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

THE JOHNS HOPKINS UNIVERSITY • APPLIED PHYSICS LABORATORY

Approved for public release; distribution unlimited.

84 07 06 014
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.
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

Approved for public release; distribution unlimited.
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

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>List of Illustrations</td>
<td>6</td>
</tr>
<tr>
<td>1.0 Background</td>
<td>7</td>
</tr>
<tr>
<td>2.0 General Discussion of RCS Data</td>
<td>8</td>
</tr>
<tr>
<td>3.0 The Capabilities of the Code from the Viewpoint of the RCS Analyst</td>
<td>8</td>
</tr>
<tr>
<td>3.1 Inputs</td>
<td>10</td>
</tr>
<tr>
<td>3.2 Outputs</td>
<td>10</td>
</tr>
<tr>
<td>3.3 Glint</td>
<td>11</td>
</tr>
<tr>
<td>4.0 Calculations</td>
<td>12</td>
</tr>
<tr>
<td>4.1 Relative-Phase RCS</td>
<td>12</td>
</tr>
<tr>
<td>4.2 Random-Phase RCS</td>
<td>12</td>
</tr>
<tr>
<td>4.3 Higher-Order Statistics</td>
<td>12</td>
</tr>
<tr>
<td>4.4 Modified Statistical Distribution</td>
<td>14</td>
</tr>
<tr>
<td>4.5 RCS Lobe Structure</td>
<td>14</td>
</tr>
<tr>
<td>4.6 Glint</td>
<td>14</td>
</tr>
<tr>
<td>5.0 User's Guide to the Code from the Viewpoint of the Computer Analyst</td>
<td>16</td>
</tr>
<tr>
<td>5.1 Introduction</td>
<td>16</td>
</tr>
<tr>
<td>5.2 RCS Program Operations</td>
<td>16</td>
</tr>
<tr>
<td>5.3 Display Options</td>
<td>18</td>
</tr>
<tr>
<td>5.4 Data Base Description</td>
<td>19</td>
</tr>
<tr>
<td>5.5 Data Base Directives</td>
<td>20</td>
</tr>
<tr>
<td>6.0 Program Listing</td>
<td>24</td>
</tr>
<tr>
<td>Appendix, A Plotting Coordinate System for the APL RCS/Statistics Code</td>
<td>75</td>
</tr>
<tr>
<td>References</td>
<td>78</td>
</tr>
</tbody>
</table>
ILLUSTRATIONS

1 Example of calculated monostatic relative-phase RCS; 1/25th scale Convair 990. ........................................... 9

2 Example of calculated monostatic mean RCS with rms spread (including lobe statistics); 1/25th scale Convair 990. ................. 9

3 Calculated width-skewness statistics for the RCS scintillation of a 1/25th scale Convair 990. ............................................. 10

4 Calculated width-kurtosis statistics for the RCS scintillation of a 1/25th scale Convair 990. ............................................. 11
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 monostatic 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).

---

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 \( \lambda/L \) apart in angle and \( c/L \) apart in frequency (where \( \lambda \) 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 \( 10^8 \) 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). 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.

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.6,4 (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 horizontal plane (minor) and the normal plane (major). “Angle” refers to the tilt of the lobe ellipse with the observation axis. N, (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 re-code the program for a more capable computer in the.

---

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 ab-
sorbing 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 scat-
tering for either radar absorbing material or dielec-
tric 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 examina-
tion of RCS offers a condensation of facts and, thus,
an economy of output not available with a determi-
nistic 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 deter-
ministic 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 sys-

3.1 Inputs

The inputs usually required for the code are wave-
length; polarization (linear, but extendable to ellip-
tical); 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 mathemat-
cal 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' con-
ceptions. 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.
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 $100^\circ$ from nose-on produce, under the random-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 $100^\circ$ 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 $\sigma/\sigma_{\text{max}}$, where $\sigma$ is the RCS for some fixed aspect, frequency, and polarization. So, for example, since $\sigma_{\text{max}} (100^\circ) = 1.14$ m$^2$ and since the mean of $\beta(x; a, b)$ is $a/(a + b) = 0.2$, the mean RCS at $100^\circ$ 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 $100^\circ$, 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 $2\pi$) 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 \( \lambda \). 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 \( i \)th component be denoted by

\[
a_i \exp(j\psi_i), \quad j = \sqrt{-1},
\]

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

\[
A = \sum_{i=1}^{N} \sqrt{\sigma_i} \exp(j\psi_i).
\]

The relative-phase RCS, \( \sigma \), for the complex target is

\[
\sigma = AA^* = \left| \sum_{i=1}^{N} \sqrt{\sigma_i} \exp(j\psi_i) \right|^2 = \sum_{i<k} \sqrt{\sigma_i} \sqrt{\sigma_k} \exp(j\psi_i) \exp(-j\psi_k).
\]

4.2 Random-Phase RCS

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

\[
\sigma_i = E[\sigma_i] = \langle AA^* \rangle = \sum a_i,
\]

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

\[
F = AA^* - E[\sigma] = 2 \sum_{i<k} a_i a_k \cos(\psi_i - \psi_k)
\]

whence the variance is

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

\[
= \left( \sum_{i=1}^{N} \sigma_i \right)^2 - \sum_{i=1}^{N} \sigma_i^2.
\]

4.3 Higher-Order Statistics

If the triple and quadruple sums are written as

\[
S_3 = \sum_{i<k<l} \sigma_i \sigma_k \sigma_l
\]

and

\[
S_4 = \sum_{i<k<l<m} \sigma_i \sigma_k \sigma_l \sigma_m,
\]

the third and fourth central moments are, respectively,

\[
\mu_3 = \langle F^3 \rangle = 12S_3
\]

and

\[
\mu_4 = \langle F^4 \rangle = 6S_2^2 + 12S_1S_3 + 132S_4.
\]
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

\[
w = \mu_2 / \nu_1^2.
\]

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

\[
\gamma'_i = \mu_i / \mu_2^2.
\]

Kurtosis is defined in the standard manner as

\[
\gamma_2 = \mu_4 / \mu_2^2 - 3.
\]

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:

\[
\gamma'_i = 4w - 2, \quad \frac{1}{2} \leq w \leq 1;
\]

Lower boundary:

\[
\gamma'_i = 0, \quad 0 \leq w \leq \frac{1}{2};
\]

Left boundary:

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

\[
\gamma'_i = \frac{2t(t + 3)}{(t + 1)(t + 2)},
\]

\[
0 \leq t \leq \infty.
\]

### 4.3.2 Width-Kurtosis

Right boundary:

\[
\gamma_2 = \frac{6n^2}{(n + 1)(n + 2)} (w + 1 / n^2 - 3 - 1 / n^2),
\]

where

\[
n = \frac{\gamma'_i + 2}{2w - \gamma'_i} \quad \text{and} \quad \gamma'_i \text{ is as above},
\]

\[
1 / 2 \leq w \leq 1, \quad \lim_{w \to 1} \gamma_2 = 6;
\]

Lower boundary:

\[
\gamma_2 = -1.5, \quad 0 \leq w \leq \frac{1}{2};
\]

Left boundary:

\[
\text{part 1: } w = 0, -1.5 \leq \gamma_2 \leq 0;
\]

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

\[
\gamma_2 = \frac{6t(t + 4)}{(t + 2)^2}.
\]

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

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

\[
0 \leq x \leq 1,
\]

where \( \Gamma(a) \) is the gamma function of argument \( a \), and \( x = a / a_{\text{max}} \). 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

\[
w = \frac{b}{a(a + b + 1)}
\]

\[
\gamma_1 = \frac{2(b - a)}{a(a + b + 2)}
\]

\[
\gamma_2 = \frac{6(a + b + 1)}{(a + b + 2)(a + b + 3)} \times \left( \frac{a}{b} \frac{b}{a + b + 1} - \frac{1}{a + b + 1} - 3 \right).
\]
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 \(c\) is the speed of light and \(S = \frac{\lambda}{c}\), 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 \(\delta\) with the \(\xi\) axis in the \(\xi\) plane:

\[
N_c = \frac{1}{\lambda S} \left( \sum A_{r.s} \right), \quad r,s = 1,2. \tag{24}
\]

where \(c\) is the speed of light and \(S = \frac{\lambda}{\mu}\) is from Eq. 10. The results are derived from Eq. 6. 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

\[
\frac{\partial \psi}{\partial X} = \frac{\partial}{\partial x} \left( kr \right) = \frac{k(X - \xi_x)}{r} = k \frac{X - \xi_x}{r}, \tag{27}
\]
and
\[
\frac{\partial \psi_i}{\partial Z} = \frac{k(Z - \xi_i)}{r_i} = k. \tag{28}
\]
The apparent source location is
\[
\xi = r \tan^{-1} \left( \frac{\partial A/\partial X}{\partial A/\partial Z} \right)
\]
\[
= \frac{\sum ika, \xi_i \exp(j\psi_i)}{\sum ika \exp(j\psi_i)} \exp(j\psi_i)
\]
\[
= \frac{\sum a_i a_j \xi_i \cos(\psi_i - \psi_j)}{\sum a_i a_j \cos(\psi_i - \psi_j)} \tag{29}
\]

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 division 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
\[
\xi = \frac{\sum a_i \xi_i}{\sum a_j}. \tag{30}
\]

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.
\[
\Delta \xi = \frac{1}{E} \sum_{i<j} a_i a_j (\xi_i - \xi_j) \cos(\psi_i - \psi_j)
\]
\[
= \frac{1}{E} \sum_{i<j} a_i a_j (\xi_i + \xi_j - 2\xi) \cos(\psi_i - \psi_j), \tag{31}
\]
and the variance of the glint is
\[
\langle \Delta \xi^2 \rangle = \frac{1}{2E^2} \sum_{i<j} a_i a_j (\xi_i + \xi_j - 2\xi)^2. \tag{32}
\]

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 \( \eta \). In addition, the variance is a tensor operator so that an off-diagonal term is needed.

Defining
\[
B_{mn} = \sum_{i<j} a_i a_j (\xi_i + \xi_j - 2\xi)^{m-n}
\]
\[
\times (\eta_i + \eta_j - 2\eta)^{m+n-2}, \quad m, n = 1, 2 \tag{33}
\]
and
\[
B = \sum_{m,n} B_{mn} a_m a_n, \tag{34}
\]

where \( a_i = \cos \alpha, a_j = \sin \alpha \), the standard deviation of glint in the \( \alpha \) direction is
\[
S_\alpha = \frac{1}{2E(\alpha)} \sqrt{\text{B}}. \tag{35}
\]

Further in analogy to the derivation of Eqs. 23 and 26, we define
\[
C_{mn} = \sum_{i<j} a_i a_j (\xi_i + \xi_j - 2\xi)^{m-n}
\]
\[
\times (\eta_i + \eta_j - 2\eta)^{m+n-2}
\]
\[
\times [(\xi_i - \xi_j)(1 + \cos \beta) + (\xi_i - \xi_j) \sin \beta]^2, \tag{36}
\]

\[
D_{mn,n} = \sum_{i<j} a_i a_j (\xi_i + \xi_j - 2\xi)^{m-n}
\]
\[
\times (\eta_i + \eta_j - 2\eta)^{m+n-2}
\]
\[
\times [(\xi_i - \xi_j)(1 + \cos \beta) - (\xi_i - \xi_j) \sin \beta]^{m+n-2}, \tag{37}
\]
\[
C = \sum_{m,n} C_{mn} a_m a_n, \tag{38}
\]

15
and

\[ D = \sum_{m,n} \sum_{r,s} D_{mnrs} a_m a_n d_r d_s \]  

(39)

Then the number of glint lobes in the \( \alpha \) direction per radian of rotation about the axis oriented in the \( \delta \) direction is

\[ N_{\alpha,\delta} = \frac{1}{2\pi S_{\alpha}} \sqrt{D}. \]  

(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,
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
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 system 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 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-

---

displayed. 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 supplied 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 \( 1 + \frac{\text{THETA}(2) - \text{THETA}(1)}{\text{THETA}(3)} \). For the first pass through the data base, the observation angle ranges from \( \text{THETA}(1) \) to \( \text{THETA}(2) \) 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, \( \text{THETA}(2) \) is larger than or equal to \( \text{THETA}(4) \), 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, \( \text{THETA}(4) \) is larger than \( \text{THETA}(2) \), the starting value, \( \text{THETA}(1) \), is replaced with \( \text{THETA}(2) \), and the ending value, \( \text{THETA}(2) \), is replaced with the minimum of \( \text{THETA}(2) + \text{NOI} \) and \( \text{THETA}(4) \). This processing loop continues as described while \( \text{THETA}(4) \) remains larger than \( \text{THETA}(2) \). All graphical 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 \textit{n} Comment-ignores the next \textit{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 \textit{n1}, \ldots, \textit{n4} Delta Angles-defines the lobe display angles. RCS lobe and glint lobe results will be displayed at angles ranging from \textit{n1} to \textit{n2} in increments of \textit{n3} degrees. The number of lobe samples, between 4 and 180, is defined by \textit{n4}. Only delta angles that lie within the theta range will be used.

GLINT \textit{n} Glint Switch-activates the glint lobe calculations if \textit{n} is set to 1 or 2 and LOBE is active. Glint data will not be computed otherwise. If \textit{n} is set to 2, the standard deviation will be added to the displacement curve.
GRAPH n1, . . . , n9
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 n1 is equal to 3 instead of 1. If n1 is negative, terminal graphs will be automatically sent to a hard-copy device. The x-axis end points are specified by n2 and n3. The number of x-axis labels and tick marks per label are specified by n4 and n5. The y-axis end points are specified by n6 and n7. The number of y-axis labels and tick marks per label are specified by n8 and n9. The next line following this instruction is always used for the title. This instruction will not affect any RCS results.

NEWWL
New Wavelength – reruns data base with new wavelength. This instruction will cause the RCS program to reexecute 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 nonspoiled 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.

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 reexecute 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).

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.\(^\dagger\)

**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 \(n_1, \ldots, n_4\)**
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 \(n_1, \ldots, n_9\)**
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 \(n_1, \ldots, n_8\)**
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\ldots**
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
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 \(r\).
CONED \(a, n, a, b, L_1, L_2\)  Truncated Elliptical Cone - defines a cone with lengths \(L_1\) and \(L_2\) from the imaginary cone tip. Cone tip to \(L_1\) 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). Parabola - defines a parabola where the parabola equation is \(p = (x^{**2} + y^{**2})/4z\).
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 \(a\).
FILE = SYO:[111, 12] RCS.FTN

### 6.0 PROGRAM LISTING

```plaintext
10  C
20  C
30  C
40  C
50  C
60  C
70  C
80  C
90  C
100 C
110 C
120 C
130 C
140 C
150 C
160 C
170 C
180 C
190 C
200 C
210 C
220 C
230 C
240 C
250 C
260 C
270 C
280 C
290 C
300 C
310 C
320 C
330 C
340 C
350 C
360 C
370 C
380 C
390 C
400 C
410 C
420 C
430 C
440 C
450 C
460 C
470 C
480 C
490 C
500 C
510 C
520 C
530 C
540 C
550 C
560 C
570 C
580 C
590 C
600 C
610 C
620 C
630 C
640 C
650 C
660 C
670 C
```

#### NOTE!

This program stores matrices by rows (not columns)

```plaintext
180 C
190 C
200 C
210 C
220 C
230 C
240 C
250 C
260 C
180 C
200 C
210 C
220 C
230 C
240 C
250 C
260 C
180 C
200 C
210 C
220 C
230 C
240 C
250 C
260 C
```

MAXRCS CONTAINS THE CURRENT MAXIMUM RCS BUFFER SIZE (181)

```plaintext
MAXRCS CONTAINS THE CURRENT MAXIMUM RCS BUFFER SIZE (181)
```
SET RCS CONSTANTS

NCMD=26
NOBJ=16
MAXRCS=181

P1=3.1415926
P2=P1/2.
RAD=P1/180.
CUTOFF=1.E-10
LINE=0
GOULD=.FALSE.

