY RX(S1,FI,S2)
656 >       S2(1)= S1(1)
657 >       S2(2)= COS(FI)*S1(2)+SIN(FI)*S1(3)
658 >       S2(3)=-SIN(FI)*S1(2)+COS(FI)*S1(3)
659 >       RETURN
660 >C
661 >        ENTRY RY(S1,FI,S2)
662 >       S2(1)= COS(FI)*S1(1)-SIN(FI)*S1(3)
663 >       S2(2)= S1(2)
664 >       S2(3)= SIN(FI)*S1(1)+COS(FI)*S1(3)
665 >       RETURN
666 >C
667 >        ENTRY RZ(S1,FI,S2)
668 >       S2(1)= COS(FI)*S1(1)+SIN(FI)*S1(2)
669 >       S2(2)=-SIN(FI)*S1(1)+COS(FI)*S1(2)
670 >       S2(3)= S1(3)
671 >       RETURN
672 >       END
673 >C
674 >       SUBROUTINE COORD(A0,B0,AP,BP,A1,B1,A2,B2)
675 >C       ------------------------------------------------------------
676 >C         THIS SUBROUTINE
677 >C         IS FULLY DESCRIBED IN:BALL,J.A.,SOME FORTRAN SUBPROGRAMS USED
678 >C         IN ASTRONOMY,TECHNICAL NOTE 1969-42,LINCOLN LABRATORY,MIT,MASS
679 >C       ************************************************************
680 >        ARSIN(X)=ATAN(X/SQRT(1-X*X))
681 >       SB0=SIN(B0)
682 >       CB0=COS(B0)
683 >       SBP=SIN(BP)
684 >       CBP=COS(BP)
685 >       SB1=SIN(B1)
686 >       CB1=COS(B1)
687 >       SB2=SBP*SB1+CBP*CB1*COS(AP-A1)
688 >       B2=ARSIN(SB2)
689 >        CB2=COS(B2)
690 >       SAA=SIN(AP-A1)*CB1/CB2
691 >       CAA=(SB1-SB2*SBP)/(CB2*CBP)
692 >C      CBB=SB0/CBP
693 >C       SBB=SIN(AP-AU)*CB0
694 >       TA202=(1.0-CAA)/SAA
695 >       A2=2.*ATAN(TA202)
696 >       RETURN
697 >       END
698 >C
699 >        SUBROUTINE REFR(ELT,ELA)
700 >C       ------------------------------------------------
701 >C       THIS SUBROUTINE ADDS REFRACTION TO THE ELEVATION ANGLE
702 >C       IF ELEVATION IS GREATER THAN 5 DEGREES.
703 >C       ASTROPHYSICAL QUANTITIES,THIRD EDITION,C.W.ALLEN,PAGE 124.
704 >C       ************************************************
705 >        REAL ELT,ELA
706 >        ELA=ELT
707 >        IF(ELT/3.14*180.LT.5.) RETURN
708 >        T=COS(ELT)/SIN(ELT)
709 >        ELA=2.82646376E-4*T-3.2482516E-7*T*T*T+ELT
710 >9       RETURN
711 >        END
712 >C
713 >        SUBROUTINE HOURS(RA,IRAH,IRAM,RAS,IRAS)
714 >C             (S.TALLQUIST)
715 >        IRAH=INT(RA)
716 >        RAM=(RA-IRAH)*60.
717 >        IRAM=INT(RAM)
718 >        RAS=(RAM-IRAM)*60.
719 >        IRAS=RAS+0.5
720 >        RETURN
721 >        END
722 >C
723 >        FUNCTION HMSTOH(H)
724 >C       ******************
725 >        XH=AINT(H)
726 >        XM=AINT((H-XH)*100.)
727 >        XS=(H-XH-XM*0.01)*10000.
728 >        HMSTOH=XH+XM/60.+XS/3600.
729 >        RETURN
730 >        END
731 >C
732 >        FUNCTION HTOHMS(H)
```

```
733 >C      ******************
734 >       XH=AINT(H)
735 >       XM=(H-XH)*60.
736 >       XS=(XM-AINT(XM))*60.
737 >       HTOHMS=XH*AINT(XM)/100.+XS/10000.
738 >       RETURN
739 >       END
740 >C
741 >       INTEGER FUNCTION LENGTH(A)
742 >       CHARACTER*1 A
743 >       L=LEN(A)
744 >       DOFOR I=1,L
745 >          J=L+1-I
746 >          IF(A(J:J).NE.' ') THEN
747 >             LENGTH=J
748 >             RETURN
749 >          ENDIF
750 >       ENDDO
751 >       LENGTH=0
752 >       RETURN
753 >       END
754 >C
```

CONTENTS OF FILE : (ANTENNA-EISCAT)LIBRARY:SYMB;1

```
  1 >       SUBROUTINE WOPEN(IF,FILENAME,ACCESS,STATUS,ISEC,MAX,IER)
  2 >C      ********************************************************
  3 >C      TO OPEN A FILE, WHICH MIGHT BE OPENED BY SOMEBODY ELSE ALSO.
  4 >C      IF OPENING IS NOT SUCCESFUL AT FIRST TIME, IT IS TRIED
  5 >C      AGAIN WITH ISEC INTERVALS UNTILL SUCCEEDED OR MAXIMUM
  6 >C      MAX TIMES. IF FILE IS NOT OPENED, IER TELLS THE REASON.
  7 >C      ********************************************************
  8 >       CHARACTER FILENAME*1, ACCESS*1,STATUS*1,MES*80
  9 >       COMMON /SITE/IMAT,CTERM; INTEGER CTERM
 10 >       DOFOR I=1,MAX
 11 >          OPEN(IF,FILE=FILENAME,ACCESS=ACCESS,STATUS=STATUS,ERR=100)
 12 >100       IER=ERRCODE
 13 >          IF (IER.EQ.0) RETURN
 14 >          IF(I.NE.MAX) CALL HOLD(ISEC,2)
 15 >       ENDDO
 16 >       WRITE(MES,'(* OPEN ERROR : *,I4,*   FILE : *,A)') IER,FILENAME
 17 >       CALL OUTMES(CTERM,MES); MES=''
 18 >       RETURN
 19 >       END
 20 >C
 21 >       SUBROUTINE OUTSTR(A)
 22 >       CHARACTER*1 A
 23 >       COMMON /SITE/IMAT,CTERM,ERRDEV; INTEGER CTERM,ERRDEV
 24 >       CALL OUTMES(CTERM,A)
 25 >       RETURN
 26 >       END
 27 >C
 28 >       SUBROUTINE OUTMES(LUNIT,MESSAGE)
 29 >C      ********************************************************
 30 >C      TO WRITE A MESSAGE TO TERMINAL OR OTHER LOGICAL UNIT LUNIT.
 31 >C      LUNIT IS RELEASED IF IT IS RESERVED BY A BACKGROUND PROGRAM.
 32 >C        LBGADDR IS THE ADDRESS OF FIRST BACKGROUND RT-PROGRAM
 33 >C        LRTADDR IS THE ADDRESS OF FIRST USER RT-PROGRAM
 34 >C      ********************************************************
 35 >       CHARACTER MESSAGE*1
 36 >       COMMON /SITE/IMAT
 37 >CR     INTEGER WHDEV,RESRV
 38 >C
 39 >       IUNIT=LUNIT; IF(IUNIT.EQ.0) IUNIT=1
 40 >CR     IPROG=0
 41 >CR     IF(IMAT.EQ.1) THEN
 42 >CR        GOTO 10
 43 >CR     ELSEIF(IMAT.EQ.2) THEN
 44 >CR        GOTO 10
 45 >CR     ELSEIF(IMAT.EQ.4) THEN
 46 >CR        LBGADDR=302028;    LRTADDR=311268
 47 >CR     ELSE
 48 >CR        GOTO 10
 49 >CR     ENDIF
 50 >C ...  FIRST TRY TO RESERVE IUNIT WITHOUT WAITING;
 51 >CR     ISTAT=RESRV(IUNIT,1,1)
 52 >CR     IF(ISTAT.NE.0) THEN
 53 >C ...     IF RESERVING UNSUCCESFUL, ASK WHO HAS IT;
 54 >CR        IPROG=WHDEV(IUNIT,1)
 55 >CR        IF(IPROG.GE.LBGADDR.AND.IPROG.LT.LRTADDR) THEN
```

