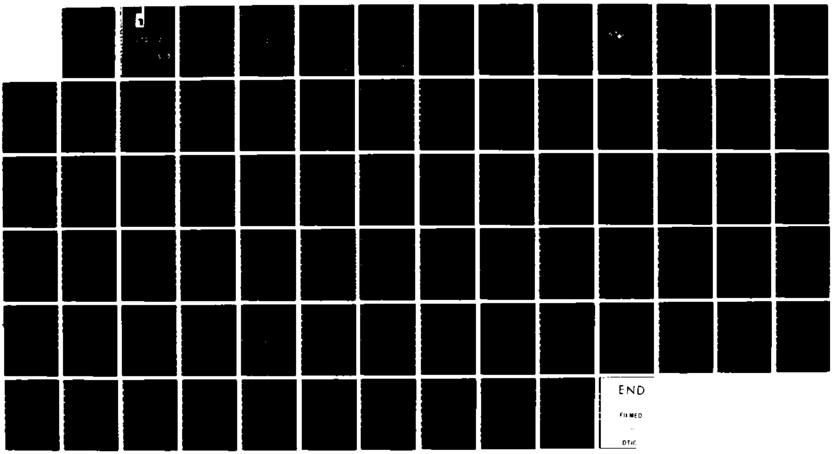
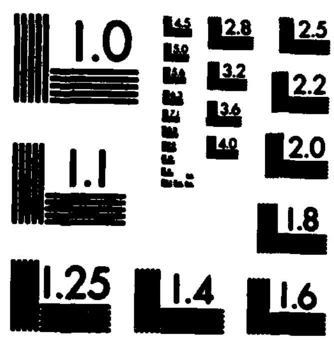
AD-A142 929 THE APL RCS (RADAR CROSS SECTION)/STATISTICS CODE DESCRIPTION ILLUSTRATIO..(U) JOHNS HOPKINS UNIV LAUREL MD APPLIED PHYSICS LAB F C PADDISON ET AL. JAN 84 1/1

UNCLASSIFIED JHU/APL/TG-1345 N00024-83-C-5301 F/G 9/2 NL





MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

AD-A142 929

(12)

JHU/APL
TG 1345
JANUARY 1984
Copy No. 12

Technical Memorandum

THE APL RCS/STATISTICS CODE DESCRIPTION, IlLUSTRATIONS OF OUTPUT, AND U. ER'S GUIDE

F. C. PADDISON
A. L. MAFFETT
J. W. FOLLIN
H W. KLIMACH

THE JOHNS HOPKINS UNIVERSITY ■ APPLIED PHYSICS LABORATORY

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE

PLEASE FOLD BACK IF NOT NEEDED FOR BIBLIOGRAPHIC PURPOSES

REPORT DOCUMENTATION PAGE

1. REPORT NUMBER JHU/APL TG 1345	2. GOVT ACCESSION NO. AD-A142 929	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) THE APL RCS/STATISTICS CODE DESCRIPTION, ILLUSTRATIONS OF OUTPUT, AND USER'S GUIDE		5. TYPE OF REPORT & PERIOD COVERED Technical Memorandum January 1984
		6. PERFORMING ORG. REPORT NUMBER JHU/APL TG 1345
7. AUTHOR (s) F. C. Paddison, A. L. Maffett, J. W. Follin, and H. W. Klimach		8. CONTRACT OR GRANT NUMBER (s) N00024-83-C-5301
9. PERFORMING ORGANIZATION NAME & ADDRESS The Johns Hopkins University Applied Physics Laboratory Johns Hopkins Road Laurel, MD 20707		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS ZT90
11. CONTROLLING OFFICE NAME & ADDRESS U.S. Air Force Rome Air Development Center Griffiss Air Force Base Rome, NY 13441		12. REPORT DATE January 1984
		13. NUMBER OF PAGES 79
14. MONITORING AGENCY NAME & ADDRESS Naval Plant Representative Office Johns Hopkins Road Laurel, MD 20707		15. SECURITY CLASS. (of this report) Unclassified
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE
16. DISTRIBUTION STATEMENT (of this Report) Approved for public release; distribution unlimited		
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)		
18. SUPPLEMENTARY NOTES		

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

radar scintillation statistics
beta parameters
RCS calculations
RCS statistical description
user's guide
data storage
radar cross s[illegible] (RCS)

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

The report is a description of a computer code that results in the first- and higher-order statistics of RCS of a complex target as a function of frequency polarization, aspect angle, and constituent parameters of the target. The premise is that a complex target can be represented by a set of simple scatterers. There is a general discussion of RCS data; the capabilities of the code from the viewpoint of the RCS analyst or data librarian; the theory of RCS, statistics, lobe structure, and glint embodied in the code; a user's guide from the viewpoint of the computer analyst; and a program listing.

DD FORM 1 JAN 73 1473

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE

JHU/APL
TG 1345
JANUARY 1984

Technical Memorandum

THE APL RCS/STATISTICS CODE DESCRIPTION, ILLUSTRATIONS OF OUTPUT, AND USER'S GUIDE

F. C. PADDISON
A. L. MAFFETT*
J. W. FOLLIN
H. W. KLIMACH*

*Consultant

THE JOHNS HOPKINS UNIVERSITY ■ APPLIED PHYSICS LABORATORY
Johns Hopkins Road, Laurel, Maryland 20707
Operating under Contract N00024-83-C-5301 with the Department of the Navy

THE JOHNS HOPKINS UNIVERSITY
APPLIED PHYSICS LABORATORY
LAUREL, MARYLAND

ABSTRACT

This report is a description of a computer code that results in the first- and higher-order statistics of RCS of a complex target as a function of frequency, polarization, aspect angle, and constituent parameters of the target. The premise is that a complex target can be represented by a set of simple scatterers. There is a general discussion of RCS data; the capabilities of the code from the viewpoint of the RCS analyst or data librarian; the theory of RCS, statistics, lobe structure, and glint embodied in the code; a user's guide from the viewpoint of the computer analyst; and a program listing.

CONTENTS

ILLUSTRATIONS

1.0 BACKGROUND

The computer code described in this report – The APL RCS/Statistics Code – represents years of development by E. Shotland, J. W. Follin, Jr., and F. C. Paddison (APL) and J. W. Crispin, Jr., A. L. Maffett, and K. M. Siegel (the University of Michigan). The results are in the form of first- and higher-order statistics for the radar cross section (RCS) of a complex target as a function of frequency, polarization, aspect angle, and constituent parameters of the target. The statistics are developed from the premise that a complex target can be broken down into a set of simple scatterers whose RCS can be more manageably calculated from available techniques (thus, the program can be conveniently updated to include the most accurate techniques currently at hand) and whose relative phases are randomly (i.e., uniformly) and independently distributed.

Methods for determining both first-order RCS statistics for a complex target and also simple component scatterer RCS data were developed by members of Siegel's Radiation Laboratory at the University of Michigan, principally Crispin and Maffett. The extensions of these methods to include the (higher-order) lobe and glint statistics were developed by Shotland and Follin under the sponsorship of the Advanced Research Projects Agency in an extension of the Advanced ALBIS Program. The (higher-order) beta statistics were developed by Follin during the Cruise Missiles Observables Program. All of these statistical methods were organized into a coherent program by Maffett, and the result was coded by H. W. Klimach. Paddison served as Program Manager for the APL efforts.

The techniques upon which the simple scatterer RCS calculations are based are, to a large extent, the methods of geometrical and physical optics with their attendant assumptions and restrictions on body smoothness and size relative to illuminating wavelength. Included in the code, however, are other methods that are appropriate, for example, to traveling-wave phenomena and wedges (with straight edges).

For regions where the ratio of body length to wavelength may become important (particularly if radar absorbing materials have been used to reduce large RCS contributions), diffraction effects have not been taken into account, but several methods are being evaluated to determine which would be most appropriate to the nature of the APL code. The methods include the geometrical and physical theories of diffraction,[1,2] the equivalent current technique,[3] and the numerical electromagnetic code.[4] The geometrical theory is based on the tracing of rays and can be described by currents induced on the illuminated portion of the structure. The physical theory adds currents induced in the shadowed area by diffraction of the incident field. The equivalent current technique adds edge currents on assumed filamentary edges. The numerical electromagnetic code is an integral-equation surface-current determination in which the structure is broken up into small cells, and the induced charge and current on each piece are calculated from the incident field and from the charge and current of each other cell. It is planned to include one or more of these methods in the APL code to account for various diffraction effects caused by some of the scattering components of a complex target.

Bistatic RCS computational capabilities are not yet fully incorporated into the code. The complexity of bistatic RCS estimation over monstatic is greatly increased by its additional aspect variables, so the statistical approach gains still more importance as a reducer of data bulk. The bistatic capability will be included in the APL code as soon as possible.

New coordinate system arrangements have been introduced recently to simplify the execution of conical-aspect views of a target (see the Appendix).

---

[1] J. B. Keller, "Geometrical Theory of Diffraction," *J. Opt. Soc. Am.*, 52, 116-130 (1962).

[2] P. Ya. Ufrimtsev, "Method of Edge Waves in the Physical Theory of Diffraction," *Izd-Vo. Sov. Radio*, 1-243 (1962); translation by U.S. Air Force Foreign Technology Division (1971).

[3] E. F. Knott and T. B. A. Senior, "Comparison of Three High-Frequency Diffraction Techniques," *Proc. IEEE* 62, 1468-1474 (1974).

[4] G. J. Burke and A. J. Poggio, *Numerical Electromagnetic Code - Method of Moments*, Naval Electronics Systems Command TD 116 (1977).

2.0 GENERAL DISCUSSION OF RCS DATA

Experimental and predicted RCS data usually are in the form of a received signal referenced to the transmitted signal, versus aspect angle. Typically, there are separate tabulations for each frequency of interest. The radars used for aircraft, missile, and space-object detection, tracking, and identification are usually at wavelengths that are small in comparison to the dimensions of the targets. Usually the targets are complex shapes. At some aspects such as broadside, the RCS of a target may be many orders of magnitude greater than at other aspects, may be dominated by a single component, and hence can be characterized simply. However, at most aspects there is a complicated interference pattern with maxima spaced roughly λ/L apart in angle and c/L apart in frequency (where λ is the wavelength, c is the velocity of light, and L is a typical length parameter). If, for example, L is 10 meters (i.e. a small missile), roughly 10^9 data points are needed to cover X band for all polarizations. A statistical description is needed to reduce the data storage and retrieval problem.

The magnitude of RCS data is not the only reason for an interest in a statistical description. The main users of these data are the hardware designer, the radar system designer, and the performance analyst. The latter two have long searched for a statistical description of the amplitude scintillation of RCS. The literature contains many attempted fits of experimental data with single-parameter statistical descriptors.

The parameters of interest in the vicinity of any one aspect are the mean RCS and its probability distribution, the lobe widths in angle and frequency, and the mean centroid and its variances.

Over the years, the technique of breaking down a complex target into a finite set of component scatterers has been evolved together with statistical descriptions of lobe structure and the appropriate descriptor of amplitude scintillation, the two-parameter beta distribution function. The computer code described in this report embodies the aforementioned predictive RCS and its statistical descriptors.

3.0 THE CAPABILITIES OF THE CODE FROM THE VIEWPOINT OF THE RCS ANALYST

The APL RCS/Statistics Code, designed for the PDP 11/60 computer, has been checked out with several missile configurations. Its computational capabilities fall into two categories, either of which, with various options, can be executed from a master code. They are:

1. First-order statistics for RCS, and
2. Lobe and higher-order statistics for RCS.

At this stage, the first- and higher-order statistics are limited to monostatic situations. However, the lobe statistics are coded in such a way as to be applicable to bistatic situations as soon as that capability is introduced.

The entire development is based on the premise that a complex scattering body can be broken down into a finite set of component scatterers (usually chosen to be elementary scatterers, some of which may correspond to simple geometric shapes).[1] If the contributions from this set of scatterers are combined with proper phases, the result is an estimate of relative-phase RCS; this option is available in the APL code. Figure 1 is an example of relative-phase RCS.

[1] J. W. Crispin, Jr., and A. L. Maffett, "Estimation of RCS for Simple Shapes," "Estimation of RCS for Complex Shapes," *Proc. IEEE* 53, (1965).

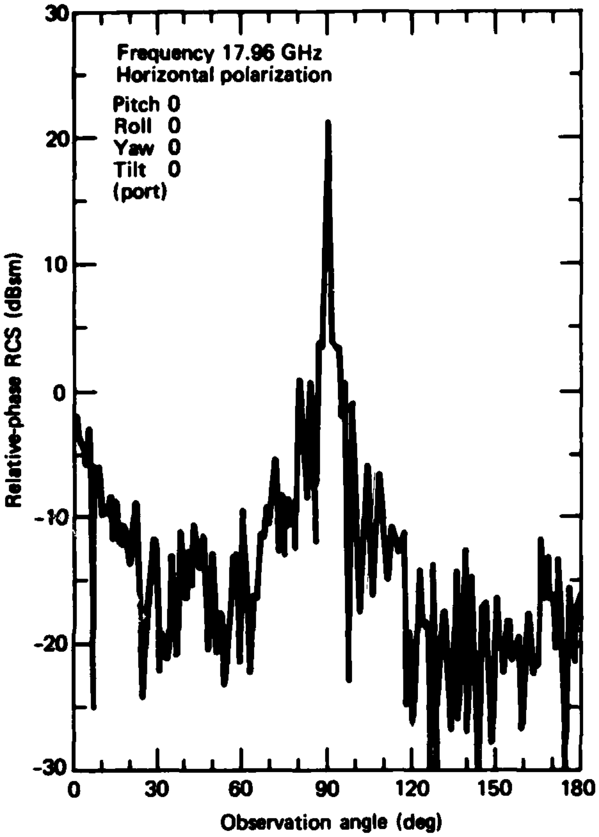


Figure 1 — Example of calculated monostatic relative-phase RCS 1/24th scale Convair 990.

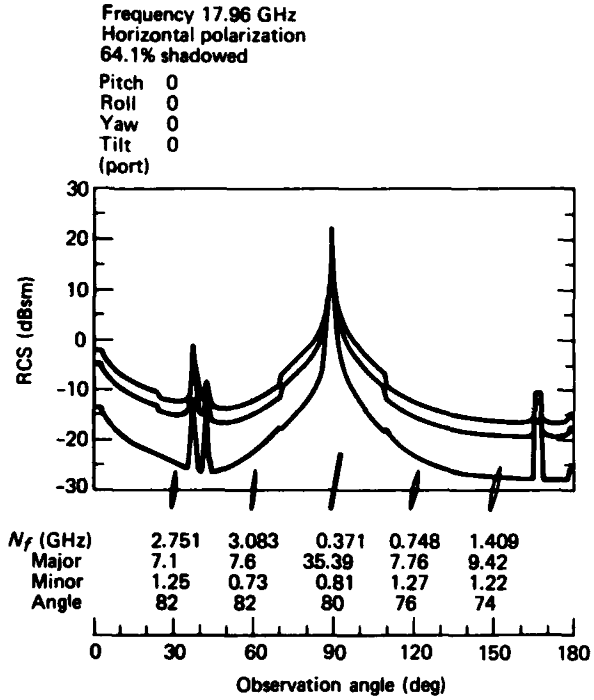


Figure 2 — Example of calculated monostatic mean RCS with rms spread (including lobe statistics); 1/25th scale Convair 990.

If it is assumed that phases among scattering components are distributed independently and randomly (i.e., uniformly, over the interval 0 to 2π), a hierarchy of statistics can be developed. Under first-order statistics fall the usual mean RCS and root-mean-square (rms) spread about the mean (see Fig. 2). Under higher-order statistics fall higher moments from which skewness and kurtosis may be derived to show the appropriate beta distribution characteristics for the RCS data under examination.[6-8] (These higher-order statistics can also be used to analyze measured RCS data; this feature is not part of the APL code.) Various properties of the autocorrelation of RCS can be used to describe lobe statistics (in both frequency and aspect).[9] Lobe characteristics are illustrated in Fig. 2 at observation angle intervals of 30°. The terms "major" and "minor" refer to half the long and short axes of the ellipse; they give the mean lobewidth at that aspect in both the horizonal plane (minor) and the normal plane (major). "Angle" refers to the tilt of the lobe ellipse with the observation axis. N_f (GHz) refers to the number of lobes of RCS per gigahertz of frequency.

At this writing, the component scatterer analyses rest largely upon physical and geometrical optics methods. However, the code could be expanded readily to include diffraction treatments and even the more exact integral-equation analyses exemplified by the numerical electromagnetic code.[4]

The program as written and described here was intended for targets of modest complexity at observing frequencies in the centimeter region. We plan to recode the program for a more capable computer in the

[6] J. W. Follin, Jr., *Statistical Properties of RCS*, JHU/APL QM-81-115 (1981).

[7] J. W. Follin, Jr., and A. L. Maffett, "RCS Scintillations and Their Statistical Description," in *Proc. Second Annual Tactical Air Surveillance Control Conf.*, Rome Air Development Center (1981).

[8] J. W. Follin, Jr., F. C. Paddison, and A. L. Maffett, "The RCS of Two Cruise Missiles and Their Statistical Description," DARPA 8th Strategic Space Symposium (1982).

[9] E. Shotland, *Statistical Analysis of Radar Target Scintillation*, JHU/APL TG 955 (1967).

near future to allow the handling of more complex targets up to the mid-millimeter range of frequencies.

The effects of a surface treatment with radar absorbing material can be taken into account through the uniform application of a reflection coefficient to the component on which the material is placed. No account is taken of the possibility of volumetric scattering for either radar absorbing material or dielectric components.

The major emphasis of the code is on a statistical examination of the RCS of a target. Considering the number of variables upon which RCS depends, a very voluminous collection of data can be amassed for a single target, even when domains of variables are rather severely restricted. A statistical examination of RCS offers a condensation of facts and, thus, an economy of output not available with a deterministic treatment of similar scope.

All of the RCS statistics are essentially unaffected by small changes in target configuration or frequency that may drastically affect the lobe structure of deterministic treatments (especially for frequencies of 1 GHz and above).

Last but not least, statistical RCS results mesh well with the requirements of detection and tracking systems for statistical formulations.

3.1 Inputs

The inputs usually required for the code are wavelength; polarization (linear, but extendable to elliptical); plane of observation; range of observation angle in the observation plane; pitch, roll, and yaw of the target; and tilt of the target rotation plane (to make comparisons with RATSCAT measurements).

For a particular complex target, a suitable mathematical model must be derived in the form of a set of component scatterers matched to the target features with the help of drawings, pictures, and artists' conceptions. The set of component scatterers must, of course, be limited to the types presently available to the APL code (some 20 as of this writing).

Computations of RCS are performed in coordinate systems generic to the various components and so must be transformed to the coordinate systems in which the statistics are formulated.

3.2 Outputs

The first-order and lobe statistics have already been illustrated in Fig. 2. Note that for the latter case, for any line through the lobe ellipse center, the distance from the center to the ellipse intercept gives the lobewidth for that direction of observation. For example, at the 30° observation angle, the ellipse has a major axis oriented 82° from the observation axis, indicating thin lobes in the horizontal plane of the target and fat lobes in the perpendicular plane.

The output for the higher-order statistics may be either printed or graphical. Let us consider the latter. One has available skewness versus width or kurtosis versus width (width, skewness, and kurtosis are defined in Section 4 of this guide). Skewness and kurtosis are plotted in Figs. 3 and 4, respectively, ver-

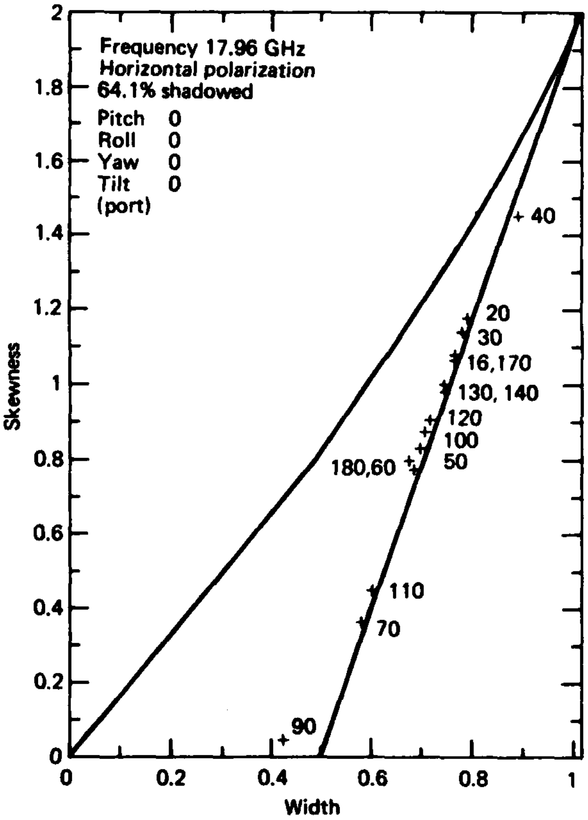


Figure 3 — Calculated width-skewness statistics for the RCS scintillation of a 1/25th scale Convair 990.

sus width for a typical target at 10° observation angle intervals. It should be noted that the data presented in Figs. 3 and 4 may be based either on experimental data or on theoretical calculations. The code can provide the theoretical values. The points are plotted within boundaries that denote the limits for beta-distributed component-RCS values. Locations denoting normal, lognormal, chi-square, and Rayleigh distri-

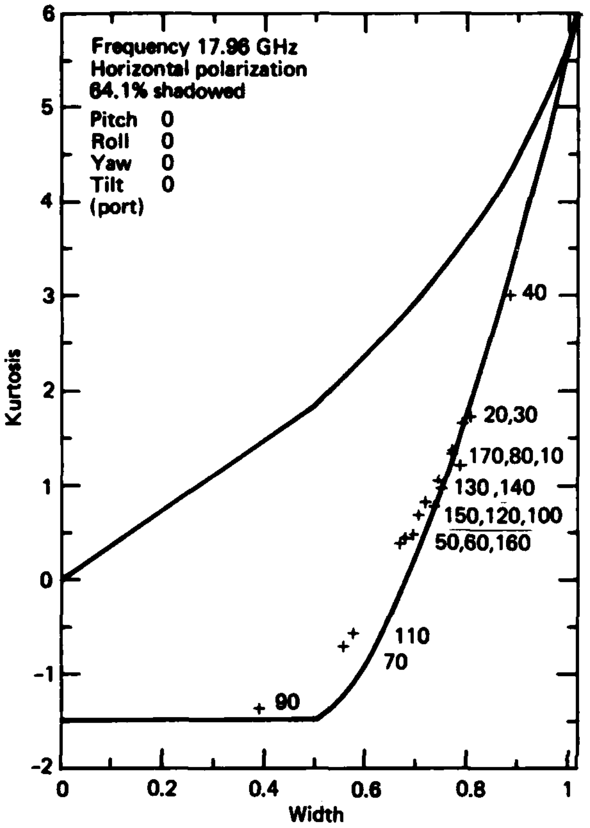


Figure 4 — Calculated width-kurtosis statistics for the RCS scintillation of a 1/25th scale Convair 990.

butions all fall *outside* of these boundaries, for both width/skewness space and width/kurtosis space.

For example, consider Fig. 4 where the observation direction 100° produces in width/kurtosis space the point (0.7,0.7). These statistics yield the parameter values

$$a = 0.96, \quad b = 3.82$$

for the beta distribution given by

$$\beta(x;a,b) = \frac{\Gamma(a+b)}{\Gamma(a)\Gamma(b)} x^{a-1}(1-x)^{b-1} \ ,$$

where $\Gamma(a)$ denotes the gamma function of an argument a. Thus, the component scatterers that contribute to the RCS at 100° from nose-on produce, under the radom-phase assumption, an RCS that is distributed according to the above beta distribution. [Note: The skewness can be calculated to be 0.88, which agrees very well with the skewness for 100° found at width 0.7 in Fig. 3. The beta distribution for the above values of a and b is skewed to the left, with a long tail to the right.]

The beta-distributed quantity x is actually σ/σ_{max}, where σ is the RCS for some fixed aspect, frequency, and polarization. So, for example, since σ_{max} (100°) = 1.14 m^2 and since the mean of $\beta(x;a,b)$ is $a/(a+b)$ = 0.2, the mean RCS at 100° is $\bar{\sigma}$ = 0.2(1.14) = 0.23 m^2 or −6.4 dBsm (dBsm is decibels relative to a square meter). Note that this value agrees with the mean RCS at 100°, which can be read on Fig. 2. The beta distribution of RCS at a fixed observation angle arises from, and depends on, the assumption that the phases among components are randomly distributed (i.e., uniformly, between 0 and 2π) for every observation angle value.

3.3 Glint

Illustrations of the output of the program predicting glint are not included in this writing; that portion of the program has yet to be validated. However, a discussion of the theory used is given in Section 4.6.

4.0 CALCULATIONS

4.1 Relative-Phase RCS

Consider a class of complex targets, each of whose members can be broken down into a finite set of component scatterers. Each component scatterer is assumed to be in the far field of the transmitter so that it is illuminated by a plane wave; it is also in the far field of the receiver so that the observed field has a plane wavefront. The major dimension of a component is assumed to be large with respect to the wavelength λ. Polarizations of the transmitter and receiver are taken to be horizontal (H) or vertical (V) in this discussion; however, they could be circular or elliptical. The possible combinations are HH, HV, VH, and VV; only the first and last are considered here.