ASSIGN LUNS AND ATTACH INPUT FILE

CALL ASNLUN(IOR,'TI',0)
CALL ASNLUN(IOW,'TI',0)
CALL ASNLUN(IOF,'TI',0)
CALL ASNLUN(IOF,'SY',0)
CALL ASNLUN(IOE,'TI',0)
CALL ASNLUN(IOE,'SY',0)
OPEN=.FALSE.

WRITE(IOW,1)
1 FORMAT(' ENTER RCS DATA FILE NAME'
READ(IOR,2,END=9999) FILE
2 FORMAT(IOAll)
CALL TRIMIO(INF,FILE)
FILE(NF+1)=0

INPUT OPTIONS

1000 C/Z = DRAW GRAPHS ON ZETA PLOTTER
1020 C/F = PRINT RESULTS FOR EACH COMPONENT
1050 C/G = GRAPH RESULTS FOR EACH COMPONENT
1040 C/GO = DRAW ALL /G OUTPUT ON ONE GRAPH (COMBINE)
1080 C/DEB = { 
1085 DEBUG=0
1070 COMBIN=0
1100 IZETA=0
IF=1
1100 IDOT=0
1110 DO 6 J=1,F,NF
1120 JJ=J
1130 IF(FILE(JJ),EQ.DOT) IDOT=JJ
1140 IF(FILE(JJ),EQ.SLASH) GO TO 7
1150 CONTINUE
1160 GO TO 6
1170 NF=JJ=1
1190 IF(FILE(JJ+1),EQ.AZ) IZETA=200
1190 IF(FILE(JJ+1),EQ.AG) DEBUG=1

25
IF(FILE(JJ+1).EQ.AP) DEBUG=2
IF(DEBUG.EQ.1.AND.FILE(JJ+2).EQ.AC) COMBINE=1
IF(JJ+2.
9 IF(DOT.GT.0) GO TO 9
C DEFAULT FILE EXTENSION TO .RCS
DO 13 J=1,4
NF=NF+1
FILE(NF)=DRC(S(J)
CONTINUE
9 FILE(NF+1)=0
OPEN(UNIT:10F,NAME=FILE,TYPE='OLD',READONLY)
C TIME=SECMS(0.0)
CALL GRINIT0,IZETA,IOG,IER)
CALL GRHY('H')
CALL GRPIC(2.25,2.25,8.75,6.25)
IF(IZETA.EQ.0) CALL GRDEV('TEK','HARD',IER)
C INITIALIZE RCS VECTOR & OTHER STUFF
C CMD=0.
IWL=1
OLDSRF=.FALSE.
GO TO 70
C
C READ NEXT LINE IN RCS DATA FILE
C
CONTINUE
C
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=9999,ERR=9993) CMD,ARR
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 TO (10,12,15,20,25,30,35,40,45,50,55,60,70,80,85,95,
500,510,520,540,550,560,570,580,590,600), J
CONTINUE
C
GO TO OBJECT SEARCH IF NO DIRECTIVE MATCH FOUND
GO TO 90
C
C "COMM" - SKIP COMMENT LINES
C
CONTINUE
C
EXECUTE APPROPRIATE DIRECTIVE
C
C
"COMM" - SKIP COMMENT LINES
C
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
12   C "COMS" - SEARCH FOR "COME"
   GO TO 3
13   IF(CMD.EQ.COME) GO TO 3
   GO TO 12
14   C "WLEN" - SET WAVELENGTH
15   WLEN=ARR(IWL)
16   IF(WLEN.LE.0.AND.IWL.EQ.1) GO TO 9991
17   IF(WLEN.LE.0) GO TO 9990
18   GO TO 3
19   C "POLAR" - SET POLARIZATION VECTOR
20   POLAR=AMAX1(1.,AMIN1(2.,ARR(I))))
21   GO TO 3
22   C "TRANS" - COMPUTE COORDINATE TRANSFORMATION MATRIX 1
23   CONTINUE
24   DO 26 J=1,9
25   CALL TRANS(TRANS,TRANS,TRANS,TRANS,TRANS,TRANS,TRANS,TRANS,TRANS)
26   CALL MATMUL(TRANS,TRANS,TRANS,TRANS,TRANS,TRANS,TRANS,TRANS,TRANS)
27   MAKE SURE TRANS IS VALID
28   DO 24 J=1,9
29   IF(ABS(MAT(J,J)-IDENT(J)).GT.1.E-3) GO TO 27
30   CONTINUE
31   GO TO 3
32   WRITE(2,29) ARR
33   GO TO 3
34   CONTINUE
35   IF(ABS(PHI)-180.) WRITE(IOW,36) PHI
36   FORMAT(' PHIR=",F6.1,' ' IS OUT OF BOUNDS')
37   GO TO 3
38   C "THETA" - SET THETA RANGE (MIN,MAX,INC) & TOTAL MAX
39   IF(OLDGRF) GO TO 43
2400> THETA(1)=ARR(1)
2410> IF(ARR(3).LE.0.) GO TO 42
2420> THETA(2)=AMAX1(THETA(1),ARR(2))
2430> THETA(3)=ARR(3)
2440> THETA(4)=ARR(4)
2450> MGRAPH=.FALSE.
2460> IF(THETA(4).GT.THETA(2)) MGRAPH=.TRUE.
2470> C
2480> NRCS=(ARR(2)-ARR(1))/ARR(3)+1.001
2490> IF(NRCS.LE.MAXRCS) GO TO 3
2500> WRITE(IOW,41)
2510> 41 FORMAT(' RCS> -- TOO MANY OUTPUT POINTS')
2520> GO TO 9993
2530> C
2540> 42 THETA(2)=ARR(1)
2550> THETA(3)=1.
2560> NRCS=1
2570> GO TO 3
2580> C
2590> 43 T=THETA(1)
2600> THETA(1)=THETA(2)
2610> THETA(2)=AMIN1(2.*THETA(1)-T,THETA(4))
2620> GO TO 3
2630> C
2640> "GRAPH" - GRAPH RESULTANT RCS VECTOR = CMD(9)
2650> C
2660> 45 CONTINUE
2670> C
2680> IF(STATS.EQ.1) GO TO 4542
2690> IF(NRCS.LE.1.OR.DEBUG.EQ.2) GO TO 3
2700> IF(IZETA.EQ.0.AND.COMIN.LE.0.AND..NOT.OLDGRF)
2710> ATTN=.FALSE.
2720> IF(CMD.EQ.CMD(9)) GO TO 4546
2730> C
2740> GRAPH DEBUG MODE
2750> M60DB=1./10**6
2760> DO 4541 J=1,NRCS
2770> IF(NRCS(J).GT.M60DB) GO TO 4540
2780> CONTINUE
2790> GO TO 4550
2800> C
2810> 4540 CALL TRAPCC(10R,ATTN)
2820> DO 4545 J=1,9
2830> ARR(J)=QRARR(J)
2840> 4545 CALL QRSCL(ARR(2),ARR(6),ARR(3),ARR(7))
2850> IF(INRPR.GT.0.AND.RPRCS.GT.0.)CALL QRSCL(ARR(2),0.,ARR(3),RPRCS)
2860> SCALE(1)=ARR(2)
2870> SCALE(2)=ARR(6)
2880> SCALE(3)=ARR(3)
2890> SCALE(4)=ARR(7)
2900> SCALE(5)=ARR(4)
2910> SCALE(6)=ARR(8)
2920> IF(COMIN.EQ.1.OR.OLDGRF) GO TO 4547
2930> C
2940> Y=GSIZE(2)
2950> IF(KLOBE.LE.0) GO TO 44
2960> Y=-.95
2970> CALL GRPLOT(GSIZE(3),GSIZE(2),0)
2980> CALL GRPLOT(GSIZE(1),GSIZE(2),1)
CALL GRAXIS(0,GSIZE(1),Y,GSIZE(3)-GSIZE(1),0.,ARR(2),ARR(3),
1 IFIX(ARR(4)),IFSIX(ARR(5)),"OBSERVATION ANGLE",80,1)
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,LPRCS,2,
1 'RCS (SM)' ,8,1)
GO TO 4547
C
CALL GRAXIS(3,GSIZE(1),GSIZE(2),DY,90.,ARR(6),ARR(7),IFIX(ARR(8)),
9 'RCS (DBSM)' ,10,1)
C
ANG=THETA(1)
DO 46 J=1,NRCS
X(V,J)=ANG
ANG=ANG+THETA(3)
YY=RCST(J)
IF(IRPR.EQ.0.) GO TO 4561
YY=CSJ(J)+2*RCST2(J)+2
IF(IRPRCS.EQ.0.) GO TO 4561
YY(J,1)=YY
GO TO 46
YV(J,1)=10.*ALOG10(AMAX1(CUTOFF,YY))
CONTINUE
CALL GRWIND(-NRCS,XV,YY,IB,0.,1.,0.)
IF(COMBIN.EQ.1.) GO TO 4568
IF(ABSIARR(J))=1.0,1,0,1
1.0,1,0,1
GO TO 46
DO 47 K=1,2
S=SQR(AMAX1(0.,RCST(J)+2-RCST2(J)))
YV(J,K)=10.*ALOG10(AMAX1(CUTOFF,RCST(J)+SIGN*S))
CALL GRWIND(-NRCS,XV,YY1,IB,0.,1.,0,1)
CONTINUE
IF(IZETA.EQ.0.) CALL GRMODE('ALPHA',2)
IF(CMD.EQ.CMDS(1)) GO TO 4548
DEBOUT(1)=CMD
IF(COMBIN.EQ.-1) DEBOUT(1)=ALL
CALL GRXT(DEBOUT,21)
GO TO 4549
READ(10,F=2,END=9993) BUF
LINE=LINE+1
IF(OLDGRAF) GO TO 4569
CALL TRIMGN(INB,BUF)
CALL GRTXT(BUF,INB)
GDAUTO=.FALSE.
IF(ARR(11)) LT .0) GDAUTO=.TRUE.
XID=GSIZE(3)+GSIZE(1))/2.
CALL GRAXY(XID,GSIZE(4)+.75,XX,YY)
CALL GRPRNT(0,XX,-1,YY,15,0.)
CALL GRTXT('MONOSTATIC FULL-SCALE ',0)
IF(IRPR.EQ.0) CALL GRXT('RCS',0)
IF(IRPR.EQ.1) CALL GRXT('RPRCS',0)
CALL GRAXY(XID,GSIZE(4)+.5,XX,YY)
CALL GRPRNT(0,XX,-1,YY,15,0.)
C
SET TERMINAL TO ALPHAM MODE AND SCALE TO INCHES
IF(IZETA.EQ.0.) CALL GRMODE('ALPHA',3)
CALL GRSCG(GSIZE(1),GSIZE(2),GSIZE(3),GSIZE(4))
X=FSIZE(3)+1.05+FSIZE(5)
Y=FSIZE(4)*.75
FREQ=.5/WLEN
CALL GRXTXT('FREQUENCY ',0)
CALL GRPRTNT(1,X,0,Y..12.0.)
CALL GRNUM('F',-1,3,FREQ)
CALL GRXTXT('GHZ',0)
CALL GRPRTNT(-1,X,0,Y..12.0.)
Y=Y-.2
CALL GRXTXT('POLARIZATION ',0)
CALL GRPRTNT(1,X,0,Y..12.0.)
CALL GRXTXT(POLHY(IFX(POLAR)),3)
CALL GRPRTNT(-1,X,0,Y..12.0.)
IF (IMGRAPH) GO TO 4542
Y=Y-.2
CALL GRXTXT('PITCH ',0)
CALL GRPRTNT(1,X,0,Y..12.0.)
CALL GRNUM('F',-1,3,PRTV(1))
CALL GRPRTNT(-1,X,0,Y..12.0.)
Y=Y-.2
CALL GRXTXT('ROLL ',0)
CALL GRPRTNT(1,X,0,Y..12.0.)
CALL GRNUM('F',-1,3,PRTV(2))
CALL GRPRTNT(-1,X,0,Y..12.0.)
Y=Y-.2
CALL GRXTXT('TAW ',0)
CALL GRPRTNT(1,X,0,Y..12.0.)
CALL GRNUM('F',-1,3,PRTV(3))
CALL GRPRTNT(-1,X,0,Y..12.0.)
Y=Y-.2
CALL GRXTXT('TILT ',0)
CALL GRPRTNT(1,X,0,Y..12.0.)
CALL GRNUM('F',-1,3,PRTV(4))
CALL GRPRTNT(-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 GRXTXT(ISIDE(1),ISIDE(1),LSIDE(ISIDE))
CALL GRPRTNT(1,X,0,Y..12.0.)
GO TO 4558
CALL GRXTXT('PHR ',0)
CALL GRPRTNT(1,X,0,Y..12.0.)
CALL GRNUM('F',-1,3,PHR)
CALL GRPRTNT(-1,X,0,Y..12.0.)
CONTINUE
IF (IMGRAPH) GO TO 4559
IF (DEBUG.NE.0) GO TO 4550
C COMPUTE AND GRAPH LOBE STATISTICS IF REQUIRED
IF(KLOBE.LE.0) GO TO 4551
CALL GRPLOT(11.,0.,0)
IGRP=1
WRITE((10L'1'),1) IGRP, 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=10L)
CALL SPAWS
OPEN(UNIT=10L, NAME='KLOBE.TMP', TYPE='OLD', RECORDSIZE=181,
2 ACCESS='DIRECT', SHARED)
IF(ISSTATS.EQ.0) GO TO 4550
GRAPH HIGHER ORDER BETA STATISTICS IF REQUIRED
CALL HOSBD(1, DUMMY, IOE, DUMMY, SLOBE, BUF, NB, 1,
1 FREQ, POLVH(IFIX(POLAR)), PERSH, PITCH, ROLL, YAW, TILT, PHR)
CLEANUP GRAPHS
IF(GDAUTO.AND.IZETA.EQ.0.AND.DEBUG.EQ.0) CALL GRTEKG
IF(DEBUG.EQ.0.AND.COMB1N.EQ.0.AND.NOT.ATTN) CALL GRTEKG
IF(IZETA.EQ.0) CALL GRPLOT(10., 8., 4., 0)
IF(IZETA.EQ.0) CALL GRMODE('ALPHA', 3)
IF(IZETA.EQ.0) CALL GRMODE('ALPHA', 3)
IF(IZETA.EQ.0) CALL GRMODE('MGRAFM') CALL GRUPPM
IF(CMD.EQ.CMD(9)) GO TO 3
COMBIN=ABS(COMB1N)
IF(ATTN) CALL DETACH(10R, IER)
IF(LP) CALL ASNLUNI() {'SYI00)
IF(LP) OPEN(UNIT=IOP, NAME='LP.LST')
IF(CMD.EQ.CMD(10)) GO TO 3
"PRINT" - PRINT RESULTANT RCS VECTOR
CONTINUE
IF(DEBUG.EQ.1.AND.CMD.EQ.CMD(10)) GO TO 4544
IF(DEBUG.EQ.1.AND.CMD.EQ.CMD(10)) GO TO 10
FORCE OUTPUT TO LINE PRINTER FOR DEBUG
IF(LP) CALL ASNLUNI(IOP, 'SYI00')
IF(LP) OPEN(UNIT=IOP, NAME='LP.LST')
IF(IO.EQ.IOP.AND.NOT.(GOLDFH.OR.LP)) CALL ATTACH(IOE, IER)
IF(IO.EQ.IOP.AND.NOT.LP) GOLDFH=True.
IF(CMD.EQ.CMD(10)) GO TO 3050
DEBUG OUTPUT PRINT MODE
M60DB=1.70N=6
DO 5054 J=1,NRCS
     IF (RCS(J).GE.M60DB) GO TO 5053
4810  5054 CONTINUE
4820  5055 WRITE(10,5055) CMD,LINE
4830  5055 FORMAT(’O’,A4,’ HAS NO RCS CONTRIBUTION (DATABASE LINE’,I3,’)’)
4840  5056 GO TO 5058
4850  C
4860  5053 CONTINUE
4870  5054 WRITE(10,56) CMD,LINE
4880  56 FORMAT(’O’,A4,’ COMPONENT OUTPUT FROM DATABASE LINE’,I3)
4890  5055 GO TO 5051
4900  5050 READ(10,END=9993) BUF
4910  5051 LINE=LINE+1
4920  C
4930  CALL TRIM0(NB,BUF)
4940  WRITE(10,54) (BUF(I),I=1,NB)
4950  54 FORMAT(’O’,120A1)
4960  C
4970  WRITE(10,57) FREQUENCY(GHZ)=’F5.1’)
4980  C
4990  IF(CMD.NE.CMDS1101) IBUG=1
5000  C
5010  IF(STATS.EQ.1) IBUG=0
5020  5051 IF(IRPR.EQ.0) WRITE(10,51)
5030  51 FORMAT(’O’ RCS OUTPUT’/2X,’ANGLE’ ,5X,’RCS’ ,5X,
5040   5’ RCS-S’/X)
5050  52 FORMAT(’O’ RP RCS OUTPUT’/2X,’ANGLE’ ,2X,’RCS(IBSM)” ,2X,
5060   5’ RCS(SM)” )
5070  C
5080  C
5090  DO 52 J=1,NRCS
5100  IF(IRPR.EQ.0) GO TO 5058
5110  Y=RC S(J)*2+RC S2(J)
5120  YYY=10. *ALOG10(AMAX1(CUTOFF,Y))
5130  WRITE(10,5059) ANG, YYY,Y
5140  5059 FORMAT(F7.1,2F10.4)
5150  GO TO 50
5160  C
5170  5058 RC S=AMAX1(CUTOFF,RCS(J))
5180  S=SQR(AMAX1(0.,RCS(J)**2-RCS2(J)))
5190  DBSM=10. *ALOG10(AMAX1(CUTOFF,RC S))
5200  DBSMS=10. *ALOG10(AMAX1(CUTOFF,RC S-S))
5210  D BSMPS=10. *ALOG10(AMAX1(CUTOFF,RC S+S))
5220  WRITE(10,53) ANG, D S M, D B S M S, D B S M P S
5230  53 FORMAT(F7.1,3F10.4,3X,2F10.4)
5240  54 IF(STATS.EQ.1) GO TO 52
5250  IF(STATS.GE.ANGINC.AN D.AM O D(ANG,STATS).NE.0) GO TO 52
5260  C
5270  C
5280  C
5290  C
5300  CALL HOSBD(IO,J,IOE,IO,SLOBE)
5310  C
5320  52 ANG=ANG+ANGINC
5330  IF(D EBUG.NE.0) GO TO 5056
5340  C
5350  C
5360  C
5370  C
5380  C
5390  C
5400  C
5410  C
5420  C
5430  C
5440  C
5450  C
5460  C
5470  C
5480  IG RPR=0
WRITE((10,'1') IPR, KLOBE, QINT, MRC, NLOBE, THETA, ALPHA, DELTA, 
1  BETA, SCALE, IZeta, IPCODE, WLEN, LBUG 
2  CLOSE(UNIT=10), 
CALL SPAWS 
OPEN(UNIT=10, NAME='LOBE_TMP', TYPE='OLD', RECORDSIZE=101, 
2  ACCESS='DIRECT', SHARED) 
IF(GOULD) WRITE(10,5052) 
5470 5052 FORMAT('1') 
IF(GOULD .AND. DEBUG.EQ.0) CALL DETACH(10,IER) 
IF(GOULD .AND. DEBUG.EQ.0) GOULD=.FALSE. 
IF(LP .AND. NOT.LPF) CLOSE(UNIT=10,DISPOSE='PRINT') 
IF(LP .AND. LPF) CLOSE(UNIT=10,DISPOSE='SAVE') 
IF(LP) CALL ASNLUNIT(10,'GD',0) 
IF(CMD.EQ.CMD(10)) GO TO 3 
CMD= CLEAR 
GO TO 70 
"PRYT" - MASTER ROTATION ANGLES 
5570 C 
5590 C 
5590 55 CONTINUE 
5600 222 FORMAT('0'/(1X,3F8.3)) 
C COMPUTE YAW->RESULT 
5610 C 
5620 C PRYT(3)=ARR(3) 
5630 C COS(ARR(3)*RADS) 
5640 5650 CALL MATPUT(PR YT,1.,0.,0.,S,-S,0.,S,C) 
5660 C MULTIPLY BY ROLL->ROLL*RESULT=RESULT 
5670 C PRYT(2)=ARR(2) 
5680 C =COS(ARR(2)*RADS) 
5690 5700 CALL MATPUT(MAT,C,-S,0.,S,C,0.,0.,0.,1.) 
5710 CALL MATMUL(MAT,PRYT,PRYT) 
5720 C MULTIPLY BY PITCH->PITCH*RESULT=RESULT 
5730 C PRYT(1)=ARR(1) 
5740 C =COS(ARR(1)*RADS) 
5750 5760 CALL MATPUT(MAT,C,0.,S,0.,1.,0.,-S,0.,C) 
5770 CALL MATMUL(MAT,PRYT,PRYT) 
5780 C MULTIPLY BY TILT->RESULT=TILT=RESULT 
5790 C PRYT(4)=ARR(4) 
5800 C =COS(ARR(4)*RADS) 
5810 5820 CALL MATPUT(MAT,C,0.,-S,0.,1.,0.,S,0.,C) 
5830 CALL MATMUL(MAT,PRYT,PRYT) 
5840 C MULTIPLY BY B->B*RESULT=RESULT->PRYT 
5850 C CALL MATPUT(MAT,0.,-1.,0.,1.,0.,0.,0.,0.,1.) 
5860 CALL MATMUL(MAT,PRYT,PR YT) 
5870 GO TO 3 
"PAUSE" - WAIT FOR A CARRIAGE RETURN 
5880 C 
5890 60 CONTINUE 
5900 60 READ(IOR,61,END=8990) J 
5930 61 FORMAT(A1) 
5940 GO TO 3 
5950 C "RAMDS" - MODIFY COMPONENT RCS 
5970 C 
5990 65 RAM(10,11)=ARR(11)/10.)
GO TO 3
C "CLEAR" - CLEAR RCS VECTOR
CONTINUE
10 IPASS=0
C 0VALID=0
070 NTOA=0
080 DO 71 J=1,MAXRCS
090 RCS2(J)=0.
100 RCS(J)=0.
110 YYK(J,1)=0.
120 YYV(J,2)=0.
130 XW(J)=0.
140 C EXTRA(J)=0.
150 IF (OPEN.EQ.1) WRITE(IO,E) DZERO, DZERO, DZERO, DZERO, DZERO, DZERO
160 CONTINUE
170 IF(CMD.EQ.CLELIKE) GO TO 3
C RESET SYSTEM VECTORS
C POLAR=1.
C SET PYRT TO IDENTITY BY DEFAULT
C DO 74 J=1,4
C PRIL(J)=IDENT(J)
C RESET LODE PARAMETERS
C KLOBE=0
C LLOBE=0
C MLOBE=0
C NLLOBE=3
C BETA=0.
C C RESET RPRCS PARAMETERS
C IPA=0
C C RESET VALID VECTOR AFTER EACH OBJECT
C VALID(1)=360.
C VALID(2)=360.
C DO 75 J=1,6
C VALID(J)=0.
C RAM=1.
C DO 75 J=1,6
C TRAN(J)=IDENT(J)
C DO 75 J=1,3
C ORGN(J)=0.
C IF(IPASS.EQ.1) GO TO 99
3020 CONTINUE
C DEBUG OUTPUT
C DEBUG IFIX(ARR(1))
C GO TO 3
NEW WAVE LENGTH

