FINAL REPORT

on

Synthesis of Multiple Shaped Beam Antenna Patterns

W. L. STUTZMAN and E. L. COFFEY

Submitted To: National Aeronautics and Space Administration
Washington, D. C.

NASA GRANT NUMBER NGR 47-004-103

August 31, 1973

Electrical Engineering Department
Virginia Polytechnic Institute and State University
Blacksburg, Virginia 24061
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Introduction.</td>
<td>1</td>
</tr>
<tr>
<td>1.1 The Need for New Approaches.</td>
<td>1</td>
</tr>
<tr>
<td>1.2 A Practical Approach to Antenna Synthesis.</td>
<td>2</td>
</tr>
<tr>
<td>1.3 Scope of the Research.</td>
<td>4</td>
</tr>
<tr>
<td>2. Mathematical Modeling of Antennas</td>
<td>6</td>
</tr>
<tr>
<td>2.1 Equivalent Currents.</td>
<td>6</td>
</tr>
<tr>
<td>2.2 Representing Antennas as Finite Apertures.</td>
<td>8</td>
</tr>
<tr>
<td>2.3 Vector Radiation Fields.</td>
<td>11</td>
</tr>
<tr>
<td>2.4 Antenna Hardware Parameter Control</td>
<td>15</td>
</tr>
<tr>
<td>3. The Iterative Sampling Method for Planar Sources.</td>
<td>17</td>
</tr>
<tr>
<td>3.1 History of the Iterative Sampling Method</td>
<td>17</td>
</tr>
<tr>
<td>3.2 Pattern Evaluation - What is an Acceptable Pattern?</td>
<td>18</td>
</tr>
<tr>
<td>3.3 The Integral Equation.</td>
<td>19</td>
</tr>
<tr>
<td>3.4 Mathematical Development of the Method</td>
<td>20</td>
</tr>
<tr>
<td>3.5 Common Antenna Types</td>
<td>24</td>
</tr>
<tr>
<td>3.5.1 Line Sources.</td>
<td>24</td>
</tr>
<tr>
<td>3.5.2 Linear Array.</td>
<td>25</td>
</tr>
<tr>
<td>3.5.3 Rectangular Aperture.</td>
<td>25</td>
</tr>
<tr>
<td>3.5.4 Rectangular Array</td>
<td>26</td>
</tr>
<tr>
<td>3.5.5 Circular Aperture</td>
<td>26</td>
</tr>
<tr>
<td>3.5.6 Arbitrary Planar Array.</td>
<td>27</td>
</tr>
<tr>
<td>3.6 Calculation of Directivity</td>
<td>28</td>
</tr>
<tr>
<td>4. Examples of Computer Antenna Synthesis.</td>
<td>32</td>
</tr>
<tr>
<td>4.1 Common Antenna Types</td>
<td>32</td>
</tr>
<tr>
<td>4.2 Linear Antenna Synthesis.</td>
<td>39</td>
</tr>
<tr>
<td>4.3 Rectangular Antenna Synthesis.</td>
<td>47</td>
</tr>
<tr>
<td>5. Conclusions</td>
<td>60</td>
</tr>
<tr>
<td>References</td>
<td>62</td>
</tr>
</tbody>
</table>

6. Appendix: The ANTSYN Computer Program
   6.1 Introduction.                                                     | A-1  |
   6.2 Program Organization.                                            | A-2  |
   6.3 User's Guide to ANTSYN.                                          | A-4  |
   6.4 Program Variables.                                               | A-8  |
   6.5 Subroutine Descriptions.                                         | A-14 |
   6.6 Statement Listing of ANTSYN.                                     | A-24 |

7. Appendix: The ANTDATA Computer Program
   7.1 Introduction.                                                     | A-56 |
   7.2 Program Organization.                                            | A-57 |
   7.3 User's Guide to ANTDATA.                                         | A-59 |
   7.4 Program Variables.                                               | A-62 |
   7.5 Subroutine Descriptions.                                         | A-63 |
   7.6 Statement Listing of ANDATA.                                     | A-74 |

8.1 Input to ANTSYN ........................................ A-108
8.2 Output from ANTSYN ...................................... A-111
8.3 Input to ANTDATA .......................................... A-123
8.4 Output from ANTDATA ...................................... A-123
1

1. Introduction

This report presents the results of a one year research study on the problem of antenna synthesis. The original goal as outlined in the proposal [1] was to develop analytical and numerical techniques which would aid in the design of multiple shaped beam antennas. During the course of the research it was decided to expand the scope of the project to include virtually any pattern type in combination with many antenna types. This, it is hoped, will increase the number of specific antenna design problems to which this method may be applied.

1.1 The Need for New Approaches

Multiple shaped beam antennas are required for synchronous orbit satellites involving advanced multi-function communications. Anticipated applications include transfer of information for biomedicine, law enforcement, adult education, etc. The satellite should be capable of point-to-point communication between any two points within the continental United States. This will be achieved using multiple satellite antenna beams and a series of ground terminals. The antenna main beams must be shaped to give appropriate illumination of the ground stations. Also, the side lobe levels must be low to minimize interference between adjacent beams. These pattern requirements are quite severe and it is a difficult procedure to find an antenna which meets the pattern specifications and is suitable for a spacecraft environment. The classical approach to determining which antenna system is most suitable is one of repeated analysis. That is, combining and modifying "off-the-shelf" antennas in many ways until an acceptable radiation pattern is obtained. The antenna system may still not be practical because of large size, narrow bandwidth, etc. When this approach is used for many different
antenna systems the "paper study" stage becomes very costly. In addition, when new antenna pattern specifications are introduced another costly "paper study" is required.

To further illustrate the magnitude of the problem, a table of some of the variables that the antenna designer works with is given below.

<table>
<thead>
<tr>
<th>Radiation Pattern Variables</th>
<th>Antenna Variables</th>
</tr>
</thead>
<tbody>
<tr>
<td>I. Main Beams</td>
<td>I. Shape</td>
</tr>
<tr>
<td>A. Number</td>
<td>A. Linear</td>
</tr>
<tr>
<td>1. Single</td>
<td>1. Linear array</td>
</tr>
<tr>
<td>2. Multiple</td>
<td>2. Line source</td>
</tr>
<tr>
<td>B. Shape</td>
<td>B. Planar</td>
</tr>
<tr>
<td>1. Nominal</td>
<td>1. Planar array</td>
</tr>
<tr>
<td>2. Shape</td>
<td>2. Planar aperture</td>
</tr>
<tr>
<td>II. Side Lobes</td>
<td>II. Size</td>
</tr>
<tr>
<td>A. Nominal</td>
<td></td>
</tr>
<tr>
<td>B. Low</td>
<td></td>
</tr>
<tr>
<td>C. Complex</td>
<td></td>
</tr>
</tbody>
</table>

There is, of course, a large number of possible combinations of pattern variables and antenna variables. In addition, there are almost endless numbers of possibilities within each category and also other possible categories. The pattern variable categories for this research project are multiple shaped main beams with complex side lobe structure.

1.2 A Practical Approach to Antenna Synthesis

The synthesis problem may be formulated as follows: Given a desired antenna pattern (which may have multiple shaped beams plus controlled side lobe structure), we wish to find antenna structures which will approximate the desired pattern within acceptable limits subject to realizability criteria. Realizability
is broadly defined as the ability of the antenna to meet the system specifications of which it is a part. Specifications are often given on the following items:

a. Ability to form the necessary number of main beams.

b. Isolation levels between beams.

c. Polarization control.

d. Power handling capability.

e. Center frequency of operation.

f. Bandwidth

g. Efficiency

h. Size

i. Weight

j. Reliability

k. Pattern control (scanning and beam reshaping for changing user needs).

For satellite systems the specifications on the above items are frequently very demanding. Thus, the antenna designer lists all possible antenna systems which are capable of meeting the specifications. This is indeed the way one must face the problem. The next step is one of determining the design details of how one excites the antenna system in order to obtain an acceptable approximation to the desired pattern. This is classically done by cut-and-try analysis. Many excitations are studied on paper or in the lab until the pattern is found or the money runs out. It is proposed here that a true synthesis (as contrasted to cut-and-try analysis) approach be explored. In other words, given the antenna type to be used (as determined from the realizability criteria) and the desired antenna pattern, determine an excitation which approximates the pattern within acceptable limits. This is done for each candidate antenna type. A general synthesis procedure capable of handling many antenna types would allow the designer to synthesize a pattern once for each antenna type instead of using a lengthy and costly cut-and-try analysis for each one. The
final stage is then one of determining which antenna type does the best job of meeting pattern and system specifications.

The antenna design problem is then described in three stages:

1. Listing the antenna types which possibly can meet system specifications.
2. Determining the excitation of each antenna type required to meet the pattern requirements.
3. Singling out the one "best" antenna system.

The first two stages are frequently blended together, but ideally they should be distinct in order to avoid missing some candidate antenna types. The first and last stages are dependent upon the antenna designer's experience and judgment. The second stage is dependent upon an accurate mathematical antenna model (experimental design is ruled out for cost reasons) and available design techniques. This project provides a general synthesis technique as a design tool, thus eliminating the cut-and-try analysis approach. Its success in terms of practical application hinges on the availability of an accurate antenna model. In other words, once an excitation is determined by the synthesis method for a given antenna type and given pattern, how does one translate this into hardware? As will be explained later in this report there are several points in the synthesis method where hardware constraints can be inserted into the solution.

1.3 Scope of the Research

This report presents the results of research into the problem of finding an excitation of a given antenna such that the desired radiation pattern is approximated to within acceptable limits. This is to be done in such a fashion that boundary conditions involving hardware limitations may be inserted into the problem. The intended application is synthesis of multiple shaped beam
antennas. Since this is perhaps the most difficult synthesis problem an antenna engineer is likely to encounter, the approach taken was to include as a by-product capability for synthesizing simpler patterns. The synthesis technique has been almost totally computerized. The computer program and its use are described in detail elsewhere in this report.

The class of antennas which may be synthesized with the computer program are those which may be represented as planar (continuous or discrete) current distributions. The technique is not limited in this sense and could indeed be extended to include, for example, the synthesis of conformal arrays or current distributions on the surface of reflectors. The antenna types which the program is set up to synthesize are the following:

<table>
<thead>
<tr>
<th>Continuous Aperture Sources</th>
<th>Arrays</th>
</tr>
</thead>
<tbody>
<tr>
<td>Line source</td>
<td>Linear Array</td>
</tr>
<tr>
<td>Rectangular Aperture</td>
<td>Rectangular Array</td>
</tr>
<tr>
<td>Circular Aperture</td>
<td>Arbitrary Planar Array</td>
</tr>
</tbody>
</table>

Pattern specifications can be virtually anything—any number of main beams, any main beam shape, or any side lobe structure. Many examples are included in this report for illustration.
2. Mathematical Modeling of Antennas

An antenna can be synthesized by totally theoretical means only if an accurate mathematical model is available initially. It is the purpose of this chapter to summarize how one can approximately represent an antenna.

2.1 Equivalent Currents

It is often convenient to use equivalent currents to obtain the radiation fields from an antenna. Suppose the source antenna is entirely enclosed by a closed surface S. Let \( \vec{E}_1 \) and \( \vec{H}_1 \) be the values of the electric and magnetic field intensities on the surface S. The fields exterior to S can be found by using the equivalent electric and magnetic surface current sources:

\[
\vec{J}_S = \hat{n} \times \vec{H}_1 \tag{2-1}
\]

\[
\vec{J}_{MS} = -\hat{n} \times \vec{E}_1 \tag{2-2}
\]

where \( \hat{n} \) is the outward normal to S. The actual sources are replaced by these equivalent sources acting in free space. The equivalent sources produce exactly the same fields external to S as the original sources. The fields internal to S produced by currents given by (2-1) and (2-2) are zero. The fields exterior to S may be found using equivalent sources \( \vec{J}_S \) and \( \vec{J}_{MS} \) in one of the following ways:

1) Use \( \vec{J}_S \) and \( \vec{J}_{MS} \) over S
2) Use \( \vec{J}_S \) over S with S a perfectly magnetic conducting surface
3) Use \( \vec{J}_{MS} \) over S with S a perfectly conducting surface

Any one of these three equivalent source configurations can be used. The
first has the disadvantage of having two sources. The second and third configurations require that calculations must be made for the source in the presence of a conductor.

If the equivalent surface \( S \) is now a plane surface (closed at infinity such that actual sources are inside), calculations are simplified. Let \( S \) be the \( z = 0 \) plane and suppose the actual sources are on the left \( (z < 0) \). The surface normal is then \( \hat{n} = \hat{z} \). The simplification arises from the fact that in methods 2) and 3) the theory of images may now be employed to replace a current (electric or magnetic) immediately in front of a conductor (magnetic or electric) by a current of double its value acting in free space. Of course, image theory gives us the correct answer only for \( z > 0 \).

Using only currents acting in free space we may now use potential integral formulations to calculate the radiation. The electric and magnetic vector potentials for far field calculations are [2]

\[
\vec{A}(r) = \mu_0 \frac{e^{-jkr}}{4\pi r} \int \int \vec{J}_S(r') e^{jkr' \cdot \hat{r}'} dx'dy' \tag{2-3}
\]

\[
\vec{A}_M(r) = \varepsilon_0 \frac{e^{-jkr}}{4\pi r} \int \int \vec{J}_{MS}(r') e^{jkr' \cdot \hat{r}'} dx'dy' \tag{2-4}
\]

where \( \hat{r}' = x'\hat{x} + y'\hat{y} \) and the coordinate system is shown below.
The currents are doubled for cases 2) and 3).

2.2 Representing Antennas as Finite Apertures

An exact solution for \( z > 0 \) is obtained if the actual fields over the whole \( z = 0 \) plane is used in (2-1) and (2-2). Also, all three formulations of Section 2.1 give the same result. In many cases the fields \( E_a \) and \( H_a \) over a specific aperture in the \( z = 0 \) plane are known or can be approximated well. The fields in the \( z = 0 \) plane and outside of the aperture are assumed to be zero. This is an assumption but is usually a necessary one to obtain a solution. Examples of antennas for which this aperture concept is useful are the horn, lens, and reflector antennas. The three equivalent current formulations now provide approximate solutions which in general do not agree with each other [2]. However, the main features of the radiation pattern are usually unaffected by these approximations.

The approximate equivalent currents over the aperture are

\[
\vec{J}_S = \hat{z} \times \vec{H}_a \tag{2-5}
\]

\[
\vec{J}_{MS} = -\hat{z} \times \vec{E}_a \tag{2-6}
\]

The expressions for the fields in the far-field region of the aperture are

\[
E_\theta = -j\omega A_\theta - j\eta \omega A_{M\phi} \tag{2-7}
\]

\[
E_\phi = -j\omega A_\phi + j\eta \omega A_{M\theta} \tag{2-8}
\]

where \( \eta = \sqrt{\mu / \varepsilon} \) and the magnetic fields are found using the plane wave relation

\[
\vec{H} = r \times \vec{E} / \eta_0 \tag{2-9}
\]
As an example suppose we use only a magnetic current and the aperture electric field is y-directed. Then

\[
\mathbf{J}_\text{M} = J_{\text{M}x} \hat{x} = 2 x E_{\text{ay}} \hat{y} = E_{\text{ay}} \hat{x} \quad (2-10)
\]

and

\[
A_{\text{Mx}} = \frac{\varepsilon_0}{4\pi r} e^{-jkr} \int \int_{\text{aperture}} 2 E_{\text{ay}} e^{jk\mathbf{r} \cdot \mathbf{r}'} dx'dy' \quad (2-11)
\]

where the factor of 2 is necessary from image theory. Now

\[
A_{\text{M}0} = \cos \theta \cos \phi A_{\text{Mx}} \quad (2-12)
\]

\[
A_{\text{M}\phi} = -\sin \phi A_{\text{Mx}} \quad (2-13)
\]

So

\[
E_{\theta} = -j\omega \varepsilon_0 A_{\text{M}\phi} = +j\omega \varepsilon_0 \sin \phi A_{\text{Mx}} \quad (2-14)
\]

\[
E_{\phi} = +j\varepsilon_0 A_{\text{M}0} = j\eta_0 \cos \theta \cos \phi A_{\text{Mx}} \quad (2-15)
\]

And

\[
A_{\text{Mx}} = \frac{\varepsilon_0}{2\pi r} e^{-jkr} \int \int_{\text{aperture}} E_{\text{ay}} e^{jk(x' \sin \theta \cos \phi + y' \sin \theta \sin \phi)} dx'dy' \quad (2-16)
\]

So

\[
E_{\theta} = \frac{jke^{-jkr}}{2\pi r} \sin \phi F_y \quad (2-17)
\]

\[
E_{\phi} = \frac{jke^{-jkr}}{2\pi r} \cos \theta \cos \phi F_y \quad (2-18)
\]

where

\[
F_y = \int \int_{\text{aperture}} E_{\text{ay}} e^{jk(x' \sin \theta \cos \phi + y' \sin \theta \sin \phi)} dx'dy' \quad (2-19)
\]
If there is an $x$-directed component of the aperture electric field the far-field expressions become

$$ E_\theta = \frac{jke^{-jkr}}{2\pi r} (P_y \sin \phi + P_x \cos \phi) \quad (2-20) $$

$$ E_\phi = \frac{jke^{-jkr}}{2\pi r} \cos \theta (P_y \cos \phi - P_x \sin \phi) \quad (2-21) $$

If an equivalent electric current is used instead of a magnetic current, equations are obtained which are duals of those above:

$$ E_\theta = \frac{jkn e^{-jkr}}{2\pi r} \cos \theta (P_y \cos \phi - P_x \sin \phi) \quad (2-22) $$

$$ E_\phi = \frac{-jkn e^{-jkr}}{2\pi r} (P_y \sin \phi + P_x \cos \phi) \quad (2-23) $$

where

$$ \vec{F} = \iiint \vec{H}_a (x',y') e^{jk(x' \sin \theta \cos \phi + y' \sin \theta \sin \phi)} dx'dy' \quad (2-24) $$

If both electric and magnetic current sources are used a combination of the preceding results is obtained [2]. This approach is not used very often because a knowledge of both aperture fields is required and the resulting number of calculations required.

These solutions are exact if a complete knowledge of the fields over the entire aperture plane is available. This is usually never possible. In fact, some assumption about the aperture fields is made in addition to the assumption that the fields are zero outside the aperture. If the aperture is connected to an infinite plane perfect conductor the formulation using magnetic current only is exact within the limits of a knowledge of the tangential electric field over the aperture. This is true because the tangential electric field is zero over the perfect conductor and thus the equivalent magnetic current is also.
As a first approximation to the aperture fields frequently the so-called physical optics approximation is used. It assumes that the aperture fields are those incident upon it from the actual source. For example, the physical-optics fields in the aperture (or mouth) of a horn antenna are those of the waveguide feed [4].

Frequently aperture antennas are not used, but rather one wishes to relate directly to a source current. In this case $\vec{J}_S$ is an actual current and its Fourier transform $\tilde{\Lambda}$ in (2-3) is used in the far-field expression (2-7). So quantitatively there is little difference from actual and equivalent current problems. The actual currents may be used in array antenna solutions.

If a current distribution can be expressed as follows

$$\vec{J}_S = \hat{x} J_{Sx}(x') J_{Sx}(y') + \hat{y} J_{Sy}(x') J_{Sy}(y')$$

(2-25)

it is referred to as being separable. In this case the two-dimensional Fourier transform, see (2-19) separates into two one-dimensional transforms. Thus each transform is that corresponding to a line source and the total pattern is the product of the patterns of two line sources. Most practical rectangular sources have separable distributions [3]. Thus, the aperture fields $\vec{E}_a$ and $\vec{H}_a$ are usually separate and render the two-dimensional integrals of (2-19) and (2-24) a product of one-dimensional integrals.

2.3 Vector Radiation Fields

In the radiation field (or far-field) the waves are locally plane and may be completely described by $\theta$ and $\phi$ components (for an antenna at the origin of a spherical coordinate system). There are also two field components in the aperture which give rise to the radiation fields. It is convenient
to describe the radiation fields in spherical coordinates and the aperture fields in Cartesian coordinates. This complicates the relationship between aperture and radiation fields. It is the purpose of this section to discuss this point.

If one uses the aperture electric field formulation the radiation fields are found from (2-20) and (2-21), which are rewritten below as

\[ E_\theta = E(r) \left[ \cos \phi F_x + \sin \phi F_y \right] \]  
(2-26)

\[ E_\phi = E(r) \left[ -\cos \theta \sin \phi F_x + \cos \theta \cos \phi F_y \right] \]  
(2-27)

where

\[ E(r) = \frac{jke^{-jk\rho}}{2\pi r} \]  
(2-28)

This can be cast in a matrix form

\[
\begin{bmatrix}
E_\theta(\theta, \phi) \\
E_\phi(\theta, \phi)
\end{bmatrix} =
\begin{bmatrix}
G_{\theta x} & G_{\theta y} \\
G_{\phi x} & G_{\phi y}
\end{bmatrix}
\begin{bmatrix}
F_x \\
F_y
\end{bmatrix}
\]  
(2-29)

where

\[ E_\theta(\theta, \phi) = \frac{E_\theta}{E(r)} \]  
(2-30)

\[ E_\phi(\theta, \phi) = \frac{E_\phi}{E(r)} \]  
(2-31)

and

\[ G_{\theta x} = \cos \phi \quad G_{\theta y} = \sin \phi \]  
(2-31)

\[ G_{\phi x} = -\cos \theta \sin \phi \quad G_{\phi y} = \cos \theta \cos \phi \]  
(2-31)

In still more compact form (2-29) becomes

\[ [E] = [G][F] \]  
(2-32)
This formulation is particularly convenient for synthesis problems. If a certain desired electric field behavior \([E]\) is known, then the corresponding desired \([F]\) is found from the solution of (2-32):

\[ [F] = [G]^{-1}[E] \]  

(2-33)

The determinant of \([G]\) is \(\cos \theta\). The inverse of \([G]\) then exists except for \(\theta = \pi/2\). This is equivalent to radiation in the plane of the source and can be avoided. \(F_x\) and \(F_y\) are related to the corresponding aperture field components \(E_{ax}\) and \(E_{ay}\) by Fourier transforms

\[
F_x(\theta, \phi) = \int \int E_{ax}(x', y') e^{jk(x' \sin \theta \cos \phi + y' \sin \theta \sin \phi)} dx' dy',
\]  

(2-34)

\[
F_y(\theta, \phi) = \int \int E_{ay}(x', y') e^{jk(x' \sin \theta \cos \phi + y' \sin \theta \sin \phi)} dx' dy',
\]  

(2-35)

from (2-19). The synthesis problem for vector fields is thus reduced to synthesizing \(F_x\) and \(F_y\) using (2-34) and (2-35). Since \(F_x\) depends only on \(E_{ax}\) and \(F_y\) depends only on \(E_{ay}\), the vector problem reduces to two scalar problems.

If the aperture fields used are magnetic fields (or electric currents) the electric fields in (2-34) and (2-35) are replaced by magnetic fields (or electric surface currents). Then \([G]\) becomes

\[
[G] = \begin{bmatrix}
\cos \theta \cos \phi & -\cos \theta \sin \phi \\
-\sin \phi & -\cos \phi
\end{bmatrix}
\]  

(2-36)

from (2-22) and (2-23).

The element pattern matrix \([G]\) may also be used to absorb element patterns of array antennas. Suppose that the principle of pattern multiplication can be used. Then
\[ F_x = \int \int I_x (x',y') \ e^{j\kappa} \ dx'dy' \]  
\[ (2-37) \]

where we let

\[ \alpha = x' \sin \theta \cos \phi + y' \sin \theta \sin \phi \]  
\[ (2-38) \]

becomes

\[ F_x = G_x \sum_{m=1}^{P} I_{xm} e^{j\alpha_m} = G_x F_{arx} \]  
\[ (2-39) \]

where

\[ \alpha_m = x'_m \sin \theta \cos \phi + y'_m \sin \theta \sin \phi \]  
\[ (2-40) \]

and \((x'_m, y'_m)\) are the element phase center locations. Similarly

\[ F_y = G_y \sum_{m=1}^{P} I_{ym} e^{j\alpha_m} = G_y F_{ary} \]  
\[ (2-41) \]

These element factors may be combined into \([G]\) giving

\[ [G_{ar}] = \begin{bmatrix} \cos \theta \cos \phi G_x & - \cos \theta \sin \phi G_y \\ - \sin \phi G_x & - \cos \phi G_y \end{bmatrix} \]  
\[ (2-42) \]

The antenna equation \((2-32)\) for the array problem becomes

\[ [E] = [G_{ar}] [F_{ar}] \]  
\[ (2-43) \]

where the \([F_{ar}]\) entries are the array factors

\[ F_{arx} = \sum_{m=1}^{P} I_{xm} e^{j\alpha_m} \]
\[ F_{ary} = \sum_{m=1}^{P} I_{ym} e^{j\alpha_m} \]  
\[ (2-44) \]

Example - A linear array of parallel short dipoles along the x-axis.

Since the current is y-directed we have

\[ F_x = 0 \]  
\[ (2-45) \]

\[ F_y = G_y F_{ary} \]  
\[ (2-46) \]
The short dipole pattern is

\[ G_y = \sin \beta \]  \hspace{1cm} (2-47)

where \( \beta \) is the spherical polar angle from the \( y \)-axis. But \( \cos \beta = \sin \theta \sin \phi \) so

\[ G_y = \sqrt{1 - (\sin \theta \sin \phi)^2} \]  \hspace{1cm} (2-48)

Now

\[ F_{ary} = \sum_{jk} I_{ym} e^{jk \alpha_m} \]  \hspace{1cm} (2-49)

where

\[ \alpha_m = x_m' \sin \theta \cos \phi \]  \hspace{1cm} (2-50)

since \( y_m' = 0 \).

2.4 Antenna Hardware Parameter Control

A mathematical model of an antenna is useful in design work when the parameter being varied in the model can be translated into hardware. In the synthesis problem we end up with an aperture distribution which will produce the desired radiation pattern. Suppose a circular aperture distribution has been synthesized. Then one must find, for example, a feed system for a reflector antenna which will produce the required field distribution over the aperture. Thus synthesis techniques are useful for a particular antenna only if its excitation is controllable in a known way. If hardware parameters (such as feed antenna size and position) are mathematical related to the aperture excitation, they may also be included in the antenna mathematical
model. The synthesis procedure then goes from pattern specification to hardware parameter output. Indeed, not many antennas are suited to this at this point in time. However, the synthesis technique presented in this report is capable accommodating several hardware limitations a priori.

Array antennas appear to be the most readily adaptable to synthesis. After specifying the desired radiation pattern, element positions, and element pattern, one obtains the required terminal currents from the synthesis technique. For a few antenna arrays the mutual coupling (or impedance) matrix is available. The required terminal voltages for each element may then be found as follows:

\[ [V] = [Z][I] \]  \hspace{1cm} (2-51)

where

\[ [V] = \begin{bmatrix} V_1 \\ V_2 \\ \vdots \end{bmatrix} = \text{terminal voltage matrix (computed from (2-51))} \]

\[ [I] = \begin{bmatrix} I_1 \\ I_2 \\ \vdots \end{bmatrix} = \text{terminal current matrix (found by synthesis)} \]

\[ [Z] = \begin{bmatrix} Z_{11} & Z_{12} & \cdots \\ Z_{21} & \cdots & \vdots \\ \vdots & \vdots & \ddots \end{bmatrix} = \text{mutual impedance matrix (known from calculations or experiment)} \]
3. **The Iterative Sampling Method for Planar Sources**

3.1 History of the Iterative Sampling Method

The iterative sampling method has been used previously for shaping the main beam [5,6] and controlling the side lobes [5,7,8,9] of line sources and uniformly spaced linear arrays. In this section the theory is extended to include any type of planar source. The method is applied to patterns which have multiple main beams that are shaped and also have controlled side lobe levels.

Many methods are available for synthesis of radiation patterns using one-dimensional sources. Although proponents of most of these methods usually claim that it is a simple matter to extend their method to two dimensions, it is, in fact, rarely simple [10]. This is supported by the fact that it is almost never done. The iterative sampling, on the other hand, has been extended to two dimensions.

The iterative sampling method allows one to suppress side lobes to very low levels over certain regions while relaxing the side lobe requirements for other regions. When applied to the multiple-beam problem for time-zone coverage, beam cutoff and low side lobes would be specified for Canadian coverage, while side lobe specifications above the horizon would be relaxed.

Using this iterative technique the designer has the option of examining the metamorphosis of pattern change (and corresponding source change) as it approaches the desired form. Thus many patterns and their corresponding source currents may be examined. The pattern which approximates the desired pattern to just within the specifications will have the least complicated
source current distribution. The iterative sampling method provides such a design.

Another interesting feature of this design approach is the possibility of using a measured pattern from an existing (or prototype) antenna system as a starting point. Then calculations can be made to reveal what changes in the source are required to make specified corrections in the pattern.

3.2 Pattern Evaluation - What is an Acceptable Pattern?

Patterns can be evaluated using one or more of several criteria. Examples are side lobe level, beam width, rate of cut-off from main beam, mean squared error (between actual and desired pattern), etc. Different synthesis methods provide patterns which perform well with respect to one of these criteria. For complex patterns involving multiple beams, shaped beams, and/or varying side lobe structure, the criteria mentioned above are inadequate. The most flexible means of pattern evaluation is that using upper and lower bounds. In other words, one specifies at any or all points of the radiation pattern how much the synthesized pattern can rise above and/or fall below the desired pattern. Thus, the designer specifies a desired pattern plus an upper and lower tolerance.