Let the signal scattered by the ith component be denoted by

$$a_i \exp(j\psi_i), \qquad j = \sqrt{-1}, \tag{1}$$

where the amplitude a_i (which is usually taken to be $\sqrt{\sigma_i}$, where σ_i is the RCS of the ith component) and the phase ψ_i may depend on aspect (for both transmitter and receiver, or on their bistatic separation β), frequency, polarization, and the constituent parameters of the component. The scattered amplitude is then

$$A = \sum_{i=1}^{N} \sqrt{\sigma_i}\, \exp(j\psi_i). \tag{2}$$

The relative-phase RCS, σ, for the complex target is

$$\sigma = AA^* = \left| \sum_{i=1}^{N} \sqrt{\sigma_i}\, \exp(j\psi_i) \right|^2$$

$$= \sum_{i,k=1}^{N} \sqrt{\sigma_i}\sqrt{\sigma_k}\, \exp(j\psi_i)\exp(-j\psi_k) \quad . \tag{3}$$

4.2 Random-Phase RCS

For the random-phase approximation, the mean RCS, $E[\sigma]$, over all random phases, is

$$\nu_1 \, \{ = E[\sigma] \} = \langle AA^* \rangle = \Sigma\, \sigma_i, \tag{4}$$

where the asterisk means complex conjugate and $\langle \cdots \rangle$ indicates an average over all phases. The deviation from the mean is

$$F = AA^* - E[\sigma]$$

$$= 2 \sum_{i<k} a_i a_k \cos(\psi_i - \psi_k)$$

$$= 2 \sum_{i<k} \sqrt{\sigma_i \sigma_k} \cos(\psi_i - \psi_k) \tag{5}$$

whence the variance is

$$\mu_2 = \langle F^2 \rangle = 2S_2 = 2 \sum_{i<k} \sigma_i \sigma_k$$

$$= \left(\sum_{i=1}^{N} \sigma_i \right)^2 - \sum_{i=1}^{N} \sigma_i^2. \tag{6}$$

4.3 Higher-Order Statistics

If the triple and quadruple sums are written as

$$S_3 = \sum_{i<k<\ell} \sigma_i \sigma_k \sigma_\ell \tag{7}$$

and

$$S_4 = \sum_{i<k<\ell<m} \sigma_i \sigma_k \sigma_\ell \sigma_m \, , \tag{8}$$

the third and fourth central moments are, respectively,[6]

$$\mu_3 = \langle F^3 \rangle = 12 S_3 \tag{9}$$

and

$$\mu_4 = \langle F^4 \rangle = 6 S_2^2 + 12 S_1 S_3 + 132 S_4 \, . \tag{10}$$

In order to examine the nature of the distributions governing RCS for a complex target, we have modified Pearson's method somewhat. Instead of examining target RCS behavior in skewness-kurtosis space, we have set up two spaces, one a width-skewness space, the other a width-kurtosis space, where width w is defined as

$$w = \mu_2/\nu_1^2 . \qquad (11)$$

In fact, to simplify the analysis somewhat, we have defined a modified skewness as

$$\gamma_1' = \mu_3/\mu_2\nu_1 . \qquad (12)$$

Kurtosis is defined in the standard manner as

$$\gamma_2 = \frac{\mu_4}{\mu_2^2} - 3 . \qquad (13)$$

From the nature of the sums S_i, $i = 1, 2, 3, 4$, above, it can be shown that the allowable regions in width-skewness (actually, read modified skewness) and width-kurtosis spaces are almost triangular and are restricted as follows (see Figs. 3 and 4):

4.3.1 Width-Skewness

Right boundary:

$$\gamma_1' = 4w - 2, \qquad \frac{1}{2} \le w \le 1 ; \qquad (14)$$

Lower boundary:

$$\gamma_1' = 0, \qquad 0 \le w \le \frac{1}{2} ; \qquad (15)$$

Left boundary:

$$w = \frac{t(t+2)}{(t+1)^2} ,$$

$$\gamma_1' = \frac{2t(t+3)}{(t+1)(t+2)} , \qquad (16)$$

$$0 \le t \le \infty .$$

4.3.2 Width-Kurtosis

Right boundary:

$$\gamma_2 = \frac{6n^2}{(n+1)(n+2)} \left(w + \frac{1}{n^2 w} - \frac{3}{n} - \frac{1}{n^2}\right) , \qquad (17)$$

where

$$n = \frac{\gamma_1' + 2}{2w - \gamma_1'} \text{ and } \gamma_1' \text{ is as above,}$$

$$\frac{1}{2} \le w \le 1 , \qquad \lim_{w \to 1} \gamma_2 = 6 ;$$

Lower boundary:

$$\gamma_2 = -1.5 , \qquad 0 \le w \le \frac{1}{2} ; \qquad (18)$$

Left boundary:

$$\text{part 1: } w = 0 , \; -1.5 \le \gamma_2 \le 0 ; \qquad (19a)$$

$$\text{part 2: } w = \frac{t(t+2)}{(t+1)^2} \quad \text{(as above)} ,$$

$$\gamma_2 = \frac{6t(t+4)}{(t+2)^2} . \qquad (19b)$$

The location of points within the restricted regions will vary with the aspect at which a complex target is illuminated and viewed. The corresponding beta distribution parameters may be calculated from those coordinates. The beta distribution is defined as

$$\beta(x) = \frac{\Gamma(a+b)}{\Gamma(a)\Gamma(b)} x^{a-1}(1-x)^{b-1} , \qquad (20)$$

$$0 \le x \le 1 ,$$

where $\Gamma(a)$ is the gamma function of argument a, and $x = \sigma/\sigma_{max}$. Then, for example, the parameters a and b can be determined from width-kurtosis space. The result can be used to find skewness, which can be checked against the location of the aspect point in width-skewness space. The equations are

$$w = \frac{b}{a(a+b+1)} ,$$

$$\gamma_1 = \frac{2(b-a)}{a(a+b+2)} ,$$

$$\gamma_2 = \frac{6(a+b+1)}{(a+b+2)(a+b+3)} \times \left(\frac{a}{b} + \frac{b}{a} - \frac{1}{a+b+1} - 3\right) . \qquad (21)$$

The values of a and b may be determined from any pair of these equations, but the first two should be used for experimental data. If one takes, for instance, the aspect of 100° (Fig. 2), one can compute $a = 0.96$ and $b = 3.82$.

In order to check the beta distribution theory against measured data, a slightly different viewpoint must be taken because the output of a measurement exercise is usually in the form of RCS versus aspect. If the RCS is sufficiently oscillatory in the chosen aspect interval (usually 10°, but ranging from 5 to 30°, depending on frequency), the random phases have had the opportunity to range over most of the intervals of definition, and so the RCS in the aspect interval should be distributed in essentially the same way that the RCS at a fixed aspect was owing to the random phase fluctuations.

4.4 Modified Statistical Distribution

An extension of the statistical theory just presented has been developed where the requirement of constant component RCS is relaxed and an individual variance and third and fourth moments about the mean are assigned. The physical regions of the statistical space resulting in this modification will be published shortly.

4.5 RCS Lobe Structure

If, instead of taking the relative phases to be random variables, one examines their variation (and hence the variation of relative-phase RCS) as a function of receiver aspect (in a bistatic situation, either aspect *or* bistatic angle), it is possible to obtain scintillation statistics. The results are derived from the RCS correlation function, using the zero crossing theorem of Rice.[10] They represent an extension of results for the monostatic case obtained by Shotland.[9]

Suppose the target is at the origin of a ξ, η, ζ coordinate system whose $\xi\zeta$ plane contains the transmitter and receiver. Then the phase term ψ_i of Eq. 1 is given by

$$\psi_i = k[\zeta_i(1 + \cos\beta) + \xi_i \sin\beta] , \qquad (22)$$

where k is the wavenumber. The results obtained are the number, N_f, of lobes of RCS per hertz of frequency:

$$N_f = \frac{1}{cS}\left(\sum_{i \neq k} \sigma_i\sigma_k[(\zeta_i - \zeta_k)(1 + \cos\beta) + (\xi_i - \xi_k)\sin\beta]^2\right) , \qquad (23)$$

where c is the speed of light and $S = \sqrt{\mu_2}$ is from Eq. 6. Also obtained are the number, N, of lobes of RCS per radian in traverse relative to a rotation axis oriented at an angle δ with the ξ axis in the $\xi\zeta$ plane:

$$N = \frac{1}{\lambda S}\left(\sum_{r,s} A_{rs} d_r d_s\right) , \quad r,s = 1,2 , \qquad (24)$$

where

$$d_1 = \sin\delta , \qquad d_2 = -\cos\delta , \qquad (25)$$

and

$$A_{rs} = \sum_{i \neq k} \sigma_i\sigma_k[(\xi_i - \xi_k)(1 + \cos\beta) - (\zeta_i - \zeta_k)\sin\beta]^{2} \times [(\eta_i - \eta_k)(1 + \cos\beta)]^{1/2} . \qquad (26)$$

The numbers N_f and N, considered as a function of δ for a fixed aspect angle, form ellipses. They can also be converted into local lobewidths of frequency and aspect, respectively, by considering their reciprocals. For any line through the ellipse center, the distance from the center to the ellipse intercept gives the lobe rate for that direction of rotation axis.

4.6 Glint

The apparent electrical location of the target reflection depends on the type of radar seeker used to measure it. For the relative-phase return (Eq. 2) with the receiver location at X, Y, Z; $X, Y << Z$; we have

$$\frac{\partial\psi_i}{\partial X} = \frac{\partial}{\partial x}(kr_i) = \frac{k(X - \xi_i)}{r_i} = k\frac{X - \xi_i}{\bar{r}} , \qquad (27)$$

[10] S. O. Rice, "Mathematical Analysis of Random Noise," in *Selected Papers on Noise and Stochastic Processes*, Nelson Wax, ed., Dover Publications, New York (1954).

and

$$\frac{\partial \psi_i}{\partial Z} = \frac{k(Z - \xi_i)}{r_i} \pm k. \qquad (28)$$

The apparent source location is

$$\xi = r \tan^{-1}\left(\frac{\partial A/\partial X}{\partial A/\partial Z}\right)$$

$$\pm \frac{\sum ika_i\, \xi_i \exp(j\psi_i)}{\sum ika_i \exp(j\psi_i)}$$

$$= \frac{\sum_{i,j} a_i a_j \xi_i \exp(j(\psi_i - \psi_j))}{\sum_{i,j} a_i a_j \exp(j(\psi_i - \psi_j))}$$

$$= \frac{1}{E} \sum_{i,j} a_i a_j \xi_i \cos(\psi_i - \psi_j), \qquad (29)$$

where the real part is taken in the last step since the RF signal is actually proportional to the real part of Eq. 2.

The di ision is normally performed by an automatic gain control (AGC) circuit with a time constant so that the numerator and denominator are averaged independently. The result in the random-phase approximation is

$$\bar{\xi} = \frac{\sum \sigma_i \xi_i}{\sum \sigma_i}. \qquad (30)$$

With a fast AGC, the fluctuations of Eq. 29 become larger but cannot be expressed in closed form. In addition, the result depends on the dynamic range of the circuits. Note that in a bistatic geometry the glint is measured relative to the receiver and is independent of bistatic angle except for the effects of shadowing of components.

Subtracting Eq. 30 from Eq. 29, we have the deviation from the mean for the slow AGC case.

$$\Delta\xi = \frac{1}{E} \sum_{i \neq j} a_i a_j (\xi_i - \bar{\xi}) \cos(\psi_i - \psi_j)$$

$$= \frac{1}{E} \sum_{i<j} a_i a_j (\xi_i + \xi_j - 2\bar{\xi}) \cos(\psi_i - \psi_j), \qquad (31)$$

and the variance of the glint is

$$\langle \Delta\xi^2 \rangle = \frac{1}{2E^2} \sum_{i<j} \sigma_i \sigma_j (\xi_i + \xi_j - 2\bar{\xi})^2. \qquad (32)$$

So far we have discussed only one component of glint, but it is clear that expressions similar to Eqs. 30 and 32 are needed for η. In addition, the variance is a tensor operator so that an off-diagonal term is needed.

Defining

$$B_{mn} = \sum_{i \neq j} \sigma_i \sigma_j (\xi_i + \xi_j - 2\bar{\xi})^{4-m-n} \times (\eta_i + \eta_j - 2\bar{\eta})^{m+n-2}, \quad m,n = 1,2 \qquad (33)$$

and

$$B = \sum_{m,n} B_{mn} a_m a_n, \qquad (34)$$

where $a_1 = \cos\alpha$, $a_2 = \sin\alpha$, the standard deviation of glint in the α direction is

$$S_\alpha = \frac{1}{2E(\sigma)} \sqrt{B}. \qquad (35)$$

Further in analogy to the derivation of Eqs. 23 and 26, we define

$$C_{mn} = \sum_{i \neq j} \sigma_i \sigma_j (\xi_i + \xi_j - 2\bar{\xi})^{4-m-n} \times (\eta_i + \eta_j - 2\bar{\eta})^{m+n-2} \times [(\zeta_i - \zeta_j)(1 + \cos\beta) + (\xi_i - \xi_j)\sin\beta]^2, \qquad (36)$$

$$D_{mnrs} = \sum_{i \neq j} \sigma_i \sigma_j (\xi_i + \xi_j - 2\bar{\xi})^{4-m-n} \times (\eta_i + \eta_j - 2\bar{\eta})^{m+n-2} \times [(\xi_i - \xi_j)(1 + \cos\beta) - (\zeta_i - \zeta_j)\sin\beta]^{4-r-s} \times [(\eta_i - \eta_j)(1 + \cos\beta)]^{r+s-2}, \qquad (37)$$

$$C = \sum_{m,n} C_{mn} a_m a_n, \qquad (38)$$

and

$$D = \sum_{m,n} \sum_{r,s} D_{mnrs}\, a_m a_n\, d_r d_s\,. \tag{39}$$

Then the number of glint lobes in the α direction per cycle is

$$N_{\alpha,f} = \frac{1}{2cS_\alpha}\sqrt{C}\,, \tag{40}$$

and the number of glint lobes in the α direction per radian of rotation about the axis oriented in the δ direction is

$$N_{\alpha,\delta} = \frac{1}{2\lambda S_\alpha}\sqrt{D}\,. \tag{41}$$

Note that Eqs. 40 and 41 are correct even though the assumption of a slow AGC has been used; only the amount of the displacement is in error.

5.0 USER'S GUIDE TO THE CODE FROM THE VIEWPOINT OF THE COMPUTER ANALYST

5.1 Introduction

The APL RCS/Statistics Code is a computer software package capable of computing and displaying the collective cross section from a group of simple components using physical optics as the main modeling basis. In addition to average cross section, it is also able to compute and display relative-phase cross section, RCS lobe and glint lobe characteristics, and higher-order beta statistics of the collective object as a function of observation angle, in either printed or graphical form. Input for the program is taken from a data base in batch mode process, and output results are displayed on a graphics terminal, plotter, or line printer. The RCS analytical basis is given in Refs. 11 and 12.

The program is currently running on a PDP 11/60 computer using the RSX-11M V3.2 operating system. The graphical peripherals available on this system are the Tektronix terminal and the ZETA plotter. The program is executed by issuing the computer command RUN RCS. It responds by prompting the user for the name of the input data base file. The data base name must then be entered, followed by any optional switches. The available options are:

/G enable graphical debugging
/GC enable graphical debugging (combined components)
/P enable printed debugging
/Z direct graphics output to plotter (default is terminal)

The program then performs the functions defined in the data base and returns to computer command mode.

This section of the report provides the user with a basic understanding of the program along with a comprehensive reference manual for actual program operations. The internal structure and program capabilities are presented in Section 5.2. Section 5.3 presents the display options, including the available debugging output. General features of the input data base, such as looping and command priorities, are described in Section 5.4. The actual RCS commands are presented in detail in Section 5.5.

5.2 RCS Program Operations

The program is designed to compute the combined RCS (called random-phase RCS) of several simple components as a function of observation angle,

[11] A. L. Maffett, *Bistatic Lobe Statistics*, JHU/APL M80-7 (Nov 1980).
[12] A. L. Maffett, *Linear Transformations Relating Theoretical and Experimental Coordinate Systems Used in RCS Determination*, JHU/APL QM-80-049 (M80-4) (Feb 1980).

wavelength, and polarization. In addition to average cross section, the program also computes relative-phase cross section, RCS lobes, average glint, and glint lobes. Finally, it may also be used to compute higher-order beta statistics.

A complex object to be analyzed in terms of RCS must first be separated into its constituent components. The basic components available to the program, such as ogives, cylinders, corner reflectors, etc., are described in Section 5.5. Each component must, at a minimum, be specified by its name, physical dimensions, and orientation in the radar coordinate system. The location of each component (or scattering center) in the radar coordinate system must also be specified if lobe or glint output is required. The geometry describing the scattering centers of each available component has been specified in Ref. 13.

The program may be divided functionally into four areas: input, cross-section computation, statistics and lobe/glint computations, and output display. The input function, which is actually in continuous operation throughout the computations, is described in detail in Section 5.4 in the context of a data base. In summary, the input must be ordered to supply the program with global parameters, followed by local component parameters, and ending with display parameters. Since knowledge of the input function or output display function is not necessary to understand the program's computational algorithms, they will not be discussed further in this section.

The program first zeros an array of accumulators and then adds in the cross-section contributions of each component read on input. This process continues until the advent of an output display instruction, which causes the accumulated results to be displayed as specified. The RCS program operates completely on one component at a time by computing the component RCS at a fixed wavelength, polarization, and radar phi angle and then varying the radar theta angle (phi and theta are the radar polar coordinates). Each cross-section value as a function of theta is stored in a separate accumulator. The range and number of theta values desired, specified on input, must not require more than 181 accumulators (a program limitation). This limitation may be circumvented in some instances as described in Section 5.3 and defined in Section 5.5.

In addition to accumulating average cross-section responses, the program also saves the square of the values of the responses in order to compute and display the standard deviation of the overall response at each theta increment. If relative-phase rather than average cross-section output is requested, it will use the two accumulator vectors to save the complex components of cross section rather than the detected sum and sum squared values. All results generated thus far may be stored in computer memory since they are relatively few in number. The lobe and/or glint options, however, require the temporary storage of large volumes of intermediate data. Those data, which consist of the cross section and three-dimensional position of each subreflector in the object for each theta angle of interest, are stored on a disk file for later processing.

The following paragraphs summarize the RCS program procedure for computing cross section for a single component. Cross-section calculations do not actually begin until a component description has been read from the input stream.

The program begins to process a component by computing a transformation matrix from component space through object space to radar space. The component-to-object transformation matrix (TRANS) will have been read previously as input. The object-to-radar transformation matrix will have been computed from previously read values of radar platform pitch, roll, yaw, and tilt (PRYT).

The program then enters the THETA loop to compute the cross section at each required theta angle. In the loop, it determines if the current radar theta and phi angles are valid for that component. If valid, they are transformed into the component coordinate system. At that point, calculations that correspond to the specific component of interest are performed.

Although they differ in detail, all groups of component calculations (i.e., subroutines) have several things in common. They must first collect their appropriate inputs and then determine if a valid response is possible from the current parameter configuration. Each component subroutine must then compute its RCS as a function of geometry, radar characteristics, and observation angle. If lobe output is also required, the component subroutines must also compute the location of the reflecting source or sources for multiple reflectors. Component position information, after being transformed to the radar coordinate system, is stored in a file for future lobe and/or glint processing.

When the component cross section has been computed, it is added to the accumulator bin appropriate to the current theta angle. The powers of these results are computed and stored in another file if optional beta statistics are desired. Beta statistics are computed in the output display section of the program using the relatively simple equations from Ref. 7. In

[13] A. L. Maffett, *Component Scattering Centers*, JHU/APL M81-1 (7 Jan 1981).

any case, the observation angle is incremented by repeating the THETA loop. After the component cross section has been computed for all required values of theta, program execution returns to the input section to obtain new component information from the data base, and the process is repeated.

Basic RCS computations pause when a display instruction is read on input. This causes the program to display its cross-section results (described in Section 5.3) and then begin lobe and/or glint calculations if required. The remainder of this section highlights the general procedure for computing lobe and glint characteristics. A complete analytical description is given in Refs. 14 and 15.

The RCS lobe calculations begin by reading temporary results stored by the main program. Each component's contribution to RCS at the observation angle of interest (specified by DELTA, a subset of THETA) is read by the lobe subprogram along with its position in space. The first parameters computed from these data are a simple cross-section sum and standard deviation. Cross-section products of component cross section and position are then computed and summed. The square root of this result divided by the standard deviation and the speed of light yields the number of lobes of radar cross section per hertz of frequency. A different set of summed cross products is used to compute lobe width.

The glint calculations begin by computing average moments for the components read in the lobe section. Sums of cross products using component cross section, position, average moments, and an intersect plane defined by the angle alpha are used to compute the glint standard deviation. The average moment arm length, along with standard deviation, as a function of observation angle (theta) is the primary output from the glint section.

Glint lobes are computed from the same set of data used to compute glint moments. In this case, an expanded set of summed cross products is computed and used in conjunction with glint standard deviation to yield the number of glint lobes per hertz of frequency with respect to observation angles delta and alpha. An even greater expanded set of summed cross products is then computed and combined with glint standard deviation to yield glint lobe width.

The RCS lobe and glint lobe computations are repeated for each required observation angle defined by the angle delta; in contrast, RCS and glint moments are defined by the observation angle theta. When computations for all values of delta have been completed, the results are displayed, and execution control is returned to the main program.

5.3 Display Options

The real power of the RCS program lies in its ability to display cross-section output results in a complete and elegant manner in both printed and graphical form. This section discusses the standard display options available in the program followed by additional debugging output. The standard display options are RCS, RCS lobes, glint, glint lobes, and higher-order statistics.

The commands that initiate the display function are GRAPH and PRINT (defined in Section 5.5). As the names imply, GRAPH produces graphical output on a terminal or a plotter, and PRINT produces printed output on a terminal or a line printer. Since the results displayed by these commands are generally the same, only differences in the data presented will be described in conjunction with the overall data available for display. Program results are not altered because of their being displayed. This allows multiple display commands to be included in the data base in order to view either accumulated results or new results if program accumulators are cleared (see the CLEAR directive).

The basic program shows monostatic full-scale RCS as a function of observation angle. Both average cross section (shown with the optional plus or minus standard deviation) and relative-phase cross section may be displayed in square meters on a logarithmic scale (i.e., in decibels relative to a square meter). Relative-phase cross section may also be shown in square meters on a linear scale. Graphical RCS output also lists certain program parameters, such as frequency, polarization, and percent shadowed, on the edge of the graph. The output from this section is always displayed as part of the GRAPH or PRINT operation.

The RCS lobe output is displayed on top of the basic cross-section output in graph mode and following it in print mode if LOBE is active. The lobe output shows the number of RCS lobes per hertz of frequency and the number of lobes per spatial increment around a circle. In graph mode, the spatial lobes are displayed inversely to yield lobe width. Samples of lobe width are computed around a circle and plotted as an ellipse centered at the observation angle of interest. The major and minor axes of the ellipse along with the orientation angle may then be found and dis-

[14] J. W. Crispin and K. M. Siegel, *Methods of Radar Cross Section Analysis*, Academic Press, New York (1968).

[15] A. L. Maffett, *Geometric and Physical Optics Formulas and Orthonormal Transformations for an RCS Computer Program*, JHU/APL M80-3 (28 Jan 1980).

played. Since these values are searched out rather than computed, lobe width must be sampled often enough (from 4 to 180 times as defined by DELTA(4)) to converge on the required accuracy.

If GLINT is active, monostatic glint is displayed as a function of all the independent parameters mentioned thus far and also the angle alpha, which defines an observation plane that cuts through the lobe structure. Since a complete graph or print list showing glint results is produced for each value of alpha, only certain values , namely 0 and 90°, are used. The varying independent parameter on a glint graph or print list is the observation angle theta. The dependent parameter represents the glint moment arm displacement in degrees; values of standard deviation are included as an option.

Monostatic glint lobe output is displayed in a manner completely analogous to cross-section lobe output. It, therefore, similarly shows the number of lobes of glint per hertz of frequency and the number of lobes per spatial angle at samples evenly spaced around a circle. The amount of glint lobe data output is, as with the RCS lobes, completely defined by the parameter array DELTA.

The final standard display option presents higher-order beta statistics computed from the basic RCS results. The type and amount of output produced by this option are very different between graph and print modes. The two plots generated in graph mode show skewness and kurtosis as a function of beta width. Distribution acceptance boundaries are drawn on each plot, followed by the actual computed data values, which are annotated with their corresponding observation angle. Many additional statistics are computed during print mode. The results are displayed within the normal RCS output listing rather than at the end.

Three mutually exclusive debugging display options are available in the RCS program. All three are intended to provide partial results corresponding to each separate component rather than the combined object. When the cross-section contribution of a component is computed, it is immediately displayed on either a printer (option /P) or a graphics output device (option /G). After it is displayed, the accumulator vectors are reset to zero, and the next component is computed and displayed on a new listing or graph. This process continues until the entire data base has been read. Prints or graphs for which there is no component contribution are not displayed. Furthermore, component display may be aborted in graph mode by entering a control −C from the terminal keyboard for each component not desired. The final debugging option, /GC, instructs the program to combine all component curves onto one graph.

Debugging display options may not be used in conjunction with lobe, glint, or higher-order statistics output. Any PRINT or GRAPH commands found in the data base are simply ignored by the program in this mode.

5.4 Data Base Description

All inputs to the RCS program must be suplied from within a data base. A data base is a file that may be created or modified by an editor. It contains all the instructions necessary to control the program. Each line in the data base, except comment and title lines, constitutes a single, complete instruction to the program. Each instruction line begins with either four or five characters, defining the directive type, followed by a space and up to nine numeric parameters. All directives (i.e., instructions) recognized by the program and their functions are defined in detail in Section 5.5.

In this section we will discuss directive interactions and usage protocols, the concepts of global and local directives followed by their relative ordering within the data base, and then looping capabilities within the data base.