```
56 >C ...          IUNIT IS PROPABLY RESERVED BY A BACKGROUND PROGRAM, STEAL IT:
57 >CR             CALL PRLS(IUNIT,1)
58 >CR             CALL RESRV(IUNIT,1,0)
59 >CR          ELSEIF(IPROG.LT.0) THEN
60 >C ...          MAYBE IUNIT IS ILLEGAL, WRITE AN ERROR MESSAGE:
61 >CR             CALL ERMON(2H60,IPROG)
62 >CR             RETURN
63 >CR          ELSE
64 >C ...          IUNIT PROPABLY RESERVED BY ANOTHER RT-PROGRAM, WAIT:
65 >CR             IPROG=0
66 >CR 10          CALL RESRV(IUNIT,1,0)
67 >CR          ENDIF
68 >CR       ENDIF
69 >C
70 >         WRITE(IUNIT,'(1X,A)') MESSAGE(1:-1)
71 >C
72 >CR       CALL RELES(IUNIT,1)
73 >CR       IF(IPROG.GT.0) THEN
74 >C ...       TRY TO GIVE IUNIT BACK TO THE PROGRAM WHO HAD IT:
75 >CR          DO FOR I=1,10
76 >CR             ISTAT=PRSRV(IUNIT,1,IPROG)
77 >CR             IF(ISTAT.EQ.0) RETURN
78 >C ...          IF SOMEBODY HAS TAKEN IT BETWEEN, KEEP TRYING 2 SECONDS:
79 >CR             CALL HOLD(10,1)
80 >CR          ENDDO
81 >C ...       THIS ERROR MESSAGE SHOULD NEWER COME:
82 >CR          CALL ERMON(2H61,ISTAT)
83 >CR       ENDIF
84 >         RETURN
85 >         END
```

```
 07.56.55     29 SEPTEMBER   1980   EISCAT SODANKYLA
VERSION 29.07.15.A

ENTER KI
PASSWORD:
OK
PROJECT NUMBER:

@(ANTE)ANTCAL-CONTROL

ANTENNA OFFSET ANALYSIS          29 SEPTEMBER 1980     EISCAT SODANKYLA
WRITE HELP IF YOU NEED IT !
ANTENNA OFFSET ANALYSIS          29 SEPTEMBER 1980     EISCAT SODANKYLA
******************************   INITIALIZED  ******************************

COMND ! DEF-ANTC-INP
INPUT FILE GENERATION FOR THE DATA GATHERING PROGRAM

CALIBRATION INPUT FILE ! (KT)ANTCAL-IN-STNDRD
FILE NAME= (KT)ANTCAL-IN-STNDRD:DATA
PERIODICALLY ? (Y/N) ! Y
INITIAL OFFSETS FROM MODEL? (Y/N) ! Y
NUMBER OF STARS (MAX 20, DIFFERENT 10) ! 16
DELAY BETWEEN STARS (MINUTES) !
 1. SOURCE INDEX ! 1
 2. SOURCE INDEX ! 1
 3. SOURCE INDEX ! 2
 4. SOURCE INDEX ! 2
 5. SOURCE INDEX ! 5
 6. SOURCE INDEX ! 5
 7. SOURCE INDEX ! 5
 8. SOURCE INDEX ! 6
 9. SOURCE INDEX ! 6
10. SOURCE INDEX ! 6
11. SOURCE INDEX ! 4
12. SOURCE INDEX ! 4
13. SOURCE INDEX ! 3
14. SOURCE INDEX ! 3
15. SOURCE INDEX ! 7

AVAILABLE STARS ARE!
1=CAS-A, 2=CYG-A, 3=3C123, 4=TAU-A, 5=VIR-A, 6=ORINEB ,-1 =1950-COORD,
15. SOURCE INDEX ! -1
STAR NAME ! 3C390
RA 1950 (H.MMSS) ! 18.4552
DE 1950 (D.MMSS) ! 79.4248
16. SOURCE INDEX ! -1 3C390 18.4552 79.4248
SWEEP SIZE (IF NOT 2.1 DEGR.) ! 1.4
SWEEP SPEED (IF NOT 0.15 DEG/SEC) !
SWEEP DIRECTION (+1=UP, -1=DOWN, 0=BOTH) !
CHECK DATA AND ANTENNA (Y/N) !
OUTPUT FILE (NOT TERM!) !
STORE FILE (OR 'NO') !!
COMND ! STAKI-ANTC
CALIBRATION INPUT FILE ! (KT)ANTCAL-IN-STNDRD
END OF FILE

FILE NAME= (KT)ANTCAL-IN-STNDRD:DATA
COMND !
        READY TO START OFFSET MEASUREMENTS

WHEN YOU WANT TO STOP GIVE! @RT ACSTOP

DISP-STATUS

 EISCAT ANTENNA CALIBRATION   1980.09.29    8.01.34  SITE 4
 LAST MEASURED OFFSETS!
        AZOFF! 0.000 (  0.00  0.00) ELOFF! 0.000 (  0.00  0.00)
 IFL=  0 ILLCNT= 0 IMPCNT= 0