IF(IWL.EQ.10) GO TO 9990
OLDGRF=.FALSE.

REWIND IOP
IF(KLOBE.GT.0) CLOSE(UNIT=10)
IF(GOULD) CALL DETACH(IOP,IER)
GOULD=.FALSE.
LINE=0
GO TO 70

THE FOLLOWING COMMANDS ARE NEW
ADDITIONS TO RCS FOR THE LOBE

"LOBE" - SET LOBE SWITCH
CONTINUE
IF(ISTATS.EQ.1.AND.ARR(1).GT.0) GO TO 573
IF(LLOBE.EQ.1) LLOBE=ARR(1)
IF(LLOBE.GT.0.AND.KLOBE.LE.0) CALL MOLOCK(10)
IF(LLOBE.GT.0.AND.KLOBE.LE.0) WRITE(10) DUMMY
IF(LLOBE.GT.0) MLOBE=LLOBE
GO TO 3
WRITE(2,502)
FORMAT(' RCS -- NPRCS AND LOBE CANNOT BOTH BE ACTIVE')
GO TO 9993
"BETA" - DEFINE BISTATIC ANGLE
BETA=ARR(1)
IF(BETA.LT.5.) BETA=0.
GO TO 3
"ORGN" - COMPONENT ORIGIN VECTOR
ORGN(1)=ARR(1)
ORGN(2)=ARR(2)
ORGN(3)=ARR(3)
GO TO 3
"DELT" - FIRST LOBE ANGULAR RANGE
DO 541 J=1,4
DELTA(J)=ARR(J)
DELTA(2)=MAX1(Delta(1),Delta(2))
DELTA(3)=MAX1(1.,DELTA(3))
GO TO 3
7190> C "ALPH" - SECOND LOBE ANGULAR RANGE
7200> C
7210> S50 DO 551 J=1,4
7220> S51 ALPHA(J)=ARR(J)
7230> GO TO 3
7240> C
7250> C***********************************************************************
7260> C END OF LOBE COMMANDS
7270> C
7280> C***********************************************************************
7290> C "RPRCS" - RELATIVE PHASE RCS
7300> C
7310> S80 CONTINUE
7320> C
7330> S50 IF (MLLOBE.GT.0.AND.ARR(1).GT.0.) GO TO 501
7340> C
7350> S50 IF (ISTATS.EQ.1.AND.ARR(1).GT.0.) GO TO 571
7360> C
7370> S50 IRPR=MINO(1,MAXI(0.,ARR(1)))
7380> C
7390> S50 IF (IRPR.EQ.0) GO TO 3
7400> C
7410> S50 RPRCS=ARR(2)
7420> C
7430> S50 LRPRCS=ARR(3)
7440> C
7450> S50 IF (LRPRCS.LE.1) LRPRCS=5
7460> C
7470> S50 LLOBE=1
7480> C
7490> S50 MLOBE=1
7500> C
7510> S50 GO TO 3
7520> C
7530> C "STATS" - HIGHER ORDER STATISTICS
7540> C
7550> S70 CONTINUE
7560> C
7570> S70 IF (IRPR.EQ.1.AND.ARR(1).GT.0.) GO TO 571
7580> C
7590> S70 IF (MLLOBE.GT.0.AND.ARR(1).GT.0.) GO TO 573
7600> C
7610> S70 ISTATS=MINO(1,MAXI(0.,ARR(1)))
7620> C
7630> S70 JSTATS=ARR(2)
7640> C
7650> S70 IF (JSTATS.LE.0.) JSTATS=10.
7660> C
7670> S70 IF (JSTATS.EQ.0.OR.OPEN) GO TO 3
7680> C
7690> S70 OPENUNIT=IOE,NAME='RCSTMP.DES',TYPE='SCRATCH',
7700> S70 Format='DIRECT',RECORDSIZE=16,MAXRECS=161
7710> C
7720> S70 OPEN=.TRUE.
7730> C
7740> S70 DO 575 J=1,181
7750> S75 WRITE(IOE)DZERO,DZERO,DZERO,DZERO,DZERO,DZERO
7760> S75 GO TO 3
7770> C
7780> C571 WRITE(2,572)
7790> C572 FORMAT(' RCS> -- RPRCS AND STAT CAN BE ABLE BE ACTIVE')
7800> C
7810> C
7820> C
7830> C573 WRITE(2,574)
7840> C574 FORMAT(' RCS> -- LOBE AND STAT CAN BE ABLE BE ACTIVE')
7850> C
7860> C
7870> C
7880> C "GLINT" - SET GLINT SWITCH
7890> C
7900> S80 GLINT=MINO(2,MAXI(0.,ARR(1)))
7910> C
7920> S80 GO TO 3
7930> C
7940> C "NEWTH" - REPEAT LOOP FOR NEU THETA ANGLE RANGE
7950> C
7960> S80 CALL CHNEWTH(KLOBE,ROUTE)
GO TO (3,80,9993),ROUTE
7800 C "GSIZE" - SET GRAPH SIZE
7810 C
7820 C
7830 DO 800 J=1,5
7840 DO 801 GSIZE(J) = ARR(J)
7850 CALL GPRIG(GSIZE(1),GSIZE(2),GSIZE(3),GSIZE(4))
7860 GO TO 3
7870 C DETERMINE IF COMMAND IS AN OBJECT
7880 C
7890 CONTINUE
7900 DO 91 J=1,NOBJ
7910 IF(CMD.EQ.OBJS(J)) GO TO 100
7920 CONTINUE
7930 WRITE(10W,92) CMD
7940 FORMAT(' RCS -- INVALID FUNCTION ',A4,'\n')
7950 GO TO 9993
7960 C
7970 C ADD OBJECT CONTRIBUTION TO RCS VECTOR
7980 C
7990 C
8000 C
8010 C
8020 C
8030 C NOTE: IPASS IS ONLY USED WITH RFLAT IN CONJUNCTION WITH LOBE
8040 IPASS=1
8050 IPASS=IPASS+1
8060 C
8070 IF(WLEN.LE.0.) GO TO 9991
8080 C WRITE(10W,92) CMD
8090 FORMAT(' RCS -- INVALID FUNCTION ',A4,'\n')
8100 GO TO 9993
8110 C
8120 C
8130 C
8140 C
8150 C
8160 C
8170 C
8180 C
8190 C
8200 IF((BUG.AND.2).EQ.0) GO TO 101
8210 WRITE(2,222) TRANS
8220 WRITE(2,222) PRYT
8230 WRITE(2,222) COMP
8240 C
8250 C
8260 C
8270 C
8280 C
8290 C
8300 C
8310 C
8320 C
8330 C
8340 C
8350 C
8360 C
8370 IF(IRCS.LE.MRCB) GO TO 131
8380 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
WRITE(10,'(I8,1X,'"SLOBE"(J,1,1),1=1,NRCS)') SLOBE(J,1,1)
DO 133 I=1,NRCS
IF(SLOBE(J,1,2).NE.0.) GO TO 134
CONTINUE
GO TO 135
NLOBE=NLOBE+4
DO 137 J=1,4
WRITE(10,'(I8,1X,'"SLOBE"(J,1,2),I=1,NRCS)') SLOBE(J,1,2)
C MAKE SECOND PASS THROUGH RFLAT FOR LOBE
IF(CMD.EQ.OBJ(13).AND.IPASS.EQ.0) I=PASS=1
GO TO 72
C DETERMINE IF VALID ANGLE FOR LOBE CONTRIBUTION
IF(LLOBE.LE.0) GO TO 132
SLOBE(1,NRCS,1)=0.
SLOBE(1,NRCS,2)=0.
C SKIP ANGLE IF NOT VALID FOR CURRENT OBJECT
DO 132 J=1,5,4
IF(VA7(J)-VALID(J+1)) 103,102,104
IF(ANG.LT.VALID(J+3).OR.ANG.GT.VALID(J+1)) GO TO 102
IF(VALID(J+2)-VALID(J+3)) 105,106,108
IF(AN7(J+2).NE.VALID(J+3)) GO TO 102
GO TO 109
IF(VALID(J+2).NE.VALID(J+3)) 107,108,109
IF(AN7(J+2).NE.VALID(J+3)) GO TO 102
GO TO 109
C CURRENT ANGLE IS NOT VALID. INCREMENT ANGLE.
GO TO 130
C CURRENT ANGLE IS VALID. FIND PRIMED ANGLES FROM RADAR COORDINATES.
NOTE! THIS PROGRAM STORES MATRICES BY ROWS (NOT COLUMNS)
CONTINUE
CANG=COS(ANG*RAD)
SANG=SIN(ANG*RAD)
CTHP=COMP(3)*SANG*CPPR+COMP(6)*SANG*SPHR+COMP(9)*CANG
TNP=ACOS(TCHP)
STHP=SIN(THP)
CPHPS=COMP(1)*SANG*CPPR+COMP(4)*SANG*SPHR+COMP(7)*CANG
SPHPS=COMP(2)*SANG*CPPR+COMP(5)*SANG*SPHR+COMP(8)*CANG
PHP=0.
IF(ABS(CPHPS).LT.1.E-5.AND.ABS(CPHPS).LT.1.E-5) GO TO 1111
PHP=ATAN2(CPHPS,CPHPS)
THE JOHNS HOPKINS UNIVERSITY
APPLIED PHYSICS LABORATORY
LAUREL, MARYLAND