The tolerance method of evaluating a synthesized pattern allows one to shape a main beam to within a fraction of a dB of the desired pattern. At the same time the side lobe region can have an upper tolerance of say 1 dB over critical portions and several dB over other regions; the lower tolerance is usually unspecified in the side lobe region because side lobes can fall anywhere below the desired level and be acceptable. It has been shown that
this means of pattern specification together with the iterative sampling method will yield synthesized patterns which include essentially all of the classical patterns which optimize only one parameter (such as side lobe level, main beam cut-off, etc.). [6]

3.3 The Integral Equation

The $\theta$ and $\phi$ components of the electric field are desired to be of a certain relative level as a function of $\theta$ and $\phi$. The desired $E_\theta(\theta, \phi)$ and $E_\phi(\theta, \phi)$ are converted into desired $F_x(\theta, \phi)$ and $F_y(\theta, \phi)$ using (2-33). The synthesis problem is to find the aperture fields $E_{ax}(x', y')$ and $E_{ay}(x', y')$ which produce sufficiently accurate approximations to the desired $F_x(\theta, \phi)$ and $F_y(\theta, \phi)$, respectively. This amounts to solving the integral equations (2-34) and (2-35). Since these two equations are identical in form we will drop subscripts which refer to polarization, while remembering that two polarizations (alone or together) are possible. The integral equation which we wish to solve is then

$$F(u, v) = \iint_{\text{aperture}} E_a(x', y') e^{jk(x'u + y'v)} \, dx'dy' \quad (3-1)$$

where

$$u = \sin \theta \cos \phi \quad (3-2)$$
$$v = \sin \theta \sin \phi \quad (3-3)$$

$E_a$ and $F$ may correspond to either component ($x$ or $y$) of the aperture field. Define, normalized coordinate variables

$$s = x'/\lambda \quad (3-4)$$
$$t = y'/\lambda$$

and source function
\[ f(s,t) = \begin{cases} \lambda^2 F_a(x',y') & \text{over the aperture} \\ 0 & \text{elsewhere} \end{cases} \]  

Substituting (3-4) and (3-5) into (3-1) gives

\[ F(u,v) = \iint f(s,t) e^{j2\pi(su + tv)} \, dsdt \]  

This integral extends over the whole st-plane and is recognized as a two-dimensional Fourier transform. The analysis problem is straightforward.

Given an aperture distribution \( f \) we can calculate \( F \) from (3-6) by integration. The synthesis problem, however, is very difficult. Suppose we are given a desired pattern \( F_d(u,v) \), which can be found from a desired electric field using (2-33). We wish to find an \( f \) (which is aperture-limited) giving an \( F \) which approximates \( F_d \) in some specified manner.

### 3.4 Mathematical Development of the Method

The iterative sampling method will be employed to find an aperture distribution which gives a pattern that approximates the desired pattern within acceptable limits as specified by upper and lower tolerances. The iterative procedure begins with an original pattern \( F^{(o)}(u,v) \) and its corresponding source distribution \( f^{(o)}(s,t) \). The source is initially of a certain type, e.g. line source, rectangular aperture, linear array, etc. It also has fixed dimensions in terms of a wavelength. These initial parameters are determined by the designer as discussed in Chapter 1. The original excitation \( f^{(o)}(s,t) \) of the antenna is one which gives a rough approximation \( F^{(o)}(u,v) \) to the desired pattern \( F_d(u,v) \). It can be found from any classical synthesis method, such as the Woodward-Lawson sampling method, or it can be an experimentally obtained pattern.
A series of corrections is added to the original pattern giving

\[ F^{(K)}(u,v) = F^{(0)}(u,v) + \sum_{i=1}^{K} \Delta F^{(i)}(u,v) \]  

(3-7)

\( K \) is the number of iterations and \( \Delta F^{(i)} \) is the \( i \)th iteration correction to the pattern. In general, each iteration is composed of a weighted sum of corrections as

\[ \Delta F^{(i)}(u,v) = \sum_n a_n^{(i)} G(u-u_n^{(i)}, v-v_n^{(i)}) \]  

(3-8)

where \( G(u-u_n^{(i)}, v-v_n^{(i)}) \) is a correction pattern centered at \( (u_n^{(i)}, v_n^{(i)}) \) and having a value of unity there. The \( \{a_n^{(i)}\} \) are weighting coefficients determined such that the current pattern is forced to equal the desired pattern at the correction point as follows

\[ a_n^{(i)} = F_d(u_n^{(i)}, v_n^{(i)}) - F^{(i-1)}(u_n^{(i)}, v_n^{(i)}) \]  

(3-9)

In other words, at the point \( (u_n^{(i)}, v_n^{(i)}) \) the amount \( a_n^{(i)} \) is added to the \( (i-1) \)th iteration pattern to obtain the desired pattern value at that point. The pattern is, of course, also changed at other points. If several corrections are applied in a given iteration of (3-8) the pattern will equal the desired pattern at the sample points only if the samples are uncorrelated. However, if the sample points are relatively far apart the correlation between samples can be very low. For a given iteration there are usually only a few corrections, frequently positioned to maintain symmetry. Thus if one abandons the idea that samples must be completely uncorrelated and replaces it with the concept that they should not be strongly correlated, the method is much more powerful and flexible. Also since the type of correction function is not based upon satisfying the property of being uncorrelated, the designer can choose one that is convenient.
For a given iteration then we have forced the pattern to be very close
(exactly equal if only one correction is used) to the desired pattern at the
sample points. The entire pattern is then recomputed and new corrections are
evaluated using (3-9). It has been found that the position of the samples
\((u_n^{(1)}, v_n^{(1)})\) which is most suitable is the location where the \((i-1)\)th iteration
pattern exceeds the tolerance by the greatest amount. Using this scheme the
number of samples is determined by the symmetry of the problem (if there is
no symmetry only one correction is applied per iteration). In this fashion
the largest corrections are applied first and the process tends toward con-
vergence. If the desired pattern specifications are too severe the iteration
procedure will converge to a certain point and then oscillate. This is not
a limitation of the method. It is rather a fundamental limitation. If a well-
behaved correction pattern \(G\) (examples are given in the next section) is used,
superdirective patterns will never be synthesized. Superdirective patterns
are to be avoided because of the accompanying complications of the source
distribution. For example, a small aperture is not capable of producing pat-
terns with an extremely sharp cut-off from the main beam unless superdirective
conditions are allowed. Using well-behaved correction functions the iterative
sampling method will not converge to a sharp cut-off desired pattern with tight
tolerances. In cases where the desired result has not been obtained one can
either use the final pattern as an approximation or start the iteration process
over again using a relaxed version of the pattern specifications.

Corresponding to each correction pattern there is a current correction
\(g_n^{(1)}(s,t)\) related to it as follows:

\[
G(u-u_n^{(1)}, v-v_n^{(1)}) = \iint_{\text{aperture}} g_n^{(1)}(s,t) e^{j(2\pi su + 2\pi tv)} \, ds \, dt \quad (3-10)
\]
The source distribution corresponding to the pattern of (3-7) is

\[ f^K(s,t) = f^0(s,t) + \sum_{i=1}^{K} \Delta f^{(i)}(s,t) \tag{3-11} \]

where

\[ \Delta f^{(i)}(s,t) = \sum_{n} a^{(i)}_n g^{(i)}_n(s,t) \tag{3-12} \]

The pattern \( F^K(u,v) \) and source \( f^K(s,t) \) are a Fourier transform pair, see (3-6). However, the only transform that has to be calculated is (3-10); all other patterns and sources are found by summing up the elementary pattern and source corrections, \( G \) and \( g \). This simplifies the required calculations greatly.

If the source is a planar array of isotropic point sources, we have

\[ f^K(s,t) = \sum_{\ell \in \mathbb{Z}} \sum_{m \in \mathbb{Z}} I^{(K)}_{\ell m} \delta(s-s_\ell, t-t_m) \tag{3-13} \]

where \( \delta \) is the Dirac delta function and \( I_{\ell m} \) are the currents for the \( \ell m \) element of the array. If the array elements are not isotropic the actual pattern is the array-element pattern times \( F(u,v) \) as discussed in Section 2.3. Let

\[ g^{(i)}_n(s,t) = \sum_{\ell \in \mathbb{Z}} \sum_{m \in \mathbb{Z}} g^{(i)}_{\ell m} \delta(s-s_\ell, t-t_m) \tag{3-14} \]

for arrays. Then (3-10) becomes

\[ G(u-u_n^{(i)}, v-v_n^{(i)}) = \sum_{\ell \in \mathbb{Z}} \sum_{m \in \mathbb{Z}} g^{(i)}_{\ell m} e^{j2\pi(s_\ell u + t_m v)} \tag{3-15} \]

For arrays substitute (3-14) into (3-12) giving

\[ \Delta f^{(i)}(s,t) = \sum_{n} a^{(i)}_n \sum_{\ell \in \mathbb{Z}} \sum_{m \in \mathbb{Z}} g^{(i)}_{\ell m} \delta(s-s_\ell, t-t_m) \tag{3-16} \]

and let

\[ \Delta f^{(i)}(s,t) = \sum_{\ell \in \mathbb{Z}} \sum_{m \in \mathbb{Z}} \Delta I^{(i)}_{\ell m} \delta(s-s_\ell, t-t_m) \tag{3-17} \]

So
\[ \Delta I_{lm}^{(1)} = \sum_n a_n^{(1)} g_{n \ell m} \]  
(3-18)

and

\[ I_{lm}^{(K)} = I_{lm}^{(0)} + \sum_{i=1}^K \Delta I_{lm}^{(i)} \]  
(3-19)

### 3.5 Common Antenna Types

In this section several common source types will be discussed. Correction functions \( G \) and \( g \) are also suggested. There are many possible functions, that one may use, including those obtained experimentally. Presented here are those functions which have been found to be applicable to many synthesis problems, are easily handled in the computer program, and which do not give superdirective patterns. The only Fourier Transform which must be performed in this method that of (3-10). Since the synthesis problem as formulated here is linear we can use the elementary functions as expansion functions to determine complex pattern and source functions (see (3-8) and (3-7), and (3-12) and (3-11)).

#### 3.5.1 Line Sources

The simplest line source is the uniformly illuminated one. A linear phase taper across the source is included to position the pattern maximum at, say, \( v_n^{(1)} \). The source correction function is

\[ g_n^{(1)}(t) = \begin{cases} 
\frac{L_{y\lambda}}{2} \exp(-j2\pi v_n^{(1)}t) & |t| \leq L_{y\lambda}/2 \\
0 & \text{elsewhere} 
\end{cases} \]  
(3-20)

where the line source has been positioned on the y-axis and is of length \( L_{y\lambda} \) wavelengths. The corresponding correction pattern is

\[ G(v-v_n^{(1)}) = \frac{\sin [L_{y\lambda}(v-v_n^{(1)})\pi]}{L_{y\lambda}(v-v_n^{(1)})\pi} \]  
(3-21)

This is the so-called \( \sin x/x \) pattern.
An excitation which gives no edge illumination is the triangular line source. Its pattern has lower side lobes but larger beam width than the uniform line source pattern. The excitation function is

\[
g_n(i)(t) = \begin{cases} 
\frac{1}{L_y} \left( 1 - 2|t|/L_y \right) \exp \left( -j2\pi \frac{(i)'}{L_y} \right) & |t| \leq L_y/2 \\
0 & \text{elsewhere}
\end{cases} 
\] (3-22)

The corresponding pattern found from (3-10) is

\[
G(v-v_n(i)) = \left[ \frac{\sin \left( \frac{L_y}{2} (v-v_n(i)) \pi/2 \right)}{L_y (v-v_n(i)) \pi/2} \right]^2
\] (3-23)

### 3.5.2 Linear Array

The uniformly illuminated, linear phase, equally spaced linear array has currents

\[
g_{nm} = \frac{i}{P} \exp \left( -j2\pi \frac{(i)'}{L_y} \right)
\] (3-24)

where \( t_m \) are the positions of the elements and equal \( m \) \( L_y \) and \( P \) is the total number of elements. The corresponding pattern is

\[
G(u-u_n(i)) = \frac{\sin \left( \frac{P(v-v_n(i)) \pi d}{L_y} \right)}{P \sin \left( (v-v_n(i)) \pi d \right)}
\] (3-25)

### 3.5.3 Rectangular Aperture

The uniformly illuminated, linear phase, rectangular aperture has excitation function

\[
g_n(s,t) = \begin{cases} 
\frac{1}{L_x} \frac{1}{L_y} \exp \left( -j2\pi \left( \frac{(i)'}{L_x} + \frac{(i)'}{L_y} \right) \right) & |s| \leq L_x/2 \\
0 & \text{elsewhere}
\end{cases} 
\] (3-26)

The pattern is

\[
G(u-u_n(i), v-v_n(i)) = \frac{\sin \left( \frac{L_x}{2} (u-u_n(i)) \right)}{L_x (u-u_n(i)) \pi} \frac{\sin \left( \frac{L_y}{2} (v-v_n(i)) \right)}{L_y (v-v_n(i)) \pi}
\] (3-27)
3.5.4 Rectangular Array

Consider a planar array which has equally spaced elements in the two principal directions. There are $P_x$ and $P_y$ numbers of elements along the x and y directions and interelement spacings of $d_{x\lambda}$ and $d_{y\lambda}$ wavelengths in the x and y directions. The element currents are

$$g_{n,m}^{(i)} = \frac{1}{P_x P_y} \exp \left[ -j2\pi (u_n^{(1)} s + v_n^{(1)} t) \right]$$  \hspace{1cm} (3-28)

The pattern is

$$G(u-u_n^{(i)}, v-v_n^{(i)}) = \frac{\sin [P_x (u-u_n^{(1)}) P_{d_{x\lambda}}] \sin [P_y (v-v_n^{(1)}) P_{d_{y\lambda}}]}{P_x \sin [(u-u_n^{(1)}) P_{d_{x\lambda}}] P_y \sin [(v-v_n^{(1)}) P_{d_{y\lambda}}]}$$  \hspace{1cm} (3-29)

3.5.5 Circular Aperture

Consider a uniform amplitude, linear phase, circular source a radius $a_\lambda$ wavelengths. The source function is

$$g_n^{(i)}(s,t) = \frac{1}{\pi a_\lambda} \exp \left[ -j2\pi (s u_n^{(1)} + t v_n^{(1)}) \right] \sqrt{s^2 + t^2} \leq a_\lambda$$  \hspace{1cm} (3-30)

The pattern for this source is, of course, found from (3-10). Since the details of this calculation have not been located in the literature, its details will be included here. First, it is more convenient to use cylindrical rather than rectangular coordinates to describe the source. Then we can write (3-30) as

$$g_n^{(i)}(\rho', \phi') = \frac{1}{\pi a_\lambda} \exp \left[ -j2\pi \rho' (\cos \phi' u_n^{(1)} + \sin \phi' v_n^{(1)}) \right] \rho' \leq a_\lambda$$  \hspace{1cm} (3-31)

The integral (3-10) over the source (3-31) is
\[ G (u-u_n (i), v-v_n (i)) \]
\[ = \int_{0}^{2\pi} \int_{0}^{\pi} \frac{1}{\pi a_{\lambda}} \exp \left\{ j 2\pi \rho' \left[ (u-u_n (i)) \cos \phi' + (v-v_n (i)) \sin \phi' \right] \right\} \rho' \, d\rho' \, d\phi' \]
\[ = \frac{1}{\pi a_{\lambda}} \int_{0}^{2\pi} \int_{0}^{\pi} \exp \left\{ j 2\pi \rho' \left( C \cos (\alpha - \phi') \right) \right\} \rho' \, d\rho' \, d\phi' \]  
(3-32)

where
\[ C = [(u-u_n (i))^2 + (v-v_n (i))^2]^{1/2} \]
(3-34)
\[ \alpha = \tan^{-1} \frac{u-u_n (i)}{v-v_n (i)} \]  
(3-35)

Now (3-33) is easily integrated as
\[ G (u-u_n (i), v-v_n (i)) = \frac{2\pi}{\pi a_{\lambda}} \int_{0}^{a_{\lambda}} J_0 (2\pi \rho' C) \rho' \, d\rho' \]
\[ = \frac{J_1 (2\pi a_{\lambda} C)}{2\pi a_{\lambda} C} \]  
(3-37)

If \( u_n (i) = v_n (i) = 0 \) we have
\[ G_n (u,v) = 2 \frac{J_1 (2\pi a_{\lambda} \sin \theta)}{2\pi a_{\lambda} \sin \theta} \]
(3-39)

which is the pattern of a uniform amplitude, zero phase, circular source. [4]

Also note that when \( u = u_n (i) \) and \( v = v_n (i) \), \( C = 0 \) and (3-38) becomes unity.

Thus \( (u_n (i), v_n (i)) \) is the pattern maximum.

3.5.6 Arbitrary Planar Array

There is a large class of antenna arrays which are not included in the previously mentioned linear and rectangular arrays. For example, the so-called
triangular array whose elements are spaced such that the fundamental lattice shape is a triangle. [11] This array provides a pattern similar to an equal size rectangular array but uses fewer elements. Also nonuniformly spaced arrays have applications. If the correction source is that of a uniform amplitude, linear phase array the pattern from (3-15) is

\[ G(u-u_n^{(i)}, v-v_n^{(i)}) = \frac{1}{M} \sum_{m=0}^{M} \exp \left\{ j2\pi \left[ s_m(u-u_n^{(i)}) + t_m(v-v_n^{(i)}) \right] \right\} (3-40) \]

There are M elements located at positions \((s_m, t_m)\) in the s,t plane.

3.6 Calculation of Directivity

The radiation pattern has been described for convenience in terms of the variables \(u\) and \(v\) instead of \(\theta\) and \(\phi\). This section discusses a few problems encountered when one wishes to calculate the directivity. The difference in the directivity between the original pattern and the final pattern (after iteration process is completed) is the gain loss. This number is usually small and may be positive or negative.

A derivation of the directivity expression using \(u\) and \(v\) coordinates has not been located in the literature, so its details are included here. The directivity is calculated as follows

\[ D = \frac{4\pi}{\Omega_A} \]

(3-41)

The beam solid angle \(\Omega_A\) is given by

\[ \Omega_A = \int_0^{2\pi} \int_0^{\pi} |F(\theta, \phi)|^2 \sin \theta \, d\theta \, d\phi \]

(3-42)

where \(|F(\theta, \phi)|\) is the field pattern normalized such that its maximum value is 1.0 and

\[ d\Omega = \sin \theta \, d\theta \, d\phi \]

(3-43)
It is frequently convenient to transform from the $\theta, \phi$ space to the $u,v$ plane using

\begin{align*}
  u &= \sin \theta \cos \phi \\
  v &= \sin \theta \sin \phi
\end{align*} \hspace{1cm} (3-44)

We are collapsing the spherical surface described by $\theta, \phi$ onto a planar surface through its equator giving a circular disk. There is an ambiguity here because points on the upper hemisphere ($\theta > \pi/2$) project onto the top of the $u,v$ disk and points on the lower hemisphere map onto the bottom of the $u,v$ disk. If we confine ourselves to only the upper hemisphere the transformation is one-to-one. In effect we modify (3-42) as

\[ \Omega_A = \int_{\theta=0}^{\theta=\pi/2} \int_{\phi=0}^{\phi=2\pi} |F(\theta,\phi)| \, d\Omega \hspace{1cm} (3-45) \]

This is assumed to contain most of the radiation. Back lobes are ignored if the antenna is in free space. If the antenna is backed by an infinite ground plane there are no back lobes and the formulation is exact (if $F(\theta,\phi)$ is exact).

The problem is to evaluate $\Omega_A$ using $F(u,v)$. This may be done in two ways. First, consider the projection of $d\Omega$ onto the $u,v$ plane; it is

\[ du \, dv = \cos \theta \, d\Omega \hspace{1cm} (3-46) \]

so

\[ d\Omega = \frac{du \, dv}{\cos \theta} \hspace{1cm} (3-47) \]

But from (3-44)

\[ \cos \theta = \sqrt{1 - u^2 - v^2} \hspace{1cm} (3-48) \]
so

\[ d\Omega = \frac{du \, dv}{\sqrt{1 - u^2 - v^2}} \quad (3-49) \]

Thus (3-45) becomes

\[ \Omega_A = \int \int_{u^2 + v^2 < 1} |F(u,v)|^2 \frac{du \, dv}{\sqrt{1 - u^2 - v^2}} \]

This result could also be obtained by a formal mathematical transformation of (3-45) as follows

\[ \Omega_A = \int \int_{u^2 + v^2 < 1} |F(u,v)|^2 \sin \theta \, J \, du \, dv \]

(3-51)

where \( J \) is the Jacobian given by

\[ J = \frac{\partial (\theta, \phi)}{\partial (u,v)} = \frac{1}{\frac{\partial (u,v)}{\partial (\theta, \phi)}} \]

(3-52)

and

\[ \frac{\partial (u,v)}{\partial (\theta, \phi)} = \begin{vmatrix} \frac{\partial u}{\partial \theta} & \frac{\partial u}{\partial \phi} \\ \frac{\partial v}{\partial \theta} & \frac{\partial v}{\partial \phi} \end{vmatrix} = \begin{vmatrix} \cos \theta \cos \phi - \sin \theta \sin \phi \\ \cos \theta \sin \phi \sin \theta \cos \phi \end{vmatrix} = \cos \theta \sin \theta \]

(3-53)

So

\[ \sin \theta \, J \, du \, dv = \sin \theta \, \frac{du \, dv}{\cos \theta \sin \theta} \]

(3-54)

\[ = \frac{du \, dv}{\sqrt{1 - u^2 - v^2}} \]

using (3-48). Substituting (3-54) into (3-51) gives the previous result (3-50).
Example: An isotropic antenna

\[ F(\theta, \phi) = \begin{cases} 1.0 & 0 \leq \theta \leq \pi/2 \\ 0 & \theta > \pi/2 \end{cases} \]

Using (3-45)

\[ \Omega_A = \int_0^{\pi/2} d\phi \int_0^{\pi} \sin \theta \, d\theta = 2\pi \]

Then (3-41) gives the directivity as \( D = 2 \). Also

\[ F(u, v) = 1.0 \quad u^2 + v^2 \leq 1 \]

Using (3-50)

\[ \Omega_A = \int \int \frac{1}{\sqrt{1 - (u^2 + v^2)}} \, du \, dv \]

Let \( r^2 = u^2 + v^2 \) then

\[ \Omega_A = \int_0^{2\pi} \int_0^1 \frac{r}{\sqrt{1 - r^2}} \, dr \, d\alpha \]

\[ = \int_0^{2\pi} \frac{1}{\sqrt{1 - r^2}} \, d\alpha \int_0^1 X^{-1/2} \left( -\frac{1}{2} \right) \, dX = 2\pi \]

where \( X = 1 - r^2 \). Again \( D = 2 \). The directivity for an isotropic antenna is 2 because one hemisphere has been neglected. We could define \( D \) as \( 2\pi/\Omega_A \) and obtain unity directivity for this isotropic antenna.
4. Examples of Computer Antenna Synthesis

The theory of chapter 3 has been computer programmed as the ANTSYN program and is discussed in detail in chapter 6. After an antenna synthesis problem has been solved using ANTSYN the results can be displayed using the ANTDATA program presented in chapter 7. The reader who intends on using these programs is referred to the appendices. In this chapter the results of several examples using the computer programs are presented. The examples given are only a small fraction of the number of antenna and pattern types which the method can handle. The important point to observe is that a wide variety of antenna shapes and pattern shapes can be synthesized using a single computer technique.

4.1 Common Antenna Types

In this section several simple antenna configurations are obtained from the computer programs. They are the patterns of the six correction functions discussed in section 3.5 and used in Subprogram PAT of the ANTSYN and ANTDATA programs. These patterns are examined for two reasons. First it serves as a program check. Many parameters are known about these patterns and can be compared to those obtained from the computer generated patterns to determine accuracy levels. Second, pattern plots of the correction functions provide a reference for visualizing synthesis capability of complex pattern shapes.

The first example is that of a uniform amplitude, uniform phase, line source. The length was chosen to be ten wavelengths. All of these patterns will change with changing antenna size, however the beam widths change in almost an inverse linear way with aperture size.
The side lobe levels do not depend on aperture size. In Fig. 4.1 is shown the pattern for this line source. (Aperture amplitude and phase distributions will be presented only when they are nonuniform.) This is the typical \( \sin \frac{x}{x} \) pattern. The linear array version of this pattern is shown in Fig. 4.2 – that of a 21 element, half-wavelength spaced, uniform amplitude, uniform phase, linear array.

A line source with a triangular amplitude taper and uniform phase is shown in Fig. 4.3. Its pattern is plotted in Fig. 4.4. Note its increased beam width and reduced side lobes relative to the uniform amplitude line source.

Next consider a rectangular aperture. For variety choose a size of \( 10\lambda \) by \( 20\lambda \). When excited with uniform amplitude and phase it has a pattern given by (3-27). The principal plane patterns are shown in Figs. 4.5 and 4.6. They are identical to patterns from line sources of the same length, e.g. Figs. 4.1 and 4.5 are the same. In Fig. 4.7 is shown a contour map of the pattern, which includes the visible region of the uv-plane. The contour levels are \( 0., -5., -10., \ldots, -40. \) dB. The contour levels may be distinguished by examining the profiles. Also the \(-35\) and \(-40\) dB contours are plotted as dashed (looking almost dotted) lines. The square region shown was divided into a grid of 151 by 151 points for plotting this figure. An excellent way to present two dimensional patterns is through the use of Fig. 4.8 which gives a three dimensional effect and provides a good feel for the pattern throughout the visible region.

The patterns of a uniformly excited rectangular array have been omitted because of their similarity to the continuous aperture patterns for element spacings of a half wavelength or less.
Figure 4.1 Pattern of a uniform amplitude, uniform phase, ten wavelength line source.

Figure 4.2 Pattern of a uniform amplitude, uniform phase, half-wavelength spaced, 21 element, linear array.
Figure 4.3 Amplitude distribution of a triangular amplitude, ten wavelength line source.

Figure 4.4 Pattern of a triangular amplitude, uniform phase, ten wavelength line source.
Figure 4.5 Profile along u-axis of the pattern from a uniform amplitude, uniform phase, 10 by 20 wavelength rectangular aperture.

Figure 4.6 Profile along v-axis of the pattern from a uniform amplitude, uniform phase, 10 by 20 wavelength rectangular aperture.
Figure 4.7 Contour map of the pattern from a uniform amplitude, uniform phase, 10 by 20 wavelength rectangular aperture.
Figure 4.8 Radiation pattern of a uniform amplitude, uniform phase, 10 by 20 wavelength rectangular aperture.

PATTERN = $\mathbb{S}$
Synthesis capability is provided for circular aperture by inclusion of an elementary pattern from such an aperture which is uniformly excited in amplitude. Figs. 4.9 and 4.10 show u and v axis profiles of the pattern from a uniform amplitude, uniform phase, five wavelength radius circular aperture. These plots are, of course, identical. A contour map of this pattern in the \( uv \)-plane for the whole visible region is shown in Fig. 4.11. The three dimensional view of the pattern is shown in Fig. 4.12.

The parameters of beam width, side lobe level and directivity have been calculated from theory and also obtained from this computer technique. They are all presented in Table 4.1 for the elementary patterns. In all cases except one the agreement is excellent. The directivity of a triangular line source is off by 8%. The reason for this is not known.

4.2 Linear Antenna Synthesis

In this section linear antennas are used to synthesize complex pattern shapes. Consider first a ten wavelength line source. Let the desired pattern and the upper and lower bounds be

<table>
<thead>
<tr>
<th>( v )</th>
<th>( F_d(v) )</th>
<th>( F_u(v) )</th>
<th>( F_L(v) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>(</td>
<td>v</td>
<td>&lt; 0.2 )</td>
<td>0. dB</td>
</tr>
<tr>
<td>( 0.4 \leq</td>
<td>v</td>
<td>&lt; 1.0 )</td>
<td>(-\infty)</td>
</tr>
</tbody>
</table>

The desired pattern is then a square beam with no side lobes but \(-40\text{ dB}\) side lobes will be tolerated. The original pattern is a Woodward-Lawson pattern with sample points at \( v = -0.2, -0.1, 0.0, 0.1, 0.2 \) and sample values of 1.0 at these points. This original pattern has excursions \(+0.86\) and \(-0.25\text{ dB}\) over the main beam and a side lobe level of \(-20\text{ dB}\) in the specified
Figure 4.9 Profile along u-axis of the pattern from a uniform amplitude, uniform phase, 10 wavelength diameter circular aperture.

Figure 4.10 Profile along v-axis of the pattern from a uniform amplitude, uniform phase, 10 wavelength diameter circular aperture.
Figure 4.11 Contour map of the pattern from a uniform amplitude, uniform phase 10 wavelength diameter circular aperture.
Figure 4.12 Radiation pattern of a uniform amplitude, uniform phase 10 wavelength diameter circular aperture.
Table 4.1 Pattern Parameters for Elementary Correction Patterns

<table>
<thead>
<tr>
<th>Antenna Type - ITYPE</th>
<th>Source Dimensions (λ)</th>
<th>Beam Width</th>
<th>Side Lobe Level (dB)</th>
<th>Directivity (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Theory</td>
<td>Computer</td>
<td>Theory</td>
</tr>
<tr>
<td>Uniform line source - 1</td>
<td>$L_y \lambda = 10$</td>
<td>0.0886</td>
<td>0.0886</td>
<td>-13.3</td>
</tr>
<tr>
<td>Uniform linear array - 2</td>
<td>$P_y = 21 \frac{d_x}{x \lambda} = 0.5$</td>
<td>0.0886</td>
<td>0.0886</td>
<td>-13.3</td>
</tr>
<tr>
<td>Triangular line source - 3</td>
<td>$L_y \lambda = 10$</td>
<td>0.128</td>
<td>0.128</td>
<td>-26.6</td>
</tr>
<tr>
<td>Uniform rectangular aperture - 4</td>
<td>$L_x \lambda = 10$</td>
<td>0.0886</td>
<td>0.0886</td>
<td>-13.3</td>
</tr>
<tr>
<td>Uniform rectangular array - 5</td>
<td>$P_x = 21 \frac{d_x}{x \lambda}$</td>
<td>0.0443</td>
<td>0.0443</td>
<td>-13.3</td>
</tr>
<tr>
<td>Uniform circular aperture - 6</td>
<td>$a \lambda = 5$</td>
<td>0.102</td>
<td>0.102</td>
<td>-17.6</td>
</tr>
</tbody>
</table>
side lobe region. The final pattern (which met the specifications) was obtained after 69 iterations. It deviated + 0.44 and - 0.42 dB over the main beam region and had a peak side lobe of -40.79 dB in the specified side lobe region. The pattern is plotted in Fig. 4.13. The corresponding current distribution is given in Fig. 4.14.