 NOW MEASURING!CAS-A  350.114  36.591
 CAS-A  AZSCAN 1 + 3.600 DEGR   24 SEC  OFFSETS!  0.06  0.51
 CAS-A  ELSCAN 1 + 2.700 DEGR   18 SEC  OFFSETS!  0.06  0.51
 CAS-A  AZSCAN 2 + 3.600 DEGR   24 SEC  OFFSETS!  0.09  0.49
 CAS-A  ELSCAN 2 + 2.700 DEGR   18 SEC  OFFSETS!  0.09  0.49
 CAS-A  AZOFF! 0.074 (350.34 36.57) ELOFF! 0.472 (350.42 36.56)
 CAS-A  AZSCAN 1 - 3.600 DEGR   24 SEC  OFFSETS!  0.07  0.47
 CAS-A  ELSCAN 1 - 2.700 DEGR   18 SEC  OFFSETS!  0.07  0.47
```

Figure D1. Example run of a definition of an input file, and start of the calibration data gathering.

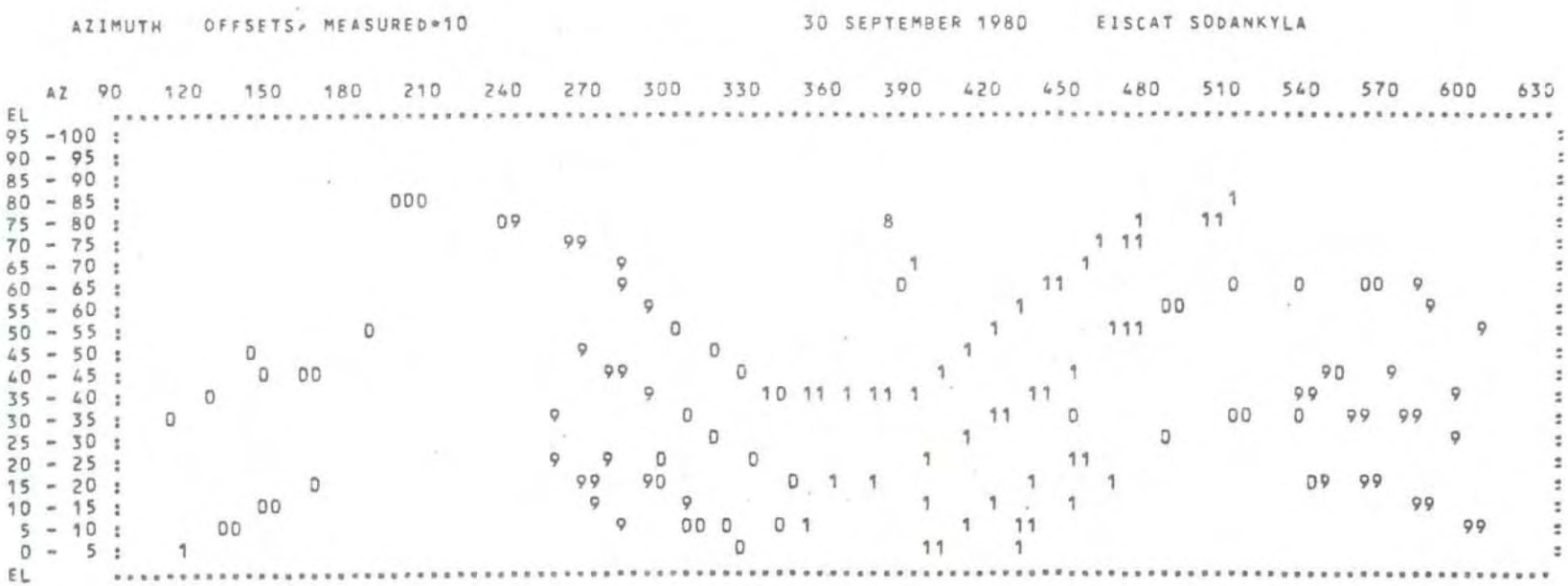
```
 10.39.00     30 SEPTEMBER   1980   EISCAT SODANKYLA
VERSION 79.07.15.A

ENTER 61
PASSWORD:
OK
PROJECT NUMBER:

@(ANTE)ANTCAL-CONTROL

ANTENNA OFFSET ANALYSIS          30 SEPTEMBER 1980     EISCAT SODANKYLA
WRITE HELP IF YOU NEED IT !
ANTENNA OFFSET ANALYSIS          30 SEPTEMBER 1980     EISCAT SODANKYLA
******************************   INITIALIZED  ******************************

COMND : STOP
ANTCAL ABORTED BY ACSTOP

COMND : OUT-FILE L-P
COMND : INPUT
INPUT FILE :
AZ OR EL OFFST FIT ? (AZ/EL) : AZ
STARTING DATE (Y,M,D,HH.MM) : 1980,9,29,0
ENDING DATE (Y,M,D,HH.MM) : ,,,,
MAXIMUM NO ITERATIONS :
MAXIMUM LAST CORRECTION :
SWEEP DIRECTION (+ / - / +-) :
 325  POINTS READ
COMND : DATA-MAP

COMND : MODEL-INP
 1 1            11 SIN(5*AZ)        1 1            11 TAN(EL)
 2 COS(AZ)      12 COS(6*AZ)        2 EL           12 TAN(EL)**2
 3 SIN(AZ)      13 SIN(6*AZ)        3 EL**2        13 TAN(EL)**3
 4 COS(2*AZ)    14 COS(7*AZ)        4 EL**3        14 SIN(EL)
 5 SIN(2*AZ)    15 SIN(7*AZ)        5 EL**4        15 SIN(2*EL)
 6 COS(3*AZ)    16 COS(8*AZ)        6 EL**5        16 SIN(3*EL)
 7 SIN(3*AZ)    17 SIN(8*AZ)        7 EL**6        17 COS(EL)
 8 COS(4*AZ)    18 COS(9*AZ)        8 EL**7        18 COS(2*EL)
 9 SIN(4*AZ)    19 SIN(9*AZ)        9 EL**8        19 COS(3*EL)
10 COS(5*AZ)    20 SW.DIR/2        10 COT(EL)      20 1/COS(EL)

GIVE IAZ1,IAZ2,IEL1,IEL2 (OR 'DEF'): DE
NFUNC( 1) = 1          *1
NFUNC( 2) = 1          *1/COS(EL)
NFUNC( 3) = 1          *TAN(EL)
NFUNC( 4) = COS(AZ)    *TAN(EL)
NFUNC( 5) = SIN(AZ)    *TAN(EL)
NFUNC( 6) = COS(AZ)    *1
NFUNC( 7) = SIN(AZ)    *1
NFUNC( 8) = COS(2*AZ)  *1
NFUNC( 9) = SIN(2*AZ)  *1

GIVE IAZ1,IAZ2,IEL1,IEL2 (OR 'DEF'): 20,20,1,1
GIVE IAZ1,IAZ2,IEL1,IEL2 (OR 'DEF'):
NFUNC( 1) = 1          *1
NFUNC( 2) = 1          *1/COS(EL)
NFUNC( 3) = 1          *TAN(EL)
NFUNC( 4) = COS(AZ)    *TAN(EL)
NFUNC( 5) = SIN(AZ)    *TAN(EL)
NFUNC( 6) = COS(AZ)    *1
NFUNC( 7) = SIN(AZ)    *1
NFUNC( 8) = COS(2*AZ)  *1
NFUNC( 9) = SIN(2*AZ)  *1
NFUNC(10) = SW.DIR/2   *1

COMND : COMPUTE
DET= 5.57E+20
COMND : MODEL-MAP
COMND : ERR-MAP
COMND : OUT,,
COMND :
```

Figure D2. Analysis of azimuth data.