All valid directives in the RCS data base fall into one of two functional categories: component definition or program control. All component definition directives are considered to be local parameters. The program control directives may be either global or local. A global parameter, once defined, retains its specified value until explicitly altered later in the data base. A local parameter only retains its assigned value until the occurrence of a component definition directive, which has the effect of resetting all local RCS program control directives and then canceling itself. The local parameters must therefore be reassigned before the occurrence of the next component directive, if required. The local RCS program control directives and their defaults or reset values are: ORGN (zero), RAMDB (zero dB), TRANS (identity matrix), and VALID (all angles).

As previously indicated, local parameters must be ordered so that all control directives related to a component directive appear before the component in the data base. Global RCS directives that define radar parameters must also appear before the component directives. The radar directives are: DELTA, GLINT, LOBE, PHIR, POLAR, PYRT, RPRCS, STATS, THETA, and WLEN. The display directives GRAPH and PRINT (either or both may be specified) must appear in the data base after the required component directives. A single title line must always be placed immediately after the display directives. The two directives NEWTH and NEWWL, which control data base looping, must always be placed at the end of the data base, if used. If both directives are used, NEWTH must precede NEWWL. The remaining directives may appear anywhere in the data base within the constraints given.

The RCS program may be instructed to reexecute the entire data base by using the directives NEWTH for new THETA and/or NEWWL for new wavelength (WLEN). The maximum number of THETA increments allowed on one pass of the data base is 181. That limit may be extended by using the NEWTH directive in conjunction with the parameter THETA(4). The actual number of increments used (NOI) is computed as 1 + (THETA(2) − THETA(1))/THETA(3). For the first pass through the data base, the observation angle ranges from THETA(1) to THETA(2) in NOI steps. If the directive NEWTH is detected, the RCS program stops current processing and starts reading the data base from the beginning. If, when the THETA directive is read, THETA(2) is larger than or equal to THETA(4), the RCS program immediately jumps to the first line following the NEWTH directive. The only directives that may follow NEWTH and NEWWL are comments. If, on the other hand, THETA(4) is larger than THETA(2), the starting value, THETA(1), is replaced with THETA(2), and the ending value, THETA(2), is replaced with the minimum of THETA(2) + NOI and THETA(4). This processing loop continues as described while THETA(4) remains larger than THETA(2). All grapical output resulting from this loop is placed on the same graph. Debugging options and the directives LOBE, GLINT, and STATS are not valid if THETA looping has been specified.

The set new wavelength directive, NEWWL, instructs the RCS program to reexecute the data base in a manner similar to NEWTH. However, in this case, each time the directive WLEN is read, the next value in the parameter list will be used as the new wavelength. For example, if the WLEN directive has three associated parameter values from a maximum of nine, the wavelength will be set to the first parameter value on pass one, the second value on pass two, and the third on pass three. The RCS program will not loop for pass four because a fourth parameter value was not specified on the WLEN command line. The only valid lines following the NEWWL directive are comment lines.

5.5 Data Base Directives

This section presents a complete list of valid RCS directives and associated parameters. The directives have been divided into two groups: RCS control and component definition. A complete description of the available RCS components is given in Refs. 11 and 12. All angles must be specified in degrees and distances in meters.

5.5.1 RCS Control Directives

CLEAR	Clear RCS – clears all accumulated radar cross-section results.
COMM *n*	Comment – ignores the next *n* lines in the data base. This instruction may be used to insert a specified number of comment lines in the data base.
COME	Comment End – ends comment block mode. This instruction will cancel the effect of COMS. The next data base line will be processed normally.
COMS	Comment Start – starts comment block mode. All data lines following this one in the data base will be ignored until the COME directive is found. These two directives (COMS and COME) provide the user with an efficient tool for ignoring blocks of data.
DELTA *n*1, . . .,*n*4	Delta Angles – defines the lobe display angles. RCS lobe and glint lobe results will be displayed at angles ranging from *n*1 to *n*2 in increments of *n*3 degrees. The number of lobe samples, between 4 and 180, is defined by *n*4. Only delta angles that lie within the theta range will be used.
GLINT *n*	Glint Switch – activates the glint lobe calculations if *n* is set to 1 or 2 and LOBE is active. Glint data will not be computed otherwise. If *n* is set to 2, the standard deviation will be added to the displacement curve.

GRAPH *n*1, . . .,*n*9 — Graph Display – collected results will be output in graphical form (see Section 5.3). For average cross-section output, plus/minus standard deviation curves will be added to the graph if the absolute value of *n*1 is equal to 3 instead of 1. If *n*1 is negative, terminal graphs will be automatically sent to a hard-copy device. The x-axis end points are specified by *n*2 and *n*3. The number of x-axis labels and tick marks per label are specified by *n*4 and *n*5. The y-axis end points are specified by *n*6 and *n*7. The number of y-axis labels and tick marks per label are specified by *n*8 and *n*9. The next line following this instruction is always used for the title. This instruction will not affect any RCS results.

GSIZE *n*1, . . . ,*n*5 — Graph Size – locates the graph boundaries on the plotter page. The lower left-hand corner of the graph will be placed at *n*1,*n*2 inches from the paper corner and the upper right-hand corner at *n*3,*n*4 inches. The column of data displayed at the right of the graph may be further displaced by *n*5 inches. The default values for *n*1 through *n*5 are 2.25, 2.25, 8.75, 6.25, and 0 inches, respectively. If used, GSIZE must appear before the GRAPH directive.

LOBE *n* — Lobe Switch – activates the cross-section lobe calculations if *n* is set to 1. Lobe data will not be computed otherwise.

NEWTH — New Theta – reruns data base with new theta angle range. This instruction will cause the RCS program to re-execute the data base from the beginning using the next theta range computed by the THETA directive. The instruction will be ignored if no new theta range is required (see THETA and Section 5.4).

NEWWL — New Wavelength – reruns data base with new wavelength. This instruction will cause the RCS program to re-execute the data base from the beginning using the next wavelength on the WLEN directive. The instruction will be ignored if no new wavelength is available (see WLEN and Section 5.4).

ORGN x,y,z — Component Origin – the component following this instruction will have its origin located at x, y, and z in radar coordinates. This instruction, useful for lobe and glint computations only, will be reset to zero after the next component.

PAUSE — Pause Program – the RCS program will stop and wait until a carriage return is received from the user terminal. This command should not be used in batch mode.

PHIR *n* — PHI Radar – sets radar phi angle to *n* degrees.

POLAR *n* — Polarization – sets system polarization to either vertical (*n* = 1) or horizontal (*n* = 2).

PRINT *n* — Print Display – collected results will be output in printed form (see Section 5.3). Printed output will be displayed on the user terminal if *n* = 0. It will be sent to a nonspooled line printer if *n* = 1, a spooled line printer if *n* = 2, and a file if *n* = 3. The next line following this instruction is always used for the title. This instruction will not affect any RCS results.

PRYT p,r,y,t — Pitch, Roll, Yaw, Tilt – defines radar rotation angles. These angles will be used to construct the radar-to-object coordinate system transformation matrix. PRYT will default to the identity matrix if not defined. If all angles are zero, the resultant matrix will have its x and y components switched.[15]

RAMDB *n* — Radar Absorption Material – modifies the effective cross section of a component. The RCS of the next component following this instruction will have *n* dB added to it before it is used. Thereafter, the value of *n* will be reset to 0 dB.

RPRCS *n*1, *n*2 — Relative Phase Switch – activates the relative phase RCS calculations if *n*1 is set to 1. Relative-phase data will not be computed otherwise. Relative-phase cross section will be displayed on a logarithmic scale unless *n*2 is specified, which defines the maximum linear value to be graphed in square meters.

STATS *n* — Statistics Switch – activates the beta higher-order statistics calculations if *n* is set to 1. Higher-order statistics will not be computed otherwise.

THETA *n*1, . . ., *n*4 — Theta Angle Range – sets radar theta range and increment. The radar observation angle, theta, will vary from *n*1 to *n*2 degrees in increments of *n*3. The total number of discrete values may not exceed 181. If the total maximum angle, *n*4, is present, theta will vary from *n*1 to *n*4 degrees in groups of (*n*2 − *n*1)/*n*3 increments. Since this option reruns the RCS program for each THETA group, the number of discrete observation angles allowed becomes unlimited. The command NEWTH must be included at the end of the input stream (but before the directive NEWWL, if present) to force the program to loop to the next THETA group (see Section 5.4). If the parameter *n*4 is specified, the following commands are not allowed: LOBE, GLINT, STATS, and debugging options.

TRANS *n*1, . . . ,*n*9 — Transformation Matrix – assigns component orientation. The nine values *n*1 through *n*9, specified in degrees, define, by row, the direction cosine matrix of the component following this instruction. The transformation matrix will be checked for validity before being used by a component and then reset to the identity matrix after being used.

VALID *n*1, . . . ,*n*8 — Valid Range – sets range of valid angles for component. The values of *n*1 through the maximum *n*8, specified in degrees, will define observation angles for which the following component is visible. The parameter values must be specified in pairs and imply the following: *n*1, *n*2 = first valid range of THETA; *n*3, *n*4 = associated valid range of PHIR (all PHIR if not present); *n*5, *n*6 = optional second valid range of THETA; *n*7, *n*8 = associated valid range of PHIR. All ranges are assumed to be counterclockwise from the first element in a pair to the second. The valid range will be reset to all angles after being used for a component.

WLEN *n*1, *n*2 . . . — Wavelength – assigns program wavelengths. This instruction sets the initial program wavelength to *n*1 meters. Each following pass through the data base in looping mode will set the program wavelength to the next *n* on the command line (pass 2 would use a wavelength of *n*2, etc.) up to a maximum of *n*9. The instruction NEWWL must be used at the end of the data base if multiple wavelength passes (i.e., looping) are desired (see Section 5.4).

5.5.2 Component Definition Directives

CAVA	a	Cavity – defines a cavity with equivalent circular area a.
CAVD	d	Cavity – defines a cavity with circular diameter d.
CFLAT	r	Circular Flat Plate – defines a flat plate with radius r.
CONE	$A,n,a,$ $L1, L2$	Truncated Elliptical Cone – defines a cone with lengths $L1$ and $L2$ from the imaginary cone tip. Cone tip to $L1$ is truncated. The elliptical cone base's major radius length is a and the major-to-minor ratio is n. The cone half-angle is defined by A.
CYLIN	a,b,L	Elliptical Cylinder – defines a cylinder with major radius, a, minor radius b, and length L.
DIHED	a,b,c	Dihedral – defines a 90° dihedral with sides a and c and height b.
ECORN	a,b,c	Elliptical Corner – defines an elliptical corner with edges a, b, and c.
LWIRE	a	Loop Wire – defines a circular wire loop with radius a.
OGIVE	R,a,b,A	Ogive – defines an ogive with radius of curvature R and center to nearest side length a. If ogive is truncated, parameter b and angle A specify the amount of truncation (b is zero otherwise).
PARAB	p	Parabola – defines a parabola where the parabola equation is $p = (x^{**}2 + y^{**}2)/-4z$.
RCORN	a,b,c	Rectangular Corner – defines a rectangular corner with edges a, b, and c.
RFLAT	a,b	Rectangular Flat Plate – defines a flat plate with sides a and b.
SOLID	a,b,c	Solid – defines an ellipsoid with radii a, b, and c.
TCORN	a,b,c	Triangular Corner – defines a triangular corner with edges a, b, and c.
TORUS	a,b	Torus – defines a torus with inner radius $a - b$ and outer radius $a + b$.
TWAVE	p,L,g	Traveling Wave – defines a traveling wave with the parameters p (relative-phase velocity), L (length), and g (reflection coefficient).
WEDGE	A,L,B	Wedge – defines a wedge with angle A and height L. Parameter B is used to limit the cross section to B/WLEN**2.
WIRE	L,a	Wire – defines a wire with length L and wire radius a.