8990  1111  CPHP=COS(TPHP)
9000  C2PHP=CHPCHPHP
9010  SPHP=SIN(TPHP)
9020  S2PHP=SPHP*SPHP
9030  223  FORMAT(4F8.3)
9040  C
9050  C
9060  C
9070  C
9080  C
9090  C
9100  C
9110  C
9120  C
9130  C
9140  C
9150  C
9160  C
9170  C
9180  C
9190  C
9200  C
9210  C
9220  C
9230  C
9240  C
9250  C
9260  1112  CPHP=COS(TPHP)
9270  SPHP=SIN(TPHP)
9280  IF(IGPBUG.AND.4.EQ.0) GO TO 110
9290  DTH=TH/RAD
9300  DPH=PH/RAD
9310  WRITE(2,223) DTH,DPH
9320  C
9330  C
9340  C
9350  110  GO TO (200,210,220,230,240,250,260,270,280,290,300,310,320,330,340,350,360,370), IOBJ
9360  C
9370  C
9380  C
9390  C
9400  200  CALL OQIVE
9410  GO TO 120
9420  C
9430  C
9440  C
9450  210  CALL CYLIN
9460  GO TO 120
9470  C
9480  C
9490  C
9500  220  CALL CONE
9510  GO TO 120
9520  C
9530  C
9540  C
9550  C
9560  C
9570  C
9580  C
9590  C
9600  240  CALL SOLID

39
GO TO 120

9610 C "TORUS" - TORUS
9620 CALL TORUS
9630 GO TO 120

9640 C "WEDGE" - WEDGE
9650 CALL WEDGE TRANS, CTH, STH, CPH, SPH
9670 GO TO 120

9680 C "WIRE" - WIRE
9690 CALL WIRE TRANS, CTH, STH, CPH, SPH
9710 GO TO 120

9720 C "LWIRE" - LOOP WIRE
9730 CALL LWIRE TRANS, CTH, STH, CPH, SPH
9750 GO TO 120

9760 C "CAVA" - CAVITY AREA
9780 CALL CAVA
9790 GO TO 120

9800 C "CAVD" - CAVITY DIAMETER
9820 CALL CAVD
9830 GO TO 120

9840 C "CFLAT" - CIRCULAR FLAT PLATE
9860 CALL CFLAT
9870 GO TO 120

9880 C "RFLAT" - RECTANGULAR FLAT PLATE
9900 CALL RFLAT IPASS
9910 IF IPASS.EQ.2 RCSC = 0.
9920 GO TO 120

9930 C "TWAVE" - TRAVELING WAVE
9950 CALL TWAVE
9960 GO TO 120

9980 C "DIHED" - DIHEDRAL
9980 CALL DIHED
10000 GO TO 120

10010 C "TCORN" - GENERAL TRIANGULAR CORNER
10030 CALL TCORN
10040 GO TO 120

10050 C "RCORN" - GENERAL RECTANGULAR CORNER
10070 CALL RCORN
10080 GO TO 120

10090 C "ECORN" - GENERAL ELLIPTICAL CORNER
10110 CALL ECORN
10120 GO TO 120

10130 C
ADD RCS CONTRIBUTIONS TO VECTORS
10160 C
10170 GO TO 120 CONTINUE
10180 IF RCSC.LT.0 WRITE (LOW, 121) CMD, ANG, RCSC

40
IF (RSCG .GT. CUTOFF) NVALID = NVALID + 1
RSC = RSC + RAM
IF (IRPR .EQ. 1) GO TO 136
RCS1 = RCS + RCS1 + RCS
RCS2 = RCS2 + RCS2 + RCS + RCS
IF (ISTATS .EQ. 0) GO TO 130
READ (IOE, 'IRCS') T1, T2, T3, T4, T5, T6
T1 = T1 + DBLE (RCS)
T2 = T2 + DBLE (RCS) * M2
T3 = T3 + DBLE (RCS) * M3
T4 = T4 + DBLE (RCS) * M4
SQ = SQRT (RCS)
T5 = T5 + SQ
T6 = DMAX1 (T6, SQ)
WRITE (IOE, 'IRCS') T1, T2, T3, T4, T5, T6
INCREMENT ANGLE
CONTINUE
ANG = ANG + ANGINC
IRCS = IRCS + 1
IF (IPASS .NE. 2) NTOTAL = NTOTAL + 1
GO TO 101
WRITE (IOE, 'IRCS') T1, T2, T3, T4, T5, T6
TERMINATE RCS PROGRAM
IF (KLOBE .GT. 0) CLOSE (UNIT = IOL, DISPOSE = 'DELETE')
IF (OPENE) CLOSE (UNIT = IOE)
IF (GOULD) CALL DETACH (IOP, IER)
TIME = SECONDS (TIME) / 60
IF (IZETA .EQ. 0) CALL GRMODE ('ALPHA', 3)
WRITE (2, 9998) TIME
FORMAT (' RCS -- ELAPSED TIME = ', F5.1, ' MINUTES')
CALL EXIT
END
SUBROUTINE CNEWTH(KLOBE, ROUTE)

INTEGER*2 ROUTE
INTEGER*2 DEBUG, COMBIN, GLINT
LOGICAL GOULD, LP, OPENE, LPF, OLDGRF, MGRAPH, GDAUTO
COMMON/SWITCH/ IOW, THETA(4), GOULD, LP, OPENE, LPF, OLDGRF, MGRAPH,
1 GDAUTO, DEBUG, COMBIN, GLINT, IBUG, LSUG, ISTATS, IZETA

ROUTE = 1

IF (THETA(4).GT.THETA(2)) GO TO 1
IF (DEBUG .NE. 0) RETURN
IF (IZETA.EQ.0 .AND. .NOT.GDAUTO) RETURN
IF (IZETA.EQ.0) CALL GRTK
IF (IZETA.NE.0) CALL GPLOT(11., 0.5, -1)
RETURN

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

WRITE(IOW, 592)
FORMAT(' RCS> -- MULTIPLE GRAPH SEGMENTS ARE NOT ALLOWED WITH: '/
1 LOBE, STATS, OR DEBUG!')
ROUTE = 3
RETURN
END
SUBROUTINE OGIVE

INTEGER*2 LOBE
COMMON/RCSCOM/CUTOFF,WLEN,PI,P12,RCSC,THP,CTHP,STHP,PHP,CPHP,SPHP,
5 R1,A,B,BETA

THPT=THP
IF(THP.GT.PI) THPT=PI-THP
IF(LOBE.GT.0) L2=SQRT(R1*R1-(B+R1-A)**2)
IF(B.GT.0.) GO TO 5
ALP=A*COSK(1.-A/R1)
RCSC=R1*R1*TRIG(SIN(ALP)**2)/4.PI)
NMA=P12-ALP
IF(THPT.LT.NMA) GO TO 2

90-ALP < THETA < 90
SIGMA=P1*R1*R1*(1.-R1-A)/(R1*STHP))
IF(LOBE.LE.0) GO TO 4
IF(PI.CT.15.0) GO TO 6
CALL TOLOBE(1,SIGMA,ANCPHP,ANSPHP,0.)
RCSC=SIGMA
RETURN

THETA = 90-ALP
CALP2=(1.-A/R1)**2
SALP=SIN(ALP)
ZALP=R1*SALP*CALP2-SQRT(CALP2*SALP**2)
WALP=ZALP*TAN(INMAJ)
CALL TOLOBE1,RCSC,WALP,CPHP,SPHP,ZALP)
RETURN

THETA < 90-ALP
Z=15.PI*COS(THPT)**2*(1.-TAN(ALP)**2*TAN(THPT)**2)**2
IF(Z.LT.CUTOFF) RETURN
SIGMA=WLEN*WLEN*TAN(ALP)**4/Z
IF(LOBE.LE.0) GO TO 1
IF(PI.LT.15.0) GO TO 6
CALL TOLOBE(1,SIGMA,0.0.,L2)
RCSC=AMIN(RCSC,SIGMA)
RETURN

TRUNCATED OGIVE (BETA IS ALWAYS < ALPHA)
BET=BETA*PAS
IF(THPT.GT.PI2-BET) GO TO 3

THETA = 0
RCSC=P1*THP**2/2
IF(PI.LT.15.0 AND LOBE.GT.0) CALL TOLOBE(1,RCSC,0.0.,L2)
IF(PI.LT.15.0) RETURN
BETA < THETA < 90-BETA
SUBROUTINE CYLIN
IMPLICIT REAL*4 (A-Z)
INTEGER*2 LOBE
COMMON/RCSCOM/CUTOFF,WLEN,P1,P2,RCSC,THP,CTHP,STHP,PHP,CPHP,SPHP,
& C2PHP,S2PHP,RAD,POLAR,LOBE,
& A,B,L

IF(STHP.LE.0.) RETURN
A2=A*A
B2=B*B

Z=(A2*CPHP*CPHP+B2*SPHP*SPHP)**1.5
RCSC=2.*WLEN/WLEN*2.**1.5

THETA'=90-

IF(ABS(D).LT.CUTOFF) GO TO 1
RETURN

SUBROUTINE CONE
IMPLICIT REAL*4 (A-Z)
INTEGER*2 LOBE
COMMON/RCSCOM/CUTOFF,WLEN,P1,P2,RCSC,THP,CTHP,STHP,PHP,CPHP,SPHP,
& C2PHP,S2PHP,RAD,POLAR,LOBE,
& A,L2
THETA'=ALPHA*

IF(STHP.LE.0.) RETURN
NU2=NU2*NU2
TALP=TAN(ALPHA)
RETURN
D=SQR(S2PHP*NU2+C2PHP)
POLANG=ATAN(-NU/(TALPM))
RCSC=-.NPI*(L2**.5-L1**.5)**2*TALP**4
RCSC=RCSC/10.*WLEN*NU2*ABS*(COS(POLANG)**3))
Z=(STHP*TALPM+NU*CTHP)**2
Z=Z*0.*PHI*STHP/G
IF(Z.GT.CUTOFF) GO TO 1
IF(LOBE.LE.0) RETURN
L=(L1+L2)/2.
CALL TOLOBE(1,RC5,L*STHP*CPHP,L2*NU*STHP*SPHP,L)
RETURN
END
C OFF - NORMAL INCIDENCE
SIGMA=1(STHP/STHP-(CTHP+STHP/NU))**2
SIGMA=SIGMA*WLEN*NU2*NU*STHP/Z
RCSC=AMIN1(RCSC,SIGMA)*L1*2)
IF(LOBE.LE.0) RETURN
CALL TOLOBE(1,AMIN1(RCSC/2.,L1*SIGMA),L1*STHR*CPHP,
1 L2*NU*STHP*SPHP,L1)
CALL TOLOBE(2,AMIN1(RCSC/2.,L2*SIGMA),L2*STHP*CPHP,
1 L2*NU*STHP*SPHP,L2)
RETURN
END
C WITH LOBE
IMPLICIT REAL*4 (A-Z)
INTEGER*2 LOBE
COMMON/RC5C/M/CUTOFF,WLEN,PI,P12,RCSC,THP,CTHP,STHP,PHP,CPHP,SPHP,
$ C2PHP,S2PHP,RAD,POLAR,LOBE,
S P
RCSC IS UNDEFINED AS THP APPROACHES PI
IF(ABS(THP).GT.70.*RAD) RETURN
RCSC=4.*PHI/CTHP**4
IF(LOBE.LE.0) RETURN
THP=TANG(THP)
CALL TOLOBE(11,RCSC,2.*PHI*TTHP*CPHP,2.*PHI*TTHP*SPHP,-PHI*TTHP*TTHP)
RETURN
END
C SUBROUTINE SOLID
C WITH LOBE
IMPLICIT REAL*4 (A-Z)
INTEGER*2 LOBE
COMMON/RC5C/M/CUTOFF,WLEN,PI,P12,RCSC,THP,CTHP,STHP,PHP,CPHP,SPHP,
$ C2PHP,S2PHP,RAD,POLAR,LOBE,
$ A,B,C
ROOM=ANB*C
D=(AN*STHP*CPHP)**2
D=D+(B*STHP*SPHP)**2
SUBROUTINE TORUS

IMPLICIT REAL*4 (A-Z)

INTEGER*2 LOSE

COMMON/RSCSCOM/CUTOFF,WLEN,PI,P12,RCSC,THP,CTHP,STHP,PHP,CPHP,SPHP,
$ C2PHP,S2PHP,RAD,POLAR,LOBE,
$ A,S

C

RCSC=6.3P1W3*WLEN/WLEN

IF(STHP.LT.CUTOFF.AND.LOSE.OT.0) CALL TOLOSE(IJRCSC,0.,0.,0.,0.,0.)

IF(STHP.LT.CUTOFF) RETURN

SIG1=P1(WMA/STHP+BHB)

IF(LOSE.GT.0) CALL TOLOSE(I1,AMNI1RCSC,81021,A*CPHP,-MSPHP,0.)

IFABS(CTHP).LE.5/(2.*N) GO TO 1

SIG2=P1(WMA/STHP-BHB)

IF(LOSE.GT.0) CALL TOLOSE(I2,AMNI1RCSC,81021,-A*CPHP,-MSPHP,0.)

RCSC=AMNI1RCSC,81021

RETURN

C

END

C

SUBROUTINE WEDEICOOR,CTH,STH,CPH,SPH

IMPLICIT REAL*4 (A-Z)

INTEGER*2 LORE

COMMON/RSCSCOM/CUTOFF,WLEN,PI,P12,RCSC,THP,CTHP,STHP,PHP,CPHP,SPHP,
$ C2PHP,S2PHP,RAD,POLAR,LOBE,
$ A,S

C

WEDEMAI=W/(WLENWLEN)

ANG=ANGLE/RAD

M=P1(WLENL/4.*WANGWANG))SIN(P1P1/(2.*WANG))#S2

T=COS(P1P1/(2.*WANG))

C=COS(P1P1/PANG+T

D=1.-T

IF(ABS(C).LE.CUTOFF.OR.ABS(D).LE.CUTOFF) RETURN

IF(POLAR.EQ.2.) GO TO 2

RCSC=2.*(COOR(3)*SPH+COOR(6)*CPH)#S2-1.

RCSC=MIN(RCSC/C+1./D)#S2

GO TO 4

RCSC=2.*(COOR(3)*CHTH+CPH+COOR(6)*CTH+SPH-COOR(9)#S 8TH)#S2-1.