The same pattern was synthesized using triangular amplitude source correction coefficients (see Figs. 4.3 and 4.4). This allows for comparison of different correction functions. The original pattern was formed using corrections located at v = -0.2, 0.0, 0.2 and of amplitude 1.0. The final pattern was obtained after only 30 iterations as compared to 69 for uniform amplitude source correction functions. The pattern deviations about the desired level of 0.0 dB over the specified main beam region were +0.30 and -0.11 dB. The peak side lobe over the specified side lobe region was -41.68 dB. Thus the synthesized pattern was obtained with fewer iterations and was more comfortably within the tolerances than the pattern synthesized using uniform amplitude source correction coefficients. The pattern for this case is shown in Fig. 4.15. Its corresponding current distribution is plotted in Fig. 4.16. Note that the current is zero at the edges. This is because the source correction function of Fig. 4.3 is zero at the aperture edges. Zero edge illumination may be desirable in some situations.

Occasionally it is desirable to use linear arrays which are not equally spaced. The ITYPE = 7 of the ANTSYN program provides for general array synthesis. This example follows that of [14] which employs a different synthesis technique. The desired pattern is a square beam with upper and lower bounds as follows
Figure 4.13 Square main beam pattern synthesized using a 10 wavelength line source with uniform amplitude source correction functions.

Figure 4.14 Current amplitude distribution required to produce pattern of Fig. 4.13.
Figure 4.15 Square main beam pattern synthesized using a 10 wavelength line source with triangular amplitude source correction coefficients.

Figure 4.16 Current amplitude distribution required to produce pattern of Fig. 4.15.
The element positions used as input to the program were found from [14] and are \( t_m = 0., \pm 0.496, \pm 0.983, \pm 1.926, \pm 2.372, \pm 3.188, \pm 3.545 \). The original pattern for this 13 element array was formed using the same Woodward-Lawson specifications as the first example of this section. The original pattern has an excursion of 4.51 dB above the desired level of 0. dB over the main (square) beam region and none below. The side lobe level is -9.2 dB. Thus, in this case, the original pattern is quite far from desired performance. The final pattern obtained from ANTSYN took 15 iterations to meet the specifications. In fact, all side lobes were below -22. dB. The pattern is shown in Fig. 4.17. This compares to a side lobe level of -18.6 dB from [14]. The element currents for the two methods are similar.

### 4.3 Rectangular Antenna Synthesis

The multibeam capability of this technique is displayed with the synthesis of a pattern with pencil beams positioned at \((0.5, 0.5), (0.5, -0.5), (-0.5, -0.5), \) and \((0.5, 0.5)\). The side lobe upper limit was specified to be -25d B in the visible region outside the main beams, i.e. (for example, the beam centered at \((0.5, 0.5)\) was specified for \(0.38<u<0.64\) and \(0.38<v<0.64\)). The other beams were specified in a symmetric fashion. The region outside of these main beam regions had an upper limit of -25dB and no lower limit.

The antenna is a 10 by 10 wavelength square aperture. The original pattern was a Woodward-Lawson pattern with a correction coefficient of 1.0 and correction locations at each of the four main beam locations given above. The final pattern was obtained after 21 iterations. Profiles through the centers of the main beams (along u for \(v=0.5\) and along v for \(u=0.5\)) are shown in

| \(|v| \leq 0.26\) | \(F_d(v)\) | \(F_u(v)\) | \(F_L(v)\) |
|------------------|------------------|------------------|------------------|
| 0. dB            | +.42 dB          | - .42 dB         |

The element positions used as input to the program were found from [14] and are \( t_m = 0., \pm 0.496, \pm 0.983, \pm 1.926, \pm 2.372, \pm 3.188, \pm 3.545 \). The original pattern for this 13 element array was formed using the same Woodward-Lawson specifications as the first example of this section. The original pattern has an excursion of 4.51 dB above the desired level of 0. dB over the main (square) beam region and none below. The side lobe level is -9.2 dB. Thus, in this case, the original pattern is quite far from desired performance. The final pattern obtained from ANTSYN took 15 iterations to meet the specifications. In fact, all side lobes were below -22. dB. The pattern is shown in Fig. 4.17. This compares to a side lobe level of -18.6 dB from [14]. The element currents for the two methods are similar.

### 4.3 Rectangular Antenna Synthesis

The multibeam capability of this technique is displayed with the synthesis of a pattern with pencil beams positioned at \((0.5, 0.5), (0.5, -0.5), (-0.5, -0.5), \) and \((0.5, 0.5)\). The side lobe upper limit was specified to be -25d B in the visible region outside the main beams, i.e. (for example, the beam centered at \((0.5, 0.5)\) was specified for \(0.38<u<0.64\) and \(0.38<v<0.64\)). The other beams were specified in a symmetric fashion. The region outside of these main beam regions had an upper limit of -25dB and no lower limit.

The antenna is a 10 by 10 wavelength square aperture. The original pattern was a Woodward-Lawson pattern with a correction coefficient of 1.0 and correction locations at each of the four main beam locations given above. The final pattern was obtained after 21 iterations. Profiles through the centers of the main beams (along u for \(v=0.5\) and along v for \(u=0.5\)) are shown in
Figure 4.17 Square main beam pattern synthesized using a 13 element, 7 wavelength, nonuniformly spaced linear array.
Figures 4.18 and 4.19. The visible region includes abscissa values between -0.866 and 0.866. Thus, the high side lobes on each end of the profiles are outside the visible region. The contour map of the region $|u|$ and $|v| \leq 1.0$ is plotted in Figure 4.20. The visible region is a circle inscribed in the square shown. The three dimensional view is given in Figure 4.21.

In the next example a rectangular beam is synthesized using a 10 by 20 wavelength rectangular array. There are 20 elements spaced 0.5 wavelength in the s-direction and 40 elements spaced 0.5 wavelength in the t-direction. The pattern specifications are:

| $|u|\leq 0.2, |u|\leq 0.05$ | $F_d(u,v)$ | $F_{II}(u,v)$ | $F_r(u,v)$ |
|-----------------------------|------------|--------------|------------|
| $0.36\leq |u|\leq 0.50$             | 0.dB       | 1.0dB       | -0.9dB     |
| $0.12\leq |v|\leq 0.50$             | -20.0      | -18.4       | unspecified|

The pattern is unspecified at all other points of the uv-plane. The gap in specifications between the main beam and side lobe regions allows the main beam to roll off. The elementary correction functions used (see 4.1) will give side lobes below -20 dB outside the side lobe region specified. The original pattern is that of a Woodward-Lawson pattern with 1.0 correction coefficients at 15 sample points which are all possible combinations of -0.2, -0.1, 0.0, 0.1, and 0.2 in u and -0.05, 0.0, and 0.05 in v. The ANTSYN computer program converged to a final pattern which met specifications after 62 iterations. The principal plane patterns are shown in Figures 4.22 and 4.23. The contour map is plotted in Figure 4.24. The contours run from 0. to -40 dB in 5 dB steps and the -35. and -40 dB contours are dotted. The three-dimensional view is shown in Figure 4.25.
Figure 4.18 A multiple beam radiation pattern profile in the \( u \)-direction for \( v = 0.5 \) synthesized using a 10 by 10 wavelength aperture antenna. The visible region is for \( |u| \leq 0.866 \).

Figure 4.19 A multiple beam radiation pattern profile in the \( v \)-direction for \( u = 0.5 \) synthesized using a 10 by 10 wavelength aperture. The visible region is for \( |v| \leq 0.866 \).
Figure 4.20 Contour map of a multiple beam, low side lobe pattern synthesized from a 10 by 10 wavelength aperture.
Figure 4.21 Multiple beam, low side lobe pattern synthesized from a 10 by 10 wavelength aperture.
Figure 4.22 Profile along u-axis of a rectangular main beam, low side lobe pattern synthesized from a 20 element, 0.5 wavelength spaced by 40 element, 0.5 wavelength spaced rectangular array.

Figure 4.23 Profile along v-axis of a rectangular main beam, low side lobe pattern synthesized from a 20 element, 0.5 wavelength spaced by 40 element, 0.5 wavelength spaced rectangular array.
Figure 4.24 Contour map of a rectangular beam, low side lobe pattern synthesized from a 20 element, 0.5 wavelength spaced by 40 element, 0.5 wavelength spaced rectangular array.

PATTERN = SL
Figure 4.25 Rectangular beam, low side lobe pattern synthesized from a 20 element, 0.5 wavelength spaced by 40 element, 0.5 wavelength spaced rectangular array.
Tighter tolerances are easily achieved. This example is a rectangular beam with the following specifications:

<table>
<thead>
<tr>
<th>(u, v)</th>
<th>F_d(u,v)</th>
<th>F_H(u,v)</th>
<th>F_L(u,v)</th>
</tr>
</thead>
<tbody>
<tr>
<td>-0.2&lt;u&lt;0.2</td>
<td>0 dB</td>
<td>0.5 dB</td>
<td>-0.5 dB</td>
</tr>
<tr>
<td>-0.05&lt;v&lt;0.05</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.34&lt;</td>
<td>u</td>
<td>&lt;0.50</td>
<td>-∞</td>
</tr>
<tr>
<td>0.12&lt;</td>
<td>v</td>
<td>&lt;0.50</td>
<td></td>
</tr>
</tbody>
</table>

The pattern is unspecified at all other points in the uv-plane. The antenna used is again 10 by 20 wavelengths, but this time it is a continuous rectangular aperture. The original pattern is a Woodward-Lawson pattern with sample points and sample values as given in Section 8.1. The ANTSYN program was run and its output is shown in Section 8.2. Coincidentally, the number of iterations required to meet specifications, 62, was identical to the previous example which had weaker specifications. The plots of the final results were obtained using the ANTDATA, whose input for this example is discussed in Section 8.3. The principal plane profiles are shown in Figures 4.26 and 4.27. Note that the main beam ripple is less than +0.5 dB and the side lobes are below -25 dB. The contour map of the pattern is shown in Figure 4.28. In this case the lowest contour shown is -30 dB. The contour interval is still 5.0 dB and the maximum contour level is 0 dB. The three dimensional plot is shown in Figure 4.29. The floor of this plot is -35 dB, i.e. values -35 dB below the main beam are suppressed.
Figure 4.26 Profile along u-axis of a rectangular main beam, low side lobe pattern synthesized from a 10 by 20 wavelength rectangular aperture.

Figure 4.27 Profile along v-axis of a rectangular main beam, low side lobe pattern synthesized from a 10 by 20 wavelength rectangular aperture.
Figure 4.28 Contour map of a rectangular main beam, low side lobe pattern synthesized from a 10 by 20 wavelength rectangular aperture.
Figure 4.29 Rectangular main beam, low side lobe pattern synthesized from a 10 by 20 wavelength rectangular aperture.
5. Conclusions

In this report we have presented a rather detailed discussion of a general technique for antenna synthesis. This general approach was adopted to allow for synthesizing perhaps the most difficult type of radiation pattern -- that of multiple shaped main beams with side lobe control. The intended application for this particular pattern is for domestic satellite antenna systems. Included are specific models for several common antenna types plus capability for synthesizing special antennas. This was done so that a paper feasibility study can be carried out for many antenna types. After such a study, there remain many engineering decisions concerning realizability (see Chapter 1) for each candidate antenna.

The examples given in Chapter 4 illustrate some of the antenna types and pattern shapes which can be handled with this method. There appears to be no limit to the variety of antennas and patterns one can use. The convergence of the iterative sampling method is not guaranteed. This is due to the simple correction functions we use. However, this selection ensures that no superdirective patterns will be synthesized. If a very narrow beam correction pattern were used, the convergence rate would increase due to increased resolution of the correction patterns. When convergence to the desired pattern is not obtained, one can relax the specifications (usually by widening the region of no-specification between the main beam and side lobe regions).

The synthesis capability can be expanded by increasing the capability of the computer programs given in the appendices. For extremely large antennas the pattern digitizing should be increased. It is now a 51 by 51
grid for one quadrant (for quadrilateral symmetry). This grid should be increased in size for antennas of, say, many tens of wavelengths in size.

The program could be made more efficient by making an array for the correction functions (e.g. PAT, SOURCE), to avoid repeated computations. This would allow one to easily put in an experimental correction function also.

The synthesis of circular apertures is currently rather slow. This is because of the Bessel function calculations which are required. Perhaps a special purpose $J_1(x)$ routine could be written to avoid the general purpose IBM SSP routine.

The ANTDATA program requires a large amount of time to plot two and three dimensional plots. An ideal solution to this would be to replace the CALCOMP plotter with a video real time display for previewing the results. A hard copy attachment to the video terminal would also be very useful. Interactive computer graphics could be explored too.
References


6. Appendix: The ANTSYN Computer Program

6.1 Introduction

The ANTSYN computer program synthesizes finite planar antennas. It is based on the theory detailed in Chapter 3. On a large scale it can be considered to consist of four major functional blocks. The main program provides control of what operations are to be performed. The subroutines comprise the remaining blocks which function as input, computation, and output. The program as presented in this report is designed to handle antennas of most shapes and sizes. However, if an unusual antenna shape or one with certain limitations arising from hardware considerations is encountered, the modular subroutine structure allows the designer to change only selected subroutines to accommodate his particular problem.

This computer program has evolved over a period of six years and had been tested thoroughly. Because it is designed for wide application, it is, however, large and complex. Thus, if the potential user intends to make any subroutine changes he should have a good grasp of the FORTRAN IV language.

The patterns and source distributions are digitized and set up as two dimensional arrays. The pattern arrays FDES, FU, FL, AND F are specified in the U and V directions at MMAX and NMAX points beginning at STARTU and STARTV and incremented in intervals of DELTAU and DELTAV. The current arrays CURR and CURI are specified in the S and T directions at MCUR and NCUR points beginning at INITLS and INITLT and incremented in intervals of DELTAS and DELTAT.
6.2 Program Organization

A block diagram of the program with all of its subroutines is shown in Fig. 6.1. As mentioned in the previous section, the main program provides control over the subroutines which fall into three categories: input, computation, and output. The organization was selected to offer maximum flexibility. The program is intended to be very general, and it does provide for synthesis of many antenna types. However, if special antenna types are to be synthesized, the subroutines SPECPT and SPSOR can be used. Also, if the original pattern, correction pattern, or source correction are experimental, the subroutines ORGPAT, PAT, and SOURCE may be replaced with a data file of some sort.

The subroutines such as ANTSYN, SEARCH, UPDATE, CHECK, and CURRENT have been developed from a considerable amount of effort and should not be changed unless one thoroughly understands the details of the entire program. The other subroutines have been written with the possibility of change in mind.

The arrays F, FDES, FU, FL, CURR, and CURI are presently dimensioned at (51, 51). The arrays US, VS, and CORCOF are dimensioned at 500. The storage used on the Virginia Tech IBM System 370/155 computer is about 200 K. Of course, any example run with the present program with dimension requirements less than those in the present program will be run by the program. If larger dimensioning is necessary the appropriate dimension statements in the program must be changed and storage allocation increased commensurately.
Figure 6.1 Block diagram of ANTSYN program
6.3 User's Guide to ANTSYN

In this section a summary of the steps one must follow when using ANTSYN is presented. The steps are listed in Table 6.1 in order. The device refers to how the step is accomplished in the program. The location refers to where in the program the step is performed. The availability is either standard or special. Standard is the way it is listed in the statement listing of Section 6.6. Special means it is to be provided by the user with the device indicated. The steps will be discussed here. Further details can be found in the variable definitions and subroutine descriptions in the following sections.

Step 1. This step is entirely optional and is included to show how one can use data storage (on-line disk in this case). The variables NUMPAT, NUMTRK, NUMSKP and IPASS are read off of the storage unit. NUMPAT is the pattern number assigned to the previous job. The program adds one to NUMPAT to form the current job pattern number. NUMTRK is the track number on disk where the previous job data was stored. NUMSKP is an array whose subscripts correspond to disk storage track numbers. If this number is 0 or 1 there is not data stored on that track. This information is used in step 10 to write onto disk.

Step 2. The pattern parameters are read in from cards under

NAMELIST/PARAM/IDISK, ISYMM, ITRMAX, DELTAU, DELTAV, STARTU, STARTV, MMAX, NMAX, MCENT, NCENT

All of these variables are to be provided on cards following the FORTRAN Namelist format.

Step 3. Next the switches for control of the print out are read in from cards under

NAMELIST/IPRINT/FDESPT, FDESPR, FDESCN, FDBPT, FDBCN, FDBPR, FORGPT, FORGCN, FORGPR, ICURPT, ICURCN, ICURPR, FCURPT, FCURCN, FCURPR, DIRECT

Only those print outs desired need to have the appropriate switch variable provided on input of this Namelist, because default for all print outs is none.
<table>
<thead>
<tr>
<th></th>
<th>Device</th>
<th>Location</th>
<th>Availability</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Job assignment</td>
<td>Auxiliary storage</td>
<td>MAIN</td>
</tr>
<tr>
<td>2.</td>
<td>Pattern parameters</td>
<td>Cards; use Namelist PARAM</td>
<td>MAIN</td>
</tr>
<tr>
<td>3.</td>
<td>Output print control</td>
<td>Cards; use Namelist IPRINT</td>
<td>MAIN</td>
</tr>
<tr>
<td>4.</td>
<td>Antenna parameters</td>
<td>Cards; use Namelist PATIN</td>
<td>INPUT</td>
</tr>
<tr>
<td></td>
<td>If ITYPE &gt;7 also use subroutine SINPUT</td>
<td>INPUT</td>
<td>Special</td>
</tr>
<tr>
<td>5.</td>
<td>Desired pattern</td>
<td>Program statements to load FDES, FU and FL</td>
<td>DESPAT</td>
</tr>
<tr>
<td></td>
<td>or</td>
<td>Cards; call subroutine READ to load FDES, FU, and FL</td>
<td>DESPAT</td>
</tr>
<tr>
<td>6.</td>
<td>Original pattern and original source</td>
<td>Cards; read NORG, US, VS, CORG to generate original state using Woodward-Lawson method</td>
<td>ORGPAT</td>
</tr>
<tr>
<td></td>
<td>or</td>
<td>Cards or program statements; if Woodward-Lawson is not used, load F, CURR, and CURI, set NORG=0</td>
<td>ORGPAT</td>
</tr>
<tr>
<td>7.</td>
<td>Special correction pattern function</td>
<td>Program statements to generate values of PAT, ITYPE &gt;7</td>
<td>SPECPT</td>
</tr>
<tr>
<td>8.</td>
<td>Special correction source function</td>
<td>Program statements to generate companion to special pattern, ITYPE &gt;7</td>
<td>SPSOR</td>
</tr>
<tr>
<td>9.</td>
<td>Special location</td>
<td>Program statements to generate source coordinates given source array subscripts, use when ITYPE &gt;7</td>
<td>SPLOC</td>
</tr>
<tr>
<td>10.</td>
<td>Job storage</td>
<td>Programming to write job data onto storage unit</td>
<td>MAIN</td>
</tr>
</tbody>
</table>
Step 4. This step provides input concerning the particular antenna to be used. It is read in on cards under

NAMELIST/PATIN/LX, LY, PX, PY, DISX, DISY, INITLS, DELTAS, FINALS, INITLS, DELTAT, FINALT, NEIMT, ARAD, ITYPE, MCUR, NCUR

See write up on Subroutine INPUT in Section 6.5 for a list of which variables must be supplied in this Namelist for the various antenna types.

Step 5. The patterns FDES, FU, and FL are to be loaded in this step. This can be done in two ways. First programming can be provided in Subroutine DESPAT to give a value to this arrays at every point. Or, a Subroutine DESPAT can be used to call READ for these arrays and then cards are read to load the arrays.

Step 6. The original pattern and original source are loaded in this step. If the Woodward-Lawson technique is satisfactory all that is necessary is to provide data cards with sample information. The first card is the number of samples NORG and uses an I5 format. The succeeding cards (NORG in number) contain UORG, VORG, AND CORG (the sample locations and values) on a 3F10.0 format. The original pattern F and current CURR and CURI may be loaded in any other fashion if the user replaces ORGPAT with programming that loads them directly or calls some device and reads them. Set NORG=0 in ORGPAT if Woodward-Lawson technique is not used.

Steps 7 and 8. If correction pattern and source functions other than the seven standard ones available are desired by the user, Subroutines SPECPT and SPSOR are to be written. Use ITYPE >7.

Step 9. If correction pattern and source functions other than the seven standard ones available are desired by the user, Subroutine SPLOC is to be written. Use ITYPE >7.
Step 10. At the completion of a program data may be stored for future use, such as with the ANTDATA program. After looking at ANTSYN printout the user can decide if further display is desired. We can then use the data stored to plot patterns, currents, etc. IDISK is used to control whether data is to be stored.
### 6.4 Program Variables

#### 6.4.1 Correspondence Between Symbols Used in the Theory and Program Variables

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Computer Program Counterpart</th>
</tr>
</thead>
<tbody>
<tr>
<td>$a(i)$</td>
<td>CORCOF( )</td>
</tr>
<tr>
<td>$a_\lambda$</td>
<td>ARAD</td>
</tr>
<tr>
<td>$d_{x\lambda}$</td>
<td>DISX</td>
</tr>
<tr>
<td>$d_{y\lambda}$</td>
<td>DISY</td>
</tr>
<tr>
<td>$f^{(K)}(s,t)$</td>
<td>CURR(M,N)</td>
</tr>
<tr>
<td>$F(i)(u,v)$</td>
<td>F(M,N)</td>
</tr>
<tr>
<td>$F_d(u,v)$</td>
<td>FDES(M,N)</td>
</tr>
<tr>
<td>$g_n(i)(s,t)$</td>
<td>SOURCE(J,K,US(L),US(L),ITYPE)</td>
</tr>
<tr>
<td>$G(u-u_n(i),v-v_n(i))$</td>
<td>PAT(U-US(L),V-VS(L),ITYPE)</td>
</tr>
<tr>
<td>$L_{x\lambda}$</td>
<td>LX</td>
</tr>
<tr>
<td>$L_{y\lambda}$</td>
<td>LY</td>
</tr>
<tr>
<td>$P_x$</td>
<td>PX</td>
</tr>
<tr>
<td>$P_y$</td>
<td>PY</td>
</tr>
<tr>
<td>$s$</td>
<td>S</td>
</tr>
<tr>
<td>$t$</td>
<td>T</td>
</tr>
<tr>
<td>$u$</td>
<td>U</td>
</tr>
<tr>
<td>$u_n(i)$</td>
<td>US( )</td>
</tr>
<tr>
<td>$v$</td>
<td>V</td>
</tr>
<tr>
<td>$v_n(i)$</td>
<td>VS( )</td>
</tr>
</tbody>
</table>
6.4.2. Definition of Some Integer Variables Used in the Program

DIRECT  Input variable controlling calculation and print out of direct-
tivities DIRORG and DIRFNL; 0 No, 1 Yes – Default is 0. Original
pattern is to be of Woodward-Lawson type.

FCURCN Input variable controlling print out of contour map of final
current distribution; 0 No, 1 Yes – Default is 0.

FCURPR Input variable controlling print out of final current distribu-
tion profile or list; 0 None, 1 Profile (S and/or T axis) for con-
tinuous sources, 1 Table of element currents for arrays, 2 List (primarily for use with ITYPE = 7) – Default is 0.

FCURPT Input variable controlling print out of a listing of the final
current distribution; 0 No, 1 Yes – Default is 0.

FDBCN Input variable controlling print out of contour map of final
pattern in dB; 0 No, 1 Yes – Default is 0.

FDBPR Input variable controlling print out of final pattern profile;
0 No, 1 Yes – Default is 0.

FDBPT Input variable controlling print out of final pattern in dB; 0
No, 1 Yes – Default is 0.

FDESFR Input variable controlling print out of desired pattern profile
table; 0 No, 1 Yes (U and/or V axis) – Default is 0.

FDESPT Input variable controlling print out of a listing of desired
pattern; 0 No, 1 Yes – Default is 0.

IC Subscript of CORCOF( ) array for latest correction.

ICURCN Input variable controlling print out of contour map of initial
current distribution; 0 No, 1 Yes – Default is 0.

ICURPR Input variable controlling print out of initial current distribu-
tion profile or list; 0 None, 1 Profile (U and/or V axis), 2 List
(primarily for use with ITYPE = 7) – Default is 0.

ICURPT Input variable controlling print out of a listing of initial cur-
cent distribution; 0 No, 1 Yes – Default is 0.

IDISK Input variable controlling output of data to disk store;
0 No, 1 Write if successful (ISUC = 1), 2 Write all final pattern data.

IPASS Optional passwork to protect disk storage

ISUC Success counter; 0 If pattern specifications have not been met,
1 If they have.
**ISYMM**
Input variable describing the symmetry of the desired pattern; 0 if no symmetry, 1 for symmetry about U-axis, 2 for symmetry about V-axis, 3 for symmetry about both U and V axes, and 4 for symmetry about U, V and both 45 degree axes.

**ITER**
Number of iterations performed.

**ITRMAX**
Input variable giving the maximum number of iterations the program is allowed.

**ITYPE**
Input variable indicating what antenna type is to be used in the synthesis, the type descriptions follow.

<table>
<thead>
<tr>
<th>ITYPE</th>
<th>Antenna Type</th>
<th>Source Illumination Used to Form Correction Pattern</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Line Source Uniform</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Equally Spaced Uniform Linear Array</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Line Source Triangular</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Rectangular Aperture Uniform</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Rectangular Array Uniform</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Circular Aperture Uniform</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>General Array Uniform</td>
<td></td>
</tr>
<tr>
<td>GT7</td>
<td>SPECPT SPSOR</td>
<td></td>
</tr>
</tbody>
</table>

**MCENT**
Input variable - First subscript of pattern array where pattern is to be normalized to 0 dB.

**MCUR**
Input variable - Number of first subscripts of CURR and CURI arrays - Usually indicates quantization in S direction.

**MMAX**
Input variable - Number of points used in U direction for pattern arrays.

**NCENT**
Input variable - Second subscript of pattern array where pattern is to be normalized to 0 dB.

**NCUR**
Input variable - Number of second subscripts of CURR and CURI arrays - Usually indicates quantization in T direction.

**NELMT**
Total number of antenna array elements - Used as input when ITYPE=7.

**NMAX**
Input variable - Number of points used in V direction for pattern arrays.

**NORG**
Input variable - Number of samples in original pattern.
NUMPAT  Pattern number - Arbitrary sequence number for identifying synthesis problems.

NUMSKP( ) Variable on disk storage. If 0 space is available on track corresponding to subscript number. If 1 track contains previously generated data.

NUMTRK Reference number of a single track on disk storage.

ORGCN Input variable controlling print out of contour map of original pattern; 0 NO, 1 YES - Default is 0.

ORGPR Input variable controlling print out of original pattern; 0 NO, 1 YES (U and/or V axis) - Default is 0.

ORGPT Input variable controlling print out of original pattern; 0 NO, 1 YES - Default is 0.

PX Input variable - Number of array elements in X-direction.

PY Input variable - Number of array elements in Y-direction.

6.4.3. Definition of Some Real Variables Used in the Program

ARAD Input variable - Radius of circular aperture source in terms of a wavelength.

CONINT Interval between contour levels of CONTUR and PATCON print outs.

CONLOW Lowest contour level of CONTUR and PATCON print outs.

CONMAX Maximum level of CONTUR and PATCON print outs.

CORCOF( ) Correction coefficient

CORG( ) Correction coefficients (or sample values) for original pattern.

CURI( , ) Imaginary part of current.

CURR( , ) Real part of current.

DELCOST Increment above and below a contour level for which a function value is said to belong to that contour when using CONTUR and PATCON print outs.

DELTAS Input variable - Increment between print out points of current distribution in S direction.

DELTAT Input variable - Increment between print out points of current distribution in T direction.
DELTAU  Input variable - Increment between comparison points in U direction. Also, Increment between pattern print out points.
DELTAV  Input variable - Increment between comparison points in V direction. Also, increment between pattern print out points.
DIRFNL  Directivity of final pattern.
DIRORG  Directivity of original pattern.
DISX    Input variable - Spacing between antenna array elements in X direction normalized to a wavelength.
DISY    Input variable - Spacing between antenna array elements in Y direction normalized to a wavelength.
F( , )  Current pattern value.
FDES( , ) Input variable - Desired pattern value.
FINALS  Input variable - Final point of current distribution print outs in S direction.
FINALT  Input variable - Final point of current distribution print outs in T direction.
FINALU  Input variable - Final point of pattern comparison and print outs in U direction.
FINALV  Input variable - Final point of pattern comparison and print outs in V direction.
FL( , )  Input variable - Lower limit on synthesized pattern.
FNORM   Factor by which pattern F( , ) is divided to normalize it to 0 dB at the point (MCENT, NCENT).
FU( , )  Input variable - Upper limit on synthesized pattern.
INITLS  Input variable - Initial point of current distribution print outs in S direction.
INITLT  Input variable - Initial point of current distribution print outs in T direction.
LX      Length of antenna in X direction in wavelengths - For continuous aperture sources this is an input variable.
LY      Length of antenna in Y direction in wavelengths - For continuous aperture sources this is an input variable.
S       Source coordinate X normalized to a wavelength.
SS( )   Antenna array element position in S direction - Input variable for ITYPE = 7.
STARTU  Input variable - Starting point in U direction for pattern comparisons and print outs.
STARTV  Input variable - Starting point in V direction for pattern comparisons and print outs.
T     Source coordinate Y normalized to a wavelength.
TT(  ) Antenna array element position in T direction - Input variable for ITYPE - 7.
U     Pattern coordinate.
UORG( ) Input variable - Positions of sample points for original pattern in U direction.
US(  ) Positions of corrections (samples) in U direction.
V     Pattern coordinate.
VORG( ) Input variable - Positions of sample points for original pattern in V direction.
VS(  ) Positions of corrections (samples) in V direction.
6.5 Subroutine Descriptions

The subroutines are discussed in the order in which they appear in Fig. 6.1.