6.0 PROGRAM LISTING

```
>>>  FILE = SY0:[111,12]RCS.FTN                 27-SEP-83   14:32:52   <<<
C*******************************************
C
C                 RRRR     CCCC     SSSS
C                 R   R   C        S
C                 RRRR    C         SSS
C                 R  R    C            S
C                 R   R    CCCC    SSSS
C
C
C       R A D A R    C R O S S    S E C T I O N
C
C*******************************************
C
C  NOTE!  THIS PROGRAM STORES MATRICES BY ROWS (NOT COLUMNS)
C
C      MAXRCS CONTAINS THE CURRENT MAXIMUM RCS BUFFER SIZE (181)
        REAL*8 T1,T2,T3,T4,T5,T6,SQ,ZERO
        REAL*4 CMDS(26),OBJS(18),RCS(181),RCS2(181),TRANS(9),VALID(8)
        REAL*4 ARR(9),PRYT(9),MAT(9),COMP(9),PRYTV(4),PRYT2(9)
        LOGICAL*2 ATTN
        LOGICAL*1 FILE(80),BUF(80),SLASH,AG,AP,OBJECT,AZ,AC,AN,DOT,DRCS(4)
        REAL*4 POLVH(2),YYV(181,2),XV(181),SLOBE(4,181,2)
        EQUIVALENCE (FILE,BUF),(YYV,SLOBE),(XV,SLOBE(1,1,2))
        REAL*4 IDENT(9),GRARR(9),SIDE(3,4),DEBOUT(6),ORGN(3),M60DB
        REAL*4 DELTA(4),ALPHA(4),SCALE(6),THETA(4),GSIZE(5)
        INTEGER*2 LSIDE(4),ROUTE
C
        INTEGER*2 DEBUG,COMBIN,GLINT
        LOGICAL*2 GOULD,LP,OPENE,LPF,OLDGRF,MGRAPH,GDAUTO
        COMMON/SWITCH/ IOW,THETA,GOULD,LP,OPENE,LPF,OLDGRF,MGRAPH,
        1 GDAUTO,DEBUG,COMBIN,GLINT,IBUG,LBUG,ISTATS,IZETA,STATS,NRCS,IOR
C
        COMMON/RCSCOM/CUTOFF,WLEN,PI,PI2,RCSC,THP,CTHP,STHP,PHP,CPHP,SPHP,
     $ C2PHP,S2PHP,RAD,POLAR,LOBE,
     $ ARR
        COMMON/RCSLOB/IOL,ORGN,COMP,SLOBE,BETA,
     $  ANG,CANG,SANG,PHR,CPHR,SPHR,KLOBE
        COMMON /RPRCS/ IRPR,IRCS,RAM,RCS,RCS2
        COMMON /GRCOM/ GRLIB
C
        DATA POLVH/'VER','HOR'/,CLEAR/'CLEA'/,COME/'COME'/,SLASH/'/'/
        DATA CMDS/'COMM','COMS','WLEN','POLA','TRAN','VALI','PHIR','THET',
     $ 'GRAP','PRIN','PRYT','PAUS','CLEA','DEBU','NEWW','RAMD','LOBE',
     $ 'BETA','ORGN','DELT','ALPH','RPRC','STAT','GLIN','NEWT','GSIZ'/
        DATA OBJS/'OGIV','CYLI','CONE','PARA','SOLI','TORU','WEDG',
     $ 'WIRE','LWIR','CAVA','CAVD','CFLA','RFLA','TWAV','DIHE','TCOR',
     $ 'RCOR','ECOR'/
        DATA IDENT/1.,0.,0.,0.,1.,0.,0.,0.,1./,AG/'G'/,AP/'P'/,AC/'C'/
        DATA DEBOUT/0.,' COM','PONE','NT O','UTPU','T'/,ALL/'ALL '/
        DATA GRARR/-1,0,180,7,3,-60,60,7,1/,AZ/'Z'/,AN/'N'/,DOT/'.'/
        DATA SIDE/'STAR','BOAR','D','PORT',0,0,'TOP',0,0,'BOTT','OM',0/
        DATA LSIDE/7,4,3,6/,DRCS/'.','R','C','S'/,ZERO/0.D0/
        DATA GSIZE/2.25,2.25,8.75,6.25,0./
C
C  DEBUG OUTPUT
C       IBUG=1=>RCS OUTPUT
C       IBUG=2=>TRANS,PRYT,COMP
C       IBUG=4=>ANGLES(RADAR&PRIME)
C
```

```
C
C      SET RCS CONSTANTS
C
          NCMD=26
          NOBJ=18
          MAXRCS=181
          PI=3.1415926
          PI2=PI/2.
          RAD=PI/180.
          CUTOFF=1.E-10
          LINE=0
          GOULD=.FALSE.
C
C
C      ASSIGN LUNS AND ATTACH INPUT FILE
C
          IOR=1
          IOW=2
          IOP=3
          IOG=4
          IOF=5
          IOL=6
          IOE=7
C
          CALL ASNLUN(IOR,'TI',0)
          CALL ASNLUN(IOW,'TI',0)
          CALL ASNLUN(IOF,'SY',0)
          CALL ASNLUN(IOP,'GD',0)
          CALL ASNLUN(IOL,'SY',0)
          CALL ASNLUN(IOE,'SY',0)
          OPENE=.FALSE.
C
          WRITE(IOW,1)
1      FORMAT(' ENTER RCS DATA FILE NAME')
          READ(IOR,2,END=9999) FILE
2      FORMAT(80A1)
          CALL TRIM80(NF,FILE)
          FILE(NF+1)=0
C
C         INPUT OPTIONS
C
C /Z      = DRAW GRAPHS ON ZETA PLOTTER
C /P      = PRINT RESULTS FOR EACH COMPONENT
C /Q      = GRAPH RESULTS FOR EACH COMPONENT
C /QC     = DRAW ALL /Q OUTPUT ON ONE GRAPH (COMBINE)
C
          DEBUG=0
          COMBIN=0
          IZETA=0
          IF=1
          IDOT=0
          DO 6 J=IF,NF
          JJ=J
          IF(FILE(JJ).EQ.DOT) IDOT=JJ
          IF(FILE(JJ).EQ.SLASH) GO TO 7
6         CONTINUE
          GO TO 8
7         NF=JJ-1
          IF(FILE(JJ+1).EQ.AZ) IZETA=200
          IF(FILE(JJ+1).EQ.AG) DEBUG=1
```

```
          IF(FILE(JJ+1).EQ.AP) DEBUG=2
          IF(DEBUG.EQ.1.AND.FILE(JJ+2).EQ.AC) COMBIN=-1
          IF=JJ+2
8         IF(IDOT.GT.0) GO TO 9
C    DEFAULT FILE EXTENTION TO  .RCS
          DO 13 J=1,4
          NF=NF+1
          FILE(NF)=DRCS(J)
13        CONTINUE
9         FILE(NF+1)=0
          OPEN(UNIT=IOF,NAME=FILE,TYPE='OLD',READONLY)
C
          TIME=SECNDS(0.0)
          CALL GRINIT(0,IZETA,IOG,IER)
          CALL GRHV('H')
          CALL GRPIC(2.25,2.25,8.75,6.25)
          IF(IZETA.EQ.0) CALL GRDEV('TEK','HARD',IER)
C
C       INITIALIZE RCS VECTOR & OTHER STUFF
C
          CMD=0.
          IWL=1
          OLDGRF=.FALSE.
          GO TO 70
C
C
C       ###############################
C       READ NEXT LINE IN RCS DATA FILE
C       ###############################
C
3       CONTINUE
          IF(OBJECT.AND.DEBUG.EQ.1) GO TO 45
          IF(OBJECT.AND.DEBUG.EQ.2) GO TO 50
C
          LINE=LINE+1
          READ(IOF,4,END=9990,ERR=9993) CMD,ARR
4       FORMAT(A4,1X,9F10.0)
C
C       DETERMINE IF CMD IS A DIRECTIVE
C
          DO 5 J=1,NCMD
          IF(CMD.EQ.CMDS(J))
       $ GO TO (10,12,15,20,25,30,35,40,45,50,55,60,70,80,85,65,
       $ 500,510,520,540,550,560,570,580,590,600), J
5       CONTINUE
C
C       GO TO OBJECT SEARCH IF NO DIRECTIVE MATCH FOUND
C
          GO TO 90
C
C
C
C       ###############################
C       EXECUTE  APPROPRIATE  DIRECTIVE
C       ###############################
C
C
C       "COMM" - SKIP COMMENT LINES
C
10      NCOMM=ARR(1)
```

```
          IF(NCOMM.LE.0) GO TO 3
          DO 11 J=1,NCOMM
          LINE=LINE+1
 11     READ(IOF,4,END=9990) CMD
          GO TO 3
 C      "COMS" - SEARCH FOR "COME"
 12     READ(IOF,4,END=9993) CMD
          LINE=LINE+1
          IF(CMD.EQ.COME) GO TO 3
          GO TO 12
 C
 C      "WLEN" - SET WAVELENGTH
 C
 15     WLEN=ARR(IWL)
          IF(WLEN.LE.0.AND.IWL.EQ.1) GO TO 9991
          IF(WLEN.LE.0) GO TO 9990
          GO TO 3
 C
 C      "POLAR" - SET POLARIZATION VECTOR
 C
 20     POLAR=AMAX1(1.,AMIN1(2.,ARR(1)))
          GO TO 3
 C
 C      "TRANS" - COMPUTE COORDINATE TRANSFORMATION MATRIX 1
 C
 25     CONTINUE
          DO 26 J=1,9
 26     TRANS(J)=COS(ARR(J)*RAD)
          CALL TRANSP(TRANS,MAT)
          CALL MATMUL(TRANS,MAT,MAT)
 C  MAKE SURE TRANS IS VALID
          DO 24 J=1,9
          IF(ABS(MAT(J)-IDENT(J)).GT.1.E-3) GO TO 27
 24       CONTINUE
          GO TO 3
 27       WRITE(2,28)
 28       FORMAT(' RCS> -- BAD TRANS DIRECTION COSINE MATRIX'/' RCS>')
          WRITE(2,29) ARR
 29       FORMAT(' RCS>   ',3F10.3)
          WRITE(2,29)
          WRITE(2,29) MAT
          GO TO 9993
 C
 C      "VALID" - SET VALID ANGLE REGIONS
 C
 30     CONTINUE
          DO 31 J=1,8
 31     VALID(J)=ARR(J)
          GO TO 3
 C
 C      "PHIR" - SET RADAR PHI ANGLE
 C
 35     PHR=ARR(1)
          IF(ABS(PHR).GT.180.) WRITE(IOW,36) PHR
 36       FORMAT(' RCS> -- PHIR=',F6.1,' IS OUT OF BOUNDS')
          GO TO 3
 C
 C      "THETA" - SET THETA RANGE (MIN,MAX,INC) & TOTAL MAX
 C
 40       IF(OLDGRF) GO TO 43
```

```
          THETA(1)=ARR(1)
          IF(ARR(3).LE.0.) GO TO 42
          THETA(2)=AMAX1(THETA(1),ARR(2))
          THETA(3)=ARR(3)
          THETA(4)=ARR(4)
          MGRAPH=.FALSE.
          IF(THETA(4).GT.THETA(2)) MGRAPH=.TRUE.
C
          NRCS=(ARR(2)-ARR(1))/ARR(3)+1.001
          IF(NRCS.LE.MAXRCS) GO TO 3
          WRITE(IOW,41)
41      FORMAT(' RCS> -- TOO MANY OUTPUT POINTS')
          GO TO 9993
C
42      THETA(2)=ARR(1)
          THETA(3)=1.
          NRCS=1
          GO TO 3
C
43        T=THETA(1)
          THETA(1)=THETA(2)
          THETA(2)=AMIN1(2.*THETA(1)-T,THETA(4))
          GO TO 3
C
C       "GRAPH" - GRAPH RESULTANT RCS VECTOR = CMD(9)
C
45      CONTINUE
C         IF(ISTATS.EQ.1) GO TO 4542
          IF(NRCS.LE.1.OR.DEBUG.EQ.2) GO TO 3
          IF(IZETA.EQ.0.AND.COMBIN.LE.0.AND..NOT.OLDGRF)
          1 CALL GRMODE('ERASE',3)
          ATTN=.FALSE.
          IF(CMD.EQ.CMDS(9)) GO TO 4546
C
C  GRAPH DEBUG MODE
          M60DB=1./10**6
          DO 4541 J=1,NRCS
          IF(RCS(J).GT.M60DB) GO TO 4540
4541      CONTINUE
          GO TO 4558
C
4540      CALL TRAPCC(IOR,ATTN)
          DO 4545 J=1,9
4545      ARR(J)=GRARR(J)
4546    CALL GRSCL(ARR(2),ARR(6),ARR(3),ARR(7))
          IF(IRPR.GT.0.AND.RPRCSQ.GT.0.)CALL GRSCL(ARR(2),0.,ARR(3),RPRCSQ)
          SCALE(1)=ARR(2)
          SCALE(2)=ARR(6)
          SCALE(3)=ARR(3)
          SCALE(4)=ARR(7)
          SCALE(5)=ARR(4)
          SCALE(6)=ARR(5)
          IF(COMBIN.EQ.1.OR.OLDGRF) GO TO 4547
C
          Y=GSIZE(2)
          IF(KLOBE.LE.0) GO TO 44
          Y=Y-.95
          CALL GRPLOT(GSIZE(3),GSIZE(2),0)
          CALL GRPLOT(GSIZE(1),GSIZE(2),1)
C
```

```
44      CONTINUE
      CALL GRAXIS(0,GSIZE(1),Y,GSIZE(3)-GSIZE(1),0.,ARR(2),ARR(3),
        1 IFIX(ARR(4)),IFIX(ARR(5)),'OBSERVATION ANGLE',99,1)
        DY=GSIZE(4)-GSIZE(2)
        IF(IRPR.EQ.0.OR.RPRCSG.EQ.0.) GO TO 4560
C
        CALL GRAXIS(3,GSIZE(1),GSIZE(2),DY,90.,0.,RPRCSG,LRPRCS,2,
        1       'RCS (SM)',8,1)
        GO TO 4547
C
4560    CALL GRAXIS(3,GSIZE(1),GSIZE(2),DY,90.,ARR(6),ARR(7),IFIX(ARR(8)),
     $ IFIX(ARR(9)),'RCS (DBSM)',10,1)
C
4547      ANG=THETA(1)
        DO 46 J=1,NRCS
        XV(J)=ANG
        ANG=ANG+THETA(3)
        YYY=RCS(J)
        IF(IRPR.EQ.0) GO TO 4561
        YYY=RCS(J)**2+RCS2(J)**2
        IF(RPRCSG.EQ.0.) GO TO 4561
        YYV(J,1)=YYY
        GO TO 46
4561    YYV(J,1)=10.*ALOG10(AMAX1(CUTOFF,YYY))
46      CONTINUE
        CALL GRWIND(-NRCS,XV,YYV,IB,0,.1,0.,1)
        IF(COMBIN.EQ.1) GO TO 4558
        IF(ABS(ARR(1)).LE.1.OR.DEBUG.EQ.1.OR.IRPR.EQ.1) GO TO 49
        SIGN=-1.
        DO 47 K=1,2
        DO 48 J=1,NRCS
        S=SQRT(AMAX1(0.,RCS(J)**2-RCS2(J)))
48    YYV(J,K)=10.*ALOG10(AMAX1(CUTOFF,RCS(J)+SIGN*S))
        CALL GRWIND(-NRCS,XV,YYV(1,K),IB,0,.1,0.,1)
47    SIGN=1.
49      IF(IZETA.EQ.0) CALL GRMODE('ALPHA',2)
        IF(CMD.EQ.CMDS(9)) GO TO 4548
        DEBOUT(1)=CMD
        IF(COMBIN.EQ.-1) DEBOUT(1)=ALL
        CALL GRTXT(DEBOUT,21)
        GO TO 4549
4548  READ(IOF,2,END=9993) BUF
        LINE=LINE+1
        IF(OLDGRF) GO TO 4559
        CALL TRIM80(NB,BUF)
        CALL GRTXT(BUF,NB)
        GDAUTO=.FALSE.
        IF(ARR(1).LT.0.) GDAUTO=.TRUE.
4549    XMID=(GSIZE(3)+GSIZE(1))/2.
      CALL GRSXY(XMID,GSIZE(4)+.75,XX,YY)
        CALL GRPRNT(0,XX,-1,YY,.15,0.)
        CALL GRTXT('MONOSTATIC FULL-SCALE ',0)
        IF(IRPR.EQ.0) CALL GRTXT('RCS',0)
        IF(IRPR.EQ.1) CALL GRTXT('RPRCS',0)
        CALL GRSXY(XMID,GSIZE(4)+.5,XX,YY)
        CALL GRPRNT(0,XX,-1,YY,.15,0.)
C
C  SET TERMINAL TO ALPHA MODE AND SCALE TO INCHES
        IF(IZETA.EQ.0) CALL GRMODE('ALPHA',3)
        CALL GRSCL(GSIZE(1),GSIZE(2),GSIZE(3),GSIZE(4))
```

```
      X=GSIZE(3)+1.05+GSIZE(5)
C
      Y=GSIZE(4)+.75
      FREQ=.3/WLEN
      CALL GRTXT('FREQUENCY ',0)
      CALL GRPRNT(1,X,0,Y,.12,0.)
      CALL GRNUM('F',-1,3,FREQ)
      CALL GRTXT(' GHZ',0)
      CALL GRPRNT(-1,X,0,Y,.12,0.)
      Y=Y-.2
      CALL GRTXT('POLARIZATION ',0)
      CALL GRPRNT(1,X,0,Y,.12,0.)
      CALL GRTXT(POLVH(IFIX(POLAR)),3)
      CALL GRPRNT(-1,X,0,Y,.12,0.)
      IF(MGRAPH) GO TO 4542
      Y=Y-.2
      PERSH=100.-100.*FLOAT(NVALID)/AMAX0(1,NTOTAL)
      CALL GRTXT('% SHADOWED ',0)
      CALL GRPRNT(1,X,0,Y,.12,0.)
      CALL GRNUM('F',-1,1,PERSH)
      CALL GRPRNT(-1,X,0,Y,.12,0.)
 4542 Y=Y-.2
      CALL GRTXT('PITCH ',0)
      CALL GRPRNT(1,X,0,Y,.12,0.)
      CALL GRNUM('F',-1,3,PRYTV(1))
      CALL GRPRNT(-1,X,0,Y,.12,0.)
      Y=Y-.2
      CALL GRTXT('ROLL ',0)
      CALL GRPRNT(1,X,0,Y,.12,0.)
      CALL GRNUM('F',-1,3,PRYTV(2))
      CALL GRPRNT(-1,X,0,Y,.12,0.)
      Y=Y-.2
      CALL GRTXT('YAW ',0)
      CALL GRPRNT(1,X,0,Y,.12,0.)
      CALL GRNUM('F',-1,3,PRYTV(3))
      CALL GRPRNT(-1,X,0,Y,.12,0.)
      Y=Y-.2
      CALL GRTXT('TILT ',0)
      CALL GRPRNT(1,X,0,Y,.12,0.)
      CALL GRNUM('F',-1,3,PRYTV(4))
      CALL GRPRNT(-1,X,0,Y,.12,0.)
      Y=Y-.2
      ISIDE=0
      IF(PHR.EQ.90.) ISIDE=1
      IF(PHR.EQ.-90.) ISIDE=2
      IF(PHR.EQ.0.) ISIDE=3
      IF(PHR.EQ.180.) ISIDE=4
      IF(ISIDE.EQ.0) GO TO 4556
      CALL GRTXT(SIDE(1,ISIDE),LSIDE(ISIDE))
      CALL GRPRNT(0,X,0,Y,.12,0.)
      GO TO 4557
 4556   CALL GRTXT('PHIR ',0)
      CALL GRPRNT(1,X,0,Y,.12,0.)
      CALL GRNUM('F',-1,3,PHR)
      CALL GRPRNT(-1,X,0,Y,.12,0.)
 4557 CONTINUE
      IF(MGRAPH) GO TO 4559
      IF(DEBUG.NE.0) GO TO 4550
C
C     COMPUTE AND GRAPH LOBE STATISTICS IF REQUIRED
```

```
C
          IF(KLOBE.LE.0) GO TO 4551
          CALL GRPLOT(11.,0.,0)
          IGRPR=1
          WRITE(IOL'1) IGRPR,KLOBE,GLINT,NRCS,NLOBE,THETA,ALPHA,DELTA,
          1 BETA,SCALE,IZETA,IPCODE,WLEN,LBUG,NB,BUF,GDAUTO,
          2 FREQ,POLVH(IFIX(POLAR)),PERSH,PITCH,ROLL,YAW,TILT,PHR,GSIZE
          CLOSE(UNIT=IOL)
          CALL SPAWNS
          OPEN(UNIT=IOL,NAME='LOBE.TMP',TYPE='OLD',RECORDSIZE=181,
          2 ACCESS='DIRECT',SHARED)
C
 4551     IF(ISTATS.EQ.0) GO TO 4550
C
C         GRAPH HIGHER ORDER BETA STATISTICS IF REQUIRED
C
          CALL HOSBD(1,DUMMY,IOE,DUMMY,SLOBE,BUF,NB,
          1       FREQ,POLVH(IFIX(POLAR)),PERSH,PITCH,ROLL,YAW,TILT,PHR)
C
C         CLEANUP GRAPHS
C
 4550     IF(GDAUTO.AND.IZETA.EQ.0.AND.DEBUG.EQ.0) CALL GRTEKG
          IF(DEBUG.EQ.1.AND.COMBIN.EQ.0.AND..NOT.ATTN) CALL GRTEKG
 4559     IF(IZETA.EQ.0) CALL GRPLOT(0.,8.4,0)
          IF(IZETA.EQ.0) CALL GRMODE('ALPHA',3)
          IF(IZETA.GT.0.AND..NOT.MGRAPH) CALL GRPLOT(11.,8.5,-1)
          IF(IZETA.GT.0.AND.MGRAPH) CALL GRUPPN
          IF(CMD.EQ.CMDS(9)) GO TO 3
 4558     COMBIN=IABS(COMBIN)
          IF(ATTN) CALL DETACH(IOR,IER)
          CMD=CLEAR
          CALL GRPLOT(0.,8.4,0)
          CALL GRMODE('ALPHA',3)
          GO TO 70
 4544     READ(IOF,2) BUF
          LINE=LINE+1
          GO TO 3
C
C       "PRINT" - PRINT RESULTANT RCS VECTOR
C
 50     CONTINUE
          IF(DEBUG.GE.1.AND.CMD.EQ.CMDS(10)) GO TO 4544
          IO=IOW
C  FORCE OUTPUT TO LINE PRINTER FOR DEBUG
          IPCODE=ARR(1)
          IF(ARR(1).GE.1.0.OR.CMD.NE.CMDS(10)) IO=IOP
          ANG=THETA(1)
          ANGINC=THETA(3)
          LP=.FALSE.
          LPF=.FALSE.
          IF(ARR(1).GE.2.0.AND.DEBUG.EQ.0) LP=.TRUE.
          IF(ARR(1).GE.3.0.AND.DEBUG.EQ.0) LPF=.TRUE.
          IF(LP) CALL ASNLUN(IOP,'SY',0)
          IF(LP) OPEN(UNIT=IOP,NAME='LP.LST')
          IF(IO.EQ.IOP.AND..NOT.(GOULD.OR.LP)) CALL ATTACH(IO,IER)
          IF(IO.EQ.IOP.AND..NOT.LP) GOULD=.TRUE.
          IF(CMD.EQ.CMDS(10)) GO TO 5050
C
C  DEBUG OUTPUT PRINT MODE
          M60DB=1./10**6
```

```
            DO 5054 J=1,NRCS
            IF(RCS(J).GT.M60DB) GO TO 5053
      5054  CONTINUE
            WRITE(IO,5055) CMD,LINE
      5055  FORMAT('0',A4,'  HAS NO RCS CONTRIBUTION (DATABASE LINE',I3,')')
            GO TO 5056
C
      5053  CONTINUE
            WRITE(IO,56) CMD,LINE
      56    FORMAT(/'0',A4,'  COMPONENT OUTPUT FROM DATABASE LINE',I3)
            GO TO 5051
      5050 READ(IOF,2,END=9993) BUF
            LINE=LINE+1
            CALL TRIM80(NB,BUF)
            WRITE(IO,54) (BUF(I),I=1,NB)
      54   FORMAT(/'0',120A1)
            FREQ=.3/WLEN
            WRITE(IO,5057) FREQ
      5057  FORMAT(' FREQUENCY(GHZ)=',F5.1)
            IBG=IBUG.AND.1
            IF(CMD.NE.CMDS(10)) IBG=1
C           IF(ISTATS.EQ.1) IBG=0
      5051   IF(IRPR.EQ.0) WRITE(IO,51)
      51   FORMAT('0 R C S     OUTPUT'/2X,'ANGLE',5X,'RCS',5X,
          $ ' RCS-S',5X,'RCS+S'/1X)
             IF(IRPR.EQ.1) WRITE(IO,57)
      57   FORMAT('0 RPRCS    OUTPUT'/2X,'ANGLE',2X,'RCS(DBSM)',2X,
          1  'RCS(SM)')
C
C
            DO 52 J=1,NRCS
            IF(IRPR.EQ.0) GO TO 5058
            Y=RCS(J)**2+RCS2(J)**2
            YYY=10.*ALOG10(AMAX1(CUTOFF,Y))
            WRITE(IO,5059) ANG,YYY,Y
      5059  FORMAT(F7.1,2F10.4)
            GO TO 58
C
      5058  RCSC=AMAX1(CUTOFF,RCS(J))
            S=SQRT(AMAX1(0.,RCS(J)**2-RCS2(J)))
            DBSM=10.*ALOG10(AMAX1(CUTOFF,RCSC))
            DBSMMS=10.*ALOG10(AMAX1(CUTOFF,RCSC-S))
            DBSMPS=10.*ALOG10(AMAX1(CUTOFF,RCSC+S))
            WRITE(IO,53) ANG,DBSM,DBSMMS,DBSMPS
      53   FORMAT(F7.1,3F10.4,3X,2F10.4)
      58    IF(ISTATS.EQ.0) GO TO 52
            IF(STATS.GE.ANGINC.AND.AMOD(ANG,STATS).NE.0) GO TO 52
C
C           COMPUTE HIGHER ORDER STATISTICS FOR BETA DISTRIBUTION
C                   HOSBD DOES NOT USE STATS INTERNALLY WITH PRINT
C
            CALL HOSBD(O,J,IOE,IO,SLOBE)
C
      52   ANG=ANG+ANGINC
            IF(DEBUG.NE.0) GO TO 5056
C
C           COMPUTE AND PRINT LOBE STATISTICS IF REQUIRED
C
            IF(KLOBE.LE.0) GO TO 5550
            IGRPR=0
```

```
          WRITE(IOL'1) IGRPR,KLOBE,GLINT,NRCS,NLOBE,THETA,ALPHA,DELTA,
          1 BETA,SCALE,IZETA,IPCODE,WLEN,LBUG
          CLOSE(UNIT=IOL)
          CALL SPAWNS
          OPEN(UNIT=IOL,NAME='LOBE.TMP',TYPE='OLD',RECORDSIZE=181,
          2 ACCESS='DIRECT',SHARED)
C
5550      IF(GOULD) WRITE(IOP,5052)
5052      FORMAT('1')
5056      IF(GOULD.AND.DEBUG.EQ.0) CALL DETACH(IO,IER)
          IF(GOULD.AND.DEBUG.EQ.0) GOULD=.FALSE.
          IF(LP.AND..NOT.LPF) CLOSE(UNIT=IOP,DISPOSE='PRINT')
          IF(LP.AND.LPF) CLOSE(UNIT=IOP,DISPOSE='SAVE')
          IF(LP) CALL ASNLUN(IOP,'GD',0)
          IF(CMD.EQ.CMDS(10)) GO TO 3
          CMD=CLEAR
          GO TO 70
C
C       "PRYT" - MASTER ROTATION ANGLES
C
55      CONTINUE
222     FORMAT('0'/(1X,3F8.3))
C  COMPUTE YAW=>RESULT
          PRYTV(3)=ARR(3)
          C=COS(ARR(3)*RAD)
          S=SIN(ARR(3)*RAD)
          CALL MATPUT(PRYT,1.,0.,0.,0.,C,-S,0.,S,C)
C  MULTIPLY BY ROLL=>ROLL*RESULT=RESULT
          PRYTV(2)=ARR(2)
          C=COS(ARR(2)*RAD)
          S=SIN(ARR(2)*RAD)
          CALL MATPUT(MAT,C,-S,0.,S,C,0.,0.,0.,1.)
          CALL MATMUL(MAT,PRYT,PRYT)
C  MULTIPLY BY PITCH=>PITCH*RESULT=RESULT
          PRYTV(1)=ARR(1)
          C=COS(ARR(1)*RAD)
          S=SIN(ARR(1)*RAD)
          CALL MATPUT(MAT,C,0.,S,0.,1.,0.,-S,0.,C)
          CALL MATMUL(MAT,PRYT,PRYT)
C  MULTIPLY BY TILT=>RESULT*TILT=RESULT
          PRYTV(4)=ARR(4)
          C=COS(ARR(4)*RAD)
          S=SIN(ARR(4)*RAD)
          CALL MATPUT(MAT,C,0.,-S,0.,1.,0.,S,0.,C)
          CALL MATMUL(PRYT,MAT,PRYT)
C  MULTIPLY BY B=>B*RESULT=RESULT=>PRYT
          CALL MATPUT(MAT,0.,-1.,0.,1.,0.,0.,0.,0.,1.)
          CALL MATMUL(MAT,PRYT,PRYT)
          GO TO 3
C
C       "PAUSE" - WAIT FOR A CARRIAGE RETURN
C
60      CONTINUE
          READ(IOR,61,END=9990) J
61      FORMAT(A1)
          GO TO 3
C
C         "RAMDB" - MODIFY COMPONENT RCS
C
65        RAM=10.**(ARR(1)/10.)
```

```
          GO TO 3
C
C      "CLEAR" - CLEAR RCS VECTOR
C
70     CONTINUE
          IPASS=0
          OBJECT=.FALSE.
          NVALID=0
          NTOTAL=0
          DO 71 J=1,MAXRCS
          RCS2(J)=0.
          RCS(J)=0.
          YYV(J,1)=0.
          YYV(J,2)=0.
          XV(J)=0.
C         EXTRA(J)=0.
          IF(OPENE) WRITE(IOE'J)DZERO,DZERO,DZERO,DZERO,DZERO,DZERO
71        CONTINUE
          IF(CMD.EQ.CLEAR) GO TO 3
C
C      RESET SYSTEM VECTORS
C
          POLAR=1.
C
C   SET PYRT TO IDENTITY BY DEFAULT
C
          DO 74 J=1,9
74        PRYT(J)=IDENT(J)
C
C         RESET LOBE PARAMETERS
C
          KLOBE=0
          LLOBE=0
          MLOBE=0
          NLOBE=-3
          BETA=0.
C
C         RESET RPRCS PARAMETERS
C
          IRPR=0
C
C      RESET VALID VECTOR AFTER EACH OBJECT
C
72     VALID(1)=-360.
          VALID(2)=360.
          DO 73 J=3,6
73     VALID(J)=0.
          RAM=1.
          DO 75 J=1,9
75        TRANS(J)=IDENT(J)
          DO 76 J=1,3
76        ORGN(J)=0.
          IF(IPASS.EQ.1) GO TO 99
          GO TO 3
C
C         DEBUG OUTPUT
C
80        IBUG=IFIX(ARR(1))
          LBUG=IFIX(ARR(2))
          GO TO 3
```

```
C
C        NEW WAVE LENGTH
C
85       IWL=IWL+1
         IF(IWL.EQ.10) GO TO 9990
         OLDGRF=.FALSE.
86       REWIND IOF
         IF(KLOBE.GT.0) CLOSE(UNIT=IOL)
         IF(GOULD) CALL DETACH(IOP,IER)