```
COMND : INPUT
INPUT FILE :
AZ OR EL OFFST FIT ? (AZ/EL) : EL
STARTING DATE (Y,M,D,HH.MM) : 1980,9,29,0
ENDING DATE (Y,M,D,HH.MM) : ,,,,
MAXIMUM NO ITERATIONS :
MAXIMUM LAST CORRECTION :
SWEEP DIRECTION (+ / - / +-) :
 325  POINTS READ
COMND : DATA-MAP

COMND : MODEL-INPU
 1 1             11 SIN(5*AZ)        1 1             11 TAN(EL)
 2 COS(AZ)       12 COS(6*AZ)        2 EL            12 TAN(EL)**2
 3 SIN(AZ)       13 SIN(6*AZ)        3 EL**2         13 TAN(EL)**3
 4 COS(2*AZ)     14 COS(7*AZ)        4 EL**3         14 SIN(EL)
 5 SIN(2*AZ)     15 SIN(7*AZ)        5 EL**4         15 SIN(2*EL)
 6 COS(3*AZ)     16 COS(8*AZ)        6 EL**5         16 SIN(3*EL)
 7 SIN(3*AZ)     17 SIN(8*AZ)        7 EL**6         17 COS(EL)
 8 COS(4*AZ)     18 COS(9*AZ)        8 EL**7         18 COS(2*EL)
 9 SIN(4*AZ)     19 SIN(9*AZ)        9 EL**8         19 COS(3*EL)
10 COS(5*AZ)     20 SW.DIR/2        10 COT(EL)       20 1/COS(EL)

GIVE IAZ1,IAZ2,IEL1,IEL2 (OR 'DEF'): DEF
NFUNC( 1) = 1          *1
NFUNC( 2) = COS(AZ)    *1
NFUNC( 3) = SIN(AZ)    *1
NFUNC( 4) = 1          *COS(EL)
NFUNC( 5) = 1          *SIN(EL)
NFUNC( 6) = 1          *COT(EL)

GIVE IAZ1,IAZ2,IEL1,IEL2 (OR 'DEF'): 20,20,1,1
GIVE IAZ1,IAZ2,IEL1,IEL2 (OR 'DEF'):
NFUNC( 1) = 1          *1
NFUNC( 2) = COS(AZ)    *1
NFUNC( 3) = SIN(AZ)    *1
NFUNC( 4) = 1          *COS(EL)
NFUNC( 5) = 1          *SIN(EL)
NFUNC( 6) = 1          *COT(EL)
NFUNC( 7) = SW.DIR/2   *1

COMND : COMPUTE
DET= 1.76E+13
COMND : MODEL-MAP
COMND : ERROR-MAP
COMND : OUT,,
COMND : EX

020433 STOP    0
@LOG
 10.52.55     30 SEPTEMBER   1980
 TIME USED IS    7 MINS   23 SECS OUT OF   13 MINS   47 SECS
--EXIT--
```

Figure D3. Analysis of elevation data.

```
INPUT AZDATA FROM FILE : (RT)ANTCAL-DATA:DATA
STARTING DATE 1980 09 29  0.00    ENDING DATE 1999 00 00  0.00    MAXITER 3   MAX LAST CORR  0.100  DIR+-
```



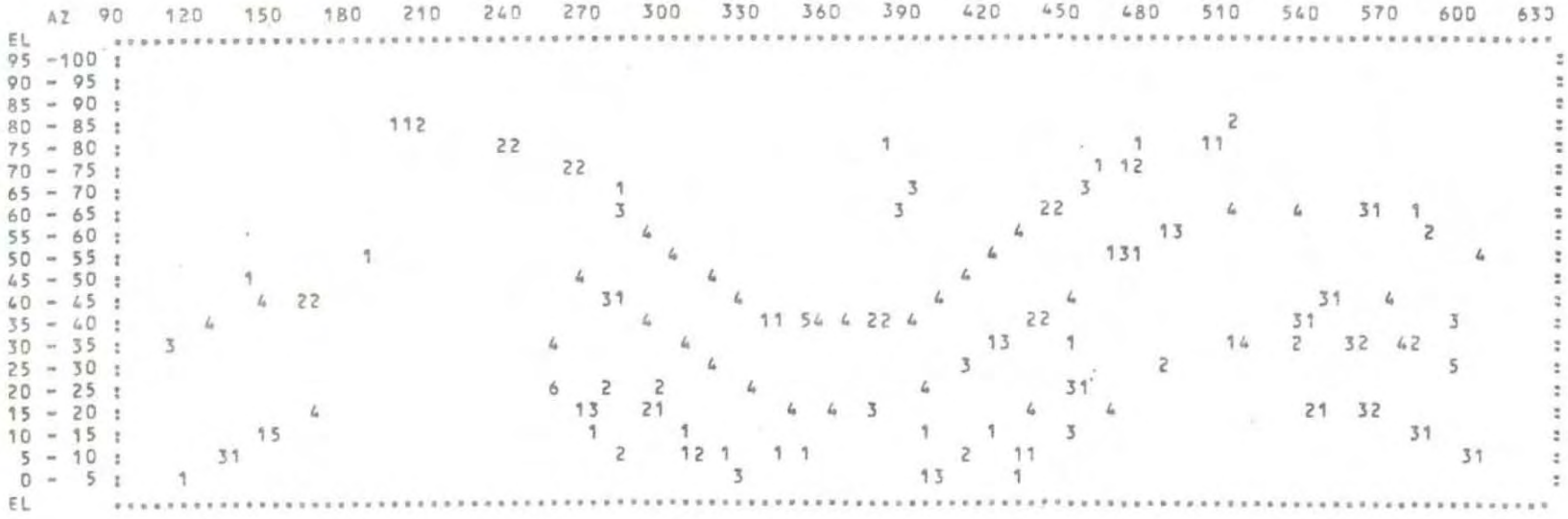
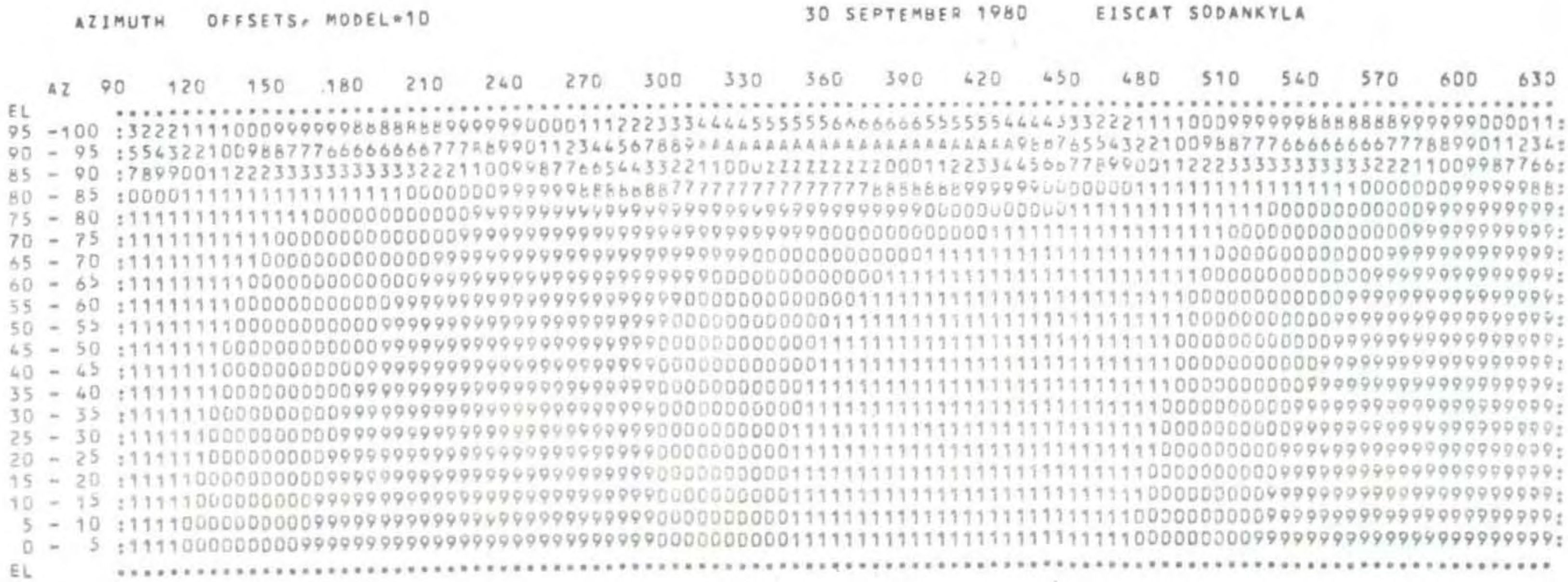


Figure D4. Map of the azimuth data. These maps are obtained by the INPUT and DATA-MAP commands.