46
SUBROUTINE WIRE(COOR,CTH,STH,CPH,SPH)

IMPLICIT REAL*4 (A-Z)
INTEGER L, A
COMMON/RCSCOM/CUTOFF,WLEN,PI,P12,RCSC,THP,CTHP,STHP,PHP,CPHP,SPHP,
C2PH,C2PHM,RAD,POLAR,LOBE,
C
REAL,COOR(10),P(3),ZK(3)

IF(STHP.LE.0.) RETURN

GAMMA=1.75
ZK(1)=COOR(8)+(-CTH)*COOR(9)+(-STH)*SPH)
ZK(2)=(-COOR(7)+(-CTH)*COOR(8)+(-STH)*CPH)
ZK(3)=COOR(7)+(-STH)*SPH)+COOR(8)+(-STH)*CPH)

IF(POLAR.EQ.1) GO TO 2

P(1)=SPH
P(2)=CPH
P(3)=CTH
GO TO 3
P(1)=CTH*CPH
P(2)=CTH*SPH
P(3)=CTH*SPH
P(3)=CTH

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))

RCSC=AMAX1((-1.,AMIN1(1.,SPS))

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)*WLEN/(GAMMA*PI*A)

IF(SIGMA.GT.0.0) RETURN

CALL TOLOBE(1,RCSC/2.,0.,0.,0.,0.)
CALL TOLOBE(2,RCSC/2.,0.,0.,0.,L/2.)
RETURN
END
SUBROUTINE LWIRE(COOR, CTH, STH, CPH, SPH)

IMPLICIT REAL*4 (A-Z)
INTEGER*2 LOBE
COMMON/RCSCOM/CUTOFF, WLEN, PI, P12, RCSC, THP, CTHP, STHP, PHP, CPHP, SPHP,
C2PHP, S2PHP, RAD, POLAR, LOBE,
REAL COOR(S)

IF(POLAR.EQ.2) GO TO 2

GAMMA=ACOSK(-COOR(2)*SPH+COOR(5)*CPH)
GO TO 3

GAMMA=ACOSK(COOR(2)*CTH+SPH+COOR(5)*CPH)
GO TO 3

K2=(1.4*PIA/WLEN)*S1P1
C02=CO2(GAMMA)**2
C02C=SIN(GAMMA)**2*C02+CTHP+C02
CALL BESJ(KA2,0,JO,.1,.IER)
IF(IER.NE.0) GO TO 9
CALL BESJ(KA2,2,J2,.5,.IER)
IF(IER.NE.0) GO TO 9
RCSC=(C02C+C02)**0.5
RCSC=RCSC/RCSC
IF(LOBE.LE.0) RETURN
IF(STHP.LE.CUTOFF) CALL TOLOBE(KA2,0,JO,.1,IER)
IF(STHP.GT.CUTOFF) CALL TOLOSE(KA2,0,JO,.1,IER)
RETURN

END
IMPLICIT REAL*4 (A-Z)
INTEGER*2 LOBE
COMMON/RSCS/RSC/CUTFF,WLEN,P1,P12,RSCC,THP,CTHP,STHP,PHP,CPHP,SPHP,
$ C2PHP,S2PHP,RAD,POLAR,LOBE,
$ D
C
IF(ABS(THP).GT.70.*WLEN) RETURN
RCSC=0.5*WLEN+P12.*WLEN/WLEN**3*WLEN
THP=TAN(THP)
IF(THP.LT.CUTOFF) CALL TOLOBE1,RCSC,0.,0.,0.
IF(THP.LT.CUTOFF) RETURN
SIGMA=.5*(2.*THP**2)/TTHP
RCSC=AMIN1(RSCC,SIGMA)
IF(LOBE.LE.0) RETURN
DS2=D*STHP**2.*RCSC
CALL TOLOSE11,RCSC,D52*STHP*CPHP,DS2*STHP*SPHP,DS2*CTHP)
RETURN
END
C
C
WITH LOBE
C
C
SUBROUTINE CAVD
C
C
END
C
C
WITH LOBE
C
C
SUBROUTINE CFLAT
C
C
SUBROUTINE RFLAT(IPASS)
C
C
WITH LOBE
C
C
IMPLICIT REAL*4 (A-Z)
INTEGER*2 LOBE
COMMON/RSCS/RSC/CUTFF,WLEN,P1,P12,RSCC,THP,CTHP,STHP,PHP,CPHP,SPHP,
$ C2PHP,S2PHP,RAD,POLAR,LOBE,
$ A
C
IF(ABS(THP).GE.P12) RETURN
RCSC=4.*WLEN/(WLEN**2+WLEN)
D=8.*WLEN**2*TAN(THP)**2
IF(D.LT.CUTOFF.AND.LOBE.GT.0) CALL TOLOSE1,RCSC,0.,0.,0.
IF(D.LT.CUTOFF) RETURN
SIGMA=AMIN1(RSCC,2.*SIGMA)
IF(LOBE.LE.0) RETURN
CALL TOLOSE12,RCSC/2.,RCSC/2.,-AMCPHP,-AMSPHP,0.)
CALL TOLOSE12,RCSC/2.,RCSC/2.,-AMCPHP,-AMSPHP,0.)
RETURN
END
C
C
WITH LOBE
C
C
IMPLICIT REAL*4 (A-Z)
INTEGER*2 LOBE,IPASS
SUBROUTINE TAVE

COMMON/RCSCOM/CUTOFF,WLEN,P1,P12,RCSC,THP,CTHP,STHP,PHP,CPHP,SHPH,
               C2PHP,S2PHP,RAD,POLAR,LOBE,
               A,B

IF(ABS(THP).GE.P12) RETURN
RCSC=PIN(0,H,W,WLEN/WLEN)*2
S2THP=S2THP*STHP
S4THP=S4THP*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/(16.*PIN**3*S4THP*S2PHP*C2PHP)
SIGMA=2.*SIGMA
RCSC=AMIN1(RCSC,2.*SIGMA)
IF(LOBE.LE.0) RETURN
RCSC4=RCSC/4.

IF(IPASS.NE.2) CALL TOLOBE(1,RCSC4,A,B,0.)
IF(IPASS.NE.2) CALL TOLOBE(2,RCSC,-A,B,0.)
IF(IPASS.EQ.2) CALL TOLOBE(1,RCSC,A,-B,0.)
IF(IPASS.EQ.2) CALL TOLOBE(2,RCSC,-A,-B,0.)
RETURN

PHI' = 0 OR 180

IF(IPASS.EQ.2) RETURN
SIGMA=H+B/(4.*PIN*S2THP)
IF(LOBE.LE.0) GO TO 1
SIGA=AMIN1(RCSC/2.,SIGMA)
CALL TOLOBE(1,1.,SIGA,0.,0.)
CALL TOLOBE(2,1.,-SIGA,0.,0.)
RETURN

PHI' = 90 OR -90

IF(IPASS.EQ.2) RETURN
SIGMA=H+A/(4.*PIN*S2THP)
IF(LOBE.LE.0) GO TO 1
SIGMA=AMIN1(RCSC/2.,SIGMA)
CALL TOLOBE(1,1.,SIGA,0.,B,0.)
CALL TOLOBE(2,1.,0.,-B,0.)
GO TO 5
RETURN

THETA' = 0

IF(IPASS.EQ.2) RETURN
IF(LOBE.LE.0) RETURN
CALL TOLOBE(1,RCSC,0.,0.,0.)
RETURN

END

WITH LOBE

IMPLICIT REAL*4 (A-Z)
INTEGER*2 LOBE
COMMON/RCSCOM/CUTOFF,WLEN,P1,P12,RCSC,THP,CTHP,STHP,PHP,CPHP,SHPH,
               C2PHP,S2PHP,RAD,POLAR,LOBE,
               P,L,GAMMA,A
4560 C IF(POLAR.EQ.1.) RETURN
4570 C KL=2.*PI/WLEN
4580 C IF(KL.LE.4.) RETURN
4590 G=3.35AMLOG10(2.*KL)-.55
4600 T=1.-PHCTHP
4610 C IF(T.LE.CUTOFF) RETURN
4620 F=STHP*SIN(KLMT/(2.*PI))/T
4630 RCSC=(GAMMA+FF/WLEN/Q)MM2/P1
4640 C IF(LOBE.GT.0) CALL TOLOBE(1,RCSC,ACPHP,ASPHP,-L/2.)
4650 RETURN
4660 END
4670 C
4680 C C
4690 C SUBROUTINE DIHED
4700 C IMPLICIT REAL*4 (A-Z)
4710 C INTEGER N LOBE
4720 C COMMON/RCSOM/CUTOFF,WLEN,P1,P12,RCSC,CTHP,P12,STHP,PHP,CPHP,SPHP,
4730 C P12,RCSC,STHP,CTHP,CTHP,RAD,POLAR,LOBE,
4740 C A,B,C
4750 C IF(PHHP.LE.0.0.OR.PHHP.GE.PI2) RETURN
4760 P14=PI/4.
4770 GAMMA=ATAN2(B,A)
4780 DU=PI2-THP
4785 C COMPUTE NORMAL COMPONENT
4790 C IF(PHHP.GT.GAMMA) GO TO 1
4800 RCSC4,NPH=AM/AM/WLEN)*2N4.5S2PHP
4810 GO TO 2
4820 C RCSC4,NPH=(AM/AM/WLEN)*2N4.5C2PHP
4830 1 IF(DU4N2.GT.CUTOFF) GO TO 4
4840 C IF(LOBE.LE.0) RETURN
4850 4 IF(STHP.LT.0.5) GO TO 3
4860 SI1A=WLEN*WLEN/RCSC/(O.N(MNBU2))H2
4870 RCSC=AMINI(RCSC,SI1A)
4880 RETURN
4890 3 RCSC=0.
4900 RETURN
4910 END
4920 C
4930 C C
4940 C SUBROUTINE TCRN
4950 C C WITH LOBE
4960 C IMPLICIT REAL*4 (A-Z)
4970 C INTEGER N LOBE
4980 COMMON/RCSOM/CUTOFF,WLEN,P1,P12,RCSC,CTHP,P12,STHP,PHP,CPHP,SPHP,
5000 C P12,RCSC,STHP,CTHP,CTHP,RAD,POLAR,LOBE,
5010 C A,B,C
5020 C IF(THP.LE.0.0.OR.THP.GE.PI2) RETURN
5030 C IF(PHHP.LE.0.0.OR.PHHP.GE.PI2) RETURN
5040 CALL ORDER(STHP,CPHP,STHP,SPHP,CTHP,L,M,N)
5050 CALL ORDER(A,B,C,T1,T2,T3)
5060 L=LL/T3
5070 MB=M/T2
IMPLICIT REAL*4 (A-Z)
INTEGER*2 LOBE

COMMON/RCSCOM/CUTOFF,WLEN,PI,P12,RCSC,THP,CTHP,STHP,PHP,CPHP,SPHP,
& C2PHP, S2PHP, RAD, POLAR, LOBE,
& 8 C2PHP, 82PHP, RAD, POLAR, LOBE,
& S A,B,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)
LAZL/T3
MB:M/T2
NC:N/T1
IF(MB.LT.NCIZ.1 GO TO 1
AREAX4.*L*MNT10TIIN
RETURN
END

SUBROUTINE RCORN

IMPLICIT REAL*4 (A-Z)
INTEGER*2 LOBE

COMMON/RCSCOM/CUTOFF,WLEN,PI,P12,RCSC,THP,CTHP,STHP,PHP,CPHP,SPHP,
& C2PHP, S2PHP, RAD, POLAR, LOBE,
& 8 C2PHP, 82PHP, RAD, POLAR, LOBE,
& S A,B,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)
LAZL/T3
MB:M/T2
NC:N/T1
IF(MB.LT.NCIZ.1 GO TO 1
AREAX4.*L*MNT10TIIN
RETURN
END

SUBROUTINE EORN
MB2 = (M/T2)^2
NC2 = (N/T1)^2
IF (NC2 .GE. LA2 + MB2) GO TO 1
AREA = (N/T1) * ATAN1 ((LA2+MB2)^2-NC2*NC2) / (4.*NC2*(L/T3)*(M/T2)))
AREA = AREA + (M/T2) * ATAN ( (LA2+NC2)^2-NC2) / (4.*MC2*(L/T3)*(N/T1)))
AREA = AREA + (L/T5) * ATAN ((MB2+NC2)^2-LA2*LA2) / (4.*LA2*(M/T2)*(N/T1)))
GO TO 2
AREA = (M/T2) * ATAN ((L/T3)*(N/T1)/(MB2+NC2-LA2))
AREA = AREA + (L/T5) * ATAN ((L/T3)*(N/T1)/(LA2-MB2+NC2))
RCSC = 4.*PI* (AREA/WLEN) ** 2
IF (LOBE .LE. 0) RETURN
D = 1./SQRT (1./ANM2+1./BNM2+1./CMW2)
CALL TOLOBE (1., RCSC, D*STHP*CPHP, D*STHP*SPHP, D*CTHP)
RETURN
END
THIS ROUTINE ASSUMES MATRICES ARE ROW STRUCTURED

SUBROUTINE TOLOB{IC,RCSL,X,Y,Z)

INTEGER*2 DEBUG,COMBIN,QLINT

LOGICAL*2 GOULDLP ,OPENE,LPFOLDGRF,MGRAPH,GAUTO

COMMON/SWITCH/,IOW,THETA(4),GOULD,LP,OPENE,LPF,OLDFGRF,MGRAPH,

1 GAUTO,DEBUG,COMBIN,QLINT,IBUG,LIBUG,LISTATS,IZETA

REAL*4 SLOBE(4,181.2i

REAL*4 ORGN,COMP,SLOBE,BETA.

COMMON/RCSLOB/IOI.ORGN,COMP,SLOBE,BETA.

THRCTHRSTHR,PHR,CPHR,SPHR,KLOBE

COMMON/RCSCOM/CUTOFF,WLEN,PI,PI2,RCSC,THP,CTHP,STHP,PHP,CPHP,SPHP,

C2PHP,S2PHP,RAD,POLAR,LOBE.

ARR

REAL*4 RCS(I,RC$2I1SIJ

COON/RPRCS/ IRPR,IRCS,RAM,RCS,RC$2

IF(LBUG.NE.O) WRITE(2,7) IC,RCSLIXY,Z

FORMATE' TOLOBE:',I1.4E11.Si

TMpEI.)UX

CALL MATVEC(COMP,TMP,TMP)

CALL MATPUT(MTRAN,CTHR*CPHR,CTHR*PHR,PR4STHR.

SPHR,-CPHR,..,-STHR*CPHR, -STHR*SPHR, -CTHR

IFISETA.LE.5.1 GO TO 1111.

CALL MATVEC(MTRAN,TMP,TMP)

IFIILBUG.ANO.2I.EQ.0J

WRITE12.4) ICTNR,PHR,X,Y.ZORGNMTRAN

5 FORMAT(IX,12,SE10.3/3E11.3/3E11.3/3E11.3)

IFIIRPR.EQ.0) GO TO 3

ARG4.4$TMPI(TMP3)I/WLEN

SQ=SQRT(RCSL*RAM)

RC5(RCS)=RC5(RCS)+SQ*CMOSIARG)

RC5(RCS)=RC5(RCS)+SQ*SIN(ARG)

IFIILBUG.ANO.1)NE.0JWRITE(2,6)IRCS,RCSL,RAM,TMP,RCS1IRCS),RC51IRCS)

FORMATI' RCSLz',13.7E11.3)

IFI(KLOBE.LE.0) RETURN

SLOBE(I,IRCS,IC)=RC5L*RAM

SLOBE(2,IRCS,IC)=TMP(1)

SLOBE(3,IRCS,IC)=TMP(2)

SLOBE(4,IRCS,IC)=TMP(3)
IF((LBUG.AND.1).NE.0) WRITE(2,2) IC,IRCS,(SLOBE(I,IRCS,IC),I=1,4)
570 2 FORMAT('TOLOBE:',212,E11.3,3X,3F8.3)
590 RETURN
600 END
610 C
620 C
630 C
640 C
SUBROUTINE ORDER(A1,A2,A3,B1,B2,B3)
650 T1=AMIN1(A1,A2,A3)
660 T3=AMAX1(A1,A2,A3)
670 B3=A1*A2+A3-T1-T3
680 B1=T1
690 B3=T3
700 RETURN
710 END
720 C
730 C
THIS ROUTINE ASSUMES MATRICES ARE ROW STRUCTURED
740 C
SUBROUTINE MATMUL(A,B,C)
770 DO 1 J=1,3
780 T(J)=T(J-1)*3
790 DO 2 K=1,3
800 J=J3+K
810 T(K+J13)=0.
820 DO 3 L=1,3
830 T(K+J13)*T(K+J13)*A(J13+L)*B(K+(L-1)*3)
840 DO 2 CONTINUE
850 DO 1 CONTINUE
860 DO 4 J=1,9
870 RETURN
880 END
890 C
900 C
THIS ROUTINE ASSUMES MATRICES ARE ROW STRUCTURED
910 C
SUBROUTINE MATVEC(A,B,C)
930 REAL*4 A(9),B(3),C(3),T(3)
940 DO 1 J=1,3
950 J3=(J-1)*3
960 T(J)=0.
970 DO 2 L=1,3
980 T(J)*T(J)+A(J13+L)*B(L)
990 DO 1 CONTINUE
1000 DO 4 J=1,3
1010 C(J)*T(J)
1020 RETURN
1030 END
1040 C
1060 REAL*4 A(9)
1070 A(11)=A11
1080 A(12)=A12
1090 A(13)=A13
1100 A(21)=A21
1110 A(22)=A22
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
E(I,J) = D(I,J)
1 RETURN
END

SUBROUTINE BESJ(X, N, BJ, D, IER)

X=ARGUMENT OF THE J BESSEL FUNCTION DESIRED
N=ORDER OF THE J BESSEL FUNCTION
 BJ=RESULTANT J BESSEL FUNCTION
D=REQUIRED ACCURACY
IER=RESULTANT ERROR CODE
0=NO ERROR
1=N IS NEGATIVE
2=X IS NEGATIVE
3=REQUIRED ACCURACY NOT OBTAINED
4=RANGE OF N COMPARED TO X NOT CORRECT

BJ=0.