         GOULD=.FALSE.
         LINE=0
         GO TO 70
C
C*******************************************
C        THE FOLLOWING COMMANDS ARE NEW
C        ADDITIONS TO RCS FOR THE LOBE
C                STATISTICS
C*******************************************
C
C        "LOBE" - SET LOBE SWITCH
C
500      CONTINUE
C        IF(IRPR.EQ.1.AND.ARR(1).EQ.0.) GO TO 3
C        IF(ISTATS.EQ.1.AND.ARR(1).GT.0.) GO TO 573
         LLOBE=ARR(1)
         IF(LLOBE.GT.0.AND.KLOBE.LE.0) CALL NOLOCK(IOL)
         IF(LLOBE.GT.0.AND.KLOBE.LE.0)
         1 OPEN(UNIT=IOL,NAME='LOBE.TMP',TYPE='UNKNOWN',RECORDSIZE=181,
         2 ACCESS='DIRECT',SHARED)
         IF(LLOBE.GT.0.AND.KLOBE.LE.0) WRITE(IOL'1) DUMMY
         IF(LLOBE.GT.0) MLOBE=LLOBE
         IF(LLOBE.GT.0) KLOBE=LLOBE
         GO TO 3
C
C501      WRITE(2,502)
C502      FORMAT(' RCS> -- RPRCS AND LOBE CANNOT BOTH BE ACTIVE')
C        GO TO 9993
C
C        "BETA" - DEFINE BISTATIC ANGLE
C
510      BETA=ARR(1)
         IF(BETA.LT.5.) BETA=0.
         GO TO 3
C
C        "ORGN" - COMPONENT ORIGIN VECTOR
C
520      ORGN(1)=ARR(1)
         ORGN(2)=ARR(2)
         ORGN(3)=ARR(3)
         GO TO 3
C
C        "DELT" - FIRST LOBE ANGULAR RANGE
C
540      DO 541 J=1,4
541      DELTA(J)=ARR(J)
         DELTA(2)=AMAX1(DELTA(1),DELTA(2))
         DELTA(3)=AMAX1(1.,DELTA(3))
         IF(DELTA(4).LT.4.0.OR.DELTA(4).GT.180.) DELTA(4)=48.
         GO TO 3
C
```

```
C       "ALPH" - SECOND LOBE ANGULAR RANGE
C
550     DO 551 J=1,4
551     ALPHA(J)=ARR(J)
        GO TO 3
C
C**********************************
C
C       END OF LOBE COMMANDS
C
C**********************************
C
C       "RPRCS" - RELATIVE PHASE RCS
C
560     CONTINUE
C       IF(MLOBE.GT.0.AND.ARR(1).GT.0.)GO TO 501
C       IF(ISTATS.EQ.1.AND.ARR(1).GT.0.) GO TO 571
        IRPR=MIN0(1,MAX1(0.,ARR(1)))
        IF(IRPR.EQ.0) GO TO 3
        RPRCSG=ARR(2)
        LRPRCS=ARR(3)
        IF(LRPRCS.LE.1) LRPRCS=5
        LOBE=1
        LLOBE=1
        MLOBE=1
        GO TO 3
C
C       "STATS" - HIGHERER ORDER STATISTICS
C
570     CONTINUE
C       IF(IRPR.EQ.1.AND.ARR(1).GT.0.) GO TO 571
C       IF(MLOBE.GT.0.AND.ARR(1).GT.0.)GO TO 573
        ISTATS=MIN0(1,MAX1(0.,ARR(1)))
        STATS=ARR(2)
        IF(STATS.LE.0.) STATS=10.
        IF(ISTATS.EQ.0.OR.OPENE) GO TO 3
        OPEN(UNIT=IOE,NAME='RCSTMP.DES',TYPE='SCRATCH',
        1       ACCESS='DIRECT',RECORDSIZE=16,MAXREC=181)
        OPENE=.TRUE.
C
        DO 575 J=1,181
575     WRITE(IOE'J)DZERO,DZERO,DZERO,DZERO,DZERO,DZERO
        GO TO 3
C
C571     WRITE(2,572)
C572     FORMAT(' RCS> -- RPRCS AND STATS CANNOT BOTH BE ACTIVE')
C       GO TO 9993
CC
C573     WRITE(2,574)
C574     FORMAT(' RCS> -- LOBE AND STATS CANNOT BOTH BE ACTIVE')
C       GO TO 9993
C
C       "GLINT" - SET GLINT SWITCH
C
580     GLINT=MIN0(2,MAX1(0.,ARR(1)))
        GO TO 3
C
C       "NEWTH" - REPEAT LOOP FOR NEW THETA ANGLE RANGE
C
590     CALL CNEWTH(KLOBE,ROUTE)
```

```
              GO TO (3,86,9993),ROUTE
C
C             "GSIZE" - SET GRAPH SIZE
C
600           DO 601 J=1,5
601           GSIZE(J)=ARR(J)
              CALL GRPIC(GSIZE(1),GSIZE(2),GSIZE(3),GSIZE(4))
              GO TO 3
C
C           DETERMINE IF COMMAND IS AN OBJECT
C
90          CONTINUE
              DO 91 J=1,NOBJ
              IOBJ=J
              IF(CMD.EQ.OBJS(J)) GO TO 100
91          CONTINUE
C
              WRITE(IOW,92) CMD
92          FORMAT(' RCS> -- INVALID FUNCTION "',A4,'"')
              GO TO 9993
C
C
C
C           *************************************
C           ADD OBJECT CONTRIBUTION TO RCS VECTOR
C           *************************************
C
C
100            IF(WLEN.LE.0.) GO TO 9991
C  NOTE:   IPASS IS ONLY USED WITH RFLAT IN CONJUCTION WITH LOBE
               IPASS=-1
99             IPASS=IPASS+1
              OBJECT=.TRUE.
              ANG=THETA(1)
              ANGMAX=THETA(2)+.001
              ANGINC=THETA(3)
              CPHR=COS(PHR*RAD)
              SPHR=SIN(PHR*RAD)
              IRCS=1
              CALL TRANSP(PRYT,PRYT2)
              CALL MATMUL(PRYT2,TRANS,COMP)
C
              IF((IBUG.AND.2).EQ.0) GO TO 101
              WRITE(2,222) TRANS
              WRITE(2,222) PRYT
              WRITE(2,222) COMP
C
C             ANG IS RADAR THETA ANGLE
C             PHR IS RADAR PHI ANGLE
C
C             IF END OF ANGLE RANGE FOR OBJECT, OUTPUT PARTIAL
C             LOBE DATA IF REQUIRED, THEN GET NEXT COMMAND.
C
C             KLOBE=>SET FROM LOBE ENTRY POINT
C             MLOBE=>LOBE IS SET FOR RUN
C             LLOBE=>LOBE IS SET FOR OBJECT
C              LOBE=>LOBE IS SET FOR CURRENT THETA
C
101         IF(IRCS.LE.NRCS) GO TO 131
               IF(IRPR.EQ.1.AND.KLOBE.EQ.0) GO TO 135
```

```
            IF(LLOBE.LE.0) GO TO 72
C  OUTPUT LOBE INFO TO TEMPORARY FILE
            NLOBE=NLOBE+4
            DO 139 J=1,4
139         WRITE(IOL'NLOBE+J) (SLOBE(J,I,1),I=1,NRCS)
            DO 133 I=1,NRCS
            IF(SLOBE(1,I,2).NE.0.) GO TO 134
133         CONTINUE
            GO TO 135
134         NLOBE=NLOBE+4
            DO 137 J=1,4
137         WRITE(IOL'NLOBE+J) (SLOBE(J,I,2),I=1,NRCS)
C
C  MAKE SECOND PASS THROUGH RFLAT FOR LOBE
135         IF(CMD.EQ.OBJS(13).AND.IPASS.EQ.0) IPASS=1
            GO TO 72
C
C           DETERMINE IF VALID ANGLE FOR LOBE CONTRIBUTION
C
131         LOBE=LLOBE
            IF(LOBE.LE.0) GO TO 132
            SLOBE(1,IRCS,1)=0.
            SLOBE(1,IRCS,2)=0.
C
C        SKIP ANGLE IF NOT VALID FOR CURRENT OBJECT
C
132          DO 102 J=1,5,4
            IF(VALID(J)-VALID(J+1)) 103,102,104
103       IF(ANG.LT.VALID(J).OR.ANG.GT.VALID(J+1)) GO TO 102
            IF(VALID(J+2)-VALID(J+3)) 105,109,106
105       IF(PHR.LT.VALID(J+2).OR.PHR.GT.VALID(J+3)) GO TO 102
            GO TO 109
106       IF(PHR.GT.VALID(J+3).AND.PHR.LT.VALID(J+2)) GO TO 102
            GO TO 109
104       IF(ANG.GT.VALID(J+1).AND.ANG.LT.VALID(J)) GO TO 102
            IF(VALID(J+2)-VALID(J+3)) 107,109,108
107       IF(PHR.LT.VALID(J+2).OR.PHR.GT.VALID(J+3)) GO TO 102
            GO TO 109
108       IF(PHR.GT.VALID(J+3).AND.PHR.LT.VALID(J+2)) GO TO 102
            GO TO 109
102       CONTINUE
C
C        CURRENT ANGLE IS NOT VALID.  INCREMENT ANGLE.
C
             GO TO 130
C
C   CURRENT ANGLE IS VALID.  FIND PRIMED ANGLES FROM RADAR COORDINATES.
C  NOTE!  THIS PROGRAM STORES MATRICES BY ROWS (NOT COLUMNS)
C
109       CONTINUE
            CANG=COS(ANG*RAD)
            SANG=SIN(ANG*RAD)
            CTHP=COMP(3)*SANG*CPHR+COMP(6)*SANG*SPHR+COMP(9)*CANG
            THP=ACOSK(CTHP)
            STHP=SIN(THP)
            CPHPS=(COMP(1)*SANG*CPHR+COMP(4)*SANG*SPHR+COMP(7)*CANG)
            SPHPS=(COMP(2)*SANG*CPHR+COMP(5)*SANG*SPHR+COMP(8)*CANG)
            PHP=0.
            IF(ABS(SPHPS).LT.1E-5.AND.ABS(CPHPS).LT.1.E-5) GO TO 1111
            PHP=ATAN2(SPHPS,CPHPS)
```

```
 1111  CPHP=COS(PHP)
         C2PHP=CPHP*CPHP
         SPHP=SIN(PHP)
         S2PHP=SPHP*SPHP
  223  FORMAT(4F8.3)
C
C
C
C
C      *****************************************
C      ADD OBJECT CONTRIBUTION FOR CURRENT ANGLE
C      *****************************************
C
C
         RCSC=0.
         IF(IOBJ.LE.6.OR.IOBJ.GE.10) GO TO 110
C
C   FIND UNPRIMED ANGLES FOR OBJECTS WHICH USE POLARIZATION
C
         CTH=PRYT2(3)*SANG*CPHR+PRYT2(6)*SANG*SPHR+PRYT2(9)*CANG
         TH=ACOSK(CTH)
         STH=SIN(TH)
         CPHS=(PRYT2(1)*SANG*CPHR+PRYT2(4)*SANG*SPHR+PRYT2(7)*CANG)
         SPHS=(PRYT2(2)*SANG*CPHR+PRYT2(5)*SANG*SPHR+PRYT2(8)*CANG)
         PH=0.
         IF(ABS(SPHS).LT.1.E-5.AND.ABS(CPHS).LT.1.E-5) GO TO 1112
         PH=ATAN2(SPHS,CPHS)
 1112  CPH=COS(PH)
         SPH=SIN(PH)
         IF((IBUG.AND.4).EQ.0) GO TO 110
         DTH=TH/RAD
         DPH=PH/RAD
         WRITE(2,223) DTH,DPH
C
C      GO TO SPECIFIED OBJECT
C
 110   GO TO (200,210,220,230,240,250,260,270,280,290,300,310,320,330,
     $ 340,350,360,370), IOBJ
C
C
C   "OGIVE" - OGIVE & TRUNCATED OGIVE
 200   CALL OGIVE
         GO TO 120
C
C   "CYLIN" - ELLIPTICAL CYLINDER
C              INPUTS=A,B,L
 210   CALL CYLIN
         GO TO 120
C
C   "CONE" - TRUNCATED ELLIPTICAL CONE & CIRCULAR CONE & STANDARD CONE
 220   CALL CONE
         GO TO 120
C
C   "PARAB" - PARABOLA
 230   CALL PARAB
         GO TO 120
C
C   "SOLID" - ELLIPSIOD & SPHEROID
C              INPUTS=A,B,C
 240   CALL SOLID
```

```
            GO TO 120
C
C   "TORUS" - TORUS
250     CALL TORUS
            GO TO 120
C
C   "WEDGE" - WEDGE
260     CALL WEDGE(TRANS,CTH,STH,CPH,SPH)
            GO TO 120
C
C   "WIRE" - WIRE
270     CALL WIRE(TRANS,CTH,STH,CPH,SPH)
            GO TO 120
C
C   "LWIRE" - LOOP WIRE
280     CALL LWIRE(TRANS,CTH,STH,CPH,SPH)
            GO TO 120
C
C   "CAVA" - CAVITY AREA
290     CALL CAVA
            GO TO 120
C
C   "CAVD" - CAVITY DIAMETER
300     CALL CAVD
            GO TO 120
C
C   "CFLAT" - CIRCULAR FLAT PLATE
310     CALL CFLAT
            GO TO 120
C
C   "RFLAT" - RECTANGULAR FLAT PLATE
320     CALL RFLAT(IPASS)
            IF(IPASS.EQ.2) RCSC=0.
            GO TO 120
C
C   "TWAVE" - TRAVELING WAVE
330     CALL TWAVE
            GO TO 120
C
C   "DIHED" - DIHEDRAL
340     CALL DIHED
            GO TO 120
C
C   "TCORN" - GENERAL TRIANGULAR CORNER
350     CALL TCORN
            GO TO 120
C
C   "RCORN" - GENERAL RECTANGULAR CORNER
360     CALL RCORN
            GO TO 120
C
C   "ECORN" - GENERAL ELLIPTICAL CORNER
370     CALL ECORN
            GO TO 120
C
C
C       ADD RCS CONTRIBUTIONS TO VECTORS
C
120     CONTINUE
            IF(RCSC.LT.0.) WRITE(IOW,121) CMD,ANG,RCSC
```

```
 121      FORMAT(' RCS> -- ',A4,F7.1,E15.6)
          IF(RCSC.GT.CUTOFF) NVALID=NVALID+1
          RCSC=RCSC*RAM
          IF(IRPR.EQ.1) GO TO 136
          RCS(IRCS)=RCS(IRCS)+RCSC
          RCS2(IRCS)=RCS2(IRCS)+RCSC*RCSC
 136      IF(ISTATS.EQ.0) GO TO 130
 C
          READ(IOE'IRCS) T1,T2,T3,T4,T5,T6
          T1=T1+DBLE(RCSC)
          T2=T2+DBLE(RCSC)**2
          T3=T3+DBLE(RCSC)**3
          T4=T4+DBLE(RCSC)**4
          SQ=SQRT(RCSC)
          T5=T5+SQ
          T6=DMAX1(T6,SQ)
          WRITE(IOE'IRCS) T1,T2,T3,T4,T5,T6
 C
 C      INCREMENT ANGLE
 C
 130    CONTINUE
          ANG=ANG+ANGINC
          IRCS=IRCS+1
          IF(IPASS.NE.2) NTOTAL=NTOTAL+1
          GO TO 101
 C
 C
 9991     WRITE(IOW,9992)
 9992     FORMAT(' RCS> -- WAVELENGTH UNDEFINED')
 C
 9993     WRITE(IOW,9994) LINE
 9994     FORMAT(' RCS> -- LAST DATA BASE LINE READ IS',I4)
 C
 9990     CONTINUE
 C
 C      TERMINATE RCS PROGRAM
 C
          IF(KLOBE.GT.0) CLOSE(UNIT=IOL,DISPOSE='DELETE')
          IF(OPENE) CLOSE(UNIT=IOE)
          IF(GOULD) CALL DETACH(IOP,IER)
          TIME=SECNDS(TIME)/60.
          IF(IZETA.EQ.0) CALL GRMODE('ALPHA',3)
          WRITE(2,9998) TIME
 9998     FORMAT(' RCS> -- ELAPSED TIME =',F5.1,' MINUTES')
 C
 9999     CALL EXIT
          END
```

```
>>>  FILE = SY0:[111,12]RCSCMD.FTN             27-SEP-83  14:34:38  <<<
      C
            SUBROUTINE CNEWTH(KLOBE,ROUTE)
      C
            INTEGER*2 ROUTE
      C
            INTEGER*2 DEBUG,COMBIN,GLINT
            LOGICAL*2 GOULD,LP,OPENE,LPF,OLDGRF,MGRAPH,GDAUTO
            COMMON/SWITCH/ IOW,THETA(4),GOULD,LP,OPENE,LPF,OLDGRF,MGRAPH,
            1 GDAUTO,DEBUG,COMBIN,GLINT,IBUG,LBUG,ISTATS,IZETA
      C
            ROUTE=1
            IF(THETA(4).GT.THETA(2)) GO TO 1
            IF(.NOT.MGRAPH) RETURN
            IF(IZETA.EQ.0.AND..NOT.GDAUTO) RETURN
            IF(IZETA.EQ.0) CALL GRTEKG
            IF(IZETA.NE.0) CALL GRPLOT(11.,8.5,-1)
            RETURN
      C
      1     IF(KLOBE.EQ.1.OR.GLINT.EQ.1.OR.ISTATS.EQ.1.OR.DEBUG.NE.0) GO TO 591
            OLDGRF=.TRUE.
            ROUTE=2
            RETURN
      C
      591   WRITE(IOW,592)
      592   FORMAT(' RCS> -- MULTIPLE GRAPH SEGMENTS ARE NOT ALLOWED WITH:'/
            1      '                 LOBE, STATS, OR DEBUG!')
            ROUTE=3
            RETURN
            END
```

```
>>>  FILE = SY0:[111,12]RCSOBJ.FTN             27-SEP-83   14:34:44  <<<
C
C  <*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*>
      SUBROUTINE OGIVE
C
C  WITH LOBE
C
      IMPLICIT REAL*4 (A-Z)
        INTEGER*2 LOBE
      COMMON/RCSCOM/CUTOFF,WLEN,PI,PI2,RCSC,THP,CTHP,STHP,PHP,CPHP,SPHP,
     $ C2PHP,S2PHP,RAD,POLAR,LOBE,
     $ R1,A,B,BETA
C
      THPT=THP
      IF(THP.GT.PI2) THPT=PI-THP
        IF(LOBE.GT.0) L2=SQRT(R1*R1-(B+R1-A)**2)
      IF(B.GT.0.) GO TO 5
3     ALP=ACOSK(1.-A/R1)
      RCSC=R1*R1*SIN(ALP)**2/(4.*PI)
      NMA=PI2-ALP
      IF(THPT.LT.NMA) GO TO 2
C
C  90-ALP < THETA' < 90
      SIGMA=PI*R1*R1*(1.-(R1-A)/(R1*STHP))
        IF(LOBE.LE.0) GO TO 4
        IF(RCSC.GT.SIGMA)  GO TO 6
        CALL TOLOBE(1,SIGMA,A*CPHP,A*SPHP,0.)
4       RCSC=SIGMA
      RETURN
C
C  THETA' = 90-ALP
6       CALP2=(1.-A/R1)**2
        SALP=SIN(ALP)
        ZALP=R1*SALP*(CALP2-SQRT(CALP2*CALP2-SALP*SALP))
        WALP=ZALP*TAN(NMA)
        CALL TOLOBE(1,RCSC,WALP*CPHP,WALP*SPHP,ZALP)
        RETURN
C
C  THETA' < 90-ALP
2     Z=16.*PI*COS(THPT)**6*(1.-TAN(ALP)**2*TAN(THPT)**2)**3
        IF(LOBE.GT.0.AND.Z.LT.CUTOFF) GO TO 6
      IF(Z.LT.CUTOFF) RETURN
      SIGMA=WLEN*WLEN*TAN(ALP)**4/Z
        IF(LOBE.LE.0) GO TO 1
        IF(RCSC.LT.SIGMA) GO TO 6
        CALL TOLOBE(1,SIGMA,0.,0.,L2)
1     RCSC=AMIN1(RCSC,SIGMA)
      RETURN
C
C   TRUNCATED OGIVE  (BETA IS ALWAYS < ALPHA)
C
5     BET=BETA*RAD
      IF(THPT.GT.PI2-BET) GO TO 3
C
C  THETA' = 0
      RCSC=PI*B*B*TAN(BET)**2
        IF(STHP.LT.CUTOFF.AND.LOBE.GT.0) CALL TOLOBE(1,RCSC,0.,0.,L2)
      IF(STHP.LT.CUTOFF) RETURN
C
C  BETA < THETA' < 90-BETA
```

```
      SIGMA=WLEN*B*TAN(THPT+BET)**2/(8.*PI*STHP)
        IF(LOBE.GT.0) CALL TOLOBE(1,SIGMA,B*CPHP,B*SPHP,L2)
      IF(BET.LE.THPT.AND.THPT.LE.PI2-BET) GO TO 1
C
C  THETA' < BETA OR THETA' > 90-BETA
      SIG=WLEN*B*TAN(THPT-BET)**2/(8.*PI*STHP)
        IF(LOBE.GT.0) CALL TOLOBE(2,SIG,B*CPHP,B*SPHP,-L2)
        SIGMA=SIGMA+SIG
      GO TO 1
      END
C
C
C  <*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*>
      SUBROUTINE CYLIN
C
C  WITH LOBE
C
      IMPLICIT REAL*4 (A-Z)
        INTEGER*2 LOBE
      COMMON/RCSCOM/CUTOFF,WLEN,PI,PI2,RCSC,THP,CTHP,STHP,PHP,CPHP,SPHP,
     $ C2PHP,S2PHP,RAD,POLAR,LOBE,
     $ A,B,L
C
      IF(STHP.LE.0.) RETURN
      A2=A*A
      B2=B*B
      Z=(A2*CPHP*CPHP+B2*SPHP*SPHP)**1.5
      RCSC=2.*PI*L*L*A2*B2/(WLEN*Z)
C
      D=8.*PI*CTHP*CTHP*Z
      IF(ABS(D).LT.CUTOFF) GO TO 1
C
C  THETA' # 90
      SIGMA=2.*WLEN*A2*B2*STHP/D
      RCSC=AMIN1(RCSC,SIGMA)
        IF(LOBE.LE.0) RETURN
        CALL TOLOBE(1,RCSC/2.,A*CPHP,B*SPHP,0.)
        CALL TOLOBE(2,RCSC/2.,A*CPHP,B*SPHP,L)
      RETURN
C
C  THETA' = 90
1       IF(LOBE.GT.0) CALL TOLOBE(1,RCSC,A*CPHP,B*SPHP,L/2.)
        RETURN
      END
C
C
C  <*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*>
      SUBROUTINE CONE
C
C  WITH LOBE
C
      IMPLICIT REAL*4 (A-Z)
        INTEGER*2 LOBE
      COMMON/RCSCOM/CUTOFF,WLEN,PI,PI2,RCSC,THP,CTHP,STHP,PHP,CPHP,SPHP,
     $ C2PHP,S2PHP,RAD,POLAR,LOBE,
     $ ALP,NU,A,L1,L2
C
      IF(CTHP.GE.0..AND.SIN(ALP*RAD).GE.STHP) RETURN
      NU2=NU*NU
      TALP=TAN(ALP*RAD)
```

```
      D=SQRT(S2PHP+NU2*C2PHP)
      POLANG=ATAN(-NU/(TALP*D))
      RCSC=8.*PI*(L2**1.5-L1**1.5)**2*TALP**4
      RCSC=RCSC/(9.*WLEN*NU2*ABS(COS(POLANG)**3))
      Z=(STHP*TALP*D+NU*CTHP)**2
      Z=Z*8.*PI*STHP*D
      IF(Z.GT.CUTOFF) GO TO 1
        IF(LOBE.LE.0) RETURN
        L=(L1+L2)/2.
        CALL TOLOBE(1,RCS,L*TALP*CPHP,L*NU*TALP*SPHP,L)
        RETURN
C
C  OFF - NORMAL INCIDENCE
1      SIGMA=((STHP/D)-(CTHP*TALP/NU))**2
      SIGMA=SIGMA*WLEN*NU2*NU*TALP/Z
      RCSC=AMIN1(RCSC,SIGMA*(L1+L2))
        IF(LOBE.LE.0) RETURN
        CALL TOLOBE(1,AMIN1(RCSC/2.,L1*SIGMA),L1*TALP*CPHP,
        1   L1*NU*TALP*SPHP,L1)
        CALL TOLOBE(2,AMIN1(RCSC/2.,L2*SIGMA),L2*TALP*CPHP,
        1   L2*NU*TALP*SPHP,L2)
      RETURN
      END
C
C
C  <*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*>
      SUBROUTINE PARAB
C
C  WITH LOBE
C
      IMPLICIT REAL*4 (A-Z)
        INTEGER*2 LOBE
      COMMON/RCSCOM/CUTOFF,WLEN,PI,PI2,RCSC,THP,CTHP,STHP,PHP,CPHP,SPHP,
     $ C2PHP,S2PHP,RAD,POLAR,LOBE,
     $ P
C
C     RCS IS UNDEFINED AS THP APPROACHES PI
        IF(ABS(THP).GT.70.*RAD) RETURN
      RCSC=4.*PI*P*P/CTHP**4
        IF(LOBE.LE.0) RETURN
        TTHP=TAN(THP)
        CALL TOLOBE(1,RCSC,2.*P*TTHP*CPHP,2.*P*TTHP*SPHP,-P*TTHP*TTHP)
      RETURN
      END
C
C
C  <*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*>
      SUBROUTINE SOLID
C
C  WITH LOBE
C
      IMPLICIT REAL*4 (A-Z)
        INTEGER*2 LOBE
      COMMON/RCSCOM/CUTOFF,WLEN,PI,PI2,RCSC,THP,CTHP,STHP,PHP,CPHP,SPHP,
     $ C2PHP,S2PHP,RAD,POLAR,LOBE,
     $ A,B,C
C
      R=A*B*C
      D=(A*STHP*CPHP)**2
      D=D+(B*STHP*SPHP)**2
```

```
      D=D+(C*CTHP)**2
      RCSC=PI*R*R/(D*D)
        IF(LOBE.GT.0) CALL TOLOBE(1,RCSC,A*STHP*CPHP,B*STHP*SPHP,
        1 C*CTHP)
      RETURN
      END
C
C
C <*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*>
      SUBROUTINE TORUS
C
C WITH LOBE
C
      IMPLICIT REAL*4 (A-Z)
        INTEGER*2 LOBE
      COMMON/RCSCOM/CUTOFF,WLEN,PI,PI2,RCSC,THP,CTHP,STHP,PHP,CPHP,SPHP,
     $ C2PHP,S2PHP,RAD,POLAR,LOBE,
     $ A,B
C
      RCSC=8.*PI**3*B*A*A/WLEN
        IF(STHP.LT.CUTOFF.AND.LOBE.GT.0) CALL TOLOBE(1,RCSC,0.,0.,0.)
      IF(STHP.LT.CUTOFF) RETURN
      SIG1=PI*(B*A/STHP+B*B)
        SIG2=0.
        IF(LOBE.GT.0) CALL TOLOBE(1,AMIN1(RCSC,SIG1),A*CPHP,B*SPHP,0.)
      IF(ABS(CTHP).LE.B/(2.*A)) GO TO 1
      SIG2=PI*(B*A/STHP-B*B)
        IF(LOBE.GT.0) CALL TOLOBE(2,AMIN1(RCSC,SIG2),-A*CPHP,-B*SPHP,0.)
    1 RCSC=AMIN1(RCSC,SIG1+SIG2)
      RETURN
      END
C
C
C <*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*>
      SUBROUTINE WEDGE(COOR,CTH,STH,CPH,SPH)
C
C WITH LOBE
C
      IMPLICIT REAL*4 (A-Z)
        INTEGER*2 LOBE
      COMMON/RCSCOM/CUTOFF,WLEN,PI,PI2,RCSC,THP,CTHP,STHP,PHP,CPHP,SPHP,
     $ C2PHP,S2PHP,RAD,POLAR,LOBE,
     $ ANGLE,L,B
      REAL COOR(9)
C
        WEDMAX=B/(WLEN*WLEN)
      ANG=ANGLE*RAD
      M=(PI*L*L/(4.*ANG*ANG))*SIN(PI*PI/(2.*ANG))**2
      T=COS(PI*PI/(2.*ANG))
      C=COS(PI*PHP/ANG)+T
      D=1.-T
      IF(ABS(C).LE.CUTOFF.OR.ABS(D).LE.CUTOFF) RETURN
C
      IF(POLAR.EQ.2.) GO TO 2
      RCSC=2.*(-COOR(3)*SPH+COOR(6)*CPH)**2-1.
      RCSC=M*(RCSC/C+1./D)**2
      GO TO 4
C
    2 RCSC=2.*(COOR(3)*CTH*CPH+COOR(6)*CTH*SPH-COOR(9)*
     $ STH)**2-1.
```

```
      RCSC=M*(RCSC/C+1./D)**2
 4    DU=ABS(PI2-ACOSK(-COOR(3)*STH*CPH-COOR(6)*STH*
     $ SPH-COOR(9)*CTH))
        RCSC=AMIN1(RCSC,WEDMAX)
      IF(DU*DU.LT.CUTOFF) GO TO 6
      SIGMA=RCSC*(WLEN/(PI*L*DU))**2/8.
      RCSC=AMIN1(RCSC,SIGMA)
        IF(LOBE.LE.0) RETURN
        CALL TOLOBE(1,RCSC/2.,0.,0.,0.)
        CALL TOLOBE(2,RCSC/2.,0.,0.,L)
      RETURN
C
 6      IF(LOBE.LE.0) RETURN
        CALL TOLOBE(1,RCSC,0.,0.,L/2.)
        RETURN
      END
C
C
C   <*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*>
      SUBROUTINE WIRE(COOR,CTH,STH,CPH,SPH)
C
C   WITH LOBE
C
      IMPLICIT REAL*4 (A-Z)
        INTEGER*2 LOBE
      COMMON/RCSCOM/CUTOFF,WLEN,PI,PI2,RCSC,THP,CTHP,STHP,PHP,CPHP,SPHP,
     $ C2PHP,S2PHP,RAD,POLAR,LOBE,
     $ L,A
      REAL COOR(9),P(3),ZK(3)
C
      IF(STHP.LE.0.) RETURN
      GAMMA=1.78
      ZK(1)=COOR(8)*(-CTH)-COOR(9)*(-STH*SPH)
      ZK(2)=-(COOR(7)*(-CTH)-COOR(9)*(-STH*CPH))
      ZK(3)=COOR(7)*(-STH*SPH)-COOR(8)*(-STH*CPH)
      IF(POLAR.EQ.2.) GO TO 2
      P(1)=-SPH
      P(2)=CPH
      P(3)=0.
      GO TO 3
 2    P(1)=CTH*CPH
      P(2)=CTH*SPH
      P(3)=-STH
C
 3    SPS=ZK(1)*P(1)+ZK(2)*P(2)+ZK(3)*P(3)
      SPS=SPS/SQRT(ZK(1)*ZK(1)+ZK(2)*ZK(2)+ZK(3)*ZK(3))
      PS=ASIN(AMAX1(-1.,AMIN1(1.,SPS)))
      RCSC=PI*L*L*COS(PS)**4/(PI2*PI2+ALOG(WLEN/(GAMMA*PI*A))**2)
        IF(STHP.LT.CUTOFF.AND.LOBE.GT.0) CALL TOLOBE(1,RCSC,0.,0.,0.)
      IF(STHP.LT.CUTOFF) RETURN
      SIGMA=(WLEN*TAN(THP)*COS(PS)**2)**2
      SIGMA=SIGMA/(16.*PI*(PI2*PI2+ALOG(WLEN/(GAMMA*PI*A*STHP))**2))
      RCSC=AMIN1(RCSC,2.*SIGMA)