```
AZIMUTH   OFFSET MODEL FIT, 325 POINTS   MSQR ERR = 0.030  MAX ERR = 0.257

 1  1          *1            *   0.055  +-0.029
 2  1          *1/COS(EL)    *  -0.060  +-0.035
 3  1          *TAN(EL)      *   0.041  +-0.022
 4  COS(AZ)    *TAN(EL)      *  -0.035  +-0.013
 5  SIN(AZ)    *TAN(EL)      *   0.000  +-0.002
 6  COS(AZ)    *1            *   0.092  +-0.008
 7  SIN(AZ)    *1            *   0.088  +-0.003
 8  COS(2*AZ)  *1            *  -0.002  +-0.004
 9  SIN(2*AZ)  *1            *  -0.000  +-0.002
10  SW.DIR/2   *1            *   0.001  +-0.003

      ERROR DISTRIBUTION
      90:
      87:
      84:
      81:
      78:
      75:
      72:
      69:                                    *
      66:                                    *
      63:                                    *
      60:                                    *
      57:                                    *
      54:                                    **
      51:                                    **2
      48:                                    ***2
      45:                                    ****
      42:                                    ****
      39:                                    ****
      36:                                    ****
      33:                                   2****
      30:                                   *****
      27:                                   *****
      24:                                   *****
      21:                                   *****
      18:                                  2*****2
      15:                                  *******
      12:                                 1*******2
       9:                               1 *********
       6:                               * *********
       3:                        1      1*1**********11 11      1
         I----+----I----+----I----+----I----+----I----+----I----+----I----+----I----+----I----+----I  MEAS.OFFST-
        -0.5      -0.4      -0.3      -0.2      -0.1       0.0       0.1       0.2       0.3       0.4       0.5  MODEL OFFST
```