**SUBROUTINE INPUT**

This subroutine provides input to the program through the card reader. The Namelist labeled PATIN is used. Of the variables in this Namelist, only certain ones are to be specified for different values of ITYPE. The variables are defined in Section 6.4. The ones which are to be provided as input for each ITYPE are listed below. Remaining variables for a given ITYPE are to be omitted from the input deck.

<table>
<thead>
<tr>
<th>ITYPE</th>
<th>Variables to be provided as input for PATIN Namelist</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>LY, INITLT, DELTAT, FINALT, ITYPE</td>
</tr>
<tr>
<td>2</td>
<td>LY, PY, DISY, ITYPE</td>
</tr>
<tr>
<td>3</td>
<td>LY, INITLT, DELTAT, FINALT, ITYPE</td>
</tr>
<tr>
<td>4</td>
<td>LX, LY, INITLS, DELTAS, FINALS, INITLT, DELTAT, FINALT, ITYPE</td>
</tr>
<tr>
<td>5</td>
<td>LX, LY, PX, PY, DISX, DISY, ITYPE</td>
</tr>
<tr>
<td>6</td>
<td>INITLS, DELTAS, FINALS, INITLT, DELTAT, FINALT, ARAD, ITYPE</td>
</tr>
<tr>
<td>7</td>
<td>ITYPE, MCUR, NCUR</td>
</tr>
</tbody>
</table>

Greater than 7 **SINPUT**, written by user for his special problem.

For ITYPE=7 the source arrays are sized with MCUR and NCUR. The total number of elements in the array should be the product of MCUR and NCUR. If the number of elements is not easily factorable, one could always use an array with MCUR=number of elements and NCUR=1. This may require some dimension statement changes in the program. Two dimensional arrays for the source are used because of their convenience with the other source types.
INITLS, DELTAS, FINALS and INITLT, DELTAT, FINALT are used for prints of the source.

SUBROUTINE SINPUT

Currently this subroutine is a dummy subprogram. Inputs for programs not included in ITYPE through 7 should use this subroutine. It is to be written and added by the user. ITYPE as used in NAMELIST/PATIN/should have a value of 8 or greater. SINPUT is called from INPUT when ITYPE is 8 or greater.

SUBROUTINE READ (F, MMAX, NMAX)

This subroutine is used to load any two dimensional array in the program by reading in values off of cards. F is any real two dimensional array. MMAX rows and NMAX columns are to be loaded. The program in its present form does not use READ but subroutines DESPAT and ORGPAT can be used to call READ to load FDES, F, FU, FL, CURR, CURI. The arrays F, CURR, and CURI are then the original pattern, real part of original current, and the imaginary part of the original current.

The arrays are read in row by row. A new row is begun by a new card. The format is 6(I3,F10.0). The integer number is a multiplier, i.e., the following real number is to be repeated that many times. For example, if MMAX were 51 and all entries in the 5th row were to be 0.0, the card corresponding to the 5th row would have 51 in columns 2 and 3 and 0 in column 13.

SUBROUTINE DESPAT (FDES, FU, FL, MMAX, NMAX, STARTU, STARTV, DELTAU, DELTAV)

The purpose of this subroutine is to return arrays FDES, FU, and FL. They are all two dimensional and are loaded with MMAX rows and NMAX columns.
There are several ways that this subroutine can be written to load these arrays. Subroutine READ can be called for each of the arrays, if card input is convenient. If the patterns can be generated from FORTRAN expressions easily, the arrays can be loaded in the subroutine by incrementing thru U and V and assigning values to the arrays. This approach often avoids the need for a large input card deck.

The values of the patterns FDES, FU, and FL are to be positive real numbers and not dB values. This is done for computing efficiency. If one wishes to work with dB values it is an easy matter to convert dB to real values in this subroutine using $20.*\text{ALOG10}(\ )$. It is best for the pattern maximum, if specified, to be close to 1.0.

SUBROUTINE ORGPAT (F, MMAX, NMAX, STARTU, STARTV, DELTAU, DELTAV, CURR, CURI, MCUR, NCUR)

This subroutine is used to initialize the pattern array F and current arrays CURR and CURI. These represent the original pattern and real and imaginary part of the original current distribution. The pattern arrays are to be specified in rows and columns starting with STARTU and STARTV and extending for MMAX and NMAX points with DELTAU and DELTAV being the separation between points. The current arrays give current values at positions in the ST plane which depend on ITYPE; see SOURCE.

The program presently loads the arrays using the Woodward-Lawson synthesis method. This amounts to a 0th iteration. First, the number of samples in the original pattern is read in as NORC on a single card under an I5 format. Next the sample positions US and VS and the correction coefficients CORG are read in on a card under 3F10.0 format. See [12] for an excellent discussion of the Woodward-Lawson method. Note that the Taylor
line source method is handled with this technique also. [13]

If the original pattern is something which is not satisfactorily represented by a Woodward-Lawson type pattern, the user can substitute for this subroutine. If the original pattern and current are experimentally obtained, the READ subroutine can be called to read in the values from cards or the arrays can be generated using analytic functions. In ORGPAT, NORG should be set to zero when not using Woodward-Lawson method to generate original state.

**SUBROUTINE ANTSYN** (ISUC, MMAX, NMAX, FDES, FU, FL, ITRMAX, ISYMM, CORCOF, IC, US, VS, STARTU, DELTAU, STARTV, DELTAV, MCENT, NCENT, ITER, FNORM, F)

This subroutine carries out the iteration procedure. The arrays FDES, FU, and FL are input and are the pattern specifications loaded by DESPAT. F is initially the original pattern found from ORGPAT. This array is changed as iterations are performed and is the current synthesized pattern state. The subroutine cycles, or iterates, until either all points of F are between corresponding points of FU and FL or the maximum number of iterations ITRMAX is exceeded before each iteration F is normalized to 1.0 at the MCENT row and NCENT column. If the pattern specifications are not met, SEARCH is called to locate where the pattern exceeds its tolerances by the greatest amount. The weighting coefficient as given in (3-9) is returned as VAL and then is loaded into CORCOF. If the correction points are close to either the U or V axis but not on either and the pattern is symmetric, VAL is adjusted because of the strong correlation between the sample and its symmetrically placed samples. ANTSYN places other corrections corresponding to the level of symmetry ISYMM. The higher the level of symmetry in the desired pattern, the higher the level of symmetry of the corrections. After each correction,
UPDATE is called to recompute the pattern; CHECK is then called to see if corrections have ever been applied at the latest sample points. The iteration is now complete and control is transferred to the beginning of ANTSYN. This is repeated until the specifications are met or ITRMAX is exceeded. Then, control is returned to the main program where results are printed out.

**SUBROUTINE SEARCH (II, JL, VAL, FDES, FU, FL, F, MMAX, NMAX, STARTU, STARTV, DELTAU, DELTAV)**

This subroutine is called by ANTSYN subroutine to locate the point where the current pattern $F$ exceeds the upper and lower limit patterns $FU$ and $FL$ by the largest amount. This point is returned from the subroutine as $II$ and $JL$ of the pattern matrices. $II$ and $JL$ are also used as input and is the first point where specifications are not met as found in ANTSYN. The search begins here to avoid searching points that were covered in ANTSYN. The $V$ axis is searched in increments of $DELTAV$ for $NMAX$ points for each $U$ value, which itself is incremented in $DELTAU$ for $NMAX$ points. The search is limited to the visible region inside the unit circle. The maximum deviation above $FU$ or below $FL$ is returned as $VAL$ as computed by (3-9). The values of $II$, $JL$, and $VAL$ are printed out and flagged with **SEARCH** so that the user can see a "time history" of the corrections applied.

**SUBROUTINE UPDATE (IC, US, VS, CORCOF, F, MMAX, NMAX, FNORM, STARTU, STARTV, DELTAU, DELTAV)**

This subroutine updates the $F$ array to keep it current. Following each correction the array is recomputed in this subroutine. In ANTSYN the coordinates of the correction point are evaluated after $II$ and $JL$ are returned from SEARCH and assigned as $UI$ and $VI$ and then as $US(IC)$ and $VS(IC)$. So $IC$ is the subscript for $US$, $VS$, and $CORCOF$ corresponding to the most recent
assignments to those arrays. IC and the whole arrays US, VS, and CORCOF are input to UPDATE. Then the pattern F is calculated using these arrays and PAT.

FUNCTION PAT (U, V, ITYPE)

This subprogram returns the value of the correction function at the point U, V. It does this for function types determined by the value of ITYPE. The seven antenna types corresponding to the numbers 1 through 7 for ITYPE as given in the integer variable definition section of this chapter are discussed in detail in Section 3.5. If the user wishes to use some other correction function, a value of ITYPE greater than 7 will make PAT call SPECPT to find a value.

FUNCTION SPECPT (U, V, ITYPE)

Currently, this is a dummy subprogram. If a correction function other than one of the seven standard ones given in PAT is required, this function is to be used. The dummy subprogram is then replaced by a function which generates values at all points (U, V). See PAT.

SUBROUTINE CHECK (IC, VAL, US, VS, CORCOF, DELTAU, DELTAV)

This subroutine is used to save computing time and storage space. After a correction has been determined by SEARCH and applied by UPDATE, CHECK is called to see if the correction point has ever been used before. All previous sample points US and VS are searched for a match to US(IC) and VS(IC). If a match is found the correction coefficient CORCOF(IC) is added to the correction coefficient previously applied at that point. The number of correction coefficients is thus reduced by one each time a match is found.
SUBROUTINE CURREN (CURR, CURI, MCUR, NCUR, US, VS, CORCOF, IC)

This subroutine calculates the final current distribution necessary to produce the final pattern F. The real and imaginary parts of the current matrix, CURR and CURI, are initially that of the original current distribution (corresponding to the original pattern) as generated in ORGPAT. The source currents are calculated by summing all corrections together with the original pattern as in (3-11) and (3-12). The correction functions for the current are obtained from the SOURCE subprogram.

COMPLEX FUNCTION SOURCE (M, N, U, V, ITYPE)

This subprogram supplies values of the correction current for loading into the current arrays CURR and CURI at the point (M, N) for pattern sample point (U, V). This subprogram has seven sources corresponding to the seven patterns of PAT and they are flagged with an ITYPE number. If antennas other than these seven standard types are required, SPSOR is used to generate it. SPSOR will be called automatically if ITYPE is greater than 7. The sources in this subprogram are the Fourier Transform mates of the patterns in PAT.

COMPLEX FUNCTION SPSOR (M, N, U, V, ITYPE)

Currently, this is a dummy subprogram. It operates in the same manner as SOURCE. It is called from SOURCE when ITYPE exceeds seven. Then the dummy subprogram should be replaced by programming which generates values of the current distribution corresponding to the correction pattern of SPECPT and with correction point (U, V). The function of SPSOR and SPECPT should be Fourier transform mates.
SUBROUTINE LOCSOR (M, N, S, T)

This subroutine is used to generate source coordinates S and T when given the subscripts M and N of the current arrays CURR and CURI. It depends, of course, on the antenna used and this is handled by the commoned variable ITYPE. For ITYPE greater than seven SPLOC is called automatically. The coordinates S and T obtained from this subroutine are used by SOURCE and in the excitation print out part of the main program.

SUBROUTINE SPLOC (M, N, S, T)

This is currently a dummy subroutine. It is called by LOCSOR when ITYPE is greater than seven. When an antenna type other than one of the seven standard types is used, the user must supply FORTRAN coding to this subroutine to perform the function of LOCSOR.

SUBROUTINE DIRCTV (CORG, UORG, VORG, NORG, US, VS, CORCOF, IC, MMAX, NMAX, DIRORG, DIRFNL)

This subroutine calculates the directivity of the original and final patterns, DIRORG and DIRFNL. The directivities are calculated as discussed in Section 3.6. The patterns are generated by adding up all weighted correction patterns. The original pattern must be of the Woodward-Lawson type. If this is not the case, programming may be changed to call ORGPAT for generation of the original pattern.

If these directivities are desired, output the variable DIRECT should be set to 1 in Namelist IPRINT.

SUBROUTINE PRINT (A, M, N, STARTU, STARTV, DU, DV)

Subroutine print is the general output subroutine. It will print out co-ordinates (U, V) and values A(I, J), 10 rows and 10 columns to a page.
U and V are calculated as follows:

\[ U = \text{STARTU} + (I - 1) \times \text{DU} \]
\[ V = \text{STARTV} + (J - 1) \times \text{DV} \]

where I and J correspond to A(I, J), the value printed.

The output format is such that for large sources the printout covers many pages. However, these pages may be pasted together to form a grid and then photo-reduced for ease of handling.

PRINT may be used for all patterns and for all sources except ITYPE = 7. To invoke PRINT code the following variables in Namelist IPRINT.

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Variable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Desired pattern</td>
<td>FDESPT = 1</td>
</tr>
<tr>
<td>Original pattern</td>
<td>FORGPT = 1</td>
</tr>
<tr>
<td>Final pattern</td>
<td>FDBPT = 1</td>
</tr>
<tr>
<td>Original current</td>
<td>ICURPT = 1</td>
</tr>
<tr>
<td>Final current</td>
<td>FCURPT = 1</td>
</tr>
</tbody>
</table>

SUBROUTINE PROFIL (DATA1, NPT, NUMPAT)

Subroutine PROFIL prints a graph of the data in DATA1 with automatic scaling using NPT(NPT<401) number of points. The abscissa is stored in DATA1 (J, 1); the ordinate is stored in DATA1 (J, 2).

Because the line printer is a discrete device, the axes will be quantized. However, the true value of the ordinate is printed to the right of the graph.

PROFIL may only be used for pattern printouts. PROFIL gives both U-axis and V-axis profiles.

To invoke subroutine PROFIL, code the following variables in Namelist IPRINT.
Pattern Variable

Original FORGPR = 1
Final FDBPR = 1

SUBROUTINE CONUR (K, L, DELCON, CONLOW, CONMAX, CONINT, A, NUMPAT)

Subroutine CONUR provides a contour map of data stored in array A (dimensioned A(51, 51). It is used primarily for two-dimensional source distributions. (Subroutine PATCON is used for two-dimensional patterns.) Contour levels between CONLOW and CONMAX differing by CONINT are printed on a K by L grid (K, L ≤ 51).

To invoke this subroutine code the following variables in Namelist IPRINT.

Source Variable

Original distribution ICURCN = 1
Final distribution FCURCN = 1

Separate contour printouts are given for real and imaginary currents.

Not intended for use with ITYPE=7 patterns.

SUBROUTINE PATCON (RDATA, MMAX, NMAX, ICODE, CONLOW, CONMAX, CONINT, STARTU, STARTV, DELTAU, DELTAV, NUMPAT, ISYMM)

Subroutine PATCON provides the user with a contour map of the desired pattern (ICODE = 0), the initial pattern (ICODE = 1), or the final pattern (ICODE = 2). Contour levels are given by CONLOW, CONMAX, and CONINT. There may be up to 10 contour levels. In addition, if the pattern at a particular point falls below CONLOW, then a MINUS sign is printed. If the pattern rises above CONMAX, a plus sign is printed. Approximate execution time of PATCON is 10 seconds.
To invoke this subroutine code the following variables in Namelist IPRINT.

<table>
<thead>
<tr>
<th>Pattern</th>
<th>ICODE</th>
<th>Variable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Desired</td>
<td>0</td>
<td>FDESCN = 1</td>
</tr>
<tr>
<td>Original</td>
<td>1</td>
<td>FORGCN = 1</td>
</tr>
<tr>
<td>Final</td>
<td>2</td>
<td>FDECN = 1</td>
</tr>
</tbody>
</table>

SUBROUTINE LIST (CURR, CURI, MCUR, NCUR)

The purpose of SUBROUTINE LIST is to print out array element coordinates and currents for the general array source (ITYPE = 7). The coordinates (S, T) are found from SUBROUTINE LOCSOR and these are printed along with the appropriate value of current.

SUBROUTINE LIST is called by coding ICURPR = 2 or FCURPR = 2 in Name-list IPRINT. ICURPR = 2 will list initial element currents while FCURPR = 2 will list final element currents.

While written primarily for sources of ITYPE = 7, LIST may be used with any source.

6.6 Statement Listing of ANTSYN

6.6. Job Control Language Statements

```
//FO66A57 JCP 32762, COFFEY
//MAIN TIME=3,REGION=220K,LINES=10,CLARS=0
//PRIORITY PRIORITY
// EXEC FORT56,LIB2=SSPLIB
//FORT5YSIN LD *
/*
//G0.F122ES1 DD DSN=ANTSANTDATA,ADV=762,UNIT=3330,VOL=SER=USERPK,DISK=SER
//G0.SYSIN UD *
/*
```

6.6.2 Source Listing
MAIN PROGRAM TO SYNTHESIZE A PATTERN FOR A GIVEN SOURCE.

VERSION 3  73/164 -- JUNE 13, 1973

WRITTEN BY:  E. L. COFFEY
              W. L. STUTZMAN

UNDER NASA GRANT:  47-004-103

LANGUAGE:  FORTRAN IV

SUBPROGRAMS REQUIRED:
  D IRCIV
  INPUT
  READ
  CRGPAT
  ANTSYN
  SEARCH
  CHECK
  UPDATE
  PAT
  SOURCE
  LOCSOR
  SPLOC
  SPECPT
  SPSOR
  SINPUT
  CURRBN
  PRINT
  PROFIL
  CONTUR
  PATCCN
  LIST
  DESPAT
  DATE
  STIME
  BESJ
  ...STANDARD FORTRAN LIBRARY SUBPROGRAMS...

INPUT/OUTPUT SUPPORT:
  FTC5F001 (SYSIN) -- CARD READER
  FTC6F001 (SYSPRINT) -- LINE PRINTER
  FTC2F001 (ANTDATA.A507C2) -- AUXILIARY STORAGE

STORAGE REQUIREMENTS: 220K
DEFINE FILE 22(35,9100,E,NREC)

INTEGER TITLE(20)
INTEGER NUMSKP(35),FDESPT,FDESCN,FDESPR
INTEGER FORGPT,FORGCN,FORPR,FDBPT,FDBCN,FDBPR,DIRECT
REAL FDES(51,51),FU(51,51),FL(51,51),F(51,51)
REAL US(500),VS(500),CORCOF(500),CURR(51,51),CURI(51,51)
REAL UCRG(100),VORG(100)
REAL INITLS,INITLT
COMPLEX SOURCE
INTEGER FCURPT,FCURCN,FCURPR
COMMON /MPROG/ ICUR,NCUR
COMMON /START/ NORG,UORG,VORG,CURG
COMMON /PAT1/ P1,P2,P3,P4,P5,P6,PI,SS(400),IT(400),RR(400)
COMMON /PAT2/ 11,12,13,14,15
COMMON /LOC/ ITYPE
DATA TITLE / 'ANTENNA SYNTHESIS PROGRAM VERSION 3 LEVEL 1',
$' E.E. ', ' DEP.' , ' T. ', /
NAMELIST /PARAM/ IDISK,ISYM,IITMAX,DECFA,DETVI,STARTU,STARTV,
$NMAX,NMAX,MCENT,MCENT
NAMELIST /IPRINT/ FDESPT,FDESPR,FDESCN,FDBPT,FDBCN,FDBPR,
$FORGPT,FORGCN,FORPR,ICURPT,ICURCN,ICURPR,FCURPT,FCURCN,FCURPR,
$DIRECT

BEGIN PROCESSING
9999 CONTINUE
READ(22,'(A500)') NUMPAT,NUMTRK,NUMSKP,IPASS
READ(22,'(A500)') NUMPAT=NUMPAT+1
CALL DATE(11,J1,K1)
CALL STIME(IT)
IHR=IT/I0000
IFR=IT-IHR*I0000
FHR=IFR/I10000
FM=FHR*60.
IMIN=FM
ISCC=(FM-IMIN)*60.
WRITE(6,'(A11,I4,A11,I4,A11,I4,A11,I4,A11)')
'ANTENNA SYNTHESIS PROGRAM VERSION 3 LEVEL 1 ',
'VPI EE DEP.T. ', ' DATE = ', 'A2', '- ', 'A2', '- ', 'A2',
'TIME = ', '12', ': ', '12', ': ', '12', ': ', ' PATTERN ', 'I4//')
A-27

DEFAULT PARAMETERS

34 IDISK=0
35 ISYM=0
36 ITRMAX=100
37 MMAX=1
38 NMAX=1
39 MCENT=1
40 NCENT=1
41 DELTAT=0.
42 DELTAU=0.
43 STARTU=0.
44 STARTV=0.
45 MCUR=1
46 NCUR=1
47 FDESPT=0
48 FDESCN=0
49 FDBPT=0
50 FDBCN=0
51 FDEPR=0
52 FDESPR=0
53 FORGPT=0
54 FORCCN=0
55 ICURPT=0
56 ICURCN=0
57 ICURPR=0
58 FCURPT=0
59 FCURCN=0
60 FCURPR=0
61 FORCPR=0
62 DIRECT=0
63 FNORM=1.0
64 ISIC=0
65 DELTAS=0.
66 DELTAT=0.

READ(5,PARAM)
WRITE(6,1521) IDISK,STARTU,MMAX,ISYM,STARTV,NMAX,ITRMAX,DELTAT,
MCENT,DELTAV,NCENT
1521 FORMAT(7X,"PROGRAM PARAMETERS"/36X,"IDISK =",11,26X,"STARTU =",12,
NMAX=",17,22X,"ITRMAX =",18,22X,"DELTAU =",19,22X,"MCENT =",20,22X,
READ(6,PRINT)
WRITE(6,1522) FDESPT, FORGPT, FDBPT, ICURPT, FCURPT,
FDECN, FORCCN, FDECN, FORCCN,
FDEPR, FORCPR, FDEPR, ICURPR, FCURPR
1522 FORMAT(1X,"PROGRAM PARAMETERS"/36X,"FDESPT =",11,5X,"FORGPT =",11,5X,
"FDBPT =",11,5X,"ICURPT =",11,5X,"FCURPT =",11,5X,
"FDECN =",11,5X,"FORCCN =",11,5X,"FDECN =",11,5X,"FORCCN =",11,
"FCURCN =",11,5X,"FORCPR =",11,5X,"FCURPR =",11,5X)
CALL INPUT

IF (ITYPE.EQ.7 .AND. ICURPR.EQ.1) ICURPR = 2
IF (ITYPE.EQ.7 .AND. FCURPR.EQ.1) FCURPR = 2
CALL LECSCR(1,1,INITLS,INITLT)
CALL LOCSOR(NCUR,NCUR,FINALS,FINALT)
IF (*CUR .NE. 1) DELTAS = (FINALS - INITLS) / (MCUR - 1)
IF (*CUR .NE. 1) DELTAT = (FINALT - INITLT) / (NCUR - 1)
CALL DESPAT(FDES,FU,FL,MMAX,NMAX,STARTU,STARTV,$DELTAU,DELTAV)

IF (FDESPT) 300, 300, 301
WRITE(6,302)
FORMAT(1H1, 1X, 'DESIRED PATTERN IN DB. ') 55X
CALL PRINT(FDES,MMAX,NMAX,STARTU,STARTV,$DELTAU,DELTAV)

300 IF (FDES = N) 303, 303, 304
304 CONTINUE
CALL PATCON(FDES,MMAX,NMAX,STARTU,STARTV,$DELTAU,DELTAV,NUMPAT,ISYM)
303 IF (FDESPT) 306, 306, 307
307 IF (MMAX .LE. 1) GO TO 306
WRITE(6,310) NUMPAT
FORMAT(1H1, 1TX, 'U-AXIS PROFILE OF DESIRED PATTERN ', 14// 12X,'U','16X,'V',15X,'FDES(U,V)',10X,'FU(U,V)',12X,'FL(U,V)')
V = STARTV + (NCENT - 1) * DELTAV
DO 309 I = 1, MMAX
U = STARTU + (I - 1) * DELTAV
WRITE(6,311) U, V, FDES(I,NCENT),FU(I,NCENT),FL(I,NCENT)
309 CONTINUE
FORMAT(1H1, 10X, 'V-AXIS PROFILE OF DESIRED PATTERN ', 14// 12X,'U',16X,'V',15X,'FDES(U,V)',10X,'FU(U,V)',12X,'FL(U,V)')
U = STARTU + (MCENT - 1) * DELTAV
DO 312 J = 1, NMAX
V = STARTV + (J - 1) * DELTAV
WRITE(6,312) U, V, FDES(MCENT,J),FU(MCENT,J),FL(MCENT,J)
312 CONTINUE
ENTER ORIGINAL PATTERN
CALL CRGPAT(F,MMAX,NMAX,STARTU,STARTV,$DELTAU,DELTAV)

OUTPUT OF ORIGINAL PATTERN

IF (FORGPT) 400, 400, 401
WRITE(6,402)
FORMAT(1H1, 1X, 'INITIAL PATTERN') 55X
CALL PRINT(F,MMAX,NMAX,STARTU,STARTV,$DELTAU,DELTAV)

400 IF (FORGPT) 403, 403, 404
404 CONTINUE
CALL PATCON(F,MMAX,NMAX,-0.5,1.3,0.2,STARTU,STARTV,$DELTAU,DELTAV)
403 IF (FORGPR) 406, 406, 407
407 CONTINUE
DO 409 J=1,401
U=(J-1)*C*C05-1. C
V=U
SUMU=0.
SUMV=C.
DO 409 K=1,NORG
SUMU=SUMU+CORG(K)*PAT(U-UORG(K),V-VORG(K),ITYPE)
SUMV=SUMV+CORG(K)*PAT(-UORG(K),V-VORG(K),ITYPE)
CONTINUE
DATA1(J,1)=U
DATA1(J,2)=SUMU
DATA2(J,1)=V
DATA2(J,2)=SUMV
CONTINUE
IF(VMAX.LE.1) GO TO 2601
WRITE(6,410)
FORVAT(1H1,25X,'U-AXIS PROFILE OF INITIAL PATTERN')
CALL PROFIL(DATA1,401,NUMPAT)
2801 IF(NMAX.LE.1) GO TO 406
WRITE(6,411)
FORVAT(1H1,25X,'V-AXIS PROFILE OF INITIAL PATTERN')
CALL PROFIL(DATA2,401,NUMPAT)
CONTINUE
ORIGINAL EXCITATION
IF(ICURPT) 500,500,501
WRITE(6,501)
IF(ICURCN) 503,503,504
WRITE(6,504)
CALL PRINT(CURR,MCUR,NCUR,INITLS,INITLT,DELTAS,DELTAT)
WRITE(6,505)
CALL CONTUR(MCUR,NCUR,CURR,INITLS,INITLT,DELTAS,DELTAT)
WRITE(6,506)
WRITE(6,507) 503,503,504
IF(MCUR.LT.1) GO TO 508
WRITE(6,508)
510 FORMAT(1H1,1X,'S AXIS PROFILE OF INITIAL CURRENT'//
$13X,'S',17X,'T',18X,'REAL',12X,'IMAGINARY',10X,'MAGNITUDE',
$12X,'PHASE')/
J=NCUR/2+1
DO 509 I=1,MCUR
CALL LOCSSR(I,J,S,T,ITYPE)
AMAG=SQRT(CURR(I,J)**2+CURR(I,J)**2)
IF(AMAG.EQ.0.) APH=0.
IF(AMAG.EQ.0.) GO TO 509
APH=ATAN2(CURR(I,J),CURR(I,J))*57.2957795
WRITE(6,511) S,T,CURR(I,J),CURR(I,J),AMAG,APH
FORMAT(9X,F8.4,9X,F8.4,10X,4(E14.7,5X))
A-30

163 508 IF(NCUR.LE.1) GO TO 506
164 WRITE(6,512)
165 512 FORMAT(1H1,10X,'T AXIS PROFILE OF INITIAL CURRENT'//
166 $13X,'S'.17X,'T',18X,'REAL',12X,'IMAGINARY',10X,'MAGNITUDE',
167 $12X,'PHASE'//)
168 IC=MCUR/2+1
169 DO 513 J=1,NCUR
170 CALL LCCSOR(I,J,S,T,ITYPE)
171 CI=CURR(I,J)
172 AMAG=SQRT(CR*CR+CI*CI)
173 IF(AMAG.EQ.0.) APH=0.
174 APH=ATAN2(CI,CR)*57.2957795
175 513 WRITE(6,511) S,T,CR,CI,AMAG,APH
176 GO TO 506
177 514 WRITE(6,515)
178 515 FORMAT(1H1///10X,'INITIAL ELEMENT CURRENTS'//5X,
179 $'J',10X,'S',15X,'T',15X,'CURR',11X,'CURR'//)
180 CALL LCCSOR(I,J,CURR,CURI,MCUR,NCUR)
181 506 CONTINUE
182 IC=0
183 WRITE(6,4747)
184 4747 FORMAT(1H1)
185 CALL ANTSYN(ISUC,MMAX,NMAX,FDES,FU,FL,ITRMAX,ISYMM,CORCOF,IC,US,
186 $VS,STARTU,DELTAV,STARTV,DELTAU,MCENT,NCENT,ITER,FNORM,F)
187 C C PRINT OUT RESULTS
188 WRITE(6,391)
189 391 FORMAT(1H1,52X,'-- FINAL COEFFICIENTS --'//45X,'J',7X,
190 $'US(J)',7X,'VS(J)',5X,'CORCOF(J)'//)
191 IF(IC.LE.0) WRITE(6,977)
192 977 FORMAT(1X,'NC ITERATIONS PERFORMED')
193 IF(IC.LT.0) GO TO 978
194 IC=998 J=1,IC
195 498 WRITE(6,35) J,US(J),VS(J),CORCOF(J)
196 35 FORMAT(3X,I3,5X,F7.4,5X,F7.4,5X,F7.4)
197 WRITE(6,497) ITER
198 497 FORMAT(1H0,9X,'NUMBER OF ITERATIONS = ',16)
199 WRITE(6,496) FNORM
200 496 FORMAT(1H0,9X,'FNORM = ',F10.5)
201 WRITE(6,976) NUMPAT
202 976 FORMAT(1H0,9X,'PATTERN NUMBER = NUMPAT = ',15)
203 C C OUTPUT FINAL PATTERN IN DB
204 978 CONTINUE
205 DO 29 J=1,MMAX
206 DO 29 K=1,NMAX
207 IF(F(J,K)) 290,289,290
208 289 F(J,K)=-2CC.
209 290 CONTINUE
210 29 CONTINUE
A-31