        IF(LOBE.LE.0) RETURN
        CALL TOLOBE(1,RCSC/2.,0.,0.,L/2.)
        CALL TOLOBE(2,RCSC/2.,0.,0.,-L/2.)
      RETURN
      END
C
C
```

```
C   <*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*>
      SUBROUTINE LWIRE(COOR,CTH,STH,CPH,SPH)
C
C   WITH LOBE
C
      IMPLICIT REAL*4 (A-Z)
        INTEGER*2 LOBE
      COMMON/RCSCOM/CUTOFF,WLEN,PI,PI2,RCSC,THP,CTHP,STHP,PHP,CPHP,SPHP,
     $ C2PHP,S2PHP,RAD,POLAR,LOBE,
     $ A
      REAL COOR(9)
C
      IF(POLAR.EQ.2.) GO TO 2
      GAMMA=ACOSK(-COOR(2)*SPH+COOR(5)*CPH)
      GO TO 3
C
2     GAMMA=ACOSK(COOR(2)*CTH*CPH+COOR(5)*CTH*SPH-COOR(8)*STH)
C
3     KA2=(4.*PI*A/WLEN)*STHP
      CG2=COS(GAMMA)**2
      SG2C=SIN(GAMMA)**2*CTHP*CTHP
      CALL BESJ(KA2,0,J0,.1,IER)
      IF(IER.NE.0) GO TO 9
      CALL BESJ(KA2,2,J2,.1,IER)
      IF(IER.NE.0) GO TO 9
      RCSC=(SG2C+CG2)*J0+(SG2C-CG2)*J2
      RCSC=PI*A*A*RCSC*RCSC
        IF(LOBE.LE.0) RETURN
        IF(STHP.LE.CUTOFF) CALL TOLOBE(1,RCSC,0.,0.,0.)
        IF(STHP.GT.CUTOFF) CALL TOLOBE(1,RCSC,A*CPHP,A*SPHP,0.)
      RETURN
9     RCS=0.
      RETURN
      END
C
C
C   <*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*>
      SUBROUTINE CAVA
C
C   WITH LOBE
C
      IMPLICIT REAL*4 (A-Z)
        INTEGER*2 LOBE
      COMMON/RCSCOM/CUTOFF,WLEN,PI,PI2,RCSC,THP,CTHP,STHP,PHP,CPHP,SPHP,
     $ C2PHP,S2PHP,RAD,POLAR,LOBE,
     $ A
C
      IF(ABS(THP).GT.70.*RAD) RETURN
      T=(4.*PI*A/(WLEN*WLEN))**1.5
      RCSC=.4*T*WLEN*WLEN
      TTHP=TAN(THP)
        IF(TTHP.LT.CUTOFF.AND.LOBE.GT.0) CALL TOLOBE(1,RCSC,0.,0.,0.)
      IF(TTHP.LT.CUTOFF) RETURN
      SIGMA=.8*PI*A/TTHP
      RCSC=AMIN1(RCSC,SIGMA)
        IF(LOBE.LE.0) RETURN
        D=SQRT(4.*A/PI)
        DS2=D*STHP/2.
        CALL TOLOBE(1,RCSC,DS2*STHP*CPHP,DS2*STHP*SPHP,DS2*CTHP)
      RETURN
```

```
      END
C
C
C   <*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*>
      SUBROUTINE CAVD
C
C   WITH LOBE
C
      IMPLICIT REAL*4 (A-Z)
        INTEGER*2 LOBE
      COMMON/RCSCOM/CUTOFF,WLEN,PI,PI2,RCSC,THP,CTHP,STHP,PHP,CPHP,SPHP,
     $ C2PHP,S2PHP,RAD,POLAR,LOBE,
     $ D
C
      IF(ABS(THP).GT.70.*RAD) RETURN
      RCSC=.05*(2.*PI*D/WLEN)**3*WLEN*WLEN
      TTHP=TAN(THP)
        IF(TTHP.LT.CUTOFF.AND.LOBE.GT.0) CALL TOLOBE(1,RCSC,0.,0.,0.)
      IF(TTHP.LT.CUTOFF) RETURN
      SIGMA=.05*(2.*PI*D)**2/TTHP
      RCSC=AMIN1(RCSC,SIGMA)
        IF(LOBE.LE.0) RETURN
        DS2=D*STHP/2.
        CALL TOLOBE(1,RCSC,DS2*STHP*CPHP,DS2*STHP*SPHP,DS2*CTHP)
      RETURN
      END
C
C
C   <*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*>
      SUBROUTINE CFLAT
C
C   WITH LOBE
C
      IMPLICIT REAL*4 (A-Z)
        INTEGER*2 LOBE
      COMMON/RCSCOM/CUTOFF,WLEN,PI,PI2,RCSC,THP,CTHP,STHP,PHP,CPHP,SPHP,
     $ C2PHP,S2PHP,RAD,POLAR,LOBE,
     $ A
C
      IF(ABS(THP).GE.PI2) RETURN
      RCSC=4.*PI*(PI*A*A)**2/(WLEN*WLEN)
      D=8.*PI*STHP*TAN(THP)**2
        IF(D.LT.CUTOFF.AND.LOBE.GT.0) CALL TOLOBE(1,RCSC,0.,0.,0.)
      IF(D.LT.CUTOFF) RETURN
      SIGMA=A*WLEN/D
      RCSC=AMIN1(RCSC,2.*SIGMA)
        IF(LOBE.LE.0) RETURN
        CALL TOLOBE(1,RCSC/2.,A*CPHP,A*SPHP,0.)
        CALL TOLOBE(2,RCSC/2.,-A*CPHP,-A*SPHP,0.)
      RETURN
      END
C
C
C   <*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*>
      SUBROUTINE RFLAT(IPASS)
C
C   WITH LOBE
C
      IMPLICIT REAL*4 (A-Z)
        INTEGER*2 LOBE,IPASS
```

```
      COMMON/RCSCOM/CUTOFF,WLEN,PI,PI2,RCSC,THP,CTHP,STHP,PHP,CPHP,SPHP,
     $ C2PHP,S2PHP,RAD,POLAR,LOBE,
     $ A,B
C
      IF(ABS(THP).GE.PI2) RETURN
      RCSC=PI*(8.*A*B/WLEN)**2
      S2THP=STHP*STHP
      S4THP=S2THP*S2THP
      IF(S4THP.LT.CUTOFF) GO TO 4
      IF(S2PHP.LT.CUTOFF) GO TO 2
      IF(C2PHP.LT.CUTOFF) GO TO 3
      SIGMA=WLEN*WLEN/(64.*PI**3*S4THP*S2PHP*C2PHP)
        SIGMA=2.*SIGMA
1     RCSC=AMIN1(RCSC,2.*SIGMA)
        IF(LOBE.LE.0) RETURN
C
        RCSC4=RCSC/4.
        IF(IPASS.NE.2) CALL TOLOBE(1,RCSC4,A,B,0.)
        IF(IPASS.NE.2) CALL TOLOBE(2,RCSC4,-A,B,0.)
        IF(IPASS.EQ.2) CALL TOLOBE(1,RCSC4,A,-B,0.)
        IF(IPASS.EQ.2) CALL TOLOBE(2,RCSC4,-A,-B,0.)
      RETURN
C
C  PHI' = 0 OR 180
2       IF(IPASS.EQ.2) RETURN
      SIGMA=B*B/(4.*PI*S2THP)
        IF(LOBE.LE.0) GO TO 1
        SIG=AMIN1(RCSC/2.,SIGMA)
        CALL TOLOBE(1,SIG,A,0.,0.)
        CALL TOLOBE(2,SIG,-A,0.,0.)
5       RCSC=AMIN1(RCSC,2.*SIGMA)
        RETURN
C
C  PHI' = 90 OR -90
3       IF(IPASS.EQ.2) RETURN
      SIGMA=A*A/(4.*PI*S2THP)
        IF(LOBE.LE.0) GO TO 1
        SIG=AMIN1(RCSC/2.,SIGMA)
        CALL TOLOBE(1,SIG,0.,B,0.)
        CALL TOLOBE(2,SIG,0.,-B,0.)
      GO TO 5
C
C  THETA' = 0
4       IF(IPASS.EQ.2) RETURN
        IF(LOBE.LE.0) RETURN
        CALL TOLOBE(1,RCSC,0.,0.,0.)
        RETURN
      END
C
C
C  <*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*>
      SUBROUTINE TWAVE
C
C  WITH LOBE
C
      IMPLICIT REAL*4 (A-Z)
        INTEGER*2 LOBE
      COMMON/RCSCOM/CUTOFF,WLEN,PI,PI2,RCSC,THP,CTHP,STHP,PHP,CPHP,SPHP,
     $ C2PHP,S2PHP,RAD,POLAR,LOBE,
     $ P,L,GAMMA,A
```

```
C
         IF(POLAR.EQ.1.) RETURN
      KL=2.*PI*L/WLEN
      IF(KL.LE.4.) RETURN
      Q=2.35*ALOG10(2.*KL)-.55
      T=1.-P*CTHP
      IF(T.LE.CUTOFF) RETURN
      F=STHP*SIN(KL*T/(2.*P))/T
      RCSC=(GAMMA*F*F*WLEN/Q)**2/PI
         IF(LOBE.GT.0) CALL TOLOBE(1,RCSC,A*CPHP,A*SPHP,-L/2.)
      RETURN
      END
C
C
C  <*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*>
      SUBROUTINE DIHED
C
      IMPLICIT REAL*4 (A-Z)
         INTEGER*2 LOBE
      COMMON/RCSCOM/CUTOFF,WLEN,PI,PI2,RCSC,THP,CTHP,STHP,PHP,CPHP,SPHP,
     $ C2PHP,S2PHP,RAD,POLAR,LOBE,
     $ A,B,C
C
      IF(PHP.LE.0..OR.PHP.GE.PI2) RETURN
         PI4=PI/4.
      GAMMA=ATAN2(B,A)
      DU=PI2-THP
C  COMPUTE NORMAL COMPONENT
      IF(PHP.GT.GAMMA) GO TO 1
      RCSC=4.*PI*(A*B/WLEN)**2*4.*S2PHP
      GO TO 2
1     RCSC=4.*PI*(C*B/WLEN)**2*4.*C2PHP
2     IF(DU**2.GT.CUTOFF) GO TO 4
         IF(LOBE.LE.0) RETURN
4      IF(STHP.LT.0.5) GO TO 3
      SIGMA=WLEN*WLEN*RCSC/(8.*(PI*B*DU)**2)
      RCSC=AMIN1(RCSC,SIGMA)
      RETURN
3     RCSC=0.
      RETURN
      END
C
C
C  <*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*>
      SUBROUTINE TCORN
C
C  WITH LOBE
C
      IMPLICIT REAL*4 (A-Z)
         INTEGER*2 LOBE
      COMMON/RCSCOM/CUTOFF,WLEN,PI,PI2,RCSC,THP,CTHP,STHP,PHP,CPHP,SPHP,
     $ C2PHP,S2PHP,RAD,POLAR,LOBE,
     $ A,B,C
C
      IF(THP.LE.0.0.OR.THP.GE.PI2) RETURN
      IF(PHP.LE.0.0.OR.PHP.GE.PI2) RETURN
      CALL ORDER(STHP*CPHP,STHP*SPHP,CTHP,L,M,N)
      CALL ORDER(A,B,C,T1,T2,T3)
      LA=L/T3
      MB=M/T2
```

```
      NC=N/T1
      IF(LA+MB.GT.NC) GO TO 1
      AREA=4.*L*M*T1/(LA+MB+NC)
      GO TO 2
 1    AREA=(LA*LA+MB*MB+NC*NC)/(LA+MB+NC)
      AREA=A*B*C*(LA+MB+NC-2.*AREA)
 2    RCSC=4.*PI*AREA*AREA/(WLEN*WLEN)
        IF(LOBE.LE.0) RETURN
        D=1./SQRT(1./A**2+1./B**2+1./C**2)
        CALL TOLOBE(1,RCSC,D*STHP*CPHP,D*STHP*SPHP,D*CTHP)
      RETURN
      END
C
C
C  <*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*>
      SUBROUTINE RCORN
C
C  WITH LOBE
C
      IMPLICIT REAL*4 (A-Z)
        INTEGER*2 LOBE
      COMMON/RCSCOM/CUTOFF,WLEN,PI,PI2,RCSC,THP,CTHP,STHP,PHP,CPHP,SPHP,
     $ C2PHP,S2PHP,RAD,POLAR,LOBE,
     $ A,B,C
C
      IF(THP.LE.0.0.OR.THP.GE.PI2) RETURN
      IF(PHP.LE.0.0.OR.PHP.GE.PI2) RETURN
      CALL ORDER(STHP*CPHP,STHP*SPHP,CTHP,L,M,N)
      CALL ORDER(A,B,C,T1,T2,T3)
      LA=L/T3
      MB=M/T2
      NC=N/T1
      IF(MB.LT.NC/2.) GO TO 1
      AREA=L*T2*(4.*T1-N/MB)
      GO TO 2
 1    AREA=4.*L*M*T1*T1/N
 2    RCSC=4.*PI*(AREA/WLEN)**2
        IF(LOBE.LE.0) RETURN
        D=1./SQRT(1./A**2+1./B**2+1./C**2)
        CALL TOLOBE(1,RCSC,D*STHP*CPHP,D*STHP*SPHP,D*CTHP)
      RETURN
      END
C
C
C  <*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*>
      SUBROUTINE ECORN
C
C  WITH LOBE
C
      IMPLICIT REAL*4 (A-Z)
        INTEGER*2 LOBE
      COMMON/RCSCOM/CUTOFF,WLEN,PI,PI2,RCSC,THP,CTHP,STHP,PHP,CPHP,SPHP,
     $ C2PHP,S2PHP,RAD,POLAR,LOBE,
     $ A,B,C
C
      IF(THP.LE.0.0.OR.THP.GE.PI2) RETURN
      IF(PHP.LE.0.0.OR.PHP.GE.PI2) RETURN
      CALL ORDER(STHP*CPHP,STHP*SPHP,CTHP,L,M,N)
      CALL ORDER(A,B,C,T1,T2,T3)
      LA2=(L/T3)**2
```

```
      MB2=(M/T2)**2
      NC2=(N/T1)**2
      IF(NC2.GE.LA2+MB2) GO TO 1
      AREA=(N/T1)*ATAN(((LA2+MB2)**2-NC2*NC2)/(4.*NC2*(L/T3)*(M/T2)))
      AREA=AREA+(M/T2)*ATAN(((LA2+NC2)**2-MB2*MB2)/
     $ (4.*MB2*(L/T3)*(N/T1)))
      AREA=AREA+(L/T3)*ATAN(((MB2+NC2)**2-LA2*LA2)/
     $ (4.*LA2*(M/T2)*(N/T1)))
      GO TO 2
 1    AREA=(M/T2)*ATAN(2.*(L/T3)*(N/T1)/(MB2+NC2-LA2))
      AREA=AREA+(L/T3)*ATAN(2.*(M/T2)*(N/T1)/(LA2-MB2+NC2))
 2    AREA=A*B*C*AREA
      RCSC=4.*PI*(AREA/WLEN)**2
        IF(LOBE.LE.0) RETURN
        D=1./SQRT(1./A**2+1./B**2+1./C**2)
        CALL TOLOBE(1,RCSC,D*STHP*CPHP,D*STHP*SPHP,D*CTHP)
      RETURN
      END
```

```
>>> FILE = SY0:[111,12]RCSSUB.FTN             27-SEP-83  14:35:49  <<<
C
C
C  THIS ROUTINE ASSUMES MATRICES ARE ROW STRUCTURED
C
      SUBROUTINE TOLOBE(IC,RCSL,X,Y,Z)
C
      INTEGER*2 DEBUG,COMBIN,GLINT
      LOGICAL*2 GOULD,LP,OPENE,LPF,OLDGRF,MGRAPH,GDAUTO
      COMMON/SWITCH/ IOW,THETA(4),GOULD,LP,OPENE,LPF,OLDGRF,MGRAPH,
      1 GDAUTO,DEBUG,COMBIN,GLINT,IBUG,LBUG,ISTATS,IZETA
C
      REAL*4 SLOBE(4,181,2)
      REAL*4 ORGN(3),COMP(9),MTRAN(3,3),TMP(3)
      COMMON/RCSLOB/IOL,ORGN,COMP,SLOBE,BETA,
     $ THR,CTHR,STHR,PHR,CPHR,SPHR,KLOBE
C
      COMMON/RCSCOM/CUTOFF,WLEN,PI,PI2,RCSC,THP,CTHP,STHP,PHP,CPHP,SPHP,
     $ C2PHP,S2PHP,RAD,POLAR,LOBE,
     $ ARR
C
      REAL*4 RCS(181),RCS2(181)
      COMMON /RPRCS/ IRPR,IRCS,RAM,RCS,RCS2
C
C     IF(LBUG.NE.0) WRITE(2,7) IC,RCSL,X,Y,Z
C7    FORMAT(' TOLOBE=',I1,4E11.3)
C
      TMP(1)=X
      TMP(2)=Y
      TMP(3)=Z
      CALL MATVEC(COMP,TMP,TMP)
      TMP(1)=TMP(1)+ORGN(1)
      TMP(2)=TMP(2)+ORGN(2)
      TMP(3)=TMP(3)+ORGN(3)
C
      CALL MATPUT(MTRAN,CTHR*CPHR,CTHR*SPHR,-STHR,
     1          SPHR,-CPHR,0.,
     2          -STHR*CPHR,-STHR*SPHR,-CTHR)
      IF(BETA.LE.5.) GO TO 1
C
1     CALL MATVEC(MTRAN,TMP,TMP)
C
      IF((LBUG.AND.2).EQ.0) GO TO 5
      WRITE(2,4) IC,THR,PHR,X,Y,Z,ORGN,MTRAN
4     FORMAT(1X,I2,5E10.3/3E11.3/3E11.3/3E11.3)
C
5     IF(IRPR.EQ.0) GO TO 3
      ARG=4.*PI*TMP(3)/WLEN
      SQ=SQRT(RCSL*RAM)
      RCS(IRCS)=RCS(IRCS)+SQ*COS(ARG)
      RCS2(IRCS)=RCS2(IRCS)+SQ*SIN(ARG)
      IF((LBUG.AND.1).NE.0)WRITE(2,6)IRCS,RCSL,RAM,TMP,RCS(IRCS),RCS2(IRCS)
6     FORMAT(' RCSL=',I3,7E11.3)
      IF(KLOBE.LE.0) RETURN
C
3     SLOBE(1,IRCS,IC)=RCSL*RAM
      SLOBE(2,IRCS,IC)=TMP(1)
      SLOBE(3,IRCS,IC)=TMP(2)
      SLOBE(4,IRCS,IC)=TMP(3)
C
```

```
          IF((LBUG.AND.1).NE.0) WRITE(2,2) IC,IRCS,(SLOBE(I,IRCS,IC),I=1,4)
2         FORMAT(' TOLOBE> ',2I2,E11.3,3X,3F8.3)
C
          RETURN
          END
C
C
C
        SUBROUTINE ORDER(A1,A2,A3,B1,B2,B3)
        T1=AMIN1(A1,A2,A3)
        T3=AMAX1(A1,A2,A3)
        B2=A1+A2+A3-T1-T3
        B1=T1
        B3=T3
        RETURN
        END
C
C
C  THIS ROUTINE ASSUMES MATRICES ARE ROW STRUCTURED
C
        SUBROUTINE MATMUL(A,B,C)
          REAL*4 A(9),B(9),T(9),C(9)
          DO 1 J=1,3
          J13=(J-1)*3
          DO 2 K=1,3
          JK=J13+K
          T(K+J13)=0.
          DO 3 L=1,3
3         T(K+J13)=T(K+J13)+A(J13+L)*B(K+(L-1)*3)
2         CONTINUE
1         CONTINUE
          DO 4 J=1,9
4         C(J)=T(J)
        RETURN
        END
C
C
C  THIS ROUTINE ASSUMES MATRICES ARE ROW STRUCTURED
C
        SUBROUTINE MATVEC(A,B,C)
          REAL*4 A(9),B(3),C(3),T(3)
          DO 1 J=1,3
          J13=(J-1)*3
          T(J)=0.
          DO 3 L=1,3
3         T(J)=T(J)+A(J13+L)*B(L)
1         CONTINUE
          DO 4 J=1,3
4         C(J)=T(J)
          RETURN
          END
C
C
        SUBROUTINE MATPUT(A,A11,A12,A13,A21,A22,A23,A31,A32,A33)
        REAL*4 A(9)
        A(1)=A11
        A(2)=A12
        A(3)=A13
        A(4)=A21
        A(5)=A22
```

```
      A(6)=A23
      A(7)=A31
      A(8)=A32
      A(9)=A33
      RETURN
      END
C
C
      SUBROUTINE TRANSP(C,E)
      REAL C(9),D(9),E(9)
      D(1)=C(1)
      D(2)=C(4)
      D(3)=C(7)
      D(4)=C(2)
      D(5)=C(5)
      D(6)=C(8)
      D(7)=C(3)
      D(8)=C(6)
      D(9)=C(9)
        DO 1 J=1,9
1       E(J)=D(J)
      RETURN
      END
C
C
C
      SUBROUTINE BESJ(X,N,BJ,D,IER)
C
C     X=ARGUMENT OF THE J BESSEL FUNCTION DESIRED
C     N=ORDER OF THE J BESSEL FUNCTION
C     BJ=RESULTANT J BESSEL FUNCTION
C     D=REQUIRED ACCURACY
C     IER=RESULTANT ERROR CODE
C     0=NO ERROR
C     1=N IS NEGATIVE
C     2=X IS NEGATIVE
C     3=REQUIRED ACCURACY NOT OBTAINED
C     4=RANGE OF N COMPARED TO X NOT CORRECT
C
      BJ=0.
      IF(N)10,20,20
10    IER=1
      RETURN
20    IF(X)33,30,31
30    IF(N.GT.0) RETURN
      BJ=1.
      RETURN
33    IER=2
      RETURN
31    IF(X-15.)32,32,34
32    NTEST=20.+10.*X-X**2/3
      GO TO 36
34    NTEST=90.+X/2.
36    IF(N-NTEST)40,38,38
38    IER=4
      RETURN
40    IER=0
      N1=N+1
      BPREV=0.
C
```

```
C     COMPUTE STARTING VALUE OF M
C
      IF(X-5.)50,60,60
50    MA=X+6.
      GO TO 70
60    MA=1.4*X+60./X
70    MB=N+IFIX(X)/4+2
      MZERO=MAX0(MA,MB)
C
C     SET UPPER LIMIT OF M
C
      MMAX=NTEST
100   DO 190 M=MZERO,MMAX,3
C
C     SET F(M),F(M-1)
C
      FM1=1.E-15
      FM=0.
      ALPHA=0.
      IF(M-(M/2)*2)120,110,120
110   JT=-1
      GO TO 130
120   JT=1
130   M2=M-2
      DO 160 K=1,M2
      MK=M-K
      BMK=2.*FLOAT(MK)*FM1/X-FM
      FM=FM1
      FM1=BMK
      IF(MK-N-1)150,140,150
140   BJ=BMK
150   JT=-JT
      S=1+JT
160   ALPHA=ALPHA+BMK*S
      BMK=2.*FM1/X-FM
      IF(N)180,170,180
170   BJ=BMK
180   ALPHA=ALPHA+BMK
      BJ=BJ/ALPHA
      IF(ABS(BJ-BPREV)-ABS(D*BJ))200,200,190
190   BPREV=BJ
      IER=3
200   RETURN
      END
C
C
        SUBROUTINE TRIM80(NB,B)
        LOGICAL*1 B(80),BL
        DATA BL/' '/
        NB=80
        DO 1 J=1,80
        IF(B(NB).NE.BL) RETURN
1       NB=NB-1
        RETURN
        END
C
C
        FUNCTION ACOSK(A)
        ACOSK=ACOS(AMAX1(-1.,AMIN1(1.,A)))
        RETURN
        END
```

```
>>>  FILE = SY0:[111,12]HOSBD.FTN               27-SEP-83   14:38:18   <<<
C*****************************************************************
C         H   H    OOO     SSSS    BBBB    DDDD
C         H   H   O   O   S        B   B   D   D
C         HHHHH   O   O    SSS     BBBB    D   D
C         H   H   O   O       S    B   B   D   D
C         H   H    OOO    SSSS     BBBB    DDDD
C*****************************************************************
C
          SUBROUTINE HOSBD(IOUT,LOOP,IOE,IOP,SLOBE,BUF,NB,
          1        FREQ,POL,PERSH,PITCH,ROLL,YAW,TILT,PHR)
          REAL*8 T1,T2,T3,T4,S1,S2,S3,S4,T12,S22,V1,U2,U3,U4,G1,G2
          REAL*8 B,N,P,RCSMAX,RCSMIN,T5,T6
          REAL*4 SLOBE(8,181),SAVE(30)
          BYTE BUF(1)
C
          INTEGER*2 DEBUG,COMBIN,GLINT
          LOGICAL*2 GOULD,LP,OPENE,LPF,OLDGRF,MGRAPH,GDAUTO
          COMMON/SWITCH/ IOW,THETA(4),GOULD,LP,OPENE,LPF,OLDGRF,MGRAPH,
          1 GDAUTO,DEBUG,COMBIN,GLINT,IBUG,LBUG,ISTATS,IZETA,STATS,NRCS,IOR
C
          LP=LOOP
          IF(IOUT.EQ.0) GO TO 20
C
C    COMPUTE REQUIRED ANGLES FOR GRAPH
C
          SLOPE=FLOAT(NRCS-1)/AMAX1(.001,THETA(2)-THETA(1))
          CONST=1.-SLOPE*THETA(1)
          ANG=STATS
          NS=0
 11       IF(ANG.GE.THETA(1)) GO TO 12
          ANG=ANG+STATS
          GO TO 11
 4        ANG=ANG+STATS
          NS=NS-1
 12       IF(ANG.GT.THETA(2)) GO TO 13
          LP=IFIX(SLOPE*ANG+CONST+.001)
          NS=NS+1
C
 20       READ(IOE'LP) T1,T2,T3,T4,T5,T6
C
          S1=T1
          T12=T1*T1
          S2=(T12-T2)/2.D0
C
          IF(S1*S2.EQ.0.D0.AND.IOUT.EQ.0) RETURN
          IF(S1*S2.EQ.0.D0) GO TO 4
C
          S3=(T12*T1-3.D0*T1*T2+2.D0*T3)/6.D0
          S4=(T12*T12-6.D0*T12*T2+3.D0*T2*T2+8.D0*T1*T3-6.D0*T4)/24.D0
C
          V1=S1
          U2=2.D0*S2
          U3=12.D0*S3
          S22=S2*S2
          U4=6.D0*S22+12.D0*S1*S3+132.D0*S4
C
          G1=6.D0*S3/(S1*S2)
          G2=-1.5D0+3.D0*S1*S3/S22+33.D0*S4/S22
C
```

```
      B=2.D0*S2/(S1*S1)
      N=S1*(S1*S2+3.D0*S3)/(2.D0*S22-3.D0*S1*S3)
      P=1.D0/(N*B)
      RCSMAX=T5**2
      RCSMIN=(2.D0*T6-T5)**2
      IF(IOUT.EQ.0) GO TO 5
C
      SLOBE(1,NS)=ANG
      SLOBE(2,NS)=B
      SLOBE(3,NS)=G1
      SLOBE(4,NS)=G2
      ANG=ANG+STATS
      GO TO 12
C
5     WRITE(IOP,1) S1,S2,S3,S4
1     FORMAT(' S1=',D11.3,' S2=',D11.3,' S3=',D11.3,' S4=',D11.3)
      WRITE(IOP,2) V1,U2,U3,U4,G1,G2
2     FORMAT(' V1=',D11.3,' U2=',D11.3,' U3=',D11.3,' U4=',D11.3,
     1 ' G1=',D11.3,' G2=',D11.3)
      WRITE(IOP,3) B,N,P,RCSMAX,RCSMIN
3     FORMAT('  W=',D11.3,'  N=',D11.3,'  P=',D11.3,' RCSMAX=',D11.3,
     1 ' RCSMIN=',D11.3/)
C
      RETURN
C
13    CONTINUE
C
C   PREPARE SKEWNESS GRAPH
C
      IF(.NOT.GDAUTO) CALL GRMODE('ALPHA',0)
      IF(GDAUTO.AND.IZETA.EQ.0) CALL GRTEKG
      IF(IZETA.EQ.0.AND..NOT.GDAUTO) READ(IOR,18) IJK
      CALL GRSAV(SAVE)
      IF(IZETA.EQ.0) CALL GRMODE('ERASE',3)
      IF(IZETA.EQ.0) CALL GRMODE('ERASE',3)
      IF(IZETA.GT.0) CALL GRPLOT(11.,8.5,-1)
C
      CALL GRHV('V')
      CALL GRAXIS(0,1.5,1.5,5.,0.,0.,1.,6,2,'WIDTH',5,2)
      CALL GRAXIS(3,1.5,1.5,7.5,90.,0.,2.,11,2,'SKEWNESS',8,2)
      CALL GRPIC(1.5,1.5,6.5,9.)
      CALL GRSCL(0.,0.,1.,2.)
C
      CALL GRSXY(4.,9.5,X,Y)
      CALL GRTXT(BUF,NB)
      CALL GRPRNT(0,X,0,Y,.15,0.)
C
      CALL GRSXY(6.3,8.5,X,Y)
      CALL GRSXY(6.3,8.3,X,DY)
      DY=Y-DY
      CALL GRTXT('FREQUENCY ',30)
      CALL GRNUM('F',-1,2,FREQ)
      CALL GRTXT(' GHZ',4)
      CALL GRPRNT(-1,X,0,Y,.08,0.)
      Y=Y-DY
      CALL GRTXT('POLARIZATION  ',50)
      CALL GRTXT(POL,3)
      CALL GRPRNT(-1,X,0,Y,.08,0.)
      Y=Y-DY
      CALL GRTXT('% SHADOWED ',20)
```

```
      CALL GRNUM('F',-1,2,PERSH)
      CALL GRPRNT(-1,X,0,Y,.08,0.)
      Y=Y-DY
      CALL GRTXT('PITCH ',20)
      CALL GRNUM('F',-1,2,PITCH)
      CALL GRPRNT(-1,X,0,Y,.08,0.)
      Y=Y-DY
      CALL GRTXT('ROLL ',20)
      CALL GRNUM('F',-1,2,ROLL)
      CALL GRPRNT(-1,X,0,Y,.08,0.)
      Y=Y-DY
      CALL GRTXT('YAW ',20)
      CALL GRNUM('F',-1,2,YAW)
      CALL GRPRNT(-1,X,0,Y,.08,0.)
      Y=Y-DY
      CALL GRTXT('TILT ',20)
      CALL GRNUM('F',-1,2,TILT)
      CALL GRPRNT(-1,X,0,Y,.08,0.)
      Y=Y-DY
      CALL GRTXT('PHIR ',20)
      CALL GRNUM('F',-1,2,PHR)
      CALL GRPRNT(-1,X,0,Y,.08,0.)
C
      CALL GRMOVE(0.,0.)
      DO 10 J=1,50
      T=FLOAT(J)/2.5
      W=T*(T+2.)/(T+1.)**2
      G=2.*T*(T+3.)/((T+1.)*(T+2.))
10    CALL GRDRAW(W,G)
      CALL GRDRAW(1.,2.)
      CALL GRDRAW(.5,0.)
C
      DO 14 J=1,NS
      CALL GRSYM(SLOBE(2,J),SLOBE(3,J),.1,0.,42)
      CALL GRTXT(' ',1)
      CALL GRNUM('I',-1,IFIX(SLOBE(1,J)),DUMMY)
14    CALL GRPRNT(-1,SLOBE(2,J),0,SLOBE(3,J),.08,0.)
C
C  PREPARE KURTOSIS GRAPH
C
      IF(.NOT.GDAUTO) CALL GRMODE('ALPHA',0)
      IF(GDAUTO.AND.IZETA.EQ.0) CALL GRTEKG
      IF(IZETA.EQ.0.AND..NOT.GDAUTO) READ(IOR,18) IJK
18    FORMAT(A2)
      IF(IZETA.EQ.0) CALL GRMODE('ERASE',3)
      IF(IZETA.GT.0) CALL GRPLOT(8.5,0.,-1)
C
      CALL GRAXIS(0,1.5,1.5,5.,0.,0.,1.,6,2,'WIDTH',5,2)
      CALL GRAXIS(3,1.5,1.5,7.5,90.,-2.,6.,9,2,'KURTOSIS',8,2)
      CALL GRSCL(0.,-2.,1.,6.)
C
      CALL GRSXY(4.,9.5,X,Y)
      CALL GRTXT(BUF,NB)
      CALL GRPRNT(0,X,0,Y,.15,0.)
C
      CALL GRSXY(6.8,8.5,X,Y)
      CALL GRSXY(6.8,8.3,X,DY)
      DY=Y-DY
      CALL GRTXT('FREQUENCY ',30)
      CALL GRNUM('F',-1,2,FREQ)
```

```
      CALL GRTXT(' GHZ',4)
      CALL GRPRNT(-1,X,0,Y,.08,0.)
      Y=Y-DY
      CALL GRTXT('POLARIZATION  ',50)
      CALL GRTXT(POL,3)
      CALL GRPRNT(-1,X,0,Y,.08,0.)
      Y=Y-DY
      CALL GRTXT('% SHADOWED ',20)
      CALL GRNUM('F',-1,2,PERSH)
      CALL GRPRNT(-1,X,0,Y,.08,0.)
      Y=Y-DY
      CALL GRTXT('PITCH ',20)
      CALL GRNUM('F',-1,2,PITCH)
      CALL GRPRNT(-1,X,0,Y,.08,0.)
      Y=Y-DY
      CALL GRTXT('ROLL ',20)
      CALL GRNUM('F',-1,2,ROLL)
      CALL GRPRNT(-1,X,0,Y,.08,0.)
      Y=Y-DY
      CALL GRTXT('YAW ',20)
      CALL GRNUM('F',-1,2,YAW)