Figure D5. Azimuth analysis. The model fitted is shown together with a histogram of the discrepancies between the model and measurements. This output is a result of the MODEL-INPUT and COMPUTE commands.



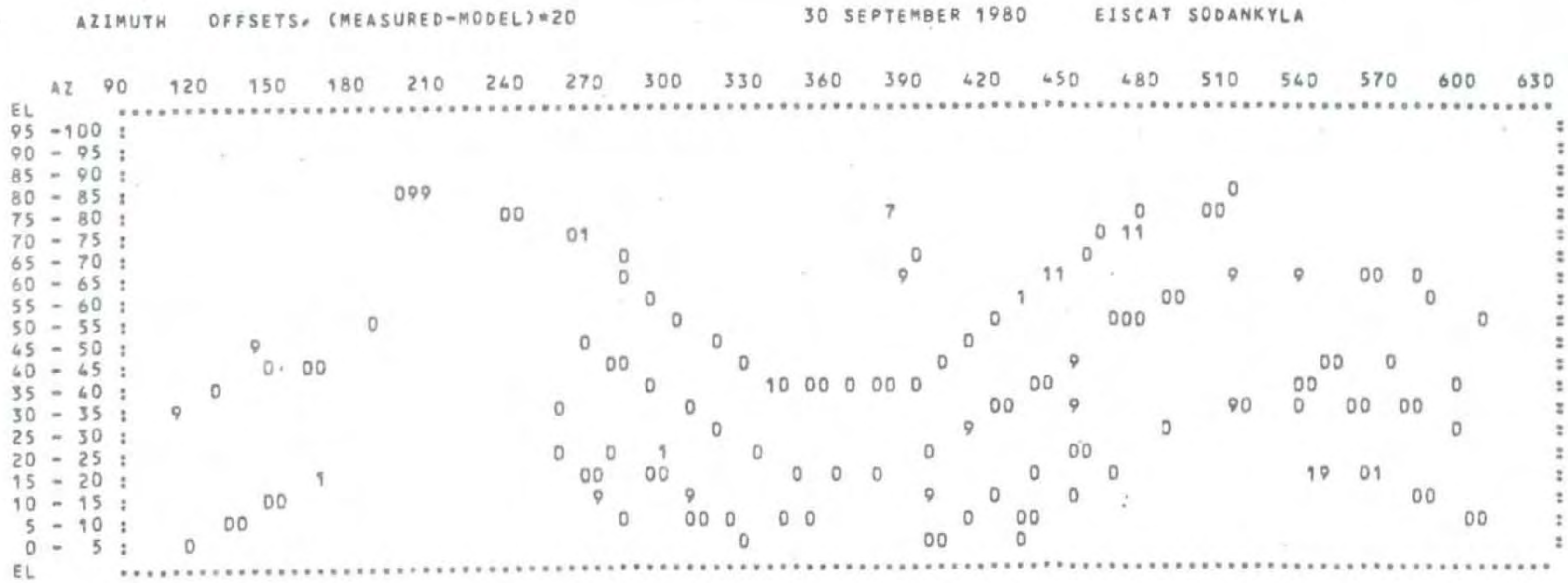


Figure D6. Map of the model fitted and a map of the differences between the model and measured data. Here, the symbols mean: A=10..19, B=20..29, 9=-1, 8=-2, 7=-3, ...

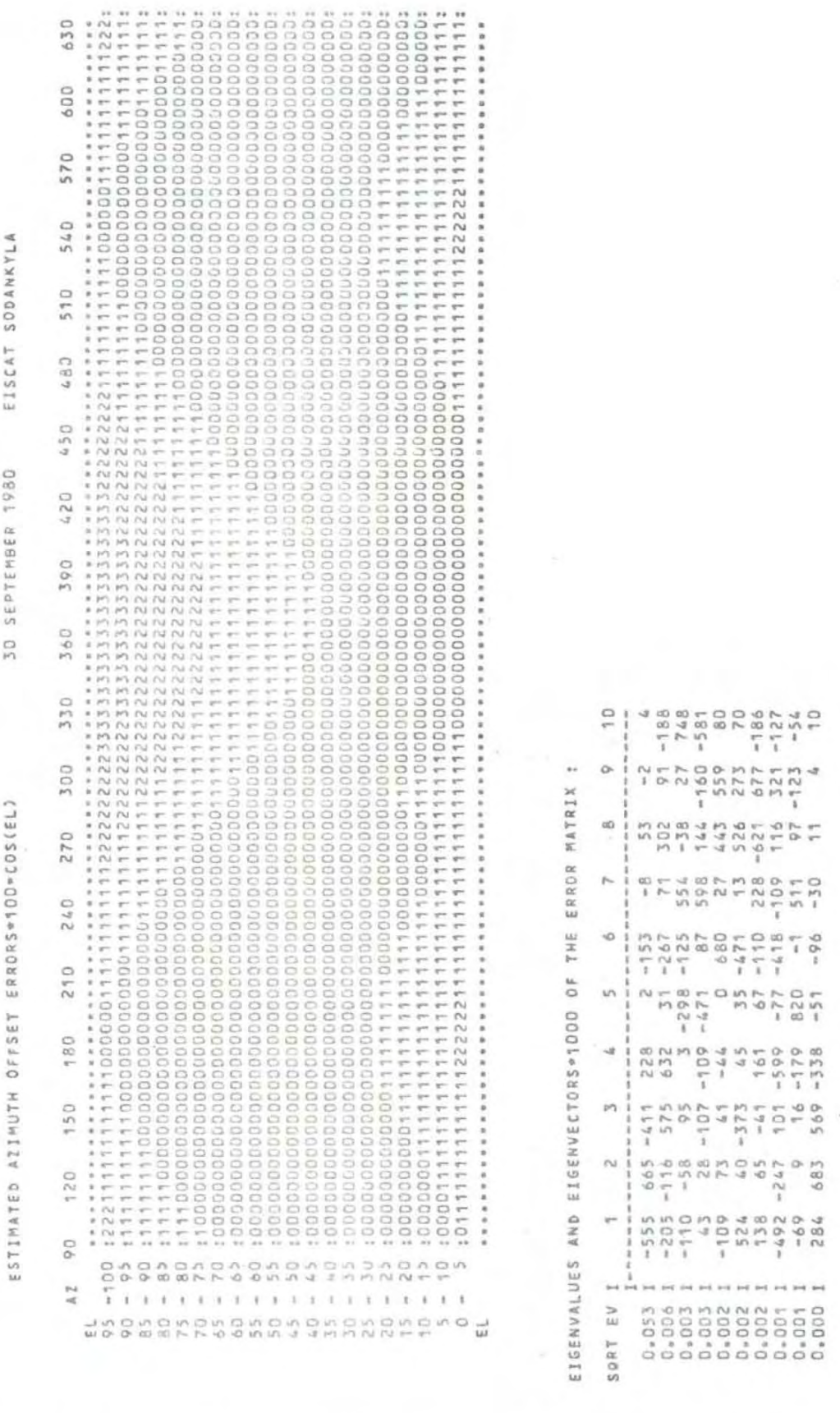
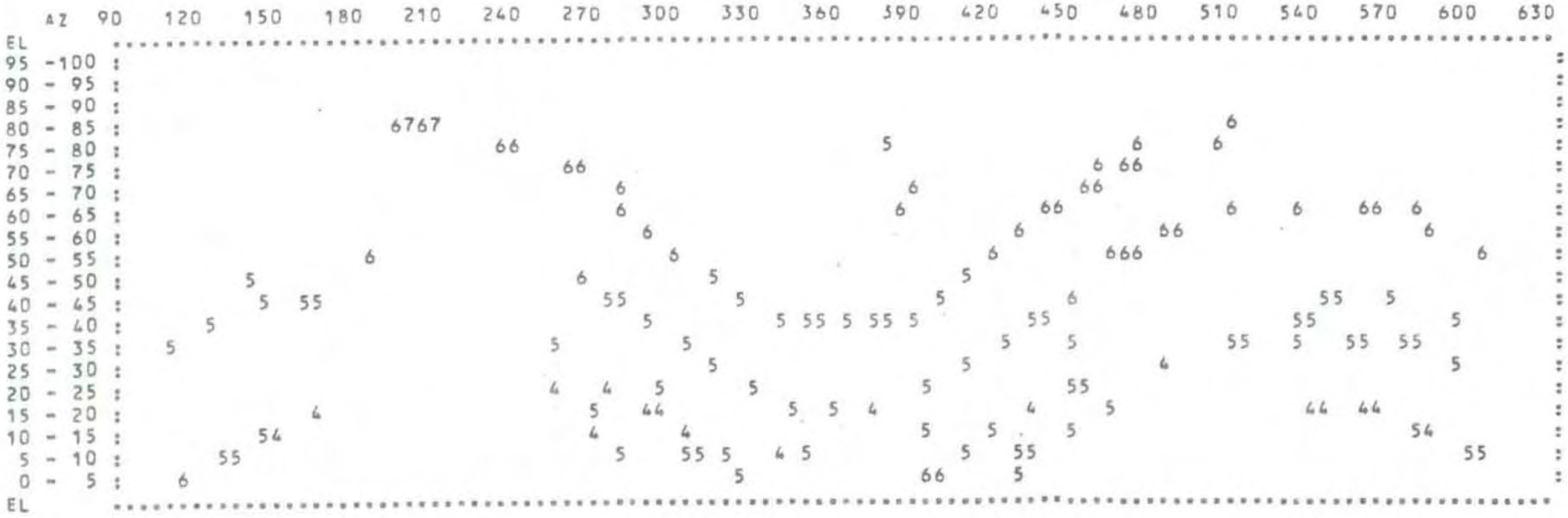


Figure D7. Estimate of the errors in the model fitted. This estimate is based on the assumption that there are no systematic errors. This map is obtained by the ERROR-MAP command.

INPUT ELDATA FROM FILE : (RT)ANTCAL-DATA:DATA
STARTING DATE 1980 09 29 0.00 ENDING DATE 1999 00 00 0.00 MAXITER 3 MAX LAST CORR 0.100 DIR+-

ELEVATION OFFSETS, MEASURED=10 30 SEPTEMBER 1980 EISCAT SODANKYLA



NUMBER OF OFFSETS MEASURED, TOTAL= 325 30 SEPTEMBER 1980 EISCAT SODANKYLA

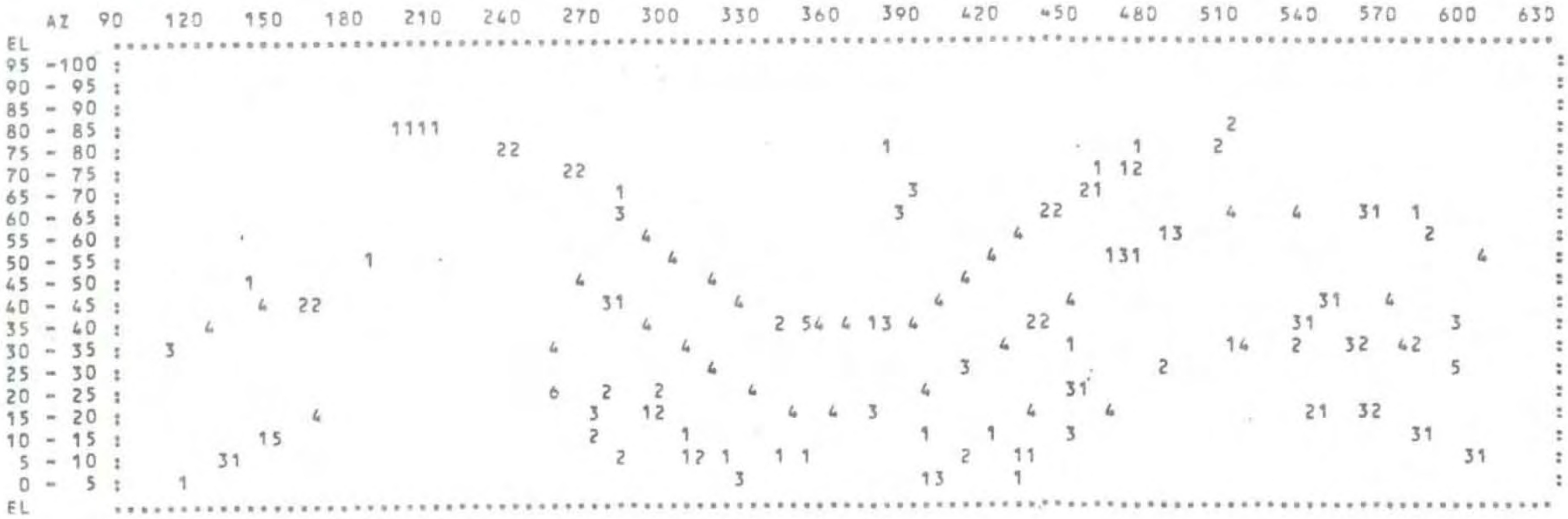
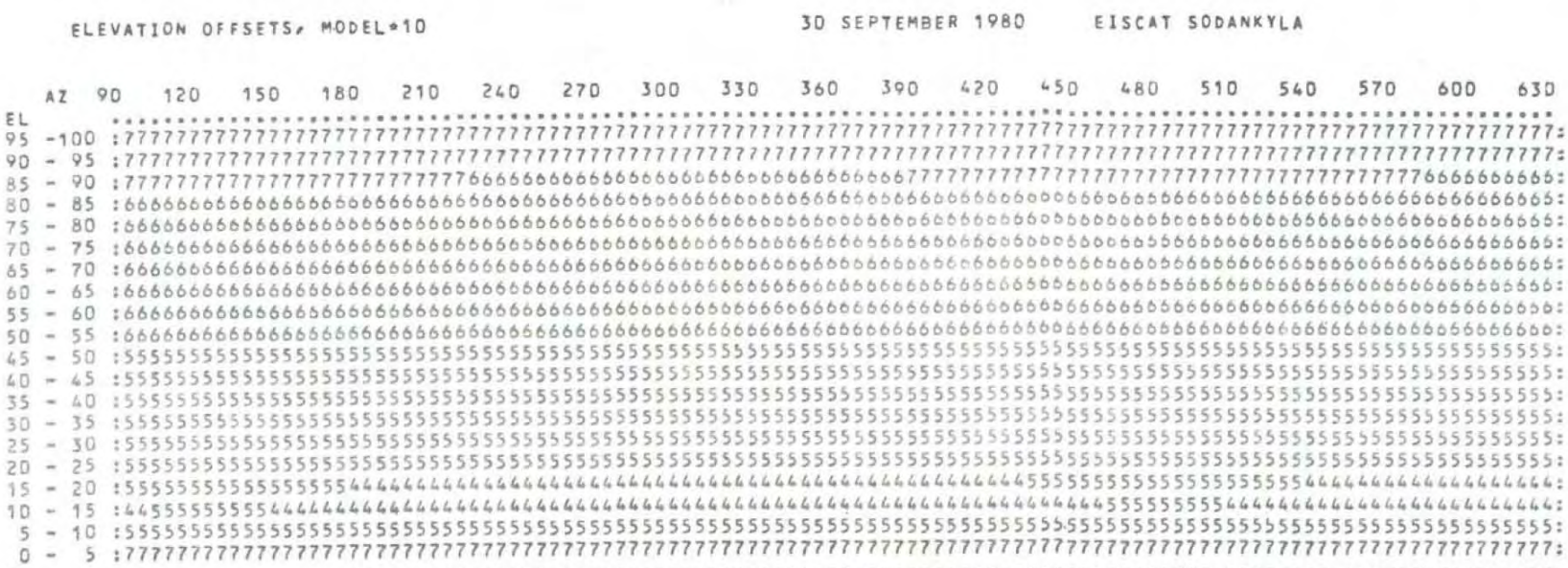


Figure D8. Plot of elevation data.