204  GO TO 29
205  290  F(J,K)=20.*ALG10(ABS(F(J,K)))
206  29 CONTINUE

C

207  IF(FDPRPT) 600,600,601
208  601 WRITE(6,602)
209  602 FORMAT(1H1/\ldots//\ldots//\ldots//\ldots//55X,'FINAL PATTERN IN OR')
210  CALL PRINT(F,MMAX,NMAX,STARTU,STARTV,DELTAU,DELTAV)

C

211  IF(FDBCN) 603,603,604
212  604 CONTINUE
213  CALL PATCON(F,MMAX,NMAX,2,-45.,0.0,5.0,STARTU,STARTV,
214     $DELTAU,DELTAV,NUMPAT,ISYMM)
215  605 IF(FDPR) 606,606,607
216  606 CONTINUE
217  DO 608 J=1,401
218     U=(J-1)*C.005-1.0
219     V=U
220     SUMU=0.
221     SUMV=0.
222     SUMU=SUMU+CORCOF(K)*PAT(U-US(K),-VS(K),ITYPE)
223     SUMV=SUMV+CORCOF(K)*PAT(-US(K),V-VS(K),ITYPE)
224  609 CONTINUE
225     DATA1(J,1)=U
226     DATA2(J,1)=V
227     DATA1(J,2)=DATA1(J,2)+SUMU
228     DATA2(J,2)=DATA2(J,2)+SUMV
229     DATA1(J,2)=DATA1(J,2)+FNORM
230     DATA2(J,2)=DATA2(J,2)+FNORM
231  608 CONTINUE
232  IF(MMAX.LE.1) GO TO 2901
233  WRITE(6,610)
234  610 FORMAT(1H1,25X,'U-AXIS PROFILE OF FINAL PATTERN')
235  CALL PROFIL(DATA1,401,NUMPAT)
236  2901 IF(NMAX.LE.1) GO TO 606
237  WRITE(6,611)
238  611 FORMAT(1H1,25X,'V-AXIS PROFILE OF FINAL PATTERN')
239  CALL PROFIL(DATA2,401,NUMPAT)
240  606 CONTINUE

C

241  IF(FCURPT+FCURPR+FCURCN.LE.0) GO TO 706
242  CALL CURREN(CURR,CURI,MCUR,NCUR,US,VS,CORCOF,IC)

C

243  IF(FCURPT) 700,700,701
244  701 WRITE(6,702)
245  702 FORMAT(1H1/\ldots//\ldots//\ldots//\ldots//55X,'FINAL CURR')
246  CALL PRINT(CURR,MCUR,NCUR,INITLS,INITLT,DELTAS,DELTAT)
247  WRITE(6,782)
248  782 FORMAT(1H1/\ldots//\ldots//\ldots//\ldots//55X,'FINAL CURI')
249  CALL PRINT(CURI,MCUR,NCUR,INITLS,INITLT,DELTAS,DELTAT)
250  704 WRITE(6,705)
251  705 CONTINUE
252    FORMAT(1H1,////10X,'FINAL CURR')
253    CALL CONTUR(MCUR,NCUR,C,COS,-0.04,0.04,0.0,CURR,NUMPAT)
254    WRITE(6,705)
255
256    CALL CONTUR(MCUR,NCUR,C,COS,-0.04,0.04,0.0,CURR,NUMPAT)
257
258    IF(FCUNPR-1) 706,707,711
259
260    CALL CURTR(MCUR,NCUR,-0.005,-0.04,0.04,0.0,CURR,NUMPAT)
261
262    IF(FCURPR-1) 706,707,711
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
IF(IOISK.EQ.1 .AND. ISUC.NE. 1) GO TO 9997
C DISK CUTPUT
GO TO 7000 J=2,35
IF(NUMSKP(J).EQ.0) GO TO 7001
7000 CONTINUE
WRITE(6,7002)
7002 FORMAT('NO DISK SPACE AVAILABLE -- DATA NOT STORED')
GO TO 9999
7001 CONTINUE
C SPACE IS AVAILABLE ON RECORD "J"
 NUMSKP(J)=1
WRITE(22,8850) NUMPAT,J,NUMSKP,IPASS
WRITE(22,8850) NUMPAT,TITLE,ISYMM,ITER,ISUC,SNORM,IDISK,
$NORG,IC,(UCRG(M),VORG(M),CORG(M),M=1,NORG),
$(US(M),VS(M),CCRCOF(M),M=IIC),ITYPE,P1,P2,P3,P4,P5,P6,
$PI,TT(M),W(1,1400),11,12,13,14,15,MCUR,NCUR
WRITE(6,7003) NUMPAT,J
7003 FORMAT('PATTERN NUMBER ',I4,' HAS BEEN STORED ON RECORD',
$'14', OF ANDATA.A507C2')
9997 GO TO 9999
END

SUBROUTINE DIRCTV(CORG,UORG,VORG,NORG,US,VS,CORCOF,IC,MMAX,NMAX,
1 CORG,UORG,DIRFNL)
C THIS SUBROUTINE CALCULATES THE DIRECTIVITY OF THE ORIGINAL PATTERN
C, DIRCRG, AND OF THE FINAL PATTERN, DIRFNL
DIMENSION CORG(100),UORG(100),VORG(100),US(500),VS(500),CORCOF(500)
COMMON /LCC/ ITYPE
FORGSC=C.
FSC=0.
FMAX1=0.
FMAX2=0.
DO 10 J=1,101
10 U=-1.0+(J-1)*0.02
CC 10 K=1,101
20 V=-1.0+(K-1)*0.02
UVSQ=L*L+V*V
F=0.
IF(UVSC.GE.1.0) GO TO 10.
1
C C
IF(NCRG.LE.0) GO TO 25
CC 20 L=1,NCRG
2C F=F+CORG(L)*PAT(U-UORG(L),V-VORG(L),ITYPE)
FORGSC=FORGSC+F**2/SCRT(1.0-UVSQ)
IF(ABS(F).GT. FMAX1) FMAX1=ABS(F)
25 CONTINUE
FSC=FCKGSQ:
FMAX2=FMAX1
IF (IC.LE.0) GO TO 10
CO 30 L=1,IC
3C F=F+CC+CCCF(L)*PAT(U-US(L),V-VS(L),ITYPE)
FSC=FSC+F**2/SQRT(1.0-U/V/SQ)
338 IF (ABS(F).GT.FMAX2) FMAX2=ABS(F)
1C CONTINUE
FORGSQ=FORGSQ0*.0004/FMAX1
FSC=FSC*C.CC/C/FMAX2
340 DIRCRG=4.0*3.14159265/FORGSQ
343 DIRFNL=4.0*3.14159265/FSC
C C
DIRC=1C.*ALOG10(DIRORG)
DIRFNL=10.*ALOG10(DIRFNL)
RETURN
ENC
SUBROUTINE INPUT
C
INTEGER PX,PY
REAL LX,LY,INITLS,INITLT
C C
COMMON /PAT1/ P1,P2,P3,P4,P5,P6,PI,SS(400),TT(400),RR(400)
COMMON /PAT2/ PI,II2,II3,II4,II5
COMMON /LOC/ ITYPE
COMMON /MPROG/ MCUR,NCUR
C C
NAMELIST /PATIN/ LX,LY,PX,PY,DIX,DIG,INITLS,DELTAS,FINALS,
$INITLT,DELTT,FINALT,NEIMT,NEIMT,ITYPE,MCUR,NCUR
C
WRITE(6,10)
10 FORMAT(///55X,'SOURCE SPECIFICATIONS'//)
PI=3.14159265
READ(5,PATIN)
IF (ITYPE.GT.7) GO TO 990
GO TO (10C,200,300,400,500,600,700), ITYPE
WRITE(6,2C) ITYPE
2C FORMAT(1HO,5X,***ERROR**** ITYPE HAS THE VALUE '111',2X,
$EXECUTION TERMINATED')
STOP
C
PI=LY
P2=INITLT
P3=DELTAT
LX=C.0
NCUR=(FINALT-INITLT)/DELTAT+1.5
MCUR=1
WRITE(6,101) LY,INITLT,FINALT,DELTAT,NCUR
101 FORMAT(1X,'ITYPE=1 -- UNIFORM LINE SOURCE'//15X,'LY = ',F7.3//' 
$15X,'INITLT,FINALT,DELTAT:',3(1X,F8.4)//15X,'NUMBER OF SAMPLE POINTS = NCUR = ',I3)
374 GO TO 999

C C
375 200 P1=LY
376 II=PY
377 LX=C.C
378 P2=DISY
379 NCUR=PY
380 MCLR=1
381 WRITE(6,201) LY,PY,DISY
382 201 FORMAT(10X,'ITYPE=2 -- UNIFORM LINEAR ARRAY'// 
$15X,'LY = ',F7.3//15X,'NUMBER OF ELEMENTS = ',I3//15X,'INTER-ELEMENT SPACING = ',F6.3)
383 GO TO 999

C C
384 300 P1=LY
385 LX=0.0
386 P2=INITLT
387 P3=DELTAT
388 NCUR=(FINALT-INITLT)/DELTAT+1.5
389 MCLR=I
390 WRITE(6,301) LY,INITLT,FINALT,DELTAT,NCUR
391 301 FORMAT(10X,'ITYPE=3 -- TRIANGULAR LINE SOURCE'// 
$15X,'LY = ',F7.3//15X,'T VARIES FROM ',F8.4,' TO ',F6.4,' ' 
$15X,'DELTAT = ',F6.3//15X,'NUMBER OF SAMPLE POINTS = NCUR = ',I3)
392 GO TO 999

C C
393 400 P1=LX
394 P2=LY
395 P3=INITLS
396 P4=INITLT
397 P5=DELTAS
398 P6=DELTAT
399 MCLR=(FINALS-INITLS)/DELTAS+1.5
400 NCUR=(FINALT-INITLT)/DELTAT+1.5
401 WRITE(6,401) LX,LY,INITLS,DELTAS,FINALS,INITLT,DELTAT,FINALT,MCUR, 
$NCUR
402 401 FORMAT(10X,'ITYPE=4 -- UNIFORM RECTANGULAR APERTURE'// 
$15X,'DIMENSIONS = LX,LY = ',F7.4,' , ',F7.4//' 
$15X,'INITLS,DELTAS,FINALS:',3(1X,F8.4,1X)// 
$15X,'INITLT,DELTAT,FINALT:',3(1X,F8.4,1X)// 
$15X,'MCUR,NCUR:',2(I3,2X))
403 GO TO 999

C C
405 500 P1=LX
406 P2=LY
407 II=PX
I2 = PY
P3 = DISX
P4 = DISY
MCUR = 11
NCUR = 12
WRITE (6, 501) LX, LY, PX, PY, DISX, DISY
501 FORMAT (LX, 'ITYPE = 5 -- UNIFORM RECTANGULAR ARRAY'//
   $15X, 'DIMENSIONS = LX, LY = ', F7.4, ', ', F7.4//
   $15X, 'NUMBER OF ELEMENTS = PX, PY = ', I3, ', ', I3//
   $15X, 'INTER-ELEMENT SPACING = DISX, DISY = ', F6.3, ', ', F6.3)
GO TO 999

C
C
601 FORMAT (1CX, 'ITYPE = 6 -- UNIFORM CIRCULAR APERTURE'//
   $15X, 'ARAD = ', F7.3//15X, 'INITLS, DELTAS, FINALS: ', 3(F8.4, 1X)//
   $15X, 'INITLT, DELTAT, FINALT: ', 3(F8.4, 1X)//
   $15X, 'MCUR, NCUR: ', 2(I3, 2X))
GO TO 999

C
C
701 FORMAT (1CX, 'ITYPE = 7 -- GENERAL ARRAY'//
   $15X, 'ELEMENT', 7X, 'SS(J)*, 14X, 'TT(J)*'
CC 702 J = 1, NELMT
READ (5, 703) SS(J), TT(J)
703 FORMAT (1F1C.C)
WRITE (6, 704) J, SS(J), TT(J)
704 FORMAT (17X, I3, 5X, 3(E14.7, 5X))
702 CONTINUE
GO TO 999
990 CALL SINPUT (PX, PY, DISX, DISY, INITLS, DELTAS, FINALS, INITLT, DELTAT, FINALT, NELMT, ARAD, ITYPE)
RETURN
SUBROUTINE READ (F, PMAX, NMAX)
DIMENSION F(51,51), I(6), VAL(6)
CO 100 J=1, PMAX
K2=0
CONTINUE
READ(5,1) (I(L), VAL(L), L=1, 6)
1 FORMAT(6(I3, F10.0))
DC 20 L=1, 6
IL=I(L)
IF(IL.EQ.0) GO TO 100
K1=K2+1
K2=K1+IL-1
DO 10 K=K1, K2
10 K=K1, K2
IC F(J, K)=VAL(L)
CONTINUE
IF(K2.LT.NMAX) GO TO 200
100 CONTINUE
RETURN
END

SUBROUTINE ORGPAT (F, PMAX, NMAX, STARTU, STARTV, DELTAU, DELTV, CURR, CUR, MCUR, NCUR)
DIMENSION FES(51, 51), FU(51, 51), FL(51, 51)
THIS LEADS THE DESIRED PATTERN AND UPPER AND LOWER LIMITS
CALL READ(FES, PMAX, NMAX)
CALL READ(FU, PMAX, NMAX)
CALL READ(FL, PMAX, NMAX)
RETURN
END

SUBROUTINE ORGPAT (F, PMAX, NMAX, STARTU, STARTV, DELTAU, DELTV, CURR, CUR, MCUR, NCUR)
REAL F(51, 51), CURR(51, 51), CURI(51, 51)
REAL UORG(100), VORG(100), CORG(100)
COMPLEX SOURCE
COMPLEX TEMP
COMMON /START/, NORG, UORG, VORG, CORG
COMMON /LOC/, ITYPE
THIS ORGPAT WILL BE "WOODWARD-LAWSON" INPUT.
CO 10 N=1, PMAX
DC 10 N=1, NMAX
IC F(M, N)=0.

REPRODUCIBILITY OF THE ORIGINAL PAGE IS POOR
DO 15 M=1,MCUR
CC 15 N=1,NCUR
15 CURR(M,N)=0.
15 CURI(M,N)=0.
WRITE(6,17)
17 FORMAT(1H1,50X,-- INITIAL COEFFICIENTS --'/45X,'J',6X,
$'UORG(J)',5X,'VORG(J)',6X,'CORG(J)'/)
REAC(5,1) NORG
1 FORMAT(15)
DO 20 IC=1,NCRG
20 FORMAT(3F10.0)
VORG(IC)=US
CORC(IC)=CORCCF
DO 30 M=1,MMAX
30 U=STARTU+(M-1)*DELTAV
DU=U-US
DO 30 N=1,NMAX
30 V=STARTV+(N-1)*DELTAV
DV=V-VS
3C H(M,N)=FCM,N)+CORCOF*PAT(DU,DV,ITYPE)
CC 40 M=1,MCUR
CC 40 N=1,NCUR
TEMP=SOURCE(M,N,US,VS,ITYPE)
CURR(M,N)=CURR(M,N)+CORCOF*REAL(TEMP)
CURI(M,N)=CURI(M,N)+CORCOF*AIMAG(TEMP)
WRITE(6,50) IC,US,VS,CORCCF
50 FORMAT(44X, T3,5XF7.4,5X,F7.4,5X,F7.4)
20 CONTINUE
RRETURN
END

SUBROUTINE ANTSYN(ISUC,MMAX,NMAX,FDES,FU,FL,ITRMAX,ISYMM,CORCOF,
$IC,US,VS,STARTU,DELTAV,STARTV,DELTAV,MCENT,NCEMT,ITER,FNORM,F)

REAL FDES(51,51),FU(51,51),FL(51,51),F(51,51)
REAL US(500),VS(500),CORCOF(500),FCRMAT(44X, T3,5XF7.4,5X,F7.4,5X,F7.4)
COMMON /SYN/

ITER=0
27 ITER=ITER+1
C NORMALIZE...

FBIG=F(MCENT,NCENT)
DO 150 M=1,MMAX
150 DO 150 N=1,NMAX
150 F(M,N)=F(M,N)/FBIG
FNORM=FNORM/FBIG
DO 151 I=1,NCRG
151 CORG(I)=CORG(I)/FBIG
IF(1C.LT.0) GO TO 153
LG 152 I=1,IC
153 CONTINUE
-- ITERATION PROCEDURE --

SET IF SPECS ARE MET.

DO 24 J=1,NMAX
U=STARTU+(J-1)*DELTAV
DO 24 K=1,NMAX
V=STARTV+(K-1)*DELTAV
UVSU=U*U+V*V
IF(UVSU.GT.T*T) GO TO 24
24 CONTINUE

IF(FDES(J,K).EQ.99.0) GO TO 24
IF(FL(J,K).LE.0.0001 .AND. ABS(F(J,K)).LE.1.E-4) GO TO 24
X1=ABS(F(J,K))
IF(X1.GT.FU(J,K)) GO TO 25
IF(FL(J,K).LE.99.0) GO TO 24
12 24 CONTINUE

ISUC=1
IC=IC+1

SPECs ARE MET -- PROCEED TO PRINTOUT.

GO TO 75C
25 CONTINUE

SPECs ARE NOT MET AT POINT (J,K)

IC=IC+1
IF(ITER/ICC*ICC.EQ.ITER) WRITE(6,7117) ITER
7117 FORMAT(10X,50X,'ITERATIONS COMPLETED')
IF(ITER-ITRMAX) 22,22,23
23 WRITE(6,34) ITRMAX
34 FORMAT(IHO,9X,'NUMBER OF ITERATIONS EXCEEDED', $10X,'PRINTOUT OF INTERMEDIATE RESULTS FOLLOWS.')
GO TO 75C
22 CONTINUE

FIND RELATIVE MAXIMUM ERROR

CALL SEARCH(J,K,VAL,FDES,FL,F,MMAX,NMAX,STARTU,STARTV,DELTAV,
$DELTAU)
IF(VAL.NE.0.0) GO TO 248
VAL EQUALS ZERO
WRITE(6,100)
100 FORMAT('ERROR IN SUBROUTINE SEARCH -- VAL=0.')
GO TO 75C
248 U1=(J-1)*DELTAV+STARTU
V1=(K-1)*DELTAV+STARTV
IF(ABS(U1).LE.0.1*DELTAV) U1=0.
IF(ABS(V1).LE.0.1*DELTAV) V1=0.
IF(LX.LT.0.0) GO TO 1000
4 IF(U1.NE.0.0 .AND. ABS(U1).LE.0.5/LX) VAL=VAL/2.
1000 IF(LX.LE.0.0 .AND. ABS(V1).LE.0.5/LY) VAL=VAL/2.

1000 IC=IC+1
IF(LY.LT.0.0) GO TO 1000
4 IF(V1.NE.0.0 .AND. ABS(V1).LE.0.5/LY) VAL=VAL/2.
IF(ISYMN .NE. 4) GO TO 1001
ITEMP = C
UV = ABS(ABS(U1) - ABS(V1))
IF(UV.EQ.0.) GO TO 1001
IF(UV*1.414.LE.LXY) VAL = VAL/2.
CONTINUE

BASIC CORRECTION -- INDEPENDENT OF ISYMM

US(IC) = U1
VS(IC) = V1
CORCOF(IC) = VAL
CALL UPDATE(IC, US, VS, CORCOF, F, MMAX, NMAX, FNORM, STARTU, STARTV $, DELTAU, DELTAV)
CALL CHECK(IC, VAL, US, VS, CORCOF, DELTAU, DELTAV)

IF(ISYMM) 26, 27, 26
26 CONTINUE
260 CONTINUE

V-AXIS AND QUADRILATERAL SYMMETRY -- ISYMM = 2, 3, 4

IF(U1.EQ.0.) GO TO 261
IC = IC + 1
US(IC) = -U1
VS(IC) = V1
CORCOF(IC) = VAL
CALL UPDATE(IC, US, VS, CORCOF, F, MMAX, NMAX, FNORM, STARTU, STARTV $, DELTAU, DELTAV)
CALL CHECK(IC, VAL, US, VS, CORCOF, DELTAU, DELTAV)
261 IF(ISYMM = 2) 259, 27, 259

U-AXIS AND QUADRILATERAL SYMMETRY -- ISYMM = 1, 3, 4

259 IF(V1.EQ.0.) GO TO 262
IC = IC + 1
US(IC) = U1
VS(IC) = -V1
CORCOF(IC) = VAL
CALL UPDATE(IC, US, VS, CORCOF, F, MMAX, NMAX, FNORM, STARTU, STARTV $, DELTAU, DELTAV)
CALL CHECK(IC, VAL, US, VS, CORCOF, DELTAU, DELTAV)
262 IF(ISYMM = 1) GO TO 27

QUADRILATERAL SYMMETRY ONLY -- ISYMM = 3, 4

IF(U1.EQ.0. OR V1.EQ.0.) GO TO 2745
IC = IC + 1
US(IC) = -U1
VS(IC) = -V1
CORCOF(IC) = VAL
CALL UPDATE(IC, US, VS, CORCOF, F, MMAX, NMAX, FNORM, STARTU, STARTV $, DELTAU, DELTAV)
CALL CHECK(IC, VAL, US, VS, CORCOF, DELTAU, DELTAV)
2745 IF(ISYMM = 1 AND ITEMP = 1) GO TO 27

FOR BIQUADRILATERAL SYMMETRY ONLY -- ISYMM = 4

ITEMP = 1
IF(U1.EQ.V1) GC TO 27
IC=IC+1
UTEMP=U1
VTEMP=V1
U1=VTEMP
V1=UTEMP
GO TO 1001
750 CONTINUE
IC=IC-1
ITER=ITER-1
RETURN
END

SUBROUTINE SEARCH(I1,J1,VAL,FDES,FU,FL,F,MMAX,NMAX,STARTU,
$STARTV,DELTAU,DELTAV)
REAL FU(51,51),FL(51,51),F(51,51),FDES(51,51)
VAL=0.
EMAX=0.
12=11
J2=J1
CC 10 J=12,MMAX
U=STARTU+(J-1)*DELTAU
20 K=J2,NMAX
V=STARTV+(K-1)*DELTAV
UVSQ=U*U+V*V
IF(UVSQ.GT.1.0) GO TO 20
FITER=ABS(F(J,K))
IF(FDES(J,K).EQ.99.0) GO TO 20
IF(FITER.GT.FU(J,K)) GO TO 2000
IF(FL(J,K).LE.1.E-4) GO TO 20
IF(FITER.GT.FL(J,K)) GO TO 20

X=FDES(J,K)
ERROR = FITER-X
IF(ABS(ERROR)-ABS(EMAX)) 20,20,21
EMAX=ERROR
VAL=SIGN(ERROR,F(J,K)*(X-FITER))
II=J
J1=K
2C CONTINUE
1C CONTINUE
WRITE(6,10C) II,J1,VAL
100 FORMAT(5X,'**SEARCH**',I8,I8,5X,F7.4)
RETURN
ENC

SUBROUTINE CHECK(IC,VAL,US,VS,CORCOF,DELTAU,DELTAV)
REAL US(5), VS(5), CORCOF(500)
IF(IC.EQ.1) RETURN
CU=0.1*DELTAU
CV=0.1*DELTAV
IC1=IC-1
U=US(IC)
W=VS(IC)
SUBROUTINE UPOATE(IC,US,VS,CORCOF,F,MMAX,NMAX,FNORM,STARTU,STARTV,$DELTAU,DELTAV)
C
DIMENSION F(51,51),US(500),VS(500), CORCOF(500)
COMMON /LOC/ ITYPE
COMMON /PAT/ P,P2,P3,P4,P4,P,P6,P,P8(400),TT(400),RR(400)
COMMON /PAT2/ 11,12,13,14,15

FUNCTION PAT(U,V,ITYPE)
C
C THIS SUBPROGRAM GIVES THE BASIC CORRECTION PATTERN F(U,V).
C
C ITYPE = 1 -- UNIFORM LINE SOURCE LOCATED AT S=0.
C 2 -- UNIFORM LINEAR ARRAY LOCATED AT S=0.
C 3 -- TRIANGULAR LINE SOURCE LOCATED AT S=0.
C 4 -- UNIFORM RECTANGULAR APERTURE.
C 5 -- UNIFORM RECTANGULAR ARRAY.
C 6 -- UNIFORM CIRCULAR APERTURE.
C 7 -- GENERAL ARRAY.

C ITYPE > 7 -- SPECIAL SOURCE (FUNCTION SPECPT(U,V,ITYPE) WILL BE CALLED.

VERSION 1  LEVEL 1

DATE OF LAST REVISION: 73/193 JULY 12, 1973

THIS WORK SUPPORTED BY NASA GRANT NGR 47-004-103

FOR FURTHER INFORMATION CONTACT:
L. STUTZMAN DEPT. OF ELEC. ENGR. 951-6624.
E. CCFFEY DEPT. OF ELEC. ENGR. 951-5494

CCMPLEX TEMP,CESP,IMAG
COMMON /PAT/ P1,P2,P3,P4,P5,P6,P7,SS(400),TT(400),RR(400)
COMMON /PAT2/ 11,12,13,14,15
IF(ITYPE.GT.7) GO TO 990
GO TO (100,200,300,400,500,600,700),ITYPE

WRITE(6,10) ITYPE
10 FORMAT(1HC,5X,'***ERROR*** ITYPE HAS THE VALUE ',ITYPE,'; EXECUTION TERMINATED')
STOP

ITYPE = 1 -- UNIFORM LINE SOURCE.
FLEN=P1
10C CONTINUE
PAT=1.0
IF(V.NE.0.) PAT = SIN(PI*P1*V)/(PI*P1*V)
GO TO 999

ITYPE = 2 -- UNIFORM LINEAR ARRAY
FLEN=P1
NELP=II
PAT=1.0
IF(V.NE.0.) PAT = SIN(PI*P1*V)/(II*SIN(PI*P1*V/I1))
GO TO 999

ITYPE = 3 -- TRIANGULAR LINE SOURCE.
FLEN=P1/2.
PAT=1.0
IF(V.NE.0.) PAT = (SIN(FLEN*PI*V)/(FLEN*PI*V))**2
GO TO 999

ITYPE = 4 -- UNIFORM RECTANGULAR APERTURE
FLS=P1
FLT=P2
ARG1=PI*P1*U
ARC2=PI*P2*V
IF(ARG1) 401,402,401
403 IF(ARG2) 404,405,403
404 PAT=SIN(ARG1)/ARG1*SIN(ARG2)/ARG2
GO TO 999
405 PAT=SIN(ARG1)/ARG1
GO TO 999
406 PAT=1.0
GO TO 999

ITYPE = 5 -- UNIFORM RECTANGULAR ARRAY
A-44

C  5CC CONTINUE
C  FLS=P1
C  FLT=P2
C  NELS=I1
C  NElt=I2
700  ARC1=PI*P1*U
701  ARC2=PI*P2*V
702  IF(ARG1) 501,502,501
703  501 IF(ARG2) 5C3,5C4,503
704  503 PAT=SIN(ARG1)/(I1*SIN(ARG1/I1))*SIN(ARG2)/(I2*SIN(ARG2/I2))$
705  GO TO 999
706  504 PAT=SIN(ARG1)/(I1*SIN(ARG1/I1))
707  GC TO 999
708  502 IF(ARG2) 5C5,506,505
709  505 PAT=SIN(ARG2)/(I2*SIN(ARG2/I2))
710  GC TO 999
711  506 PAT=1.0
712  GO TO 999

C  ITYPE = 6 -- UNIFORM CIRCULAR APERTURE.
C  600 C=SCRT(U*U+V*V)
C  A=P1
714  IF(C.EQ.C.) GO TO 601
715  X=2.*PI*PI*C
716  CALL BESJ(X,1,BJ,0.0001,IER)
717  PAT=2.*BJ/X
718  GC TO 999
719  601 PAT=1.0
720  GC TO 999

C  ITYPE = 7 -- GENERAL ARRAY
C  700 IMAG=(C.C,1.C)
722  NELMT=I1*I2
723  TEMP=(C.C,C.O)
724  DO 701 J=1,NELMT
725  TEMP=TEMP+1.0*CEXP(IMAG*2.*PI*(U*SS(J)+V*TT(J)))
726  701 CONTINUE
727  PAT=REAL(TEMP)/NELMT
728  GC TO 999
729  99C PAT=SPECPT(U,V,ITYPE)
730  999 RETURN
731  END

COMPLEX FUNCTION SOURCE(M,N,U,V,ITYPE)
C  THIS SUBPROGRAM CALCULATES THE CURRENT AT POINT (M,N) DUE TO
C  THE PATTERN AT POINT (U,V).
C  ITYPE = 1 -- UNIFORM LINE SOURCE LOCATED AT S=0.
C  ? -- UNIFORM LINEAR ARRAY LOCATED AT S=0.
TRIANGULAR LINE SOURCE LOCATED AT S=0.

UNIFORM RECTANGULAR APERTURE.

UNIFORM RECTANGULAR ARRAY.

UNIFORM CIRCULAR APERTURE.

SPECIAL SOURCE (FUNCTION CPSOR(M,N,U,V,ITYPE) WILL BE CALLED.)

ITYPE > 7 -- SPECIAL SOURCE (FUNCTION CPSOR(M,N,U,V,ITYPE) WILL BE CALLED.)

VERSION 1 LEVEL 1

DATE OF LAST REVISION: 73/166 JULY 12, 1973

THIS WORK SUPPORTED BY NASA GRANT NGR 47-004-103.

FOR FURTHER INFORMATION CONTACT:
W. L. STUTZMAN DEPT. OF ELEC. ENGR. 951-6624.
E. L. COFFEY DEPT. OF ELEC. ENGR. 951-5494.