      CALL GRPRNT(-1,X,0,Y,.08,0.)
      Y=Y-DY
      CALL GRTXT('TILT ',20)
      CALL GRNUM('F',-1,2,TILT)
      CALL GRPRNT(-1,X,0,Y,.08,0.)
      Y=Y-DY
      CALL GRTXT('PHIR ',20)
      CALL GRNUM('F',-1,2,PHR)
      CALL GRPRNT(-1,X,0,Y,.08,0.)
C
      CALL GRMOVE(0.,0.)
      DO 15 J=1,50
      T=FLOAT(J)/2.5
      W=T*(T+2.)/(T+1.)**2
      Q=6.*T*(T+4.)/(T+2.)**2
15    CALL GRDRAW(W,Q)
      CALL GRDRAW(1.,6.)
      W=1.
      DO 16 J=1,50
      W=W-.01
      Z=4.*W-2.
      ZN=(Z+2.)/(2.*W-Z)
      ZN2=ZN*ZN
      Q=6.*ZN2*(W+1./(ZN2*W)-3./ZN-1./ZN2)/((ZN+1.)*(ZN+2.))
16    CALL GRDRAW(W,Q)
      CALL GRDRAW(0.,-1.5)
C
      DO 17 J=1,NS
      CALL GRSYM(SLOBE(2,J),SLOBE(4,J),.1,0.,42)
      CALL GRTXT(' ',1)
      CALL GRNUM('I',-1,IFIX(SLOBE(1,J)),DUMMY)
17    CALL GRPRNT(-1,SLOBE(2,J),0,SLOBE(4,J),.08,0.)
      CALL GRRES(SAVE)
      CALL GRHV('H')
      RETURN
      END

>>>  FILE = SY0:[111,12]SPAWNS.FTN            27-SEP-83  14:36:42  <<<
      SUBROUTINE SPAWNS
      INTEGER*2 EXSTAT(8)
      DATA TASK/6RMCR.../
      CALL SPAWN(TASK,,,11,,EXSTAT,,'RUN LOBE/TASK=LOBE',18,,IDSW)
      CALL STOPFR(11)
      RETURN
      END
```

```
>>>  FILE = SY0:[111,12]LOBE.FTN                 27-SEP-83  14:36:48  <<<
C*****************************************************************
C
C        L         OOO     BBBB    EEEEE
C        L        O   O    B   B   E
C        L        O   O    BBBB    EEEE
C        L        O   O    B   B   E
C        LLLLL     OOO     BBBB    EEEEE
C
C
C        COMPUTE RCS LOBE STATISTICS
C
C*****************************************************************
C
         REAL*4 SLOBE(4,181),SIG(4,180),NF,ND(181,3),SAVE(30)
         REAL*4 THETA(4),ALPHA(4),DELTA(4),SCALE(6),B(2,2),C(2,2)
         REAL*4 D(2,2,2,2),AMN(2),DRS(2),TMPARR(2,2),GSIZE(5)
         INTEGER*2 GLINT,RR,SS
         LOGICAL*2 GDAUTO
         BYTE BUF(80)
C
         DID(X)=44577.*COS(1.5707963-ATAN(X/685.8))
C
         IOW=2
         CALL ASNLUN(IOW,'TI',0)
         IOL=1
         CALL ASNLUN(IOL,'SY',0)
         IOT=5
         CALL ASNLUN(IOT,'SY',0)
C
         OPEN(UNIT=IOL,NAME='LOBE.TMP',TYPE='OLD',RECORDSIZE=181,
         1 ACCESS='DIRECT',READONLY,SHARED)
C
         READ(IOL'1) IOUT,KLOBE,GLINT,NRCS,NLOBE,THETA,ALPHA,DELTA,
         1 BETA,SCALE,IZETA,IPCODE,WLEN,LBUG,NB,BUF,GDAUTO,
         2 FREQ,POL,PERSH,PITCH,ROLL,YAW,TILT,PHR,GSIZE
         LBUG4=LBUG.AND.4
         LBUG8=LBUG.AND.8
C
         IF(DELTA(3).EQ.0.) WRITE(IOW,8)
  8      FORMAT(' LOBE> -- DELTA HAS NOT BEEN DEFINED')
         IF(DELTA(3).EQ.0.) GO TO 38
C
         IF(LBUG4.NE.0) WRITE(2,227) IOUT,KLOBE,GLINT,NRCS,NLOBE,IZETA,IPCODE,
         1 THETA,DELTA,ALPHA,SCALE,BETA,WLEN
  227    FORMAT(1X,7I4/1X,8F8.2/1X,8F8.2/1X,8F8.2)
C
         IO=IOW
         IF(IOUT.EQ.1) GO TO 4
C
C   INITIALIZE LOBE FOR PRINTED OUTPUT
C
         IF(IPCODE.EQ.0) GO TO 7
         IO=3
         CALL ASNLUN(IO,'GD',0)
         GO TO 7
C
C   INITIALIZE LOBE FOR GRAPHICAL OUTPUT
C
  4      IOG=4
```

```
      CALL GRINIT(0,IZETA,IOG,IER)
      CALL GRHV('H')
      CALL GRPIC(GSIZE(1),GSIZE(2),GSIZE(3),GSIZE(4))
      IF(IZETA.EQ.0) CALL GRDEV('TEK','HARD',IER)
C
C
C  LBUG DEFINITIONS:
C                LBUG=1=>TOLOBE TRANSFORMATIONS
C                LBUG=2=>TOLOBE OUTPUT
C                LBUG=4=>INPUT & LOBE OUTPUT
C                LBUG=8=>GLINT OUTPUT
C
7     NANG=(DELTA(2)-DELTA(1))/DELTA(3)+1.0001
      IF(IOUT.EQ.0.OR.IZETA.NE.0) WRITE(IO,222) NANG
222   FORMAT('0>> COMPUTE RCS LOBE STATISTICS FOR',I3,'  THETA ANGLES <<')
C
C     COMPUTE CONSTANT TERMS
C
      SNBET=SIN(BETA)
      CSBET1=COS(BETA)+1.
      SOL=.2998E9
      PI=3.1415926
      TWOPI=PI*2.
      RAD=PI/180.
      IF(IOUT.EQ.1) CALL GRSCL(SCALE(1),SCALE(2),SCALE(3),SCALE(4))
C
C
C     PERFORM ENTIRE SET OF LOBE STATISTIC COMPUTATIONS
C     (READ ENTIRE TEMPORARY FILE EACH TIME)
C     FOR EACH REQUIRED DELTA ANGLE
C
      ANG=DELTA(1)
21    IF(ANG.GE.THETA(1)) GO TO 22
      ANG=ANG+DELTA(3)
      NANG=NANG-1
      IF(NANG.LE.0) STOP 'LOBE ERROR'
      GO TO 21
C
22    FIRSTA=ANG
      SLOPE=FLOAT(NRCS-1)/AMAX1(.01,THETA(2)-THETA(1))
      CONST=1.-SLOPE*THETA(1)
      IF(LBUG4.NE.0) WRITE(2,228) SLOPE,CONST,ANG
228   FORMAT(' SLOPE,CONST,ANG=',3F10.3)
C
      DO 100 LL=1,NANG
C
C  DETERMINE NEXT INDEX INTO LOBE ARRAY
      IF(ANG.GT.THETA(2)) GO TO 101
      L=IFIX(SLOPE*ANG+CONST+.001)
C
C  COMPUTE SUM AND STANDARD DEVIATION
C
      NLB=-3
      SUM=0.
      STD=0.
      N=0
C
1     NLB=NLB+4
      IF(NLB.GT.NLOBE) GO TO 3
      READ(IOL'NLB+1) (SLOBE(1,J),J=1,L)
```

```
           IF(SLOBE(1,L).EQ.0.) GO TO 1
           DO 19 K=2,4
   19      READ(IOL'NLB+K) (SLOBE(K,J),J=1,L)
C
           IF(LBUG4.NE.0) WRITE(2,221) (SLOBE(I,L),I=1,4)
  221      FORMAT(' RCS,POSN=',4E11.3)
C
           N=N+1
           DO 2 J=1,4
    2      SIG(J,N)=SLOBE(J,L)
C
           SUM=SUM+SIG(1,N)
           STD=STD+SIG(1,N)**2
           IF(N.LT.180) GO TO 1
           IF(N.EQ.180) WRITE(2,9)
    9      FORMAT(' RCS> -- LOBE WILL USE ONLY THE FIRST 180 COMPONENTS')
C
C
    3      IF(N.LE.1) GO TO 100
           IF((IOUT.EQ.0.OR.IZETA.NE.0).AND.LL.EQ.1) WRITE(2,24) N
   24      FORMAT(10X,'>> USING',I4,'  SUBCOMPONENTS <<')
           STD=SQRT(SUM*SUM-STD)
           IF((LBUG4).NE.0) WRITE(2,223) N,SUM,STD
  223      FORMAT(' N,SUM,STD=',I5,2E11.3)
C
C
C                   COMPUTE  R C S  LOBE  I
C
C
           G=0.
           N1=N-1
C
           DO 5 I=1,N1
           I1=I+1
           DO 5 J=I1,N
    5      G=G+SIG(1,I)*SIG(1,J)*((SIG(4,I)-SIG(4,J))*CSBET1+
     $       (SIG(2,I)-SIG(2,J))*SNBET)**2
           NF=SQRT(G)/(SOL*STD)
           IF((LBUG4).NE.0) WRITE(2,224) NF
  224      FORMAT(' NF=',4E11.3)
           NF=NF*1.E9
C
C
C                   COMPUTE  R C S  LOBE  II
C
C
C
C    COMPUTE A MATRIX (A(1,2)=A(2,1))
C
           A11=0.
           A12=0.
           A22=0.
C
           DO 6 I=1,N1
           I1=I+1
           DO 6 J=I1,N
C
           T1=SIG(1,I)*SIG(1,J)
C
           T2=(SIG(2,I)-SIG(2,J))*CSBET1-(SIG(4,I)-SIG(4,J))*SNBET
```

```
C
          T3=(SIG(3,I)-SIG(3,J))*CSBET1
C
          A11=A11+T1*T2*T2
          A12=A12+T1*T2*T3
          A22=A22+T1*T3*T3
C
6         CONTINUE
          A21=A12
C
C
C   COMPUTE F(DELTA) VS DELTA
C
          DELT=360./DELTA(4)
C         DELT=2.
          DEL=0.
          IDEL=1
          IF(LBUG4.NE.0) WRITE(2,225) DELT,A11,A12,A21,A22
225       FORMAT(' DELT=',E11.3/' A=',2E11.3/3X,2E11.3)
C
10        IF(DEL.GT.359.9) GO TO 11
          ND(IDEL,1)=DEL
          D1=SIN(DEL*RAD)
          D2=-COS(DEL*RAD)
          T1=A11*D1+A12*D2
          T2=A21*D1+A22*D2
          A=D1*T1+D2*T2
          ND(IDEL,2)=SQRT(A)/(WLEN*STD)
C
          DEL=DEL+DELT
          IDEL=IDEL+1
          GO TO 10
C
11        ND(IDEL,1)=360.
          ND(IDEL,2)=ND(1,2)
C
          DO 18 J=1,IDEL
18        ND(J,1)=AMOD(ND(J,1)+90.,360.)
C
          IF(IOUT.EQ.1) GO TO 15
C
C  PRINT NF AND ND
C
          WRITE(IO,12) ANG,NF
12        FORMAT('ORCS LOBE OUTPUT',5X,'THETA=',F6.1,'   NF(GHZ)=',F6.3/
     $     5X,'DELTA',5X,'ND'/)
C
          DO 13 J=1,IDEL
13        WRITE(IO,14) ND(J,1),ND(J,2)
14        FORMAT(1X,F9.2,F10.3)
          GO TO 100
C
C  GRAPH NF AND 1/ND
C
15        CONTINUE
          FMAX=0.
          FMIN=1000.
          DO 16 J=1,IDEL
          ND(J,2)=1./(ND(J,2)*RAD)
          FMAX=AMAX1(FMAX,ND(J,2))
```

```
         IF(FMAX.EQ.ND(J,2)) FANG=ND(J,1)
         FMIN=AMIN1(FMIN,ND(J,2))
         ND(J,2)=ND(J,2)
16       CONTINUE
         IF(FANG.GE.180.) FANG=FANG-180.
         IF(FANG.GT.90.) FANG=FANG-180.
C
         CALL GRRXY(ANG,0.,X,Y)
         Y=GSIZE(2)-.95
         CALL GRSAV(SAVE)
         CALL GRINIT(1,0,0,IER)
         IF(LL.NE.1) GO TO 17
C
         CALL GRTXT('NF(GHZ)',0)
         CALL GRPRNT(+1,2.1,-1,Y+.53,.1,0.)
         CALL GRTXT('MAJOR',0)
         CALL GRPRNT(+1,2.1,-1,Y+.38,.1,0.)
         CALL GRTXT('MINOR',0)
         CALL GRPRNT(+1,2.1,-1,Y+.23,.1,0.)
         CALL GRTXT('ANGLE',0)
         CALL GRPRNT(+1,2.1,-1,Y+.08,.1,0.)
C
17       CALL GRNUM('F',-1,3,NF)
         CALL GRPRNT(0,X,-1,Y+.53,.10,0.)
         CALL GRNUM('F',-1,2,FMAX)
         CALL GRPRNT(0,X,-1,Y+.38,.1,0.)
         CALL GRNUM('F',-1,2,FMIN)
         CALL GRPRNT(0,X,-1,Y+.23,.1,0.)
         CALL GRNUM('F',-1,2,FANG)
         CALL GRPRNT(0,X,-1,Y+.08,.1,0.)
C
         CALL GRCP('POLAR')
         Y=GSIZE(2)
         CALL GRPIC(X,Y,X+1.,Y+1.)
         CALL GRSCL(0.,0.,(SCALE(3)-SCALE(1))/6.5,360.)
         CALL GRPIC(0.,Y-.28,11.,Y+.5)
         CALL GRWIND(-IDEL,ND(1,2),ND(1,1),IDUM,0,DUM,DUM,IDUM)
         CALL GRUPPN
         CALL GRRES(SAVE)
C
100      ANG=ANG+DELTA(3)
101      CONTINUE
C
         IF(GLINT.EQ.0) CALL EXIT
C
C**********************************************************
C
C
C                COMPUTE  GLINT  LOBE  III
C
C
         OPEN(UNIT=IOT,NAME='LOBE.SCR',TYPE='SCRATCH')
C
         DO 38 IALP=1,91,90
C
C   PAGE PLOT
         IF(IOUT.EQ.0) GO TO 50
         IF(.NOT.GDAUTO) CALL GRMODE('ALPHA',0)
         IF(IZETA.EQ.0.AND.GDAUTO) CALL GRTEKG
         IF(IZETA.EQ.0.AND..NOT.GDAUTO) READ(IOW,47) IJK
```

```
47        FORMAT(A2)
          IF(IZETA.GT.0) CALL GRPLOT(11.,8.5,-1)
          CALL GRINIT(1,0,0,IER)
C
50        ALP=FLOAT(IALP-1)
          REWIND IOT
C
C    LOOP ON THETA INCREMENTS
          ANG=THETA(1)
          DO 35 L=1,NRCS
          IF(ALP.EQ.0.) GO TO 23
C
          READ(IOT,20) N,XBAR,YBAR,SUM,B
          IF(GLINT.EQ.1.OR.N.LE.1) GO TO 37
          GO TO 39
C
23        NLB=-3
          XBAR=0.
          YBAR=0.
          SUM=0.
          STD=0.
          N=0
C
31        NLB=NLB+4
          IF(NLB.GT.NLOBE) GO TO 30
          READ(IOL'NLB+1) (SLOBE(1,J),J=1,L)
          IF(SLOBE(1,L).EQ.0) GO TO 31
          DO 32 K=2,4
32        READ(IOL'NLB+K) (SLOBE(K,J),J=1,L)
C
          N=N+1
          DO 33 J=1,4
33        SIG(J,N)=SLOBE(J,L)
C   COMPUTE  E[sigma]
          SUM=SUM+SIG(1,N)
C   COMPUTE  SUM(sigma*greek-X)
          XBAR=XBAR+SIG(1,N)*SIG(2,N)
C   COMPUTE  SUM(sigma*greek-Y)
          YBAR=YBAR+SIG(1,N)*SIG(3,N)
          GO TO 31
C
30        IF(N.EQ.0) WRITE(IOT,20) N
          IF(N.EQ.0) GO TO 37
C   COMPUTE  greek-Xbar
          XBAR=XBAR/SUM
C   COMPUTE  greek-Ybar
          YBAR=YBAR/SUM
C
          STD=0.
          IF(GLINT.EQ.1.OR.N.EQ.1) WRITE(IOT,20) N,XBAR,YBAR
          IF(GLINT.EQ.1.OR.N.EQ.1) GO TO 37
C
          B(1,1)=0.
          B(1,2)=0.
          B(2,1)=0.
          B(2,2)=0.
C
          N1=N-1
          DO 34 I=1,N1
          I1=I+1
```

```
      DO 34 J=I1,N
C
      T1=SIG(1,I)*SIG(1,J)
      T2=SIG(2,I)+SIG(2,J)-2.*XBAR
      T3=SIG(3,I)+SIG(3,J)-2.*YBAR
C
      B(1,1)=B(1,1)+T1*T2*T2
      B(1,2)=B(1,2)+T1*T2*T3
      B(2,1)=B(1,2)
      B(2,2)=B(2,2)+T1*T3*T3
34    CONTINUE
      WRITE(IOT,20) N,XBAR,YBAR,SUM,B
20    FORMAT(A2,7A4)
C
39    A1=COS(ALP)
      A2=SIN(ALP)
C
C   COMPUTE GLINT STANDARD DEVIATION
C
      BB=B(1,1)*A1*A1+B(2,1)*A2*A1+B(1,2)*A1*A2+B(2,2)*A2*A2
      STD=SQRT(BB)/(2.*SUM)
      IF(LBUGS.NE.0) WRITE(2,226) N,SUM,STD,XBAR,YBAR,ANG,B
226   FORMAT('ON,SUM,STD=',I3,2E11.3/
      1 ' XBAR,YBAR,ANG=',3E11.3/
      1 ' B=',4E11.3)
C
37    ND(L,1)=ANG
      ND(L,3)=STD
      IF(ALP.EQ.90.) GO TO 36
C
      ND(L,2)=XBAR
      GO TO 35
C
36    ND(L,2)=YBAR
C
35    ANG=ANG+THETA(3)
C
      IF(IOUT.EQ.1) GO TO 51
C
C   PRINT GLINT DISPLACEMENT VECTOR
C
      WRITE(IO,53) ALP
53    FORMAT('ORCS GLINT OUTPUT     ALPHA=',F6.1/
      1 ' ANGLE  DISPLACEMENT  DIS-S     DIS+S'/)
      DO 54 LP=1,NRCS
      T1=ND(LP,2)-ND(LP,3)
      T2=ND(LP,2)+ND(LP,3)
54    WRITE(IO,55) ND(LP,1),ND(LP,2),T1,T2
55    FORMAT(F6.1,F11.3,F10.3,F10.3)
      GO TO 52
C
C   GRAPH GLINT DISPLACEMENT VECTOR
C
51    IF(IZETA.EQ.0) CALL GRMODE('ERASE',3)
C
      CALL GRPLOT(GSIZE(3),GSIZE(2),0)
      CALL GRPLOT(GSIZE(1),GSIZE(2),1)
      DAX=GSIZE(3)-GSIZE(1)
      CALL GRAXIS(0,GSIZE(1),GSIZE(2)-.95,DAX,0.,SCALE(1),SCALE(3),
      1 IFIX(SCALE(5)),IFIX(SCALE(6)),'OBSERVATION ANGLE',99,1)
```

```
C
      DAY=GSIZE(4)-GSIZE(2)
      CALL GRAXIS(3,GSIZE(1),GSIZE(2),DAY,90.,-360.,360.,5,2,
     1 'DISPLACEMENT (DEG)',99,2)
C
      IF(IZETA.EQ.0) CALL GRMODE('ALPHA',2)
      CALL GRTXT(BUF,NB)
      CALL GRPRNT(0,5.5,-1,7.,.15,0.)
      CALL GRTXT('MONOSTATIC GLINT FOR ALPHA=',0)
      CALL GRNUM('F',-1,1,ALP)
      CALL GRPRNT(0,5.5,-1,6.75,.15,0.)
      IF(IZETA.EQ.0) CALL GRMODE('ALPHA',3)
C
C  DISPLAY RADAR PARAMETERS
C
      X=9.0
      Y=7.
      DY=.2
      CALL GRTXT('FREQUENCY ',30)
      CALL GRNUM('F',-1,2,FREQ)
      CALL GRTXT(' GHZ',4)
      CALL GRPRNT(-1,X,0,Y,.12,0.)
      Y=Y-DY
      CALL GRTXT('POLARIZATION  ',50)
      CALL GRTXT(POL,3)
      CALL GRPRNT(-1,X,0,Y,.12,0.)
      Y=Y-DY
      CALL GRTXT('% SHADOWED ',20)
      CALL GRNUM('F',-1,2,PERSH)
      CALL GRPRNT(-1,X,0,Y,.12,0.)
      Y=Y-DY
      CALL GRTXT('PITCH ',20)
      CALL GRNUM('F',-1,2,PITCH)
      CALL GRPRNT(-1,X,0,Y,.12,0.)
      Y=Y-DY
      CALL GRTXT('ROLL ',20)
      CALL GRNUM('F',-1,2,ROLL)
      CALL GRPRNT(-1,X,0,Y,.12,0.)
      Y=Y-DY
      CALL GRTXT('YAW ',20)
      CALL GRNUM('F',-1,2,YAW)
      CALL GRPRNT(-1,X,0,Y,.12,0.)
      Y=Y-DY
      CALL GRTXT('TILT ',20)
      CALL GRNUM('F',-1,2,TILT)
      CALL GRPRNT(-1,X,0,Y,.12,0.)
      Y=Y-DY
      CALL GRTXT('PHIR ',20)
      CALL GRNUM('F',-1,2,PHR)
      CALL GRPRNT(-1,X,0,Y,.12,0.)
C
C  DRAW AVERAGE DISPLACEMENT CURVE
      CALL GRPIC(GSIZE(1),GSIZE(2),GSIZE(3),GSIZE(4))
      CALL GRSCL(SCALE(1),-360.,SCALE(3),360.)
C     CALL GRVEC(-NRCS,ND(1,1),ND(1,2),DUM,0,DUM,DUM,DUM)
C
      CALL GRMOVE(ND(1,1),DID(ND(1,2)))
      DO 70 JRCS=2,NRCS
   70 CALL GRDRAW(ND(JRCS,1),DID(ND(JRCS,2)))
C
```

```
          IF(GLINT.EQ.1) GO TO 52
C
C  GRAPH DISPLACEMENT +/- STD
          CALL GRMOVE(ND(1,1),DID(ND(1,2)+ND(1,3)))
          DO 49 J=2,NRCS
49        CALL GRDRAW(ND(J,1),DID(ND(J,2)+ND(J,3)))
          CALL GRMOVE(ND(1,1),DID(ND(1,2)-ND(1,3)))
          DO 59 J=2,NRCS
59        CALL GRDRAW(ND(J,1),DID(ND(J,2)-ND(J,3)))
          CALL GRUPPN
          CALL GRRES(SAVE)
C
C
C                 COMPUTE GLINT LOBE IV
C
C
C   REPEAT STD CALCULATIONS FOR ELLIPSES
C
C   LOOP ON DELTA INCREMENTS
52        ANG=FIRSTA
          DO 45 LL=1,NANG
C
C  DETERMINE NEXT INDEX INTO LOBE ARRAY
          IF(ANG.GT.THETA(2)) GO TO 38
          L=IFIX(SLOPE*ANG+CONST+.001)
C
          NLB=-3
          XBAR=0.
          YBAR=0.
          SUM=0.
          STD=0.
          N=0
C
41        NLB=NLB+4
          IF(NLB.GT.NLOBE) GO TO 40
          READ(IOL'NLB+1) (SLOBE(1,J),J=1,L)
          IF(SLOBE(1,L).EQ.0) GO TO 41
          DO 42 K=2,4
42        READ(IOL'NLB+K) (SLOBE(K,J),J=1,L)
C
          N=N+1
          DO 43 J=1,4
43        SIG(J,N)=SLOBE(J,L)
C  COMPUTE  E[sigma]
          SUM=SUM+SIG(1,N)
C  COMPUTE  SUM(sigma*greek-X)
          XBAR=XBAR+SIG(1,N)*SIG(2,N)
C  COMPUTE  SUM(sigma*greek-Y)
          YBAR=YBAR+SIG(1,N)*SIG(3,N)
          GO TO 41
C
40        IF(N.LE.1) GO TO 45
C  COMPUTE  greek-Xbar
          XBAR=XBAR/SUM
C  COMPUTE  greek-Ybar
          YBAR=YBAR/SUM
C
          DO 48 MM=1,2
          DO 48 NN=1,2
          B(MM,NN)=0.
```

```
           C(MM,NN)=0.
           DO 48 RR=1,2
           DO 48 SS=1,2
    48     D(MM,NN,RR,SS)=0.
     C
           N1=N-1
           DO 44 I=1,N1
           I1=I+1
           DO 44 J=I1,N
     C
           T1=SIG(1,I)*SIG(1,J)
           T2=SIG(2,I)+SIG(2,J)-2.*XBAR
           T3=SIG(3,I)+SIG(3,J)-2.*YBAR
           T4=((SIG(4,I)-SIG(4,J))*CSBET1+(SIG(2,I)-SIG(2,J))*SNBET)**2
           T5=(SIG(2,I)-SIG(2,J))*CSBET1-(SIG(4,I)-SIG(4,J))*SNBET
           T6=(SIG(3,I)-SIG(3,J))*CSBET1
     C
           TMPARR(1,1)=T2*T2
           TMPARR(1,2)=T2*T3
           TMPARR(2,1)=TMPARR(1,2)
           TMPARR(2,2)=T3*T3
           DO 44 MM=1,2
           DO 44 NN=1,2
           TEMP=T1*TMPARR(MM,NN)
           B(MM,NN)=B(MM,NN)+TEMP
           C(MM,NN)=C(MM,NN)+TEMP*T4
     C
           D(MM,NN,1,1)=D(MM,NN,1,1)+TEMP*T5*T5
           D(MM,NN,1,2)=D(MM,NN,1,2)+TEMP*T5*T6
           D(MM,NN,2,1)=D(MM,NN,1,2)
           D(MM,NN,2,2)=D(MM,NN,2,2)+TEMP*T6*T6
    44     CONTINUE
     C
           A1=COS(ALP)
           A2=SIN(ALP)
     C
     C   COMPUTE GLINT STANDARD DEVIATION
           BB=B(1,1)*A1*A1+B(2,1)*A2*A1+B(1,2)*A1*A2+B(2,2)*A2*A2
           STD=SQRT(BB)/(2.*SUM)
           CC=C(1,1)*A1*A1+C(2,1)*A2*A1+C(1,2)*A1*A2+C(2,2)*A2*A2
     C
     C   COMPUTE N(delta,f)
     C
           NF=SQRT(CC)/(2.*STD*SOL)
           NF=NF*1.E9
     C
           IF(LBUGS.NE.0) WRITE(2,220) N,SUM,STD,XBAR,YBAR,B,C,D
    220    FORMAT('0N,SUM,STD=',I3,2E11.3/
           1 ' XBAR,YBAR=',2E11.3/
           1 ' B=',4E11.3/
           1 ' C=',4E11.3/
           1 4(' D=',4E11.3/))
     C
     C
     C              COMPUTE GLINT LOBE V
     C
     C
           AMN(1)=A1
           AMN(2)=A2
           DELT=360./DELTA(4)
```

```
              DEL=0.
              IDEL=1
      C
      60      IF(DEL.GT.359.9) GO TO 61
              ND(IDEL,1)=DEL
              DRS(1)=SIN(DEL*RAD)
              DRS(2)=-COS(DEL*RAD)
      C
              DD=0.
              DO 69 MM=1,2
              DO 69 NN=1,2
              DO 69 RR=1,2
              DO 69 SS=1,2
      69      DD=DD+D(MM,NN,RR,SS)*AMN(MM)*AMN(NN)*DRS(RR)*DRS(SS)
      C
              ND(IDEL,2)=SQRT(DD)/(2.*WLEN*STD)
      C
              DEL=DEL+DELT
              IDEL=IDEL+1
              GO TO 60
      C
      61      ND(IDEL,1)=360.
              ND(IDEL,2)=ND(1,2)
      C
              DO 68 J=1,IDEL
      68      ND(J,1)=AMOD(ND(J,1)+90.,360.)
      C
              IF(IOUT.EQ.1) GO TO 65
      C
      C   PRINT NF AND ND
      C
              WRITE(IO,62) ANG,NF,ALP
      62      FORMAT('ORCS GLINT OUTPUT',5X,'THETA=',F6.1,'   NF(GHZ)=',F8.3/
           $   5X,'DELTA',5X,'ND',5X,'ALPHA=',F6.1/)
      C
              DO 63 J=1,IDEL
              TEMP=1./(ND(J,2)*RAD)
      63      WRITE(IO,64) ND(J,1),ND(J,2),TEMP
      64      FORMAT(1X,F9.2,2F10.3)
              GO TO 45
      C
      C   GRAPH NF AND 1/ND
      C
      65      CONTINUE
              FMAX=0.
              FMIN=1000.
              DO 66 J=1,IDEL
              ND(J,2)=1./(ND(J,2)*RAD)
              FMAX=AMAX1(FMAX,ND(J,2))
              IF(FMAX.EQ.ND(J,2)) FANG=ND(J,1)
              FMIN=AMIN1(FMIN,ND(J,2))
      C       ND(J,2)=ND(J,2)*.1
      66      CONTINUE
              IF(FANG.GE.180.) FANG=FANG-180.
              IF(FANG.GT.90.) FANG=FANG-180.
      C
              CALL GRRXY(ANG,0.,X,Y)
              Y=GSIZE(2)-.95
              CALL GRINIT(1,0,0,IER)
              IF(LL.NE.1) GO TO 67
```

```
      C
                CALL GRTXT('NF(GHZ)',0)
                CALL GRPRNT(+1,2.1,-1,Y+.53,.1,0.)
                CALL GRTXT('MAJOR',0)
                CALL GRPRNT(+1,2.1,-1,Y+.38,.1,0.)
                CALL GRTXT('MINOR',0)
                CALL GRPRNT(+1,2.1,-1,Y+.23,.1,0.)
                CALL GRTXT('ANGLE',0)
                CALL GRPRNT(+1,2.1,-1,Y+.08,.1,0.)
      C
      67        CALL GRNUM('F',-1,3,NF)
                CALL GRPRNT(0,X,-1,Y+.53,.10,0.)
                CALL GRNUM('F',-1,2,FMAX)
                CALL GRPRNT(0,X,-1,Y+.38,.1,0.)
                CALL GRNUM('F',-1,2,FMIN)
                CALL GRPRNT(0,X,-1,Y+.23,.1,0.)
                CALL GRNUM('F',-1,2,FANG)
                CALL GRPRNT(0,X,-1,Y+.08,.1,0.)
      C
                CALL GRCP('POLAR')
                Y=GSIZE(2)
                CALL GRPIC(X,Y,X+1.,Y+1.)
                CALL GRSCL(0.,0.,(SCALE(3)-SCALE(1))/6.5,360.)
                CALL GRPIC(X-1.,Y-.28,X+1.,Y+.5)
                CALL GRWIND(-IDEL,ND(1,2),ND(1,1),IDUM,0,DUM,DUM,IDUM)
                CALL GRUPPN
                CALL GRRES(SAVE)
      C
      C
      45        ANG=ANG+DELTA(3)
      38        CONTINUE
                CLOSE(UNIT=IOL)
                CLOSE(UNIT=IOT)
                CALL EXIT
                END
```