IF(N)10,20,20
10 IER=1
RETURN
20 IFRS,30,31
30 IF(N.GT.0) RETURN
11 BJ=1.
12 RETURN
33 IER=2
RETURN
31 IF(X-15.)32,32,33
32 NTEST=20.+10.*X-X**2/3
GO TO 30
34 NTEST=90.+X/2.
36 IF(N-NTEST)40,38,38
38 IER=4
RETURN
40 IER=0
49 N=N+1
DPREV=0.
C
COMPUTE STARTING VALUE OF M

1780  C
1785  C  IF(X-.150,60,60
1790  50  MA=X+5
1800  60  GO TO 70
1810  60  MA=1.4MX+60/X
1820  70  MB=M*IF(X(X)/4+2
1830  MZERO=MAXO(MA,MB)
1840  C
1850  C  SET UPPER LIMIT OF M
1860  C
1870  C  MMAX=TEST
1880  100  DO 190 M=MZERO,MMAX,3
1890  C
1900  C  SET F(M,F(M-1)
1910  C
1920  C  FM1=1.E-15
1930  0  FM=0.
1940  C  ALPHA=0.
1950  C  IF(M-(M/2)+2)120,110,120
1960  110  JT=-1
1970  120  GO TO 130
1980  120  JT=1
1990  130  M2=M-2
2000  DO 160 K=1,M2
2010  C
2020  C  MK=M-K
2030  C  BMK=.5*FLOAT(MK)*FM1/X-FM
2040  C  FM=FM1
2050  C  IF((MK)150,140,150
2060  140  BJ=BMK
2070  150  JT=-JT
2080  160  S=1+JT
2090  160  ALPHA=ALPHA+BMK
2100  BMK=2.*FM1/X-FM
2110  C  IF(N(1)150,170,180
2120  170  BJ=BMK
2130  180  ALPHA=ALPHA+BMK
2140  C  BJ=BJ/ALPHA
2150  C  IF(ABS(BJ-BPREV)-ABS(D*BJ))200,200,190
2160  190  BPREV=BJ
2170  200  IER=3
2180  200  RETURN
2190  END
2200  C
2210  C  SUBROUTINE TRIMEO(NB,B)
2220  C  LOGICAL1 B(60),BL
2230  C  DATA BL/' '/
2240  C  NB=60
2250  C  DO 1 J=1,60
2260  1 IF(B(NB):NE.BL) RETURN
2270  1 NB=NB-1
2280  1 RETURN
2290  END
2300  C
2310  C  FUNCTION ACOSK(A)
2320  C  ACOSK=ACOS(MAX11-1.,MIN11.A))
2330  C  RETURN
2340  END

57
FILE = SYO:[111,12]HOSBD.FTN 27-SEP-83 14:30:18 <<<

SUBROUTINE HOSBD(IOUT, LOOP, IOE, IOPE, SLOPE, BUF, NB, 
1 FREQ, POLPER8H, PITCHSROLL, YAW, TILT, PHR)

REAL*8 T1, T2, T3, T4, SI, S2, S3, S4, T12, S22, V1, U5, U9, U4, Q1, Q2

REAL*4 BPNP, RC8M1AX, NC8UIN, T5, T6

REAL*8 SLOBEIS, SLOPE, STL8E(I, 101), SAVE(30)

BYTE BUF(I)

INTEGER*2 DEBUG, COMIN, GLINT

LOGICAL*2 GOULD, LP, OPENE, LPF, OLDGRF, MGRAPH, GDAUTO

COMMON/SWITCH/ IOE, THETA(14), GOULD, LP, OPENE, LPF, OLDGRF, MGRAPH,

1 GDAUTO, DEBUG, COMIN, GLINT, IBUG, LSBUG, ISTATS, IZETA, STATS, NRC8, IOE

C

LP=LOOP

IF(IOE.EQ.0) GO TO 20

C COMPUTE REQUIRED ANGLES FOR GRAPH

SLOPE=FLOAT(NRC8-1)/AMAX(1.001, THETA(2)-THETA(1))

CONST=1.0-SLOPE*THETA(1)

ANG=STATS

NS=0

IF(ANG.GE.THETA(1)) GO TO 12

ANG=ANG+STATS

GO TO 11

ANG=ANG+STATS

NS=NS+1

IF(ANG.GT.THETA(2)) GO TO 13

LP=FIX(SLOPE*ANG+CONST*.001)

NS=NS+1

READ(IOE,LP) T1, T2, T3, T4, T5, T6

S1=T1

T12=T1*T1

S2=(T12-T2)/2.DO

C

IF(S1*NS2.EQ.0.DO.AND.IOE.EQ.0) RETURN

IF(S1=NS2.EQ.0.DO) GO TO 4

S3=(T12*T1-3.DO*T12*T2+2.DO*T3)/6.DO


V1=S1

U2=2.DO*S2

470 U3=12.DO*S3

480 S22=82*S2

490 U4=6.DO*S2U2+12.DO*NS1*NS3+32.DO*NS4

500 C

Q1=6.DO*NS3/(S1*NS2)

520 Q2=-1.6DO+3.DO*NS1*NS3/62+3.DO*NS4/82

830 C

60
B=2.DO+S2/(S1*S1)
N=S1*(S1+S2+S3)/((2.DO*S22-3.DO*S1*S3))
P=1.DO/(NNB)
RCMAX=T5*XN2
RCMIN=(2.DO*T6-T5)*XN2
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
WRITE(IOP,1) S1,S2,S3,S4
FORMAT(1'S1=',D11.3,' S2=',D11.3,' S3=',D11.3,' S4=',D11.3)
WRITE(IOP,2) V1,U2,U3,U4,G1,G2
1 ' G1=',D11.3,' G2=',D11.3)
WRITE(IOP,3) B,N,N,P,RCMAX,RCMIN
FORMAT(1' B',D11.3,' N=',D11.3,' P=',D11.3,' RCXMAX=',D11.3,' RCXMIN=',D11.3)
RETURN

C
CONTINUE

C
PREPARE SKENNESS GRAPH

C
IF(I.EQ.0) CALL GRMODE('ALPHA',0)
IF(IZETA.EQ.0.AND.IZETA.EQ.0) CALL GRTKGT
IF(I.EQ.0.AND.IZETA.EQ.0.AND...NOT.GDOUT) READ(1OR,18) JK
CALL GRMAY(SAVE)
IF(I.EQ.0) CALL GRMODE('ERASE',3)
IF(I.EQ.0) CALL GRMODE('ERASE',3)
IF(I.EQ.0) CALL GRPLOT(11.,.5,.5,-1)
CALL GRH(V'
CALL GRAXIS(1.5,1.5,5.,0.,1.,0.,'WIDN',.5,2)
CALL GRAXIS(1.5,1.5,5.,.5,.5,.5,.5,'SKEENNESS',.5,2)
CALL GRPIC(1.,1.,1.,1.)
CALL GRSCY(0.,0.,1.,1.,)
CALL GRXY(4.,9.5,X,Y)
CALL GRTXY(BUF,MB)
CALL GRPNT(0.,X,Y,.,18.,0.)
CALL GRXY(18.,9.5,X,Y)
CALL GRXY(0.,9.5,30.,Y)
DT=1-DY
CALL GRTXTFREQUENCY',20)
CALL GRNUM('F',.5,.5,1.FREQ)
CALL GRTXT('GHZ',4)
CALL GRPNT(-1.,X,Y,.,0.,0.)
Y=1-DY
CALL GRTXT('POLARIZATION',50)
CALL GRTXY(POL,9)
CALL GRPNT(-1.,X,Y,.,0.,0.)
Y=1-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,PHIR)
CALL GRPRNT(-1,X,0,Y,.08,0.)
CALL GRMOVE(0.,0.)
DO 10 J=1.50
X=FLOAT(J)/2.5
W=W*(T+2.)/(T+1.)**2
X=W*(T-3.)/(T+1.)**2
CALL GRDRAW(W,0)
CALL GRDRAW(1.,2.)
CALL GRDRAW(-.5,0.)
DO 14 J=1.50
CALL GSXYM(SLOBE(2,J),SLOBE(3,J),.1,0.,.42)
CALL GSXYT('Y',1)
CALL GRNUM('F',-1,1,IFIX(SLOBE(1,J)),DUMMY)
CALL GRPRNT(-1.,SLOBE(2,J),0.,SLOBE(3,J),.08,0.)
PREPARE KURTOSIS GRAPH
CALL GRMODE('ALPHA',0)
CALL GRMODE('ERASE',8)
CALL GRAXIS(0,1.5,1.5,5.,0.,.1,6.2,'WIDTH',.5,2)
CALL GRAXIS(3,1.5,1.5,5.,60.,-2.,6.,.2,'KURTOSIS',.8,2)
CALL GRSCALE(-.2,.1,.6.)
CALL GSXY(4.,.6,.5,X,Y)
CALL GRTXT('BUF',MB)
CALL GRPRNT(0.,X,0,Y,.18,0.)
CALL GSXY(-.6,.6,.5,X,Y)
CALL GSXY(-.6,.6,.3,X,DY)
DY=DY-DY
CALL GRTXT('FREQUENCY',.20)
CALL GRNUM('F',-.1,2,FREQ)
CALL GRTEXT('GHZ', .4)
CALL GRPANT(-1, X, 0, Y, .08, 0.)
Y = Y-DY
CALL GRTEXT('Polarization', .50)
CALL GRTEXT('POL', 3)
CALL GRPANT(-1, X, 0, Y, .08, 0.)
Y = Y-DY
CALL GRTEXT('X SHADOWED', .20)
CALL GRNUM('F', -1, 2, PERSH)
CALL GRPANT(-1, X, 0, Y, .08, 0.)
Y = Y-DY
CALL GRTEXT('Pitch', .20)
CALL GRNUM('F', -1, 2, PITCH)
CALL GRPANT(-1, X, 0, Y, .08, 0.)
Y = Y-DY
CALL GRTEXT('Roll', .20)
CALL GRNUM('F', -1, 2, ROLL)
CALL GRPANT(-1, X, 0, Y, .08, 0.)
Y = Y-DY
CALL GRTEXT('Tilt', .20)
CALL GRNUM('F', -1, 2, TILT)
CALL GRPANT(-1, X, 0, Y, .08, 0.)
Y = Y-DY
CALL GRTEXT('Phir', .20)
CALL GRNUM('F', -1, 2, PHR)
CALL GRPANT(-1, X, 0, Y, .08, 0.)
C
CALL GRMOVE(0, 0)
DO 15 J = 1, 50
T = T*FLOAT(J)/2.5
W = W*(T+2.)/(T+1.)**2
G = G*.W/T**4./T+2.*2
DO 10 W = W-.01
Z = Z#.W-.2.
W = W#.Z+2./12.#W-Z)
Z = Z#.Z+Z
Z = Z#.Z+Z
Q = G#.Z+Z-.W*(Z2#.Z-.Z1#.Z)/(Z#1#.Z+Z2#.Z+Z#3#.Z)
CALL GRTEXT(W, G)
CALL GRDRAW(W, G)
W = W
dO 10 J = 1, 50
Z = Z-.W-.01
Z = Z#.W-.2.
W = W#.Z+2./12.#W-Z)
Z = Z#.Z+Z
Z = Z#.Z+Z
Q = G#.Z+Z-.W*(Z2#.Z-.Z1#.Z)/(Z#1#.Z+Z2#.Z+Z#3#.Z)
CALL GRTEXT(W, G)
CALL GRDRAW(W, G)
C
DO 17 J = 1, 50
K = K#.SLOBE(2, J), SLOBE(4, J), 1, 0, 42)
CALL GRTEXT('Y', 3)
CALL GRTEXT('Y', 3)
CALL GRNUM('F', -1, 1, SLOBE(1, J), DUMMY)
CALL GRPANT(-1, SLOBE(2, J), 0, SLOBE(4, J), .08, 0.)
CALL GRRE6(SAVE)
CALL GRHV('H')
RETURN
END
FILE = SYQ:[111,123]LOBE.FTN

COMPUTE RCS LOBE STATISTICS

OPEN UNIT=IOUT, NAME='LOBE.TMP', TYPE='OLD', RECORDSIZE=181, ACCESS='DIRECT', READONLY, SHARED

READ(IOL'1), IOUT, KLOBE, GLINT, NRCS, MLLOBE, THETA, ALPHA, DELTA,
1 Beta, SCALE, IZETA, IPCODE, WLEN, LBUG, NR, BUF, GAUTO,
2 FREQ, POL, PERSH, PITCH, ROLL, YAW, TILT, PHR, GSIZE
LBug4=LBug, AND.4
LBug6=LBug AND.8
IF(Delta(3).EQ.0.) GO TO 30
FORMAT('LOBE -- DELTA HAS NOT BEEN DEFINED')
IF(Delta(3).EQ.0.) GO TO 30
IF(LBug4.NE.0) WRITE(2,227) IOUT, KLOBE, GLINT, NRCS, MLLOBE, IZETA, IPCODE,
1 THETA, DELTA, ALPHA, SCALE, BETAN, WLEN
FORMAT(1X,714/1X,9F8.2/1X,9F8.2/1X,9F8.2)
10=IOW
IF(IOUT.EQ.1) GO TO 4
INITIALIZE LOBE FOR PRINTED OUTPUT

IF(IPCODE.EQ.0) GO TO 7
10=3
CALL ASNLUN(I0, 'GD', 0)
GO TO 7

INITIALIZE LOBE FOR GRAPHICAL OUTPUT

10=4
CALL QINIT(O,IZETA,LOG,YER)
CALL QMV('H')
CALL QPIQG(SIZE1,SIZE2,SIZE3,SIZE4)
    IF(IZETA.EQ.0) CALL QDEVI('TK','HARD',IER)
C
         LBUG DEFINITIONS:
         LBUG=1=>TOLOBE TRANSFORMATIONS
         LBUG=2=>TOLOBE OUTPUT
         LBUG=4=>INPUT & LOBE OUTPUT
         LBUG=5=>GLINT OUTPUT
         LBUG=7
         NANG=(DELTA(2)-DELTA(1))/DELTA(1)+1.0001
         IF(|OUT.EQ.0 OR IZETA.NE.0) WRITE(10,222) NANG
         CALL GSC1(SIZE1,SIZE2,SIZE3,SIZE4)

COMPUTE CONSTANT TERMS
C
         SBET=SIN(BETA)
         CSBET=COS(BETA)+1.
         SOL=29989
         IF=3.1415926
         TWOP=IF/180.
         RAD=PI/180.
         IF(OUT.EQ.0) CALL QSC1(SIZE1,SIZE2,SIZE3,SIZE4)

PERFORM ENTIRE SET OF LOBE STATISTIC COMPUTATIONS
(READ ENTIRE TEMPORARY FILE EACH TIME)
FOR EACH REQUIRED DELTA ANGLE

C
         ANG=DELTA(1)
         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
         FIRSTA=ANG
         SLOPE=FLOAT(NRC-1)/AMAX(1.,THETA(2)-THETA(1))
         CONST=1.-SLOPE*THETA(1)
         IF(LBUG4.NE.0) WRITE(2,222) SLOPE,CONST,ANG
         CALL GSC1(SIZE1,SIZE2,SIZE3,SIZE4)
         FORMAT(' SLOPE,CONST,ANG=)',3F10.3)

DO 100 LL=1,NANG

C
         DETERMINE NEXT INDEX INTO LOBE ARRAY
         IF(ANG.GT.THETA(2)) GO TO 101
         L=IFIX(SLOPE*ANG+CONST+.001)