COMMON /PAT1/ P1,P2,P3,P4,P5,P6,P1,SS(400),TT(400),RR(400)
COMMON /PAT2/ 11,12,13,14,15

CALL LOCSCR(M,N,S,T)
IF(ITYPE.GT.7) GO TO 990
GO TO (100,200,300,400,500,600,700),ITYPE

WRITE(6,10) ITYPE
10 FORMAT(1HC,5X,***ERROR*** ITYPE HAS THE VALUE '111', '2X, EXECUTION TERMINATED')

ITYPE = 1 -- UNIFORM LINE SOURCE

100 CONTINUE
FLN=P1
SOURCE=CEXP(-IMAG*PI*2.*T*V)/PI
GO TO 999

ITYPE = 2 -- UNIFORM LINEAR ARRAY

200 CONTINUE
FLN=P1
SOURCE=CEXP(-IMAG*2.*PI*V*T)/PI
GO TO 999

ITYPE = 3 -- TRIANGULAR LINE SOURCE

300 CONTINUE
FLN=P1
CON=ABS(2.*T/P1)
751 SOURCE = 2.*PI*CEXP(-IMAG*2.*PI*T*V)*(1.-CCN)
752 IF(CON.GT.1) SOURCE = (0.0,0.0)
753 GO TO 999

C
C
C

ITYPE = 4 -- UNIFORM RECTANGULAR APERTURE
C
C

4CC CONTINUE
C
C
C

FLS = P1
C
C
C

FLT = P2
C
C
C

SOURCE = CEXP(-IMAG*2.*PI*(S*U+V*T))/(P1*P2)
C
C
C

GO TO 999

C
C
C

ITYPE = 5 -- UNIFORM RECTANGULAR ARRAY
C
C
C

500 CONTINUE
C
C
C

FLS = P1
C
C
C

FLT = P2
C
C
C

SOURCE = CEXP(-IMAG*2.*PI*(S*U+V*T))/(P1*P2)
C
C
C

GO TO 999

C
C
C

ITYPE = 6 -- UNIFORM CIRCULAR APERTURE
C
C
C

600 RHC = SCRT(S*S+T*T)
C
C
C

A = P1
C
C
C

SOURCE = (C.C,C.C)
C
C
C

IF(RHC.LE.P1) SOURCE = CEXP(-IMAG*2.*PI*(S*U+T*V))/(2.*PI*P1**2)
C
C
C

GO TO 999

C
C
C

ITYPE = 7 -- GENERAL ARRAY
C
C
C

700 CONTINUE
C
C
C

SOURCE = CEXP(-IMAG*2.*PI*(U*S+V*T))/(11*I2)
C
C
C

GO TO 999

C
C
C

SOURCE = LCCSOR(M,N,U,V,ITYPE)
C
C
C

999 RETURN

C
C
C

SUBROUTINE LCCSOR(M,N,U,V)
C
C
C

INTEGER PX,PY
C
C
C

REAL INITLS,INITLT
C
C
C

COMMON /PAT1/ P1,P2,P3,P4,P5,P6,PI,SS(400),TT(400),RR(400)
C
C
C

COMMON /PAT2/ 11,12,13,14,15
C
C
C

COMMON /LCC/ ITYPE
C
C
C

IF(ITYPE.GT.7) GO TO 990
C
C
C

GO TO (100,200,300,400,500,600,700),ITYPE
C
C
C

WRITE(6,10) ITYPE
C
C
C

10 FORMAT(1HC,5X,'***ERROR*** ITYPE HAS THE VALUE ',I11,'*; EXECUTION TERMINATED')
C
C
C

STOP
C 100 CONTINUE
C INITLT=P2
C DELTAT=P3

782 S=0.
783 T=P2+(N-1)*P3
784 GO TO 999

C

785 200 CONTINUE
C
C

786 S=0.
787 T=(N-11/2-1)*P2
788 IF(11/2*2.EQ.11) T=T+0.5*P2
789 GO TO 999

C

790 300 GO TO 100
C

791 400 CONTINUE
C INITLS=P3
C INITLT=P4
C DELTAS=P5
C DELTAT=P6

792 S=P3+(M-1)*P5
793 T=P4+(N-1)*P6
794 GO TO 999

C

795 500 CONTINUE
C PX=I1
C PY=I2
C DISX=P3
C DISY=P4

796 S=(M-11/2-1)*P3
797 T=(N-12/2-1)*P4
798 IF(11/2*2.EQ.11) S=S+0.5*P3
799 IF(12/2*2.EQ.12) T=T+0.5*P4
800 GO TO 999

C

801 600 GO TO 400
C

802 700 CONTINUE
803 NELMT=(M-1)*I2+N
804 S=SS(NELMT)
805 T=TT(NELMT)
806 GO TO 999

C

807 990 CALL SPLOC(M,N,S,T)
808 999 RETURN
809 END
SUBROUTINE CURRENT(CURR, CURI, MCUR, NCUR, US, VS, CURCOF, IC)

THIS SUBROUTINE CALCULATES THE FINAL CURRENT DISTRIBUTION NECESSARY TO PRODUCE THE FINAL PATTERN F(U,V).


COMPLEX SOURCE, TEMP
REAL CURR(51,51), CURI(51,51), US(500), VS(500), CURCOF(500)
REAL UORG(100), VORG(100), CURG(100)
COMMON / START/ NORG, UORG, VORG, CURG
COMMON / LOC/ ITYPE
DO 100 M=1, MCUR
DO 100 N=1, NCUR
CURR(M,N)=0.
100 CURI(M,N)=0.
DO 200 M=1, MCUR
DO 200 N=1, NCUR
DO 200 I=1, NORG
TEMP=SOURCE(M,N, US(I), VS(I), ITYPE)
CURR(M,N)=CURR(M,N)+CURG(I)*REAL(TEMP)
CURI(M,N)=CURI(M,N)+CURG(I)*AIMAG(TEMP)
200 CONTINUE
IF (IC.LT.?) RETURN

SUBROUTINE PRINT(A, M, N, STARTU, STARTV, DU, DV)

SUBROUTINE PRINT IS THE GENERAL OUTPUT SUBROUTINE --
IT WILL PRINT OUT CO-ORDINATES (U,V) AND VALUES A(I,J)
10 ROWS AND 10 COLUMNS TO A PAGE.

DIMENSION A(51,51), U(51), V(51)
WRITE(6,6969)
6969 FORMAT (1H1)
CC 10 J=1,51
629 U(J)=STARTU+(J-1)*DU
630 V(J)=STARTV+(J-1)*DV
N2=N/10.+0.99
M2=M/10.+0.99
A-49

833 DC 100 J1=1,M2
834 DC 200 K=1,N2
835 M3=1+(J1-1)*10
836 M4=M3+9
837 IF(M4.GT.M) M4=M
838 N3=1+(K-1)*10
839 N4=N3+9
840 IF(N4.GT.N) N4=N

PRINT A HEADING

WRITE(6,20) (V(I),I=N3,N4)
2C FORMAT(1H1,16X,F6.3,9(4X,F6.3))
WRITE(6,30)

PRINT A PAGE

K2=(M4-M3+1)*6
DO 400 J=1,K2
J2=J/6
IF(J2*6-J) 27,28,27
J3=J2+M3-1
WRITE(6,29) U(J3),(A(J3,I),I=N3,N4)
CC TO 4000
27 WRITE(6,31)

400 CONTINUE
IF(N4.EQ.N .AND. M4.EQ.M) GO TO 300
2CC CONTINUE
1CC CONTINUE
3CC RETURN
29 FORMAT(3X,F6.3,14X,F6.3,5X,10(F9.4,1X))
31 FORMAT(13X,'1',10H-')
}

SUBROUTINE PRCFIL(DATA1,NPT,NUMPAT)

INTEGER SF
INTEGER OUTPUT(101)
INTEGER BLANK,PLUS,SLASH,STAR
REAL DATA(401,2),BOUND(1C1)
REAL DATA1(401,2)
DATA BLANK,PLUS,SLASH,STAR / ',+,'|','* '/
47 CONTINUE

47 CONTINUE

IF(NPT.GT.600) GO TO 999
BIG=-1.0E10
SMALL = 1.0E10
CC 1,J=1,NPT

SUBROUTINE PRCFIL(DATA1,NPT,NUMPAT)
876 IF(DATA(J,2).LT.-60.0) DATA(J,2)=-60.0
877 IF(DATA(J,2).LT.SMALL) SMALL=DATA(J,2)
878 IF(DATA(J,2).GT.BIG) BIG=DATA(J,2)
879 1 CONTINUE
880 DIFF=ABS(BIG-SMALL)
881 SF = C
882 IF(DIFF.LT.100.) GO TO 21
883 IF(CIFF.LT.100.) GO TO 20
884 DO 2 J=1,1C
885 IF(CIFF.LT.100.) GO TO 2
886 SF=J
887 GO TO 20
888 2 CONTINUE
889 4C0 WRITE(6,1CC)
890 100 FORMAT('0 YOUR DATA IS TOO LARGE FOR THIS PROGRAM.')
891 RETURN
892 1C DO 3 J=1,1C
893 K=11-J
894 IF(DIFF.LT.100.) GO TO 3
895 SF=-K
896 GO TO 20
897 3 CONTINUE
898 GO TO 4C0
899 2C DO 4 J=1,NPT
900 4 DATA(J,2) = DATA(J,2)*10.**(SF)

C CALCULATE BOUNDS

901 21 SCALE=DIFF/100.
902 DO 5 J=1,1C
903 K=J-1
904 5 BOUND(J)=(BIG-K*SCALE)*10.**(SF)

C PRINT TITLE

905 WRITE(6,64C) NUMPAT
906 64C FORMAT(26X,'PATTERN NUMBER',15//)
907 IF (SF.EQ.C) GO TO 200
908 WRITE(6,4004) SF
909 4004 FORMAT(53X, 'SCALE FACTOR IS 10**',12/)
910 WRITE(6,65C) (BOUND(J),J=1,101,20)
911 65C FORMAT(/X,5(F7.3,13X),F7.3,2X,'REAL',5X,'C9.1')
912 DO 6 J1=1,NPT
913 J=NPT+1-J1
914 60 K=1,1C
915 5C CUTPUT(K)=BLANK
916 IF((J1-1)/10-(J-1)) 62,61,62
917 61 CC 40 K=1,101,10
918 4C CUTPUT(K)=PLUS
919 GO TO 87
920 62 CUTPUT(1)=SLASH
921 CUTPUT(101) = SLASH
SUBROUTINE CCNTUR(K,L,CCNLOW,CCNMAX,CCNINT,NUMPAT)
C***PRINT2*********************************************************************
C THIS SUBPROGRAM GIVES A CONTOUR MAP OF THE MATRIX A
C K AND L ARE THE MAXIMUM VALUES OF I AND J
C IF K=L=51 OR 101 AXES WILL BE SET UP AS FOR A PATTERN PLOT
C DELCON=DELTA(INCREMENT) BETWEEN CONTOURS FOR CONUR SUBROUTINE
C CCNLOW=LOWEST CONTOUR LEVEL
C CCNMAX=HIGHEST CONTOUR LEVEL
C CCNINT=CONTOUR INTERVAL
C NUMPAT=PATTERN NUMBER
C
DIMENSION A(51,51)
DIMENSION ALPHA(10)
DIMENSION CCN AT LEAST L
DIMENSION COL(101)
DATA ALPHA/1HC,1H1,1H2,1H3,1H4,1H5,1H6,1H7,1H8,1H9/
DATA BLANK,CUT/1H,1H/.
C
IF(K.LE.1.CR.L.LE.1) RETURN
CCNINT=CCNINT
CCNMAX=CCNMAX
CCNLOW=CCNLOW
CELCON=CELCON
WRITE(6,87) NUMPAT
87 FCMPAT(IHC,"FCR THE PATTERN NUMBERED",I5)
IF(CCINT) 99,99,100
99 BIC=-1.E27
99 SMALL=1.E27
99 GO 98 I=1,K
99 GO 98 J=1,L
99 IF(A(I,J) .GT. BIG) BIG=A(I,J)
99 IF(A(I,J) .LT. SMALL) SMALL=A(I,J)
99 CONTINUE
CONINT=(BIG-SMALL)/10.
DELCON=C.5*CONINT
CCALOW=SMALL+DELCON
CONMAX=BIG-DELCON

WRITE(6,71) DELCON,CONLOW,CONMAX,CONINT
FORMAT(1HO,)*DELCON=*,F10.5,3X,CONLOW=*,F10.5,3X,CONMAX=*,

PRINT LEVEL DESIGNATIONS
MCHAR=ABS((CONMAX-CONLOW)/CONINT+1.1)
CON=CONMAX+CONINT
ICON=M-1
CON=CON-CONINT
WRITE(6,72) ICON,CON
FORMAT(1HC,'CONTOUR LEVEL ',12,'=',F10.5)
CONTINUE

WRITE HEADING
DO 32 J=1,101
COL(J)=BLANK
CONTINUE
32
DO 35 J=1,101,2
COL(J)=DCT
CONTINUE
35
WRITE(6,200)
FORMAT(1H1)
N1=L*2
IF(N1.GT.101) N1=101
WRITE(6,101) (COL(J),J=1,N1)
FORMAT(/1HC,14X,101A1)
CO I=1,K
CE 31 J=1,101
CCL(J)=BLANK
CONTINUE
31
J2=-1
CO 2 J=1, L
J2=J2+2
IICON=-1
CON=CONMAX+CONINT
CO 50 M=1,MCHAR
IICON=ICON+1
CON=CON-CONINT
IF(A(I,J).GT.(CON+DELCN)) GO TO 50
IF(A(I,J).LT.(CON-DELCN)) GO TO 50
IF(A(I,J).LE.51) CCL(J2)=ALPHA(ICON+1)
IF(A(I,J).GT.51) COL(J)=ALPHA(ICON+1)
GO TO 2
A-53

1012 5C  CONTINUE
1013 2  CONTINUE
1014 WRITE(6,140) (COL(J1),J1=1,101)
1015 14C  FORMAT(1H13X,*',1C1A1)
1016 1  CONTINUE
1017 RETURN
1018 END

1019 SUBROUTINE PATCON(RDATA,MMAX,NMAX,ICODE,CONLOW,CONMAX,CONINT,
$STARTU,STARTV,DELTAU,DELTAV,NUMPAT,ISYMM)
C
1020 REAL RDATA(51,51),UAXIS(11),LOW(12),HIGH(12)
1021 INTEGER CUTPUT(101),LEVEL(12),BLANK
1022 DATA BLANK/' '/
1023 DATA LEVEL/'C',' ','21,'3','40,1','5','6','7','8','9','1','-','1'/
1024 CALL DATE(I,J,K)
1025 WRITE(6,10) I,J,K,NUMPAT
1026 10 FORMAT(1H1,* PATTERN CONTOUR SUBPROGRAM',34X,'DATE = ',A2,'-',A2
$,-',A2,3CX,'PATTERN NUMBER',15///////)
1027 IF(ICCDE.EQ.0) WRITE(6,11)
1028 IF(ICCDE.EQ.1) WRITE(6,12)
1029 IF(ICCDE.EQ.2) WRITE(6,13)
11 11 FORMAT(42X,'CONTOUR PLOT OF THE DESIRED PATTERN '////)
1031 12 FORMAT(46X,'CONTOUR PLOT OF THE INITIAL PATTERN'////////)
1032 13 FORMAT(45X,'CONTOUR PLOT OF THE FINAL PATTERN IN DB.'////)

C 1033 FINALU=STARTU+(MMAX-1)*DELTAU
1034 FINALV=STARTV+(NMAX-1)*DELTAV
1035 U1=STARTU
1036 U2=FINALU
1037 V1=STARTV
1038 V2=FINALV
1039 NCOUNT=MMAX
1040 NCOUNT=NMAX

C 1041 IF(ISYMM-1) 70,30,20
1042 20 UBIG=AMAX1(AHS(STARTU),ABS(FINALU))
1043 U1=-UBIG
1044 U2=UBIG
1045 MCOUNT=2*NCOUNT-1
1046 IF(ISYMM.EQ.2) GO TO 70

C 1047 30 VBIG=AMAX1(AHS(STARTV),ABS(FINALV))
1048 V1=-VBIG
1049 V2=VBIG
1050 MCOUNT=2*NCOUNT-1

C 1051 70 CONTINUE
C
C ESTABLISH LOWER AND UPPER LIMITS
NUMCON=(CONMAX-CONLO)/CONINT+1.5

CONTINUE

LOW(11)=-1.E3C
HIGH(12)=1.E30
HIGH(11)=LOW(1)
LOW(12)=HIGH(NUMCON)
MSKIP=100/(MCOUNT-1)
NSKIP=100/(NCOUNT-1)

CU=(U2-U1)/10.
DO 40 I=1,11
40 WRITE(6,41) (UAXIS(I),I=1,11)
42 FFORMAT(I3X,11(F7.4,3X)/16X,11(F7.4,3X))

CU=(U2-U1)/100.
CV=(V2-V1)/100.
N1=NSKIP-1
CONTINUE

DO 51 K=1,101
51 OUTPUT(K)=BLANK

U=U1+(N-1)*DU

IF(U*L+V/V.GT.1.0) GO TO 60

IJ=1
IK=1
J=(U-STARTU)/DELTAV+1.5
K=(V-STARTV)/DELTAV+1.5
IF(IG.GE.1.AND. IG.LE.MM) IJ=0
IF(IK.GE.1.AND. IK.LE.NMAX) IK=0

101 IF(IJ) 102,202,201
102 IF(IK) 300,1000,300

200 IF(ISYMM=-1) 6C,6C,201
201 J=1.5-(U-STARTU)/DELTAV
202 IF(IJ) 6C,202,60

300 IF(ISYMM.EQ.0 .OR. ISYMM.EQ.2) GO TO 60
301 K=1.5-(V-STARTV)/DELTAV
302 IF(IK) 60,1000,60

1CC F=FDATA(J,K)
1001 IF(F.LE.LGW(1)) GO TO 1001
1002 IF(F.GT.HIGH(NUMCON)) GO TO 1002
SUBROUTINE LIST(CURR, CURI, McUR, NCUR)

THIS SUBROUTINE LISTS ARRAY ELEMENT COORDINATES AND CURRENTS


DIMENSION CURR(51,51), CURI(51,51)
DO 10 M=1, MCUR
DO 10 N=1, NCUR
J=(M-1)*NCUR+N
CALL LOCSCRM(M, N, S, T)
WRITE(6, 100) J, S, T, CURR(M, N), CURI(M, N)
10 CONTINUE
100 FORMAT(3X, 14, 5X, 4(E14.7, 2X))
RETURN
ENC
7. Appendix: The ANTDATA Computer Program

7.1 Introduction

When dealing with two-dimensional antenna patterns data display becomes a very important phase of an antenna study. The ANTDATA computer program was written to accomplish this purpose. It is used for publication quality graphical display of patterns and source distributions. These plots are in one (profile), two (contour) and three dimensional forms. The program is written in FORTRAN IV and has been used on an IBM 370/155 with an on-line CALCOMP drum plotter.

This program is for support of the ANTSYN program. There are several subroutines of ANTSYN which provide data output, e.g. PRINT, PROFIL, CONTUR, PATCON and LIST. These may be sufficient for many needs and they do supply quantitative information. However, after synthesizing patterns using ANTSYN if further data display is desired ANTDATA can be used. In this way only those plots which are of interest to the designer are plotted. ANTSYN provides a preview capability for ANTDATA. Both programs could be combined. But when they are separate the program sizes are about 220 K for ANTSYN and 220 K for ANTDATA instead of one 440 K program. Also after previewing the results of ANTSYN, the user can easily select which (if any) of the plot options in ANTDATA he wishes to exercise.

ANTDATA is currently set up to use the correction positions and coefficients from ANTSYN to reconstruct the pattern and source distribution using some of the ANTSYN subroutines. This is done to minimize storage space. If storage is no problem the program could be altered to work directly from pattern and current arrays. Although, one must then use the resolution (array dimensions) used in ANTSYN, which may not be sufficient to see all of the detailed structure in the plots.
The original pattern is based on a Woodward-Lawson pattern. If the user wishes to use a different original pattern, he could write a subroutine, ORGPAT, and use it to initialize the pattern magnitude array A. The corrections found in ANTSYN and passed to ANTDATA would then be used to form the final pattern A as programmed here.

7.2 Program Organization

Again a modular structure using several subroutines has been used to allow for modifications. The main program generates the pattern and current arrays and controls which plots are made. Fig. 7.1 shows a block diagram of the program organization. Subroutines PAT, SPECPT, SOURCE, SPSOR, LOCSPR, and SPLOC are used in generating the pattern and source arrays and are also used in ANTSYN. PLOT1, PLOT1C and PLOT1P are used to plot profiles (cuts through one plane) of the pattern magnitude in dB, the source magnitude, and the source phase. PLOT2 and its subroutines (CNLAL and PLOTL) are used to draw accurate contour maps of the pattern magnitude in dB, the source magnitude, and/or the source phase. PLOT3 and its subroutines (THREE2, THREE3, THREE4, and THREE5) are used to plot the pattern magnitude in dB, the current magnitude, and/or the current phase with a three dimensional effect.

In the next section a description of how the user controls which plots are obtained is discussed. In Section 7.4 a list of important program variable definitions is given. Section 7.5 has descriptions of the subroutines shown in Fig. 7.1. Finally, Section 7.6 is a statement listing of the ANTDATA program.
Figure 7.1 Block diagram of ANTDATA program.
7.3 User's Guide to ANTDATA

The following steps are what the user must consider when he uses ANTDATA. When an input data card must be supplied it will be underlined.

**Step 1.** Specify pattern number and location in storage.

Read NUMPAT and NUMTRK from a card under a 1415 format. This is the pattern number of the job submitted to ANTDATA. NUMTRK is the track number of disk storage where the data for this pattern is thought to be stored. The program will look at that track first. If the pattern number on that track does not match NUMPAT, all tracks will be searched. If the pattern is found on an unexpected track or not found at all, messages will be printed out.

This step can be altered if the input form is different. For instance pattern data could be read in using cards.

**Step 2.** Array size.

Read MMAX and NMAX from a card under a 1415 format. These are the sizes of the pattern magnitude, current magnitude, and current phase arrays, all loaded into $A(,)$, for PLOT 2 and PLOT 3. For the examples presented in Chapter 4 MMAX and NMAX were 151.

**Step 3.** Number of correction coefficients.

The variables ITEMP and ITEMP1 are read from disk storage. ITEMP is the number of correction coefficients of the original pattern. ITEMP1 is the number of correction coefficients for the final pattern, not including the original ones.

**Step 4.** Pattern data.

Data concerning the original and final patterns are read off of disk storage. They are NUMPAT, TITLE, ISYMM, ITER, ISUC, FNORM, IDISK, NORC, IC, ($UORG(J), VORG(J), CORG(J), J=1, ITEMP), (US(J), VS(J), CORCOF(J), J=1, ITEMP1), ITYPE, P1, P2, P3, P4, P5, P6, PI, (SS(J), TT(J), J=1, 400), I1, I2, I3, I4, I5, NCUR, NCUR.
Refer to the statement listing of subroutine INPUT of the ANTSYN program for a meaning of P1, P2, ... and I1, I2, ..... These vary with ITYPE.

If the original pattern is not of the Woodward-Lawson type the ORGPAT subroutine of ANTSYN could be used to load the original pattern and then corrections added to it to form the final pattern for use with ANTDATA.

**Step 5. Options for pattern magnitude plots.**

Read OPT1U, OPT1V, OPT2, and OPT3 from a card under a 4I1 format. Use zeros for no plot and ones for plot.

**Step 6. U profile location.**

Read CONST from a card under a 8F10.0 format. This is the value of V where the profile is made. In other words, the profile is parallel to the U-axis with a value of V equal to CONST. If CONST is zero the profile is on the U-axis. Use only if OPT1U=1.

**Step 7. V profile location.**

Read CONST from a card under a 8F10.0 format. This is the value of U where the profile is made. In other words, the profile is parallel to the V-axis with a value of U equal to CONST. If CONST is zero the profile is on the V-axis. Use only if OPT1V=1.

**Step 8. Parameters for PLOT2 and PLOT3 of pattern.**

Read LOWCON and DASH from a card under a 8F10.0 format. Use only if OPT2 and/or OPT3 is 1.

**Step 9. Pattern contour parameters.**

Read CONLOW, CONMAX, and CONINT from a card under a 8F10.0 format. Use only if OPT2 is 1.

**Step 10. Options for current magnitude plots.**

Read OPT1U, OPT1V, OPT2, and OPT3 from a card under a 4I1 format. Use zeros for no plot and ones for plot. OPT1U and OPT1V now refer to S and T profiles of the current magnitude.
Step 11. S profile location.

Read CONST from a card under a 8F10.0 format. This is the value of T where the profile is made. Use only if OPT1U now is 1.

Step 12. T profile location.

Read CONST from a card under a 8F10.0 format. This is the value of S where the profile is made. Use only if OPT1V now is 1.

Step 13. Parameters for PLOT2 and PLOT3 of current magnitude.

Read LOWCON and DASH from a card under a 8F10.0 format. Use only if OPT2 and/or OPT3 is 1.


Read CONLOW, CONMAX, and CONINT from a card under a 8F10.0 format. Use only if OPT2 is 1.

Step 15. Options for current phase plots.

Read OPT1U, OPT1V, OPT2, and OPT3 from a card under a 411 format. Use zeros for no plot and ones for plot. OPT1U and OPT1V now refer to profiles of current phase in the S and T directions.

Step 16. Parameters for PLOT2 and PLOT3 of current phase.

Read LOWCON and DASH from a card under a 8F10.0 format. Use only if OPT2 and/or OPT3 is 1.


Read CONLOW, CONMAX, and CONINT from a card under a 8F10.0 format. Use only if OPT2 is 1.

Step 18. Go to Step 1 if another job is to be run.
7.4 Program Variables

Many variables used in this program were also used in ANTSYN and their definitions are found in Section 6.4.

7.4.1 Definition of Some Important Integer Variables Used in ANTDATA

ITEMP = Number of original correction coefficients, CORG.

ITEMP1 = Number of correction coefficients (not including original ones), CORCOF.

MMAX = Number of points of first subscript of arrays of pattern magnitude, current magnitude, and current phase used in PLOT2 and PLOT3.

NMAX = Number of points of second subscript of arrays of pattern magnitude, current magnitude, and current phase.

OPT1U = Plot control for subroutines PLOT1, PLOT1C, and PLOT1P. It controls profile plots of the pattern magnitude in the U direction, the current magnitude in the S direction, and/or the current phase in the S direction. If it is 1 a plot is made, otherwise no plot is made.

OPT1V = Plot control for subroutines PLOT1, PLOT1C and PLOT1P. It controls profile plots of the pattern magnitude in the V direction, the current magnitude in the T direction and/or the current phase in the T direction. If it is 1 a plot is made, otherwise no plot is made.

OPT2 = Plot control for PLOT2 subroutine. It controls contour plots of pattern magnitude, current magnitude, and current phase. If it is 1 or greater a plot is made, otherwise no plot is made.

OPT3 = Plot control for PLOT3 subroutine. It controls three dimensional plots of pattern magnitude, current magnitude, and current phase. If it is 1 or greater a plot is made, otherwise no plot is made.

7.4.2 Definition of Some Important Real Variables Used in ANTDATA

A( , ) = Two dimensional array with MMAX by NMAX entries. It must be dimensioned to handle these entries. It is used for the pattern magnitude in dB, the current magnitude, and the current phase.
CONINT = The interval between contour levels for PLOT2 subroutine.
CONLOW = The lowest contour level for PLOT2 subroutine.
CONMAX = The highest contour level for PLOT2 subroutine.
CONST = The amount a profile is displaced from an axis (U, V, S, or T).
DASH = The contour level for PLOT2 subroutine equal to and below which all contours will be dashed. Above this value contours will be solid.
LOWCON = The floor of the PLOT3 subroutine. Three dimensional plots will have all values below LOWCON set to zero. This is used to "clean up" the plot.

7.5 Subroutine Descriptions

The subprograms PAT, SPECPT, SOURCE, SP6OR, LOC6OR, and SP6LOC have been discussed in Section 6.5. The remaining subprograms of ANTDATA are briefly described in this section. The contour and three dimensional plotting packages were obtained from other individuals and are so referenced. The one-dimensional plots were written by the authors and S. Kauffman. All of the plotting packages were written for use on the Virginia Tech CALCOMP plotter and as such require the use of some local plot subroutines. These are also explained.

7.5.1 ANTDATA Plot Subroutines

SUBROUTINE PLOT1
Purpose:
To produce a profile plot of far field pattern magnitude vs. an appropriate variable (U, V or THETA).
Usage:
CALL PLOT1 (PSTRT, PEND, IP, CODE, CONST, NUMPAT)
Description of Parameters:

- **PSTRT**: Abscissa of first point to be plotted.
- **PEND**: Abscissa of last point to be plotted.
- **IP**: Number of points to be plotted. This must be less than the dimension of array PTS. A reasonable choice is 4001.
- **CODE**: Labeling code. If CODE = 0 the horizontal axis will be left blank and the value stored in CONST will be reproduced at the bottom of the plot in the form "THETA=CONST". If CODE=1, then the horizontal axis will be labeled "+V" and "-V" and the value stored in CONST will be reproduced as "U=CONST." If CODE=2, the horizontal axis will be labeled "+U" and "-U" and CONST will be reproduced as "V=CONST."
- **CONST**: Label constant.
- **NUMPAT**: Pattern number.

Remarks:

i. PSTRT and PEND may span any interval. However, PTS(1) must correspond to PSTRT, and PTS(IP) must correspond to PEND.

ii. Before each subroutine call, PTS must be loaded with the appropriate data points. PTS must be in dB, with points equally spaced from PSTRT to PEND.

COMMON Blocks Required: COMMON /PLT1/ PTS

Subroutines and Function Subprograms Required: FACTOR, PLOT, SYMBOL, NUMBER, AXIS.

SUBROUTINE PLOTIC

Purpose:

To produce a profile plot of line source or aperture current distribution magnitude vs. an appropriate variable (S, T or THETA).