APPENDIX

A PLOTTING COORDINATE SYSTEM FOR THE APL RCS/STATISTICS CODE

Background

The following remarks are applicable to the APL RCS/Statistics Code described in this document. In consequence of the code's sequential development over a rather long time, several peculiarities exist. Because they are not explicitly pointed out in the body of the text, their significance may be overlooked, particularly in regard to recent demands on the code that were not anticipated during its development.

The pecularities have mainly to do with coordinate systems. In early work, a master coordinate system (using unprimed variables) was chosen arbitrarily with the z axis pointing forward, the x axis to port, and the y axis up. That master system was connected to the component coordinate systems by transformations called TRANS in the code.

Since a comparison of theoretical calculations with experimental data is always desirable, and since the RATSCAT facility is perhaps the chief supplier of such data, it became advisable to introduce the RATSCAT standard into our master system. RATSCAT uses the convention z axis forward, x axis up, and y axis pointing to starboard. Rather than change the code structure, the RATSCAT master system was connected to our master system by a simple linear transformation. The shadowing of one component by another was now defined in the RATSCAT system (the instructions are called VALID in the code). Since the target aspect modifications of pitch, roll, yaw, and tilt are also available in the RATSCAT system it was decided to introduce these transformations into the code.

Statistical information about the lobing structure of an RCS pattern can be regained provided that the component scatterer locations can be specified. When this development was completed, the locations were specified by giving the position of the origin of each component system in the RATSCAT system (this instruction is called ORGN in the code).

Now we have the requirement to describe RCS over an arbitrary planar cut or over a conical cut about an arbitrary cone axis.

Modifications

Because shadowing and component origins are specified in the RATSCAT, or radar-fixed, coordinate system, the pitch, roll, yaw, and tilt transformations are disallowed for other than small angular excursions. But they are exactly the transformations one would like to use to satisfy the requirement for arbitrary planar and conical cuts. The best way to satisfy the requirement is to eliminate the original pitch, roll, yaw, and tilt capability and replace it with a new system called PLOTTING. The target-fixed system stands fast while the plotting system axes (originally coincident with the radar-fixed axes) are "pitched," "rolled," or "yawed" to a new position. Thus, the original terminology is retained but with the meaning just defined. (Actually, one can think of the target itself being pitched, rolled, or yawed, but in an angular direction opposite to that of the axes movement.)

In addition to this change, the original unprimed coordinate system is eliminated. Henceforth, TRANS expresses a transformation from the primed to the RATSCAT system (in a manner exactly analogous to the way TRANS previously expressed a transformation from the primed to the unprimed system).

Transformations

There are, therefore, three systems to deal with: the plotting or display system, *D*; the target-fixed or RATSCAT system, *R*; and the primed system of a component, *P*.

D is connected with *R* by a linear transformation, *C*:

$$\hat{x}_D = c_{11}\hat{x}_R + c_{12}\hat{y}_R + c_{13}\hat{z}_R$$

$$\hat{y}_D = c_{21}\hat{x}_R + c_{22}\hat{y}_R + c_{23}\hat{z}_R$$

$$\hat{z}_D = c_{31}\hat{x}_R + c_{32}\hat{y}_R + c_{33}\hat{z}_R$$

where the caret denotes unit vector.

The user specifies a region given by angular intervals for the display polar and azimuth angles θ_D and ϕ_D, respectively. For them, he must calculate the corresponding angles in the R system (where shadowing is defined) as follows:

$$\cos\theta_R = c_{13}\sin\theta_D\cos\phi_D + c_{23}\sin\theta_D\sin\phi_D + c_{33}\cos\theta_D$$

$$\cos\phi_R = \frac{c_{11}\sin\theta_D\cos\phi_D + c_{21}\sin\theta_D\sin\phi_D + c_{31}\cos\theta_D}{\sin\theta_R}$$

(note the subscript R in the denominator above)

$$\sin\phi_R = \frac{c_{12}\sin\theta_D\cos\phi_D + c_{22}\sin\theta_D\sin\phi_D + c_{32}\cos\theta_D}{\sin\theta_R}$$

(and here too).

If

$$c_{11}\sin\theta_D\cos\phi_D + c_{21}\sin\theta_D\sin\phi_D + c_{31}\cos\theta_D = 0,$$

then $\phi_R = \pm\pi/2$ when

$$c_{12}\sin\theta_D\cos\phi_D + c_{22}\sin\theta_D\sin\phi_D + c_{32}\cos\theta_D$$

is greater than or less than zero, respectively.

If

$$c_{12}\sin\theta_D\cos\phi_D + c_{22}\sin\theta_D\sin\phi_D + c_{32}\cos\theta_D = 0,$$

then $\phi_R = 0$ or π when

$$c_{11}\sin\theta_D\cos\phi_D + c_{21}\sin\theta_D\sin\phi_D + c_{31}\cos\theta_D$$

is greater than or less than zero, respectively.

If $\theta_R = 0$, then ϕ_R is indeterminate because the direction of interest is identical with $\hat{z}_R$.

As was indicated above, these formulas represent the same process used previously to connect the P and U systems (in the same order as R and D are connected above). It is now identical with the process that, henceforth, connects the P and R systems. Furthermore, since the U system is now eliminated, the transformation.

$$B = \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

must also be eliminated.

To recapitulate, in the revised arrangement there are only the three coordinate systems: the display system, D; the target-fixed, or RATSCAT, system, R; and the primed, or component-fixed, system, P. Henceforth, the three instructions TRANS, VALID, and ORGN will be referred to the R system.

Pitch, Roll, Yaw

The linear transformation C can be expressed as any one of, or a combination of, pitch, roll, or yaw, denoted by matrixes P, R, and Y, respectively. Taking the original position of D to be coincident with R and moving D while keeping R fixed, the angles are defined as follows:

Pitch through an angle p is a rotation of D about $\hat{y}_R$, positive from $\hat{z}_R$ toward $\hat{x}_R$. C takes the special form P, where

$$P = \begin{bmatrix} \cos p & 0 & -\sin p \\ 0 & 1 & 0 \\ \sin p & 0 & \cos p \end{bmatrix}.$$

Roll through an angle r is a rotation of D about $\hat{z}_R$, positive from $\hat{x}_R$ toward $\hat{y}_R$. C takes the special form R where

$$R = \begin{bmatrix} \cos r & \sin r & 0 \\ -\sin r & \cos r & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

Yaw through an angle y is a rotation of D about $\hat{x}_R$, positive from $\hat{y}_R$ toward $\hat{z}_R$. C takes the special form Y, where

$$Y = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos y & \sin y \\ 0 & -\sin y & \cos y \end{bmatrix}.$$

When applied successively, P, R, and Y each represents, in some order (these transformations are *not* commutative), a rotation of the coordinate system from its *last* position to a *new* position. The complete transformation C, equal to a product of P, R, and Y in some order, represents a transformation from the R system to the D system.

Examples

Consider a 90° pitch of axes, followed by a 180° roll, with zero yaw. Then

$$C = RP = \begin{bmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 & 0 & -1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 \\ 0 & -1 & 0 \\ 1 & 0 & 0 \end{bmatrix}.$$

Note that

$$PR = \begin{bmatrix} 0 & 0 & -1 \\ 0 & -1 & 0 \\ -1 & 0 & 0 \end{bmatrix} \neq RP.$$

For example, with the conditions $\theta = 60°$, $0° \leq \phi \leq 180°$, the transformation RP produces a portside viewing, from front to rear, over a conical cut at a 30° elevation above the horizontal plane. In contrast, the transformation PR produces a portside viewing, from rear to front, over a conical cut at a 30° declination below the horizontal plane.

REFERENCES

[1]J. B. Keller, "Geometrical Theory of Diffraction," *J. Opt. Soc. Am.*, **52**, 116-130 (1962).

[2]P. Ya. Ufrimtsev, "Method of Edge Waves in the Physical Theory of Diffraction," *Izd-Vo. Sov. Radio*, 1-243 (1962); translation by U.S. Air Force Foreign Technology Division (1971).

[3]E. F. Knott and T. B. A. Senior, "Comparison of Three High-Frequency Diffraction Techniques," *Proc. IEEE* **62**, 1468-1474 (1974).

[4]G. J. Burke and A. J. Poggio, *Numerical Electromagnetic Code—Method of Moments*, Naval Electronics Systems Command TD 116 (1977).

[5]J. W. Crispin, Jr., and A. L. Maffett, "Estimation of RCS for Simple Shapes," "Estimation of RCS for Complex Shapes," *Proc. IEEE* **53**, (1965).

[6]J. W. Follin, Jr., *Statistical Properties of RCS*, JHU/APL QM-81-115 (1981).

[7]J. W. Follin, Jr., and A. L. Maffett, "RCS Scintillations and Their Statistical Description," in *Proc. Second Annual Tactical Air Surveillance Control Conf.*, Rome Air Development Center (1981).

[8]J. W. Follin, Jr., F. C. Paddison, and A. L. Maffett, "The RCS of Two Cruise Missiles and Their Statistical Description," DARPA 8th Strategic Space Symposium (1982).

[9]E. Shotland, *Statistical Analysis of Radar Target Scintillation*, JHU/APL TG 955 (1967).

[10]S. O. Rice, "Mathematical Analysis of Random Noise," in *Selected Papers on Noise and Stochastic Processes*, Nelson Wax, ed., Dover Publications, New York (1954).

[11]A. L. Maffett, *Bistatic Lobe Statistics*, JHU/APL M80-7 (Nov 1980).

[12]A. L. Maffett, *Linear Transformations Relating Theoretical and Experimental Coordinate Systems Used in RCS Determination*, JHU/APL QM-80-049 (M80-4) (Feb 1980).

[13]A. L. Maffett, *Component Scattering Centers*, JHU/APL M81-1 (7 Jan 1981).

[14]J. W. Crispin and K. M. Siegel, *Methods of Radar Cross Section Analysis*, Academic Press, New York (1968).

[15]A. L. Maffett, *Geometric and Physical Optics Formulas and Orthonormal Transformations for an RCS Computer Program*, JHU/APL M80-3 (28 Jan 1980).

INITIAL DISTRIBUTION EXTERNAL TO THE APPLIED PHYSICS LABORATORY*

The work reported in JHU/APL TG 1345 was done under Navy Contract N00024-83-C-5301 and is related to Task ZT90, which is supported by the U. S. Air Force Rome Air Development Center.

ORGANIZATION	LOCATION	ATTENTION	No. of Copies
DEPARTMENT OF DEFENSE			
DTIC	Alexandria, VA	Accessions	12
DEPARTMENT OF THE NAVY			
Defense Advanced Research Projects Agency	Arlington, VA	STO, W. Kurowski	1
Office of Naval Research	Washington, DC	R. W. Klementz	1
NAVSEASYSCOM	Washington, DC	SEA-9961	2
NAVAIRSYSCOM	Washington, DC	AIR-950D	2
NAVPLANTREPO	Laurel, MD		1
DEPARTMENT OF THE AIR FORCE			
Rome Air Development Center	Griffiss AFB, NY	J. H. Michels (OCTM)	1
AD/YITQ	Eglin AFB, FL	B. Colt	1
AFWAC/AAWP, EW Division	Wright-Patterson AFB, OH	W. F. Bahret	1
HQ FTD (SQDA)	Wright-Patterson AFB, OH	V. D. Dykes	1
AFCMO/SAE	Kirtland AFB, NM	I. S. Hodges	1
AFEWC/EAAE	San Antonio, TX	J. Bennett	1
Data Management Branch		T. Spear	1
HQ 6585 Test Group (RX)	Holloman AFB, NM	F. Anburgey	1
		J. F. Lutz	1
		D. Stein	1
CONTRACTORS			
Teledyne Micronetics	San Diego, CA	President	1
Delco Electronics	Santa Barbara, CA	R. Hvamini	1
Martin-Marietta Aerospace Corp.	Orlando, FL	C. W. Maness (MP-200)	1
Hughes Aircraft Co. Space & Communciations Group	El Segundo, CA		1
Boeing Aircraft Co.	Seattle, WA	T. Dalby	1
General Dynamics Convair Division	San Diego, CA	F. W. H. Anding (MS-SS-6130)	1
Decision-Science Applications Inc.	Arlington, VA	J. F. Belyea	1
Dare Technology Inc.	La Jolla, CA		1
Science Applications Inc.	Santa Barbara, CA	G. R. Curry	1
McDonnel Douglas Astronautics	St. Louis, MO	E. Chubb	1
A. L. Maffett	Dexter, MI		1
H. W. Klimach	Canton, MI		1

Requests for copies of this report from DoD activities and contractors should be directed to DTIC, Cameron Station, Alexandria, Virginia 22314 using DTIC Form 1 and, if necessary, DTIC Form 55.

END

FILMED

9-84

DTIC