C
         COMPUTE SUM AND STANDARD DEVIATION

         N=n-3
         SUM=0.
         STD=0.
         IF(N.LT.0) GO TO 3

         MDL=MDL+1
         N=MDL
         READ(10,L,MDL+1) (SLOBE(1,J),J=1,L)
1150 IF(SLOBE(1,L),EQ.0.) GO TO 1
1160 DO 19 K=2,4
1170 10 READ('NLB+K') (SLOBE(K,J),J=1,L)
1180 C
1190 IF(LBUG4.NE.0) WRITE(2,221) (SLOBE(1,L),L=1,4)
1200 221 FORMAT(1'RCS,POSN=',4E11.3)
1210 C
1220 N=N+1
1230 DO 2 J=1,4
1240 2 SIG(J,N)=SLOBE(J,N)
1250 C
1260 SUM=SUM+SIG(1,N)
1270 STD=STD+SIG(1,N)**2
1280 IF(N.LT.180) GO TO 1
1290 IF(N.EQ.180) WRITE(2,0)
1300 0 FORMAT(' RCS -- LOBE WILL USE ONLY THE FIRST 180 COMPONENTS')
1310 C
1320 C
1330 3 IF(N.LE.1) GO TO 100
1340 IF((IOUT.EQ.0,OR,IZETA.NE.0).
1350 AND,LL.EQ.1) WRITE(2,224) N
1360 224 FORMAT(10X,'USING',14,' SUBCOMPONENTS ')
1370 STD=SQRT(SUM/SUM-STD)
1380 IF(LBUG4.NE.0) WRITE(2,223) N,STD
1390 223 FORMAT(1*N,SUM,STD=',15.2E11.3)
1400 C
1410 C
1420 C
1430 C
1440 G=0.
1450 N1=N-1
1460 C
1470 DO 5 I=1,N1
1480 5 I=I+1
1490 DO 6 J=1,N1
1500 6 G=G+SIG(I,J)*SIG(I,J)+SIG(4,I)-SIG(4,J))**2
1510 S=(SIG(I,J)-SIG(I,J))**2
1520 NF=SQRT(1/G/(SOL/STD))
1530 IF(LBUG4.NE.0) WRITE(2,224) NF
1540 224 FORMAT(1*N=1.E0)
1550 C
1560 C
1570 C
1580 C
1590 C
1600 C
1610 C
1620 C
1630 C
1640 COMPUTE RCS LOBE II
1650 C
1660 A11=0.
1670 A12=0.
1680 A22=0.
1690 C
1700 DO 6 I=1,N1
1710 6 I=I+1
1720 T1=SIG(I,I)*SIG(I,J)
1730 C
1740 T2=(SIG(I,J)**2)**2
1750 COMPUTE A MATRIX (A(1,2)=A(2,1))

1750 C T3=(SIG(3,1)-SIG(3,J))*C3BET1
1760 C
1770 C
1780 A11=A11+T1*T2*T3
1790 A12=A12+T1*T2*T3
1800 A22=A22+T1*T2*T3
1810 C CONTINUE
1820 C
1830 C COMPUTE F(DELTA) VS DELTA
1840 C
1850 DELT=560./DELT(A)
1860 DELT=50.
1870 DELT=1.
1880 IF(LBUG4,NE.0) WRITE(2,225) DELT,A11,A12,A21,A22
1890 FORMAT(' DELT= ',E11.3,' A11= ',E11.3,' A12= ',E11.3)
1900 IF(LBUG4,GE.0) WRITE(2,226) DELT,A11,A12,A21,A22
1910 IF(LBUG4,GT.359.0) GO TO 11
1920 ND(IDEL,1)=DEL
1930 D1=SIN(DEL*RAD)
1940 D2=COS(DEL*RAD)
1950 T1=A11*D1+A12*D2
1960 T2=A21*D1+A22*D2
1970 A=D1*T1+D2*T2
1980 ND(IDEL,2)=SQRT(A)/(WLEN*STD)
1990 IF(IDEL.1) GO TO 10
2000 C DEL=DEL+DELT
2010 C ND=IDEL+1
2020 C GO TO 10
2030 C
2040 C
2050 C
2060 C PRINT NF AND ND
2070 C
2080 C WRITE(10,12) ANG,NF
2090 C FORMAT(' ORCS LOBE OUTPUT',5X,'THETA= ',F6.1,' NF(GHZ)= ',F6.3/
2100 C 9 5X,'DELTA',5X,'ND'/)
2110 C
2120 C DO 15 J=1,IDEL
2130 C ND(J,1)=AMOD(ND(J,1)+360.,360.)
2140 C IF(IOUT.EQ.1) GO TO 15
2150 C
2160 C PRINT NF AND ND
2170 C
2180 C WRITE(10,12) ANG,NF
2190 C FORMAT(' ORCS LOBE OUTPUT',5X,'THETA= ',F6.1,' NF(GHZ)= ',F6.3/
2200 C 9 5X,'DELTA',5X,'ND'/)
2210 C
2220 C DO 15 J=1,IDEL
2230 C ND(J,1)=AMOD(ND(J,1)+360.,360.)
2240 C IF(IOUT.EQ.1) GO TO 15
2250 C GRAPH NF AND 1/ND
2260 C
2270 C
2280 C CONTINUE
2290 C
2300 C
2310 C
2350 IF(FMAX.EQ.ND(J,2)) FANG=ND(J,1)
2360 FMIN=MIN(FMIN,ND(J,2))
2370 ND(J,2)=ND(J,2)
2380 CONTINUE
2390 IF(FANG.GE.180.) FANG=FANG-180.
2400 IF(FANG.GT.80.) FANG=FANG-180.
2410 CALL GRRXY(ANG,0.,X,Y)
2420 Y=SIZE(2)-.95
2430 CALL GRSAY(SAVE)
2440 CALL GRKNT(1,0,0,IER)
2450 IF(IERR.NE.1) GO TO 17
2460 CALL GRTXT('NF(GHZ)',0)
2470 CALL GRPRNT(+1,2,1,-1,Y+.53,1,0.)
2480 CALL GRPRNT(+1,2,1,-1,Y+.38,1,0.)
2490 CALL GRXTX('MAJOR',0)
2500 CALL GRPRNT(+1,2,1,-1,Y+.23,1,0.)
2510 CALL GRPRNT(+1,2,1,-1,Y+.08,1,0.)
2520 CALL GRPRNT(+1,2,1,-1,Y+.08,1,0.)
2530 CALL GRPRNT(+1,2,1,-1,Y+.08,1,0.)
2540 CALL GRPRNT(+1,2,1,-1,Y+.08,1,0.)
2550 CALL GRPRNT(+1,2,1,-1,Y+.08,1,0.)
2560 CALL GRPRNT(+1,2,1,-1,Y+.08,1,0.)
2570 CALL GRNUMI('F',-1,3,NF)
2580 CALL GRPRNT(0,X,-1,Y+.53,10,0.)
2590 CALL GRNUMI('F',-1,2,FMAX)
2600 CALL GRPRNT(0,X,-1,Y+.38,1,0.)
2610 CALL GRNUMI('F',-1,2,FMIN)
2620 CALL GRPRNT(0,X,-1,Y+.23,1,0.)
2630 CALL GRNUMI('F',-1,2,FANG)
2640 CALL GRPRNT(0,X,-1,Y+.08,1,0.)
2650 CALL GRPIC('POLAR')
2660 Y=SIZE(2)
2670 CALL GRPIC(X,Y,X+1.,Y+1.)
2680 CALL GRSCL(0.,0.,(SCALE(3)-SCALE(1))/8.5,360.)
2690 CALL GRPIC(0.,Y-.28,11.,Y+.5)
2700 CALL GRWIND(-IDEL,ND(1,2),ND(1,1),IDUM,0,DUM,DUM,IDUM)
2710 CALL GRUPN
2720 CALL GRRES(SAVE)
2730 ANG=ANG+DELTA(3)
2740 CONTINUE
2750 IF(GLINT.EQ.0) CALL EXIT
2760 C COMPUTE GLINT LOBE III
2770 C OPEN(UNIT=IOT,NAME='LOBE.SCR',TYPE='SCRATCH')
2780 C DO 38 IALP=1,91,90
2790 C PAGE PLOT
2800 C IF(OUT.EQ.0) GO TO 50
2810 C IF(IzetA.EQ.0.AND.GDAUTO) CALL GRMODE('ALPHA',0)
2820 C IF(IzetA.EQ.0.AND.GDAUTO) CALL GRTEK
2830 C IF(IzetA.EQ.0.AND..NOT.GDAUTO) READ(LOW,47) IJK
FORMAT(A2)

IF(IZETA.GT.0) CALL GRPLOT(11.,0.,5.,-1)

CALL GRINIT(1,0,0,IER)

ALP=FLOAT(ALP-1)

REWIND IOT

LOOP ON THETA INCREMENTS

ANG=THETA(1)

DO 35 L=1,NRCS

IF(ALP.EQ.0.) GO TO 23

READ(IOT,20) N,XBAR,YBAR,SUM,B

IF(GLINT.EQ.1.OR.N.LE.1) GO TO 37

GO TO 39

3000 C

3010 C

3020 C

3030 C

3040 C

3050 C

3060 C

3070 C

3080 C

3090 C

3100 C

3110 C

3120 C

3130 C

3140 C

3150 C

3160 C

3170 C

3180 C

3190 C

3200 C

3210 C

3220 C

3230 C

3240 C

3250 C

3260 C

3270 C

3280 C

3290 C

3300 C

3310 C

3320 C

3330 C

3340 C

3350 C

3360 C

3370 C

3380 C

3390 C

3400 C

3410 C

3420 C

3430 C

3440 C

3450 C

3460 C

3470 C

3480 C

3490 C

3500 C

3510 C

3520 C

3530 C
DO 34 J=11,N
3580 C
3590 C T1=SIG(1,1)+SIG(1,J)
3600 C T2=SIG(2,1)+SIG(2,J)-2.*XBAR
3610 C T3=SIG(3,1)+SIG(3,J)-2.*YBAR
3620 C
3630 B(1,1)=B(1,1)+T1*T2*T2
3640 B(1,2)=B(1,2)+T1*T2*T3
3650 B(2,1)=B(2,1)+T1*T2*T2
3660 B(2,2)=B(2,2)+T1*T3*T3
3670 CONTINUE
3680 WRITE(IO,T20) N,XBAR,YBAR,SUM,B
3690 FORMAT(A2,7A4)
3700 C
3710 A1=COS(ALP)
3720 A2=SIN(ALP)
3730 C COMPUTE GLINT STANDARD DEVIATION
3750 STD=SQRT(BB)/(2.*SUM)
3760 IF(LBUGS.NE.0) WRITE(2,226) N,SUM,STD,XBAR,YBAR,ANG,B
3770 FORMAT('ON,SUM,STD=',13,2E11.3/
3780 1 ' XBAR,YBAR,ANG=',5E11.3/
3790 1 ' B=',4E11.3)
3800 C
3810 ND(L,1)=ANG
3820 ND(L,3)=STD
3830 IF(ALP.EQ.90.) GO TO 35
3840 C
3850 ND(L,2)=XBAR
3860 GO TO 35
3870 C
3880 ND(L,2)=YBAR
3890 C
3900 ANG=ANG+THETA(3)
3910 C
3920 IF(IOUT.EQ.1) GO TO 51
3930 C PRINT GLINT DISPLACEMENT VECTOR
3940 C
3950 WRITE(IO,55) ALP
3960 FORMAT('ORCS GLINT OUTPUT ALPHAS',F6.1/
3970 1 ' ANGLE DISPLACEMENT DIS-S DIS+S'/1
3980 DO 64 LP=1,NRC
3990 T1=ND(LP,2)-ND(LP,3)
4000 T2=ND(LP,2)+ND(LP,3)
4010 WRITE(IO,65) ND(LP,1),ND(LP,2),T1,T2
4020 FORMAT(F6.1,F11.3,F10.3,F10.3)
4030 GO TO 52
4040 C GRAPH GLINT DISPLACEMENT VECTOR
4050 C
4060 IF(I1ZETA.EQ.0) CALL GRMODE('ERASE',3)
4070 C
4080 CALL GRPLOT(GSIZE(3),GSIZE(2),0)
4090 CALL GRPLOT(GSIZE(3),GSIZE(2),1)
4100 DAX=GSIZE(3)-GSIZE(1)
4110 CALL GRAXIS(IO,GSIZE(3),GSIZE(2),.95,DAX,0.,SCALE(1),SCALE(3),
4120 1 IFIX(SCALE(5)),IFIX(SCALE(6)),'OBSERVATION ANGLE',90,1)
DISPLAY RADAR PARAMETERS

CALL GRXTXT('FREQUENCY', 30)
CALL GRNUM('F', -1, 2, FREO)
CALL GRPRINT(-1, X, 0, Y, 12, 0.)
Y = Y - DY
CALL GRXTXT('POLARIZATION', 50)
CALL GRXTXT('PITCH', 20)
CALL GRNUM('F', -1, 2, PITCH)
CALL GRPRINT(-1, X, 0, Y, 12, 0.)
Y = Y - DY
CALL GRXTXT('ROLL', 20)
CALL GRNUM('F', -1, 2, ROLL)
CALL GRPRINT(-1, X, 0, Y, 12, 0.)
Y = Y - DY
CALL GRXTXT('YAW', 20)
CALL GRNUM('F', -1, 2, YAW)
CALL GRPRINT(-1, X, 0, Y, 12, 0.)
Y = Y - DY
CALL GRXTXT('TILT', 20)
CALL GRNUM('F', -1, 2, TILT)
CALL GRPRINT(-1, X, 0, Y, 12, 0.)
Y = Y - DY
CALL GRXTXT('PHIR', 20)
CALL GRNUM('F', -1, 2, PHIR)
CALL GRPRINT(-1, X, 0, Y, 12, 0.)

DRAW AVERAGE DISPLACEMENT CURVE

CALL GRPIC(GSIZE(4), GSIZE(2), DAY, 90., -360., 360., 5.2)
CALL GRXTXT('DISPLACEMENT (DEG)', 'DAY, 92')

IF (ZETA.EQ.0) CALL GRMODE('ALPHA', 2)
CALL GRXTXT(BUF, ND)
CALL GRPRINT(0, 5.5, -1.7, 15, 0.)
CALL GRXTXT('MONOSTATIC GLINT FOR ALPHA', 0)
CALL GRNUM('F', -1, 1, ALP)
CALL GRPRINT(0, 5.5, -1, 6.75, 15, 0.)
IF (ZETA.EQ.0) CALL GRMODE('ALPHA', 3)

CALL GRXTXT('FREQUENCY', 30)
CALL GRNUM('F', -1, 2, FREO)
CALL GRPRINT(-1, X, 0, Y, 12, 0.)
Y = Y - DY
CALL GRXTXT('POLARIZATION', 50)
CALL GRXTXT('PITCH', 20)
CALL GRNUM('F', -1, 2, PITCH)
CALL GRPRINT(-1, X, 0, Y, 12, 0.)
Y = Y - DY
CALL GRXTXT('ROLL', 20)
CALL GRNUM('F', -1, 2, ROLL)
CALL GRPRINT(-1, X, 0, Y, 12, 0.)
Y = Y - DY
CALL GRXTXT('YAW', 20)
CALL GRNUM('F', -1, 2, YAW)
CALL GRPRINT(-1, X, 0, Y, 12, 0.)
Y = Y - DY
CALL GRXTXT('TILT', 20)
CALL GRNUM('F', -1, 2, TILT)
CALL GRPRINT(-1, X, 0, Y, 12, 0.)
Y = Y - DY
CALL GRXTXT('PHIR', 20)
CALL GRNUM('F', -1, 2, PHIR)
CALL GRPRINT(-1, X, 0, Y, 12, 0.)