Usage:

CALL PLOTIC (PSTRT, PEND, IP, CODE, CONST, NUMPAT)
Description of Parameters:

PSTRT - Abscissa of first point to be plotted.
PEND - Abscissa of last point to be plotted.
IP - Number of points to be plotted. This must be less than the dimension of array PTS. A reasonable choice is 4001.
CODE - Labeling code. If CODE = 0, the horizontal axis will be left blank and the value stored in CONST will be reproduced at the bottom of the plot in the form "THETA = CONST." If CODE = 1, then the horizontal axis will be labeled "+T" and "-T" and the value stored in CONST will be reproduced as "S=CONST." If CODE = 2, the horizontal axis will be labeled "+S" and "-S" and CONST will be reproduced as "T=CONST."
CONST - Label constant.
NUMPAT - Pattern number.

Remarks:

1. The vertical axis is automatically scaled from 0.0 to 0.05, 0.0 to 0.1, 0.0 to 0.2, 0.0 to 0.5, 0.0 to 1.0 depending on the range of the data in PTS.

2. Before each subroutine call, PTS must be loaded with appropriate data points none of which must be less than zero.

COMMON Blocks Required: COMMON /PLT1/, PTS

Subroutines and Function Subprograms Required: FACTOR, PLOT, SYMBOL, NUMBER, AXIS.

SUBROUTINE PLOT1P

Purpose:

To produce a profile plot of line source or aperture current distribution phase (in degrees) vs. an appropriate variable (S, T or THETA).

Usage:

CALL PLOT1P (PSTRT, PEND, IP, CODE, CONST, NUMPAT)
Description of Parameters:

- **PSTRT** - Abscissa of first point to be plotted.
- **PEND** - Abscissa of last point to be plotted.
- **IP** - Number of points to be plotted. This must be less than the dimension of array PTS. A reasonable choice is 4001.
- **CODE** - Labeling code. If CODE = 0, the horizontal axis will be left blank and the value stored in CONST will be reproduced at the bottom of the plot in the form "THETA=CONST." If CODE = 1, the horizontal axis will be labeled "+T" and "-T" and the value stored in CONST will be reproduced as "S=CONST." If CODE = 2, the horizontal axis will be labeled "+S" and "-S" and CONST will be reproduced as "T=CONST."
- **CONST** - Label constant.
- **NUMPAT** - Pattern number.

Remarks:

1. Before each subroutine call, PTS must be loaded with appropriate data points in degrees (-180 ≤ PTS ≤ 180).

**COMMON Blocks Required:** COMMON /PLT1/ PTS

**Subroutines and Functions Required:** FACTOR, PLOT, SYMBOL, NUMBER, AXIS

**SUBROUTINE PLOT2**

Purpose:

To draw a contour map of data in array A.

Usage:

CALL PLOT2 (N, M, CONLOW, CONMAX, CONINT, NUMPAT, DASH)

Description of Parameters:

- **N** - Number of points to be plotted in horizontal direction.
- **M** - Number of points to be plotted in vertical direction.
- **CONLOW** - Lowest contour level to be plotted.
CONMAX - Highest contour level to be plotted.
CONINT - Interval between contour levels.
NUMPAT - Pattern number.
DASH - Contour levels below DASH will be dashed rather than solid.

Remarks:

1. If CONINT = 0 or CONLOW = CONMAX, the subroutine will determine the contour levels to be plotted.

COMMON Blocks Required: COMMON /ARRAY/ A

Subroutine and Function Subprograms Required: CNLAL, PLOT, FACTOR, SYMBOL, NUMBER

Reference: D. A. Vossler, E. S. Robinson

SUBROUTINE CNLAL

Purpose:
To determine the maximum and minimum of array X. To calculate the increment that will give 10 equally spaced contours between the maximum and minimum of array X.

Usage:
CALL CNLAL (N, M, CNTRLO, CMAX, CNTRAL, NC)

Description of Parameters:

N - Number of points in horizontal direction.
M - Number of points in vertical direction.
CNTRLO - Least value of array X.
CMAX - Greatest value of array X.
CNTRAL - ABS(CMAX-CNTRLO)/10.
NC - IF NC=0: CNTRLO and CMAX are returned.
     IF NC=1: CNTRLO, CMAX, and CNTRAL are returned.
SUBROUTINE PLOTL

Purpose:

To plot a straight line between two points.

Usage:

CALL PLOTL(X1,Y1,X2,Y2,SCALE)

Description of Parameters:

X1 - Abscissa of starting point.
Y1 - Ordinate of starting point.
X2 - Abscissa of end point.
Y2 - Ordinate of end point.
SCALE - Scale factor used in converting (X1,Y1) and (X2,Y2) to proper plot size.

Remark:

PLOTL is equivalent to the following two statements:
CALL PLOT(SCALE*X1+2.,SCALE*Y1+0.25,3)
CALL PLOT(SCALE*X2+2.,SCALE*Y2+0.25,2)
Where PLOT is a standard VPI plot subroutine.

COMMON Blocks Required: None.
Subroutine and Function Subprograms Required: None.

SUBROUTINE PLOT3

Purpose:

To draw a perspective view of a contoured surface.

Description of Parameters and Important Variables:

N - Number of data points along first axis.
M - Number of data points along the second axis.
Numpat  -  Pattern number (for labeling)

K  -  Code that tells whether to draw the grid lines:
   K=1: Along the N-Dimension only.
   K=2: Along the M-Dimension only.
   K=3: Along both dimensions.

SDists  -  Distance from surface to eye when perspective is calculated -- SKists > .6 usually won't show any distortion due to Parallax.

Yaw  -  (In degrees) How far the object is turned away from the viewer.

Pitch  -  (In degrees) How the surface is lowered or raised at the front edge. (Positive pitch tends to expose the top of the figure.)

Size  -  (In inches) The size of the cube that encloses the figure.

Kode  -  "Hidden Line" switch. If Kode=0, do not draw hidden lines... If Kode=1, all hidden lines are plotted.

Mgn  -  Whether to draw the outline of the cube to help orient the viewer. MGN=0: Do not draw any outline of the cube. MGN=1: Draw the outline of the cube separate from the figure. MGN=2: Draw the outline of the cube superimposed on the surface plot. MGN=3: Draw only the three edges of the cube that meet at the origin, superimposed on the surface plot.

Scale  -  How tall to make the surface relative to the height of the cube. SCALE=0: Do not scale the data at all but trust the user that the data is/not so high that it runs off the paper. SCALE=1: Scale the data so the top of the data just touches the top of the cube. SCALE=0.3: Scale the data so the top of the surface is three-tenths as high as the cube.

Remarks:

i. It is very expensive to draw opaque surfaces, because the program has to determine the visibility of every point, the computer time doubles or triples...Depending on how many line segments are partially visible.

ii. The contents of array A are destroyed in computation.

Common blocks required:

Common /Array/ A
Common /Three6/ ANGA, ANGB, HV, V, SH, SV
Common /Three7/ SL, SM, SN, CX, CY, CZ, QX, QY, QZ, SD
Subroutine and Function Subprograms Required: THREE2, THREE3, THREE3, THREE4, THREE5, PLOT, FACTOR, SYMBOL, NUMBER.

Reference: Howard Jesperson, Iowa State University.

SUBROUTINE THREE2

Purpose:

To find the corners of a three-dimensional rotated cube.

Usage:

CALL THREE2(X, Y, Z, XP, H, V, KODE)

Description of Parameters:

(X,Y,Z) — Vectors of length 2. Position of rotated vertices.
XP — Height above paper.
(H,V) — Vectors of length 10. Location of projected vertices on paper.
KODE — Dummy variable

COMMON Blocks Required: None

Subroutine and Function Subprograms Required: THREE4

SUBROUTINE THREE4

Purpose:

To find the location of a point in the rotated cube.

Usage:

CALL THREE4(X, Y, Z, XP, YP, ZP, KODE)

Description of Parameters:

(X,Y,Z) — Coordinates of point to be located.
XP — Height above paper of point.
(YP,ZP) — Coordinates of projection on paper.
COMMON Blocks Required:

COMMON /THREE6/ ANGA, ANGB, HV, D, SH, SV  
COMMON /THREE7/ SL, SM, SN, CX, CY, CZ, QX, QY, QZ, SD

Subroutine and Function Subprograms Required: None.

SUBROUTINE THREE3
Purpose:
To plot a perspective of a three-dimensional figure.

Usage:
CALL THREE3(X, Y, N, M, H, V, K, KODE)

Description of Parameters:

X - Vector of length 2
Y - Vector of length 2
N - Number of points in first direction
M - Number of points in second direction
H, V - Vectors of length 10...Coordinates of projected vertices of cube.
K - Grid Line Code (See Subroutine PLOT3)
KODE - Hidden Line Switch (See Subroutine PLOT3)

COMMON Blocks Required:

COMMON /THREE6/ ANGA, ANGB, HV, D, SH, SV  
COMMON /THREE7/ SL, SM, SN, CX, CY, CZ, QX, QY, QZ, SD  
COMMON /ARRAY/ A

Subroutine and Function Subprograms Required: THREE4, THREE5, PLOT

SUBROUTINE THREE5
Purpose:
To see if a point on the projected three-dimensional figure is visible.

Usage:
CALL THREE5(XI, YJ, M, W, P, KODE)
Description of Parameters:

- **XI** - Abscissa of the projected point.
- **YJ** - Ordinate of the projected point.
- **M** - Number of horizontal points.
- **N** - Number of vertical points.
- **P** - PLOT CODE; IF $P = -1$ INVISIBLE TO VISIBLE
- **KODE** - Hidden Line Code (See Subroutine PLOT3)

COMMON Blocks Required:

- COMMON /ARRAY/ A
- COMMON /THREE6/ ANGA, ANGB, HV, D, SH, SV
- COMMON /THREE7/ SL, SM, SN, CX, CY, CZ, QX, QY, QZ, SD

Subroutine and Function Subprograms Required: None.

7.5.2 Virginia Tech Subroutines

**VPI UTILITY SUBPROGRAMS**

<table>
<thead>
<tr>
<th>Subprograms</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>DATE</td>
<td>To return the current month, day, and year.</td>
</tr>
<tr>
<td>STIME</td>
<td>To return the time of day in ten thousandths of an hour (Integer Format)</td>
</tr>
<tr>
<td>TIMEON</td>
<td>To set the interval timer to zero</td>
</tr>
<tr>
<td>TIMECK</td>
<td>To return the amount of CPU time used in hundredths of seconds since the last call to TIMEON.</td>
</tr>
</tbody>
</table>

Reference: VIRGINIA POLYTECHNIC INSTITUTE AND STATE UNIVERSITY, "COMPUTING CENTER USER'S GUIDE," VOL. 7, "UTILITY PROGRAMS."

**VPI PLOTTER SUBROUTINES**

<table>
<thead>
<tr>
<th>Subroutine</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>AXIS</td>
<td>To draw a labeled axis of a desired length with annotated tic marks every inch.</td>
</tr>
</tbody>
</table>
FACTOR  To scale the plot in both the X and Y directions.
NUMBER  To draw a floating point number.
PLOT    To move the pen from one point to another, to draw a line between points, to establish a new origin, and to signal the end of a plot.

SYMBOL To plot a string of alphanumerical characters.

Reference: VIRGINIA POLYTECHNIC INSTITUTE AND STATE UNIVERSITY, "COMPUTING CENTER USER'S GUIDE," VOL. 6, "THE PLOTTER."

INTEGER FUNCTION ICVT
Purpose:
    To convert an integer to character format internal coding.
Usage:
    ICHAR=ICVT(NUM)
Remark:
    This function was originally written in assembler. Object deck is read in under the SYSLIN dataset.
7.6 Statement Listing of ANTDATA

7.6.1 Job Control Language

Listed below are the JCL statements required for the ANTDATA program to run on the Virginia Tech IBM 370/155 system.

```
//B0663PL3 JOB 507C2,COFFEY
/ *MAIN TIME=19,LINES=3,REGION=250K,CARDS=0
/ *PRIORITY PRIORITY
/ *FORMAT PL, FORMS=PFGURAGV, PEN=XXFIAE, DDNAME=CALCUMP
/ EXEC FORTGCG, PARM.GO='PAPER=39,PTIME=119', EP=MAIN
//FORT.SYSIN DD *
/ *
//GO.SYSLIB DD
// DD DSN=VPI.PLOTLIB, DISP=SHR
// DD DSN=VPI.SSPLIB, DISP=SHR
//GO.SYSLIN DD
// DD *
/ *
//GO.FT22F001 DD DSN=ANTDATA.A507C2,UNIT=3330, VOL=SER=UPSERPK, DISP=SHR
//GO.FT06F001 DD SYSOUT=A, DCB=(BLKSIZE=133, RECFM=F)
//GO.SYSIN DD *
/ *
```

7.6.2 Source Listing

Listed below are the FORTRAN IV statements of the ANTDATA program.
PROGRAM DESCRIPTION:
ANTDATA IS THE OUTPUT PROGRAM USED IN CONJUNCTION WITH ANTSYN2.
BY SPECIFYING APPROPRIATE PARAMETERS ANTDATA WILL GIVE A ONE,
TWO, OR THREE DIMENSION PLOT OF THE PATTERN (IN OD.), THE SOURCE
MAGNITUDE, AND THE SOURCE PHASE.

INPUT/OUTPUT:
THE MAJORITY OF INPUT IS TAKEN FROM ANTSYN2 VIA DIRECT-ACCESS
UNIT 22 (ANTDATA.A507C2). PARAMETERS AND JOB OPTIONS ARE SUPPLIED
THROUGH UNIT 5 (SYSIN). ALL OUTPUT IS CHANNELLED TO UNIT 6
(SYSPRINT) AND THE PLOTTER (PLOT1).

VERSION 1 73 - 094 -- APRIL 5, 1973

WRITTEN BY: W. L. STUTZMAN
S. R. KAUFFMAN
E. L. COFFEY

UNDER NASA GRANT: 47-004-103

ADDITIONAL SUBPROGRAMS REQUIRED:
ICVT -- ASSEMBLER LANGUAGE SUBPROGRAM TO CONVERT AN INTEGER
TO A2 CHARACTER FORMAT.

DEFINE FILE 22(35,9100,E,NREC)

DIMENSION A(151,151),PTS(4001),US(500),VS(500),CURCOF(500)
DIMENSION UOR(100),VOR(100),COR(100)
DIMENSION AU(151),AV(151)
INTEGER TITLE(20)
REAL INITLS,NITLT
REAL LOWCON
INTEGER OPT1V,OPT2,OPT3,PX,PY
COMPLEX CTEMP,CI
COMMON /PLT1/ PTS
COMMON /ARRAY/ A
COMMON /PAT1/ P1,P2,P3,P4,P5,P6,P1,SS(400),TT(400),RR(400)
COMMON /PAT2/ 11,12,13,14,15
COMMON /LOC/ ITYPE
IPAGE=0
PI=3.14159265
CI=CMPLX(0.0,1.0)
CALL STIME(ITIME)
TIME=0.
9999 CONTINUE
CALL Tipecn
IPAGE=IPAGE+1
CALL DATE(ll,Jl,K1)
CALL STIME(IT)
IHR=IT/10000
IFR=IT-IHR*10000
FM=IFR/10000.
IMIN=FM
ISEC=(FM-IMIN)*60
IHR=ICVT(IHR)
IMIN=ICVT(IMIN)
ISEC=ICVT(ISEC)
IPG=ICVT(IPAGE)
WRITE(6,1) ll,Jl,K1,IHR,IMIN,ISEC,IPG
1  FORMAT(llH1,2X,'ANTDATA VERSION 1 LEVEL 2',
$8X,0VPI EE DEPT.,5X, 'DATE = ',A2,'-',A2,'-',I,A2,
$5X, 'TIME = ',A2,'.',A2,'.',A2,'.',A2,'.',A2)
READ PATTERN NUMBER AND TRACK -- VERIFY THAT PATTERN IS ACTUALLY STORED
READ(5,10,END=999) NUMPAT,NUMTRK
10  FORMAT(1415)
IF(NUMPAT.EQ.0) GO TO 999
WRITE(6,704) NUMPAT
704  FORMAT(' PLOT OUTPUT FOR PATTERN ',15,': ')
READ(22,20) NUM
20  FORMAT(A4)
IF(NUM.EQ.NUMPAT) GO TO 51
GO TO 30 I=2,25
READ(22*I,20) NUM
30  CONTINUE
NUMPAT IS NOT ON DISK
WRITE(6,60) NUMPAT
60  FORMAT(1H0,'PATTERN NUMBER ',15, ' WAS NOT LOCATED -- PROGRAM HALT')
GO TO 999
NUMPAT FOUND ON UNEXPECTED TRACK
WRITE(6,60) NUMPAT,NUMTRK,I
60  FORMAT(1H0,'PATTERN NUMBER ',15, ' WAS NOT FOUND ON TRACK ',12,
$0 BUT WAS LOCATED ON TRACK ',12)
NUMTRK=I
51  CONTINUE
REGIN PROCESSING
READ(5,10) PMAX,NMAX
READ(22*NUMTRK,7C) ITEMP,ITEMP1
TO FORMAT(104X,2A4)
READ(22,NUMTRK,101) NUMPAT, TITLE, ISYMM, ITER, ISUC, FNORM, IVISK, NCRG, IC, (UORG(J), VORG(J), CORG(J), J=1, ITER), (US(J), VS(J), CORCOF(J), J=1, TEMP), (SS(J), TT(J), J=1, TEMP1), ITYPE, P1, P2, P3, P4, P5, P6, P7, (SS(J), TT(J), J=1, TEMP1), ITYPE, P1, P2, P3, P4, P5, P6, P7, (SS(J), TT(J), J=1, TEMP1), ITYPE, P1, P2, P3, P4, P5, P6, P7, (SS(J), TT(J), J=1, TEMP1), ITYPE, P1, P2, P3, P4, P5, P6, P7, (SS(J), TT(J), J=1, TEMP1), ITYPE, P1, P2, P3, P4, P5, P6, P7, (SS(J), TT(J), J=1, TEMP1), ITYPE, P1, P2, P3, P4, P5, P6, P7, (SS(J), TT(J), J=1, TEMP1), ITYPE, P1, P2, P3, P4, P5, P6, P7, (SS(J), TT(J), J=1, TEMP1), ITYPE, P1, P2, P3, P4, P5, P6, P7, (SS(J), TT(J), J=1, TEMP1), ITYPE, P1, P2, P3, P4, P5, P6, P7, (SS(J), TT(J), J=1, TEMP1), ITYPE, P1, P2, P3, P4, P5, P6, P7

101 FORMAT(75A4,11(200A4))

C C READ OPTIONS FOR PATTERN MAGNITUDE
C
READ(5,29) OPT1U, OPT1V, OPT2, OPT3
29 FORMAT(4(11))

IF(OPT1U-1) 80, 81, 80
81 CONTINUE

READ(5,31) CONST

IF(MMAX.LE.1) GO TO 80
80 DO 90 J=1,4001

U=(J-1)*0.0005-1.0
90 SUM=0.

DO 91 K=1,NCRG

91 SUM=SUM+CORC(K)*PAT(U-UORG(K), CONST-VORG(K), ITYPE)

IF(IC.LE.0) GO TO 90
90 DO 91 K=1,IC

91 SUM=SUM+CORC(K)*PAT(U-US(K), CONST-VS(K), ITYPE)

CONTINUE

PT5(J)=2C.*ALOG10(ABS(SUM))

WRITE(6,92) CONST

92 FORMAT(*CV-AXIS PROFILE PLOT REQUESTED -- V = ',F6.3)

CALL PLOT1(-1.0,1.0,4001,2,CONST,NUMPAT)

IF(OPT1V-1) 82, 83, 82
82 CONTINUE

READ(5,31) CONST

IF(MMAX.LE.1) GO TO 82
82 DO 90 J=1,4001

V=(J-1)*0.0005-1.0
90 SUM=0.

DO 91 K=1,NCRG

91 SUM=SUM+CORC(K)*PAT(V-VORG(K), CONST-VORG(K), ITYPE)

IF(IC.LE.0) GO TO 90
90 DO 91 K=1,IC

91 SUM=SUM+CORC(K)*PAT(V-VS(K), CONST-VS(K), ITYPE)

CONTINUE

PT5(J)=2C.*ALOG10(ABS(SUM))

WRITE(6,93) CONST

93 FORMAT(*CV-AXIS PROFILE PLOT REQUESTED -- U = ',F6.3)

CALL PLOT1(-1.0,1.0,4001,1,CONST,NUMPAT)

IF(OPT2+OPT3) 85, 85, 84
84 CONTINUE

C C GENERATE PATTERN ARRAY
C

READ(5,31) LOWCON, DASH

IF(MMAX.LE.1 OR NMAX.LE.1) GO TO 239

100 DELTAU=2.0/(MMAX-1)
101 DELTAV=2.0/(NMAX-1)
102 WRITE(6,761) LOWCON, LOWCON
103 FORMAT(*PATTERN IS NOW BEING GENERATED. IF PATTERN < ',F7.2,

90 PATTERN = ',F7.2)
104 IF(ITYPE.GT.5) GO TO 5000

C C 701 FORMAT(*C PATTERN IS NOW BEING GENERATED. IF PATTERN < ',F7.2,

90 PATTERN = ',F7.2)
104 IF(ITYPE.GT.5) GO TO 5000

C C 701 FORMAT(*C PATTERN IS NOW BEING GENERATED. IF PATTERN < ',F7.2,

90 PATTERN = ',F7.2)
104 IF(ITYPE.GT.5) GO TO 5000

C C 701 FORMAT(*C PATTERN IS NOW BEING GENERATED. IF PATTERN < ',F7.2,
LOAD UP AU AND AV

DO 2000 I=1,NMAX
U=(I-1)*DELTAU
2000 AU(I)=PAT(U,O.,ITYPE)
DO 2010 J=1,NMAX
V=(J-1)*DELTAV
2010 AV(J)=PAT(O.,V+ITYPE)

BEGIN

U=-1.0-DELTAU
DO 2040 M=1,NMAX
U=U+DELTAU
2040 V=-1.0-DELTAV
DO 2050 N=1,NMAX
V=V+DELTAV
TEMP=C.

CONTINUE

K=1,NORG
I=ABS(U-UORG(K))/DELTAU+1.5
J=ABS(V-VORG(K))/DELTAV+1.5
TEMP=TEMP+CORG(K)*AU(I)*AV(J)
IF(IC.LE.C) GO TO 2020
DO 2040 K=1,IC
I=ABS(U-US(K))/DELTAU+1.5
J=ABS(V-VS(K))/DELTAV+1.5
TEMP=TEMP+CORCOF(K)*AU(I)*AV(J)
CONTINUE
A(M,N)=20.*ALOG10(ABS(TEMP))
GO TO 239

CONTINUE

CONTINUE

IF(MMAX.LE.1.OO OR NMAX.LE.1.OO) GO TO 230
READ(5,31) CONLOW,CONMAX,CONINT
31 FORMAT(6FI0.0)
IF(MMAX.LE.1.OO OR NMAX.LE.1.OO) GO TO 230
DO 257 M=1,NMAX
DO 257 N=1,NMAX
IF(A(M,N).LT. LOWCON) A(M,N)=LOWCON
257 CONTINUE
WRITE(6,220) CONLOW,CONMAX,CONINT
A-79

154 220 FORMAT('CONTOUR PLOT OF PATTERN REQUESTED')
     $1.\text{\textdollar}\text{\textdollar}$  LOWEST CONTOUR = ',F7.2/
     $1.\text{\textdollar}\text{\textdollar}$  HIGHEST CONTOUR = ',F7.2/
     $1.\text{\textdollar}\text{\textdollar}$  CONTOUR INTERVAL = ',F7.2)

155 CALL PLOT2(MMAX,NMAX,CONLOW,CONMAX,CONINT,NUMPAT,NASH)
6   21C IF(OPT3) 230,230,231
157 231 WRITE(6,24C)
158 240 FORMAT(1HO,'THREE-DIMENSIONAL PLOT OF PATTERN REQUESTED')
159 CALL PLOT3(MMAX,NMAX,NUMPAT)
160 23C CONTINUE
161 IF(MMAX.LE.1.OR.NMAX.LE.1) WRITE(6,23C)
162 23 FORMAT('ONE DIMENSIONAL')
163 85 CONTINUE

CEND OF PATTERN

C

164 IA=0
165 IF(ITYPE.EQ.1) GO TO 401
166 IF(ITYPE.EQ.3) GO TO 401
167 IF(ITYPE.EQ.4) GO TO 401
168 IF(ITYPE.EQ.6) GO TO 401
169 4CC WRITE(6,4C2)
170 402 FORMAT(1HO,'THE SOURCE IS AN ARRAY -- THIS PGM IS FOR CONTINUOUS SOURCES ONLY')
171 IA=1
172 401 CONTINUE
173 P3TEMP=P3
174 P5TEMP=P5
175 P6TEMP=P6
176 IF(ITYPE.EQ.1) 404,403,404
177 404 IF(ITYPE.EQ.3) 405,403,405
178 403 CONTINUE

C ITYPE= 1 OR 3
179 INITLS=0.
180 DELTAS=0.
181 FINALS=0.
182 INITLT=P2
183 FINALT=P2+P1
184 P3=P1/4000.
185 CELTAT=P3
186 GO TO 410
187 405 IF(ITYPE.EQ.4) 407,406,407
188 406 CONTINUE

C ITYPE=4
189 INITLS=P3
190 FINALS=P3+P1
191 INITLT=P4
192 FINALT=P4+P2
193 P5=P2/4000.
194 P6=P2/4000.
195 CELTAT=P6
196 DELTAS=P5
197 GO TO 41C
198 407 CONTINUE
199 IF(ITYPE.EQ.6) 410,409,410
A-80

\begin{verbatim}
200  409 INITLS=P3
201  FINALS=P3+2.*P1
202  INITLT=P4
203  FINALT=P4+2.*P1
204  P5=P1/2000.
206  DELTAT=P6
207  DELTAS=P5
208  \textbf{CONTINUE}
209  READ(5,29) OPTIU,OPIV,OPT2,OPT3
210  \textbf{IF(OPTIU-1) 302,301,302}
211  \textbf{CONTINUE}
212  \textbf{READ(5,31) CONST}
213  \textbf{IF(IA.EQ.1) GO TO 3000}
214  \textbf{IF(INMAX.LE.1) GO TO 302}
215  J=1
216  \textbf{IF(DELTAT.NE.0.) J=1.5+(CONST-INITLT)/DELTAT}
217  DO 303 I=1,4001
218  CTEMP=(0.0,0.0)
219  \textbf{IF(NORG.LE.0) GO TO 304}
220  DO 305 K=1,NORG
221  CTEMP=CTEMP+CORG(K)*SOURCE(I,J,UORG(K),VORG(K),ITYPE)
222  \textbf{CONTINUE}
223  \textbf{READ(5,31) CONST}
224  \textbf{IF(IA.EQ.1) GO TO 3000}
225  \textbf{IF(INMAX.LE.1) GO TO 322}
226  \textbf{CONTINUE}
227  \textbf{READ(5,31) CONST}
228  \textbf{IF(IA.EQ.1) GO TO 3000}
229  \textbf{IF(INMAX.LE.1) GO TO 322}
230  \textbf{CONTINUE}
231  \textbf{READ(5,31) CONST}
232  \textbf{IF(IA.EQ.1) GO TO 3000}
233  \textbf{IF(INMAX.LE.1) GO TO 322}
234  J=1
235  \textbf{CONTINUE}
236  \textbf{IF(DELTAS.NE.0.) I=1.5+(CONST-INITLT)/DELTAS}
237  DO 313 J=1,4001
238  CTEMP=(0.0,0.0)
239  \textbf{CONTINUE}
240  DO 315 K=1,NORG
241  CTEMP=CTEMP+CORG(K)*SOURCE(I,J,UORG(K),VCRG(K),ITYPE)
242  \textbf{CONTINUE}
243  \textbf{READ(5,31) CONST}
244  \textbf{IF(IA.EQ.1) GO TO 3000}
245  \textbf{CONTINUE}
246  \textbf{READ(5,31) CONST}
247  \textbf{IF(IA.EQ.1) GO TO 3000}
248  \textbf{CONTINUE}
249  \textbf{CONTINUE}
250  \textbf{CONTINUE}
251  \textbf{CONTINUE}
252  P3=P3TEMP
253  P5=P5TEMP
254  P6=P6TEMP
255  MCUR=51
256  NCUR=51
\end{verbatim}
IF(DOPI2+CPT3) 320,320,321
CONTINUE
READ(5,31) LOWCON,DASH
IF(IA.EQ.1) GO TO 333

C GENERATE CURRENT MAGNITUDE ARRAY
C
DO 330 M=1,MCUR
DO 331 N=1,NCUR
CALL LOCSOR(M,N,S,T)
CTEMP=0.
DC 339 K=1,NORG
339 CTEMP=CTEMP+CORG(K)*SOURCE(M,N,UORG(K),VORG(K),ITYPE)
DC 332 K=1,IC
332 CTEMP=CTEMP+CRCOF(K)*SOURCE(M,N,US(K),VS(K),ITYPE)
A(P,N)=CABS(CTEMP)
CONTINUE
CONTINUE
CONTINUE
IF(OPT2) 350,350,351
READ(5,31) CONLOW,CONMAX,CONINT
IF(IA.EQ.1) GO TO 360
IF(VMAX.LE.1.OR.NMAX.LE.1) GO TO 360
WRITE(6,340) CONLOW,CONMAX,CONINT
CALL PLOT2(MCUR,NCUR,CONLOW,CONMAX,CONINT,NUMPAT,DASH)
340 IF(CPT3) 360,360,361
360 IF(IA.EQ.1) GO TO 360
WRITE(6,355)
355 FORMAT(1HO,'THREE DIMENSION PLOT OF CURRENT MAGNITUDE REQUESTED'
$'LOWEST CONTOUR = ',F7.4/
$' HIGHEST CONTOUR = ',F7.4/
$' CONTOUR INTERVAL = ',F7.4)
360 CALL PLOT3(MCUR,NCUR,NUMPAT)
365 CONTINUE
366 IF(VMAX.LE.1.OR.NMAX.LE.1) WRITE(6,23)
367 CONTINUE
C END OF CURRENT MAGNITUDE
C
C READ OPTIONS FOR CURRENT PHASE
C
READ(5,29) OPT1U,OPTIV,OPT2,OPT3
IF(OPT2+CPT3) 520,520,521
CONTINUE
IF(IA.EQ.1) GO TO 533
READ(5,31) LOWCON,DASH

C GENERATE CURRENT PHASE
C
DO 530 M=1,MCUR
DO 531 N=1,NCUR
CALL LOCSOR(M,N,S,T)
CTEMP=0.
DC 549 K=1,NORG
549 CTEMP=CTEMP+CORG(K)*SOURCE(M,N,UORG(K),VORG(K),ITYPE)
549 CONTINUE
300 DO 532 K=1,IC
301 532 CTEMP=CTEMP+CORCOF(K)*SOURCE(M,N,US(K),VS(K),ITYPE)
302  CREAL = REAL(CTEMP)
303  CIMAG = AIMAG(CTEMP)
304  A(M,N) =ATAN2(CIMG,CREAL)*180./PI
305 531 CONTINUE
306 53C CONTINUE
307 533 CONTINUE
308 IF(OPT2) 550,550,551
309 551 READ(5,31) CONLOW,CONMAX,CONINT
310 IF(IA.EQ.1) GO TO 560
311 IF(PMAX.LE.1.OR.NMAX.LE.1) GO TO 560
312 WRITE(6,560) CONLOW,CONMAX,CONINT
313 54C FORMAT('CONTOUR PLOT OF CURRENT PHASE REQUESTED /
$'  \$     LOWEST CONTOUR = 'F7.2/
$     HIGHEST CONTOUR = 'F7.2/
$     CONTOUR INTERVAL = 'F7.2)
314  CALL PLOT2 (MCUR,NCUR,CONLOW,CONMAX,CONINT,NUMPAT,DASH)
315 55C IF(OPT3) 560,560,561
316 561 IF(IA.EQ.1) GO TO 560
317 WRITE(6,555)
318 555 FORMAT('THREE DIMENSION PLOT OF CURRENT PHASE REQUESTED')
319  CALL PLOT3(MCUR,NCUR,NUMPAT)
320 56C CONTINUE
321 IF(PMAX.LE.1.OR.NMAX.LE.1) WRITE(6,23)
322 52C CONTINUE
323  CALL TIMECK(ISEC)
324  FMIN=ISEC/6000.
325  WRITE(6,897) FMIN
326 897 FORMAT('EXECUTION TIME: ',F6.2,' MINUTES.').
327  TIME=TIME+FMIN
328 GO TO 9999
329 999 WRITE(6,600)
330 60C FORMAT('*** END OF EXECUTION *** /)
331  CALL PLOT(0.0,0.0,-4)
332  WRITE(6,898) TIME
333 898 FORMAT('TOTAL EXECUTION TIME: ',F7.2,' MINUTES.').
334  CALL STIME(JTIME)
335  IT=JTIME-ITIME
336  FMIN=IT/10000.*60.
337  WRITE(6,899) FMIN
338 899 FORMAT('TOTAL ELAPSED TIME: ',F7.2,' MINUTES.').
339  STOP
340 ENC
341 SUBROUTINE PLOT1(PSTRT,PEND,IP,CODE,CONST,NUMPAT)

SUBROUTINE PLOT1

WRITTEN BY: S. R. KAUFMAN

DATE: 73-113 APRIL 23, 1973
INPUT:

PSRT  -- BEGINNING OF PLOT
PEND  -- END OF PLOT
IP     -- NUMBER OF POINTS TO BE PLOTTED
CODE   -- LABELLING VARIABLE. IF CODE=0: LABEL='IHEA = ';
         IF CODE=1: LABEL='U = '; IF CODE = 2: LABEL = 'T = '
CONST  -- CONSTANT PARAMETER FOR LABEL
NUMPAT -- NUMBER OF PATTERN FOR LABEL.