```
ELEVATION OFFSET MODEL FIT,  325 POINTS   MSQR ERR = 0.018  MAX ERR = 0.099

 1  1          *1          *   0.428  +-0.023
 2  COS(AZ)    *1          *  -0.002  +-0.002
 3  SIN(AZ)    *1          *   0.003  +-0.001
 4  1          *COS(EL)    *  -0.100  +-0.016
 5  1          *SIN(EL)    *  -0.227  +-0.017
 6  1          *COT(EL)    *   0.015  +-0.001
 7  SW.DIR/2   *1          *  -0.019  +-0.002

   ERROR DISTRIBUTION
 120:
 116:
 112:
 108:
 104:
 100:
  96:
  92:                                                  3
  88:                                                  *
  84:                                                  *
  80:                                                  *
  76:                                                 2*
  72:                                                 **
  68:                                                 **
  64:                                                 **3
  60:                                                 ***
  56:                                                 ***
  52:                                                 ***
  48:                                                 ***
  44:                                                 ***
  40:                                                 ***1
  36:                                                 ****
  32:                                                 ****
  28:                                                3****
  24:                                                *****
  20:                                                *****
  16:                                                ******
  12:                                                ******
   8:                                                ******
   4:                                      1  2 ********* 1  1
     I----+----I----+----I----+----I----+----I----+----I----+----I----+----I----+----I----+----I  MEAS.OFFST-
    -0.5      -0.4      -0.3      -0.2      -0.1       0.0       0.1       0.2       0.3       0.4       0.5  MODEL OFFST
```

Figure D9. Elevation model fit.



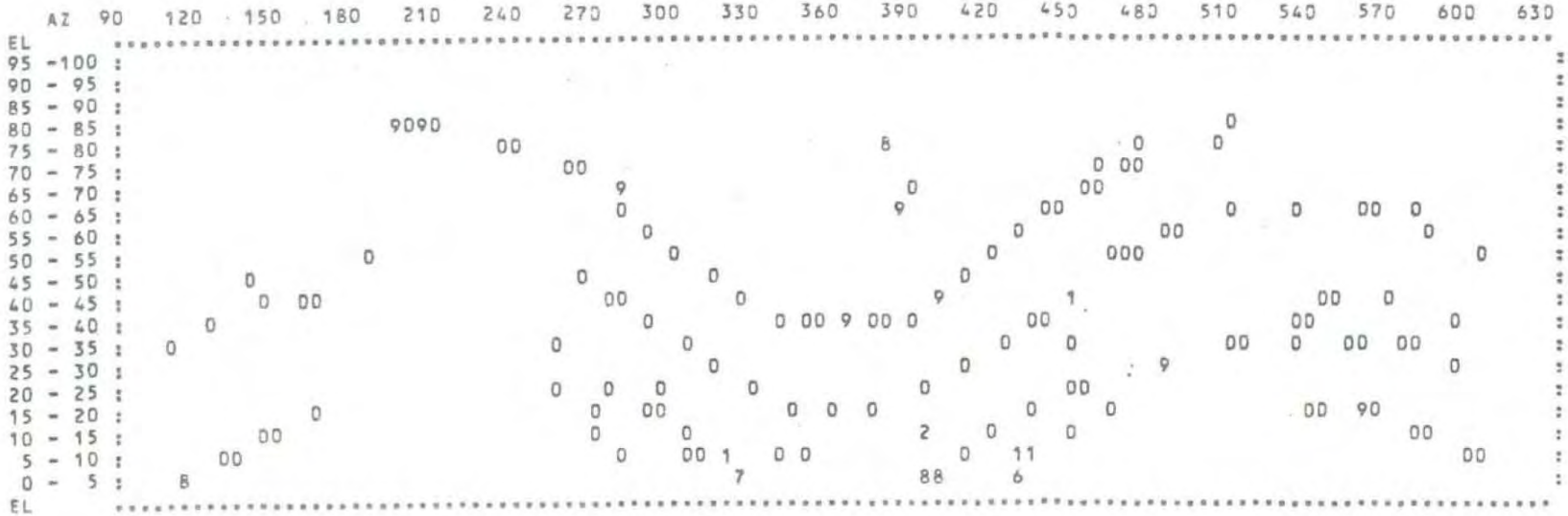
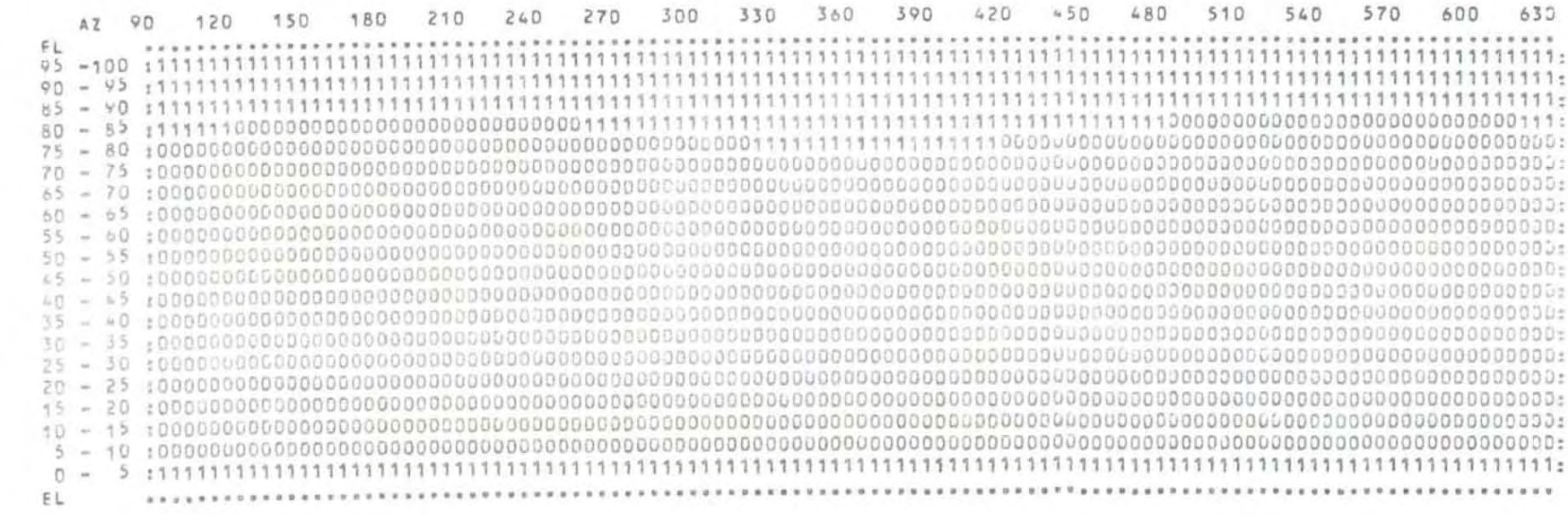


Figure D10. Plot of the elevation model fitted.

ESTIMATED ELEVATION OFFSET ERRORS*100 30 SEPTEMBER 1980 EISCAT SODANKYLA



EIGENVALUES AND EIGENVECTORS*1000 OF THE ERROR MATRIX :

SQRT EV	I	1	2	3	4	5	6	7
0.033	I	703	22	3	-494	-510	-15	14
0.005	I	7	141	-4	-700	694	62	43
0.002	I	106	325	115	156	37	-41	917
0.001	I	118	849	288	170	28	-50	-384
0.001	I	55	-330	939	-1	69	33	-8
0.001	I	665	-204	-139	430	494	-241	-91
0.000	I	185	9	-46	161	71	965	-5

Figure D11. Estimates of the errors of elevation model.

EISCAT publications

F. du Castel, O. Holt, B. Hultqvist, H. Kohl and M. Tiuri:
A European Incoherent Scatter Facility in the Auroral Zone (EISCAT).
A Feasibility Study ("The Green Report") June 1971. (Out of print).

O. Bratteng and A. Haug:
Model Ionosphere at High Latitude, EISCAT Feasibility Study, Report No. 9.
The Auroral Observatory, Tromsö July 1971. (Out of print).

A European Incoherent Scatter Facility in the Auroral Zone, UHF System and Organization ("The Yellow Report"), June 1974.

EISCAT Annual Report 1976.(Out of print).

P.S. Kildal and T. Hagfors:
Balance between investment in reflector and feed in the VHF cylindrical antenna.
EISCAT Technical Notes No. 77/1, 1977.

T. Hagfors:
Least mean square fitting of data to physical models.
EISCAT Technical Notes No. 78/2, 1978.

T. Hagfors:
The effect of ice on an antenna reflector.
EISCAT Technical Notes No. 78/3, 1978.

T. Hagfors:
The bandwidth of a linear phased array with stepped delay corrections.
EISCAT Technical Notes No. 78/4, 1978.

Data Group meeting in Kiruna, Sweden, 18-20 Jan. 1978
EISCAT Meetings No. 78/1, 1978

EISCAT Annual Report 1977