CALL GRMOV(ND(1, 1), DID(ND(1, 2)))
DO 70 JRCS = 3, NRC
CALL GRDRAW(ND(JRCS, 1), DID(ND(JRCS, 2)))
70

69
```
4760 IF(GLINT.EQ.1) GO TO 52
4770 C GRAPH DISPLACEMENT +/- STD
4780 CALL GRMOVE(ND(1,1),ND(1,2)+ND(1,3))
4790 DO 49 J=2,NRC
4800 CALL GRDRAW(ND(J,1),ND(J,2)+ND(J,3))
4810 CALL GRMOVE(ND(1,1),ND(1,2)-ND(1,3))
4820 DO 59 J=2,NRC
4830 CALL GRDRAW(ND(J,1),ND(J,2)-ND(J,3))
4840 CALL GRUPPN
4850 CALL GRRES(SAVE)
4860 C COMPUTE GLINT LOBE IV
4870 C REPEAT STD CALCULATIONS FOR ELLIPSES
4880 C LOOP ON DELTA INCREMENTS
4890 52 ANG=FIRSTA
4900 DO 45 LL=1,NANG
4910 C DETERMINE NEXT INDEX INTO LOBE ARRAY
4920 C IF(ANG.GT.THETA(2)) GO TO 35
4930 L=FIX(SLOPE*ANG+CONST+.001)
4940 C NLB=NLB+1
4950 IF(NLB.GT.NLOBE) GO TO 40
4960 READ(IOL'NLB+1)(SLOBE(J,1),J=1,L)
4970 IF(SLOBE(1,L).EQ.0) GO TO 41
4980 DO 42 K=2,4
4990 READ(IOL'NLB+K)(SLOBE(K,J),J=1,L)
5000 C N=N+1
5010 DO 43 J=1,K
5020 SIG(J,N)=SLOBE(J,L)
5030 C COMPUTE E[sigma(X)]
5040 SUM=SIG(1,N)
5050 C COMPUTE SUM(sigma[gamma-X])
5060 XBAR=XBAR+SIG(1,N)*SIG(2,N)
5070 C COMPUTE SUM(sigma[gamma-Y])
5080 YBAR=YBAR+SIG(1,N)*SIG(3,N)
5090 GO TO 41
5100 C IF(N.LE.1) GO TO 45
5110 C COMPUTE greek-Xbar
5120 XBAR=XBAR/SUM
5130 C COMPUTE greek-Ybar
5140 YBAR=YBAR/SUM
5150 DO 48 MM=1,2
5160 DO 49 NN=1,2
5170 B(MM,NN)=0.
```
C(MM,NN) = 0.
DO 46 RR = 1, 2
DO 48 SS = 1, 2
430> C
440> DIM(MM,NN,RR,SS) = 0.
450> C
460> M1 = N - 1
470> DO 44 I = 1, M1
480> 11 = I + 1
490> DO 44 J = 11, N
510> C
520> T1 = SIG(1,1) + SIG(1,J)
530> T2 = SIG(2,1) + SIG(2,J) - 2 * XBAR
540> T3 = SIG(3,1) + SIG(3,J) - 2 * YBAR
550> T4 = (SIG(4,1) - SIG(4,J)) * CSBET1 + (SIG(2,1) - SIG(2,J)) * SNBET1 * N2
560> T5 = SIG(2,1) - SIG(2,J) * CSBET1 - (SIG(4,1) - SIG(4,J)) * SNBET1
570> CONTINUE
580> C
590> TMPARR(1,1) = T2 * T2
600> TMPARR(1,2) = T2 * T3
610> TMPARR(2,1) = TMPARR(1,2)
620> TMPARR(2,2) = T3 * T3
630> DO 44 MM = 1, 2
640> DO 44 NN = 1, 2
650> TEMP*T3*TMPARR(MM,NN)
660> B(MM,NN) = B(MM,NN) + TEMP
670> C
680> DIM(MM,NN) = DIM(MM,NN) + TEMP* T4
690> C
700> D(MM,NN,1,1) = D(MM,NN,1,1) + TEMP* T5*T5
710> D(MM,NN,1,2) = D(MM,NN,1,2) + TEMP* T5*T6
720> D(MM,NN,2,1) = D(MM,NN,2,1) + TEMP* T6*T5
730> D(MM,NN,2,2) = D(MM,NN,2,2) + TEMP* T6*T6
740> CONTINUE
750> C
760> A1 = COS(ALP)
770> A2 = SIN(ALP)
780> C
790> COMPUTE GLINT STANDARD DEVIATION
800> BB = B(1,1) * NA1 * A1 + B(1,2) * NA2 * A1 + B(2,1) * NA1 * A2 + B(2,2) * NA2 * A2
810> STD = SQRT(BB)/12. * SUM
820> CC = C(1,1) * NA1 * C(2,1) * NA2 * C(2,2) * NA1 * A2 + C(2,2) * NA2 * A2
830> COMPUTE N(deltal,f)
840> NF = SQRT(CC)/(2. * STD * SOL)
850> NF = NF/1. E9
860> C
870> IF(LBUGS .NE. 0) WRITE(2, 220) N, SUM, STD, XBAR, YBAR, B, C, D
880> FORMAT('ON, SUM, STD=', 15, 2E11.3/
890> 1 ' XBAR, YBAR=', 10E11.3/
900> 10 ' B=', 10E11.3/
910> 10 ' C=', 10E11.3/
920> 10 ' DELTA=', 10E11.3/
930> C
940> COMPUTE GLINT LOBE V
950> A(NM) = A1
960> AMN(2) = A2
970> DELTA = 360./DELTA(4)
5950> DEL=0.
5960> IDEL=1
5970> C
5980> 60 IF(DEL.GT.359.9) GO TO 61
5990> ND(IDEL,1)=DEL
6000> DRS(1)=SIN(DEL*RAD)
6010> DRS(2)=-COS(DEL*RAD)
6020> C
6030> DD=0.
6040> DO 66 J=1,IDE
6050> ND(J,1)=300.
6060> GO TO 60
6070> C
6080> 80 IF(JOUT.EQ.1) GO TO 65
6090> C
6100> WRITE(0,62) ANG,NF,ALP
6110> 62 FORMAT('ORCS GLINT OUTPUT',5X,'THETA=',F6.1,' NF(GHZ)='',F6.3/
6120> ' $ 5X,'DELTA'=',5X,'ND'=',5X,'ALPHA='',F6.1/)
6130> C
6140> DO 66 J=1,IDE
6150> TEMP=1./ND(J,1)*RAD)
6160> WRITE(10,64) ND(J,1),ND(J,2),TEMP
6170> 64 FORMAT(1X,F9.2,2F10.3)
6180> GO TO 46
6190> C
6200> C GRAPH NF AND ND
6210> C
6220> WRITE(0,62) ANG,NF,ALP
6230> 62 FORMAT('ORCS GLINT OUTPUT',5X,'THETA=',F6.1,' NF(GHZ)='',F6.3/
6240> ' $ 5X,'DELTA'=',5X,'ND'=',5X,'ALPHA='',F6.1/)
6250> C
6260> DO 66 J=1,IDE
6270> TEMP=1./ND(J,2)*RAD)
6280> WRITE(10,64) ND(J,1),ND(J,2),TEMP
6290> 64 FORMAT(1X,F9.2,2F10.3)
6300> GO TO 46
6310> C
6320> C CONTINUE
6330> FMAX=0.
6340> FMIN=1000.
6350> DO 66 J=1,IDE
6360> ND(J,2)=1./ND(J,2)*RAD)
6370> FMAX=MAX(FMAX,ND(J,2))
6380> IF(ND(J,2).LT.FMIN) FANG=ND(J,1)
6390> C
6400> ND(J,2)=ND(J,2)*1.
6410> CONTINUE
6420> IF(FANG.GE.180.) FANG=FANG-180.
6430> IF(FANG.GT.90.) FANG=FANG-90.
6440> CALL ORXY(ANG,0.,X,Y)
6450> Y=8SIZE(2)-.95
6460> CALL ORXY(1,0,0,IER)
6470> IF(IERR.NE.1) GO TO 67
0550 C
0560 CALL GRTXT('NF(GHZ)',0)
0570 CALL GRPRNT(+1,2,1,-1,Y+.53,.1,0.)
0580 CALL GRTXT('MAJOR',0)
0590 CALL GRPRNT(+1,2,1,-1,Y+.36,.1,0.)
0600 CALL GRTXT('MINOR',0)
0610 CALL GRPRNT(+1,2,1,-1,Y+.23,.1,0.)
0620 CALL GRTXT('ANGLE',0)
0630 CALL GRPRNT(+1,2,1,-1,Y+.06,.1,0.)
0640 C
0650 CALL GRTXT('F',-1,3,NF)
0660 CALL GRPRNT(0,X,-1,Y+.53,.1,0.)
0670 CALL GRTXT('F',-1,2,FMAX)
0680 CALL GRPRNT(0,X,-1,Y+.36,.1,0.)
0690 CALL GRTXT('F',-1,2,FMIN)
0700 CALL GRPRNT(0,X,-1,Y+.23,.1,0.)
0710 CALL GRTXT('F',-1,2,ANG)
0720 CALL GRPRNT(0,X,-1,Y+.06,.1,0.)
0730 C
0740 CALL GRCP('POLAR')
0750 Y=SIZE(2)
0760 CALL GRPIC(X,Y,X+1,Y+1)
0770 CALL GRBCL(0,0.,(SCALE(3)-SCALE(1))/0.5,360.)
0780 CALL GRPIC(X-1,Y-20,X+1,Y+5)
0790 CALL GRWIND(-1,DEL,ND(1,2),ND(1,1),IDUM,0,DUM,DUM,IDUM)
0800 CALL GRUPPN
0810 CALL GRRES(SAVE)
0820 C
0830 C
0840 AMG=AMG+DELTA(3)
0850 CONTINUE
0860 CLOSE(UNIT=10L)
0870 CLOSE(UNIT=10T)
0880 CALL EXIT
0890 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 peculiarities 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.

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 \( \theta_D \) and \( \phi_D \), respectively. For them, he must calculate the

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).
corresponding angles in the $R$ system (where shadowing is defined) as follows:

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

$$\cos \phi_R = \frac{c_{11} \sin \theta_D \cos \phi_D + c_{21} \sin \theta_D \sin \phi_D + c_{12} \cos \theta_D}{\sin \theta_R}$$

(note the subscript $R$ in the denominator above)

$$\sin \phi_R = \frac{c_{12} \sin \theta_D \cos \phi_D + c_{22} \sin \theta_D \sin \phi_D + c_{13} \cos \theta_D}{\sin \theta_R}$$

(And here too).

If

$$c_{11} \sin \theta_D \cos \phi_D + c_{21} \sin \theta_D \sin \phi_D + c_{12} \cos \theta_D = 0,$$

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

$$c_{12} \sin \theta_D \cos \phi_D + c_{22} \sin \theta_D \sin \phi_D + c_{13} \cos \theta_D$$

is greater than or less than zero, respectively.

If

$$c_{12} \sin \theta_D \cos \phi_D + c_{22} \sin \theta_D \sin \phi_D + c_{13} \cos \theta_D = 0,$$

then $\phi_R = 0$ or $\pi$ when

$$c_{11} \sin \theta_D \cos \phi_D + c_{21} \sin \theta_D \sin \phi_D + c_{13} \cos \theta_D$$

is greater than or less than zero, respectively.

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

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 matrices $P$, $R$, 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 $\hat{y}_D$, positive from $\hat{z}_D$ toward $\hat{x}_D$. $C$ 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 $\hat{z}_D$, positive from $\hat{x}_D$ toward $\hat{y}_D$. $C$ 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 $\hat{x}_D$, positive from $\hat{y}_D$ toward $\hat{z}_D$. $C$ 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 \textit{not} commutative), a rotation of the coordinate system from its \textit{last} position to a \textit{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.

\textbf{Examples}

Consider a $90^\circ$ pitch of axes, followed by a $180^\circ$ 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 $\theta = 60^\circ$, $0^\circ \leq \phi \leq 180^\circ$, the transformation $RP$ produces a portside viewing, from front to rear, over a conical cut at a $30^\circ$ elevation above the horizontal plane. In contrast, the transformation $PR$ produces a portside viewing, from rear to front, over a conical cut at a $30^\circ$ declination below the horizontal plane.
REFERENCES

**INITIAL DISTRIBUTION EXTERNAL TO THE APPLIED PHYSICS LABORATORY**

The work reported in JHU/APL TR-134 was done under Navy Contract N00014-83-C-5001 and is related to Task 5190, which is supported by the U.S. Air Force Rome Air Development Center.

<table>
<thead>
<tr>
<th>ORGANIZATION</th>
<th>LOCATION</th>
<th>ATTENTION</th>
<th>No. of Copies</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEPARTMENT OF DEFENSE</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DTIC</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DEPARTMENT OF THE NAVY</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Defense Advanced Research</td>
<td>Alexandria, VA</td>
<td>Accession</td>
<td>12</td>
</tr>
<tr>
<td>Project Agency</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Office of Naval Research</td>
<td>Arlington, VA</td>
<td>NTO, W. Kurowski</td>
<td>1</td>
</tr>
<tr>
<td>NAVSEA/SYSCOM</td>
<td>Washington, DC</td>
<td>R. W. Klemmer</td>
<td>1</td>
</tr>
<tr>
<td>NAVPLANTREPO</td>
<td>Washington, DC</td>
<td>SEA-9991</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>Laurel, MD</td>
<td>AIR-9903</td>
<td>2</td>
</tr>
<tr>
<td>DEPARTMENT OF THE AIR FORCE</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Rome Air Development Center</td>
<td>Griffis AFB, NY</td>
<td>J. H. Michel (OCTM)</td>
<td>1</td>
</tr>
<tr>
<td>ADY/TINTO</td>
<td>Eglin AFB, FL</td>
<td>R. Coh</td>
<td>1</td>
</tr>
<tr>
<td>AFWAC/AWNP, EW Division</td>
<td>Wright-Patterson AFB, OH</td>
<td>W. J. Baltz</td>
<td>1</td>
</tr>
<tr>
<td>HQ FTD (SQDA)</td>
<td>Wright-Patterson AFB, OH</td>
<td>V. D. Udeks</td>
<td>1</td>
</tr>
<tr>
<td>AFMC/SAE</td>
<td>Kirtland AFB, NM</td>
<td>J. Stegner</td>
<td>1</td>
</tr>
<tr>
<td>AFWC/EAEE</td>
<td>San Antonio, TX</td>
<td>T. Spear</td>
<td>1</td>
</tr>
<tr>
<td>Data Management Branch</td>
<td>Holloman AFB, NM</td>
<td>E. B. Aurbart</td>
<td>1</td>
</tr>
<tr>
<td>HQ 585 Test Group (RX)</td>
<td></td>
<td>J. E. Aar</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>D. Sion</td>
<td>1</td>
</tr>
<tr>
<td>CONTRACTORS</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Teledyne Micronetics</td>
<td>San Diego, CA</td>
<td>President</td>
<td>1</td>
</tr>
<tr>
<td>Delco Electronics</td>
<td>Santa Barbara, CA</td>
<td>R. Hyamini</td>
<td>1</td>
</tr>
<tr>
<td>Martin-Marietta Aerospace Corp.</td>
<td>Orlando, FL</td>
<td>C. W. Maness</td>
<td>1</td>
</tr>
<tr>
<td>(M/620)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hughes Aircraft Co.</td>
<td>El Segundo, CA</td>
<td>T. Cline</td>
<td>1</td>
</tr>
<tr>
<td>Space &amp; Communications Group</td>
<td>Seattle, WA</td>
<td>T. Dalby</td>
<td>1</td>
</tr>
<tr>
<td>Boeing Aircraft Co.</td>
<td>San Diego, CA</td>
<td>W. H. Anning</td>
<td>1</td>
</tr>
<tr>
<td>General Dynamics</td>
<td>Arlington, VA</td>
<td>J. E. Healy</td>
<td>1</td>
</tr>
<tr>
<td>Convair Division</td>
<td>La Jolla, CA</td>
<td>G. R. Curgis</td>
<td>1</td>
</tr>
<tr>
<td>Decision Science Applications, Inc.</td>
<td>Santa Barbara, CA</td>
<td>F. Chubb</td>
<td>1</td>
</tr>
<tr>
<td>Dare Technology Inc.</td>
<td>St. Louis, MO</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Science Applications, Inc.</td>
<td>Deters, MI</td>
<td></td>
<td></td>
</tr>
<tr>
<td>McDonnell Douglas Astronautics</td>
<td>Canton, MI</td>
<td></td>
<td></td>
</tr>
<tr>
<td>A. L. Malek</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>H. W. Klimach</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

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.