INTEGER NAME(2), CODE
DIMENSION PTS(4001)
COMMON /PLT1/ PTS
CALL FACTOR(0.5)
CALL PLOT(8.91.,-3)
IF(CODE.GT.0) GO TO 3
CALL SYMBOL(-1.2,-6.6,8raitheta = 0.,8)
CALL NUMBER(-.3,-.8,-2,CONST,0.,3)
GO TO 6
3 IF(CODE.GT.1) GO TO 4
CALL SYMBOL(-1.,-.8,-2,1HU,0.,1)
CALL SYMBOL(-.9,-.8,-2,3H = 0.,3)
CALL NUMBER(-.2,-.8,-2,CONST,0.,3)
CALL SYMBOL(-2.6,-.4,-2,2H-2,0.,2)
CALL SYMBOL(2.4,-.4,-2,2H+V,0.,2)
GO TO 6
4 IF(CODE.GT.2) GO TO 5
CALL SYMBOL(-1.,-.8,-2,1HU,0.,1)
CALL SYMBOL(-.9,-.8,-2,3H = 0.,3)
CALL NUMBER(-.2,-.8,-2,CONST,0.,3)
CALL SYMBOL(-2.6,-.4,-2,2H-2,0.,2)
CALL SYMBOL(2.4,-.4,-2,2H+U,0.,2)
6 CONTINUE
PDOL=(PSRT-PEND)/IP
PTIC=(ABS(PSRT-PEND))/10.]
CALL AXIS(-5000.,1H,1400.,0.,IP,PTIC)
PSRT=PSRT+(6.*PTIC)+.00001
PTIC2=PTIC+.00001
CALL AXIS(1000.,1H,14000.,0.,PSRT,PTIC)
71 CALL PLOT(-1.,0.,3)
72 CALL PLOT(+1.,0.,3)
73 CALL PLOT(0.,9.8,2)
74 CALL PLOT(0.,9.8,2)
75 CALL PLOT(0.,9.8,2)
76 CALL SYMBOL(-.05,-.4,.2,1MO,0.,1)
77 X=0.05
78 CC 10 J=1,6
79 Y=0.5*(J-1)+0
80 CALL PLOT(-X,Y,3)
81 10 CALL PLOT(X,Y,2)
82 CALL PLOT(0.,0.,0.,3)
83 IF(PTS(1).LE.-50.) PTS(1)=-50.
84 FS=((PTS(1))/10.)*0.5
SUBROUTINE PLOTIC(PSTRT, PEND, IP, CODE, CONST, NUMPAT)

INTEGER NAME(2), CODE
CALL FACTOR(0.5)
CALL PLOT(-5., FS, 3)
CC 1 IW1=1, IP
THETS=((PSTRT-(IW1*POEL))*5.)/(ABS(PSTRT))
FDBS=((PTS(IW1))/10.)*+5.5
IF(FDBS.LT.0.5) GO TO 1
CALL PLOT(THETS, FDBS, 2)
CONTINUE
CALL SYMBOL(-5.0, -0.8, 0.125, IOHPATTERN = ,0., 1J)
FNUM=FLCATINUMPAT)
CALL NUMBER(-3.87, -0.8, 0.125, FNUM, 0.9, -1)
CALL AXES(-5.5, 0.5, 17FAR FIELD PATTERN, 17, 5.0, 90., -50., 10.)
CALL PLOT(8., -1.0, -3)
RETURN
END

SUBROUTINE PLOTIC

WRITTEN BY: S. R. KAUFMAN

DATE: 73-113 APRIL 24, 1973

INPUT:

PSTRT -- BEGINNING OF PLOT
PEND -- END OF PLOT
IP -- NUMBER OF POINTS TO BE PLOTTED
CODE -- LABELLING VARIABLE. IF CODE=0: LABEL IS 'THETA = '
CODE=1: LABEL IS 'S = '
CODE=2: LABEL IS 'I = '
CONST -- CONSTANT PARAMETER FOR LABEL.
NUMPAT -- PATTERN NUMBER FOR LABEL.

CALL FACTOR(0.5)
CALL PLOT(-5.0, -0.8, 0.125, IOHPATTERN = ,0., 1J)
FNUM=FLCATINUMPAT)
CALL NUMBER(-3.87, -0.8, 0.125, FNUM, 0.9, -1)
CALL AXES(-5.5, 0.5, 17FAR FIELD PATTERN, 17, 5.0, 90., -50., 10.)
CALL PLOT(8., -1.0, -3)
416  GO TO 6
417  IF (CODE.GT.2) GO TO 5
418  CALL SYMBOL(-1.0,-0.8,0.2,1,0.0,1)
419  CALL SYMBOL(-0.9,-0.8,0.2,3,0.0,3)
420  CALL NUMBER(-0.2,-0.8,0.2,CONST,0.0,3)
421  CALL SYMBOL(-2.6,-0.4,0.2,2,5,0.0,2)
422  CALL SYMBOL(2.4,-0.4,0.2,2,5,0.0,2)
423  CONTINUE
424  PDEL=(PSTRT-PEND)/IP
425  PTIC=ABS(PSTRT-PEND)/10.0
426  CALL AXIS(-5.0,0.0,1,H1,1,4.0,0.0,PSTRT,PTIC)
427  PSTRE=PSTRT+(6.0*PTIC+0.00001)
428  PTIC2=PTIC+0.00001
429  CALL AXIS(1.0,0.0,1,H1,1,4.0,0.0,PSTRE,PTIC2)
430  CALL PLOT(-1.0,0.0,3)
431  CALL PLOT(1.0,0.0,2)
432  CALL PLOT(0.0,0.0,3)
433  CALL PLOT(0.0,5.0,2)
434  CALL PLOT(0.0,5.0,3)
435  CALL SYM(-0.5,-0.4,0.2,1,H0,0.0,1)
436  X=0.05
437  Y=0.5+(J-1)*1.0
438  CALL PLOT(-X,Y,3)
439  CALL PLOT(X,Y,2)
440  CALL PLOT(X,Y,2)
441  CALL PLOT(0.0,0.0,3)
442  PSTRE=5.0*PSTRT
443  GMAX=0.0
444  DO 1 I=1,IP
445  IF (PTS(I).GT.GMAX) GMAX=PTS(I)
446  CONTINUE
447  IF (GMAX.GT.0.5) ASCLE=1.
448  IF (GMAX.LE.0.5) ASCLE=0.5
449  IF (GMAX.LE.0.2) ASCLE=0.2
450  IF (GMAX.LE.0.1) ASCLE=0.1
451  IF (GMAX.LE.0.05) ASCLE=0.05
452  PTS=(PTS(I))/ASCLE*5.+0.5
453  CALL PLOT(-5.0,PTS,3)
454  DO 7 I=1,IP
455  THETA=(PSTRT-(I*PDEL))
456  THETS=THETA/(ABS(PSTRT))/5.
457  APTS=((PTS(I))/ASCLE)*5.+0.5
458  CALL PLOT(THETS,APTS,2)
459  CONTINUE
460  IF (GMAX.GT.0.5) ATIC=0.2+0.0001
461  IF (GMAX.GT.0.5) ATIC=0.1+0.0001
462  IF (GMAX.LE.0.20) ATIC=0.04+0.0001
463  IF (GMAX.LE.0.1) ATIC=0.02+0.0001
464  IF (GMAX.LE.0.05) ATIC=0.01+0.0001
465  CALL AXIS(-5.5,0.5,16,HSOURCE MAGNITUDE,16.5,0.98,0.0,0.0,ATIC)
466  CALL SYMBOL(-5.0,-0.8,0.125,10,HCPATTERN = ,0.,10)
467  FNUM=FLOAT(NUMPAT)
468  CALL NUMBER(-3.5,-0.8,0.125,FNUM,0.,-1)
469  CALL PLOT(0.0,9.0,-3)
470  RETURN
471  ENC
SUBROUTINE PLOT1P(PSTRT, PEND, IP, CODE, CONST, NUMPAT)

SUBROUTINE PLOT1P

WRITTEN BY: S. R. KAUFMAN

DATE: 73-113 APRIL 24, 1973

INPUT:

PSTRT -- BEGINNING OF PLOT
PEND -- END OF PLOT
IP -- NUMBER OF POINTS TO BE PLOTTED (IP < 4062)
CODE -- LABELLING PARAMETER. IF CODE = 0: LABEL IS 
        'THETA = '; IF CODE = 1: LABEL IS 'S = '; IF CODE = 2
        LABEL = 'T = '.
CONST -- CONSTANT PARAMETER IN LABEL.
NUMPAT -- PATTERN NUMBER FOR LABEL.

INTEGER NAME(2), CODE
DIMENSION PTS(4001)
COMMON /PLTL/PTS
CALL FACTOR(0.5)
CALL PLCT(8., 1., -3)
IF(CODE.GT.0) GO TO 3
CALL SYMBOL(-1.2, -6., 2, 8, 'THETA = ', 0., 8)
CALL NUMBER(.3, -.8, .2, CONSTRAINT, 0., 3)
GO TO 6
3 IF(CODE.GT.1) GO TO 4
CALL SYMBOL(-1., -.8, 2, 1, 'HS', 0., 1)
CALL SYMBOL(-.9, -.8, .2, 3, ' = ', .0, 3)
CALL NUMBER(-.2, -.8, 2, CONSTRAINT, 0., 3)
CALL SYMBOL(-2.6, -.4, 2, '+'T', 0., 2)
CALL SYMBOL(2.4, -.4, 2, '+T', 0., 2)
GO TO 6
4 IF(CODE.GT.2) GO TO 5
CALL SYMBOL(-1., -.8, 2, 1, 'HT', 0., 1)
CALL SYMBOL(-.9, -.8, .2, 3, ' = ', .0, 3)
CALL NUMBER(-.2, -.8, 2, CONSTRAINT, 0., 3)
CALL SYMBOL(-2.6, -.4, 2, '-T', 0., 2)
CALL SYMBOL(2.4, -.4, 2, '-T', 0., 2)
GO TO 6
5 PCSEL = (PSTRT- PEND)/IP
PTIC = (ABS(PSTRT - PEND))/10.0
CALL AXIS(-5.0, 0.0, 1, 1, 0.0, 0.0, PSTRT, PTIC)
PSTRE = PSTRT + (6.0*PTIC + 0.00001
PTIC2 = PTIC + 0.00001
CALL AXIS(1.0, 1, 0.0, 4.0, 0.0, PSTRE, PTIC2)
CALL PLCT(-1., 0., 3)
CALL PLCT(1., 0., 2)
CALL PLOT(1., 0., 3)
CALL PLOT(0., 5.8, 2)
CALL PLOT(0., 0., 3)
A-87

506  CALL SYMBOL(-.05,-.4,.2,lHO,0.,1)
507  CALL PLOT(0.,0.,0.,3)
508  X=0.05
509  GO TO J=1,9
510  Y=0.5+(J-1)*1.0
511  CALL PLOT(-X,Y,3)
512  IC CALL PLOT(X,Y,2)
513  CALL PLOT(0.,0.,0.,3)
514  DC 1 IW1=1,IP
515  THETA=(PSTRT-(IW1*PDEL))
516  THETS=(THETA/(ABS(PSTRT)))*5.
517  PANGS=PTS(IW1)/180.*4.+4.5
518  IF(IW1.EQ.1)CALL PLOT(THETS,PANGS,3)
519  IF(IW1.EQ.1)GO TO 1
520  CALL PLOT(THETS,PANGS,2)
521  1 CONTINUE
522  CALL AXIS(-5.5,0.5,14HAPERTURE PHASE,14,8.,90.,-180.,45.)
523  CALL SYMBOL (-5.0,-0.8,0.125,10HPATTERN = 0.010)
524  FNLM=FLOAT(NUPPAT)
525  CALL NUMBER(-3.5,-0.8,0.125,FNUM,0.,-1)
526  CALL PLOT(8.,-1.,-3)
527  5 RETURN
528  END

SUBROUTINE PLOT2(N,M,CONLOW,CONMAX,CONINT,NUPPAT,DASH)
C
C   A = N BY M MATRIX OF DATA POINTS
C   CONLOW = LOWEST CONTOUR TO BE PLOTTED
C   CONMAX = HIGHEST CONTOUR TO BE PLOTTED
C   CONINT = INTERVAL BETWEEN CONTOURS
C   WORDS = TEXT OF PLOT LABEL
C   NCHAR= NUMBER OF CHARACTERS IN PLOT LABEL
C   CONTOURS BELOW -40. ARE PLOTTED AS DASHED LINES
C
C
C
530  DIMENSION A(151,151),RA(151),RB(151),X(151),Y(151)
531  COMMON /ARRAY/ A
532  CALL PLOT(8.,0.,-3)
533  CALL FACTOR (0.7)
534  MS=M
535  NS=N
536  RATIO=MS/NS
537  SCALE=10.
538  ANM=AMAX0(N-1,M-1)
539  IF(RATIO-1.0)1,2
540  1 SX=ANM
541  SY=RATIO*ANM
542  GC TO 3
543  2 SX=1./RATIO*ANM
544  SY=ANM
545  3 SMAX=AMAX1(SX,SY)
546  SS=SX/SMAX
547  SYS=SY/SMAX
548  IF(CONINT)4,4,5
549  4 CALL CNLAL(N,MS,CNTRL0,CMAX,CNTRL,0)
550  GC TO 7
A-88

551 CENTER = CENTRAL
552 IF(CNTRMAX.EQ.CNTRLOW)GO TO 6
553 CNTRLO = CONLOW
554 GO TO 7
556 CALL CNLAL(N,M,CNTRLO,CNTRMAX,CNTRAL,1)
557 7 CONTINUE
558 CONLOW = CNTRLO
559 CONMAX = CNTRMAX
560 CENTRAL = CENTRAL
561 CALL PLOTL(SS,SYS,0.,SYS,SCALE)
562 CALL PLOTL(0.,0.,SS,0.,SCALE)
563 CALL PLCTL(0.,0.,SS,SYSSCALE)
564 CALL PLOTL(1.,0.,0.,0.,SCALE)
565 CALL PLOT(1.00,0.25,3)
566 CALL PLOT(0.60,0.25,2)
567 CALL PLOT(0.60,0.25,2)
568 CALL PLOT(1.00,0.25,2)
569 CALL PLOT(1.00,0.25,2)
570 CALL SYMBOL(0.88,0.45,0.12,10,HPATTERN = 90,10)
571 FNUM = NUMPAT
572 CALL NUMBER(0.88,2.075,0.12,FNUM,90.,-1)
573 125 YCCNA = 1.0/SMAX
574 DELTAX = SX/FLOAT(N-1)
575 X(1) = 0.0
576 Y(1) = 0.0
577 RB(1) = A(1,1)
578 DO 27 J = 2,N
579 RB(J) = A(J,1)
580 27 X(J) = X(J-1) + DELTAX
581 DELTAY = SY/FLOAT(M-1)
582 DO 28 J = 2,M
583 28 Y(J) = Y(J-1) + DELTAY
584 DO 118 K = 2,M
585 RA(J) = RB(J)
586 DO 118 J = 2,N
587 118 RB(J) = A(J,K)
588 DO 118 J = 2,N
589 35 ASSIGN 112 TO L
590 RR = RA(J)
591 XX = X(J)
592 YY = Y(K-1)
593 37 RL = RR
594 XL = XX
595 YL = YY
596 39 IF(RL - RA(J-1)) 41,40,40
597 40 IF(RL - RB(J)) 42,50,50
598 41 RL = RA(J-1)
599 XL = X(J-1)
600 YL = Y(K-1)
601 GO TO 40
602 42 RL = RB(J)
603 XL = X(J)
604 YL = Y(K)
605 GO TO 50
606 50 RS = RR
607 XS = XX
A-89

608     YS = YY
609     IF (RS - RA(J-1)) 52, 52, 53
610     52 IF (RS - RB(J)) 60, 60, 54
611     53 RS = RA(J-1)
612     XS = X(J-1)
613     YS = Y(K-1)
614     GO TO 52
615     54 RS = RB(J)
616     XS = X(J)
617     YS = Y(K)
618     GO TO 60
619     60 RM = HR
620     XM = XX
621     YM = YY
622     IF (RM - RS) 62, 62, 61
623     61 IF (RM - RL) 70, 62, 62
624     62 RM = RA(J-1)
625     XM = X(J-1)
626     YM = Y(K-1)
627     IF (RM - RS) 64, 64, 63
628     63 IF (RM - RL) 70, 64, 64
629     64 RM = RB(J)
630     XM = X(J)
631     YM = Y(K)
632     70 YCS = YS*YCONA
633     YCM = YM*YCONA
634     YCL = YL*YCONA
635     71 YS = YS - SY
636     YM = YM - SY
637     YL = YL - SY
638     72 XCS = XS/SMAX
639     XCM = XM/SMAX
640     XCL = XL/SMAX
641     RC = CNTRL0
642     80 IF (RC = CMAX) GO TO 110
643     IF (RC = CMAX) GO TO 110
644     81 IF (RM = NE. RS) GO TO 91
645     82 IF (RL = NE. RM) GO TO 100
646     91 IF (RC - RS) 100, 95, 92
647     92 IF (RC - RM) 96, 93, 94
648     93 XPA = XCM
649     YPA = YCM
650     GO TO 99
651     94 IF (RC - RL) 106, 103, 110
652     95 Q = 0.0
653     GO TO 97
654     96 Q = (RC - RS)/(RM - RS)
655     97 XPA = XCS - Q*(XCS - XCM)
656     YPA = YCS - Q*(YCS - YCM)
657     98 Q = (RC - RS)/(RL - RS)
658     XPB = XCS - Q*(XCS - XCL)
659     YPB = YCS - Q*(YCS - YCL)
660     100 IF (RC = DASH) 10115, 10115, 10116
661     10115 XPH = 0.5*(XPA - XPB)
662     YPB = 0.5*(YPA - YPB)
663     IF (ABS (XPA - XPB) = .001) 5001, 5002, 5002
664     5001 IF (ABS (YPA - YPB) = .001) 100, 5002, 5002
SUBROUTINE CNTRAL(N,M,CNTRL0,CMAX,CNTRAL,NC)

DIMENSION X(151,151)

COMMON /ARRAY/ X

XMAX=X(I,1)

XMIN=X(I,1)

GO 10 J=1,M

GO 10 I=1,N

10 XMIN=AMIN(XMAX,X(I,J))

IF(NC.EQ.1) GO TO 40

IF(XMAX.EQ.0.) GO TO 20

SN=XMIN/XMAX

IF(SN).GT.20 GO TO 30

20 XCON=ABS(XMAX)

IF(ABS(XMIN).GT.ABS(XMAX)) XCON=ABS(XMIN)

CNTRL0=CNTRAL+INT(XMIN/CNTRAL)

RETURN

30 XCON=ABS(XMAX-XMIN)

CNTRL0=CNTRAL+INT(XMIN/CNTRAL)

RETURN

40 CMAX=CNTRAL+INT(XMAX/CNTRAL)

CNTRL0=CNTRAL+INT(XMIN/CNTRAL)

RETURN

END
**SUBROUTINE PLOT3**

**PURPOSE:** TO DRAW A PERSPECTIVE VIEW OF A CONTOURED SURFACE.

**DESCRIPTION OF PARAMETERS AND IMPORTANT VARIABLES:**

- **N** - NUMBER OF DATA POINTS ALONG FIRST AXIS.
- **M** - NUMBER OF DATA POINTS ALONG THE SECOND AXIS.
- **NUMPAT** - PATTERN NUMBER (FOR LABELLING).
- **K** - CODE THAT TELLS WHETHER TO DRAW THE GRID LINES:
  - K=1: ALONG THE N-DIMENSION ONLY.
  - K=2: ALONG THE M-DIMENSION ONLY.
  - K=3: ALONG BOTH DIMENSIONS.
- **CISTS** - DISTANCE FROM SURFACE TO EYE WHEN PERSPECTIVE IS CALCULATED -- SLISTS > 6 USUALLY WON'T SHOW ANY DISTORTION DUE TO PARALLAX.
- **YAW** - (IN DEGREES) HOW FAR THE OBJECT IS TURNED AWAY FROM THE VIEWER.
- **PITCH** - (IN DEGREES) HOW THE SURFACE IS LOWERED OR RAISED AT THE FRONT EDGE. (POSITIVE PITCH TENDS TO EXPOSE THE TOP OF THE FIGURE).
- **SIZE** - (IN INCHES) THE SIZE OF THE CUBE THAT ENCLOSES THE FIGURE.
- **KODE** - "HIDDEN LINE" SWITCH. IF KODE=0 DO NOT DRAW HIDDEN LINES...IF KODE=1, ALL HIDDEN LINES ARE PLOTTED.
- **SCALE** - HOW TALL TO MAKE THE SURFACE RELATIVE TO THE HEIGHT OF THE CUBE. SCALE=0: DO NOT SCALE THE DATA AT ALL.

REMARKS.

I. IT IS VERY EXPENSIVE TO DRAW OPAQUE SURFACES, BECAUSE THE PROGRAM HAS TO DETERMINE THE VISIBILITY OF EVERY POINT, THE COMPUTER TIME DOUBLES OR TRIPLES...DEPENDING ON HOW MANY LINE SEGMENTS ARE PARTIALLY VISIBLE.

II. THE CONTENTS OF ARRAY A ARE DESTROYED IN COMPUTATION.

COMMON BLOCKS REQUIRED:

```
COMMON /ARRAY/ A
COMMON /THREE6/ ANG,ANGB,HV,D,SH,SV
COMMON /THREE7/ SL,SM,SN,CX,CY,CZ,OX,OY,OZ,SD
```

SUBROUTINE AND FUNCTION SUBPROGRAMS REQUIRED:

```
THREE2
THREE3
THREE4
THREE5
PLOT
FACTOR
SYMBOL
NUMBER
```

REFERENCE: HOWARD JESPERSON, IOWA STATE UNIVERSITY.

MODIFIED FOR USE AT VPI BY: ROBERT C. KEPHART.
S. R. KAUFFMAN
W. L. STUTZMAN
E. L. COFFEY

SUBROUTINE PLOT3(N,M,NUMPAT)

```
C**N**= N BY M MATRIX OF DATA POINTS
C**N**= PLOTS= PLOT LABELING
C**N**= NCHAR= NUMBER OF CHARACTERS IN THE PLOT LABEL+SPACES
```

```
COMMON /ARRAY/ A
COMMON /THREE6/ ANG,ANGB,HV,D,SH,SV
COMMON /THREE7/ SL,SM,SN,CX,CY,CZ,OX,OY,OZ,SD
```
DIMENSION H(10), V(10), X(2), Y(2), Z(2), XP(8), A(151, 151)

K = 2
SOISTS = 6.0
PITCH = 30.
YAW = 45.
SIZE = 10.
KOCF = 0
MFN = 0
SCALE = 1.

CALL FACTOR(1.1)
CALL PLOT(8.0, -2.0)
CALL PLOT(0.4, 0.0, 2.0)
CALL PLOT(0.4, 8.0, 2.0)
CALL PLOT(0.0, 0.0, 2.0)

CALL SYMBOL(0.3, 1.0, 0.12, 10)

FNLM = FLOAT(NUMPAT)

CALL NUMBER(0.3, 2.1, 30, 90, 12, FNUM, 90, -1)

ANGA = (YAW + 270.) * .0174532
ANGB = PITCH * .0174532

IF (ABS(SN) .NE. 1.0) GO TO 10

WRITE(6, 20)

GO TO 2150

CONTINUE

SD = 1.0 / SCRT(1.0 - SN ** 2)

T = MAXC(M, N)

D = M ** 2 + N ** 2 + T ** 2

SCL = SOISTS * D

CX = -SL * SCL

CY = -SM * SCL

CZ = -SN * SCL

CALL SUBR(0.3, 1.0, 0.12, 10)

GL TO 2060

WRITE(6, 100) CX, CY, CZ

WRITE(6, 100) GX, GY, GZ

FORMAT(1X, 3F15.3)
Z(1) = A(I,J)
CALL 1000 J = 1, N
CALL 1000 K = 1, M
Z(1) = AMAXI(Z(1), A(J,K))
Z(2) = AMIN1(Z(2), A(J,K))

1000 CONTINUE
RANGE = |Z(2) - Z(1)|
DCL = 1.0
IF(Scale .NE. 0) DOL = T/RANGE*Scale
C SCALE THE SURFACE TO MAKE A "CUBE".
DO 30 I = 1, N
DO 30 J = 1, M
A(I,J) = (A(I,J) - I) * DOL
30 CONTINUE
Z(1) = C.C
Z(2) = T
2000 CALL THREE2 ( X, Y, Z, XP, H, V, KODE)
DO 2130 I = 1, 8
H(I) = (XP(I) - OX) * SM - (H(I) - OY) * SL * SD
V(I) = (V(I) - OZ) * SD
2130 CONTINUE
2100 H(10) = H(11)
H(9) = H(1)
DO 1001 J = 1, 8
H(9) = AMAXI(H(9), H(J))
H(10) = AMAXI(H(10), H(J))
1001 CONTINUE
1002 V(9) = V(11)
V(10) = V(1)
DO 1002 J = 1, 8
V(9) = AMAXI(V(9), V(J))
V(10) = AMAXI(V(10), V(J))
1002 CONTINUE
IF(MGN .EQ. 0) GO TO 2140
S = V
IF(MGN .EQ. 1) S = 1.5
SH = S/ (H(10) - H(9))
SV = S/ (V(10) - V(9))
SH = SIGN(A MINI(SH*SV), SH)
SV = SIGN(SH, SV)
IF(MGN .EQ. 1) CALL PLOT (0.0, 2., -3)
CALL SYMBOL(H(1) - H(9))*SH, (V(1) - V(9))*SV, 14, "0", 0., 1
CALL SYMBOL(H(3) - H(9))*SH, (V(3) - V(9))*SV, 14, "M", 0., 1
CALL SYMBOL(H(2) - H(9))*SH, (V(2) - V(9))*SV, 14, "Z", 0., 1
CALL SYMBOL(H(5) - H(9))*SH, (V(5) - V(9))*SV, 14, "N", 0., 1
CALL PLOT(-0.05, -0.3)
CALL PLOT(H(1) - H(9))*SH, (V(1) - V(9))*SV, 3
CALL PLOT(H(2) - H(9))*SH, (V(2) - V(9))*SV, 2
CALL PLOT(H(1) - H(9))*SH, (V(1) - V(9))*SV, 2
CALL PLOT(H(3) - H(9))*SH, (V(3) - V(9))*SV, 2
CALL PLOT(H(1) - H(9))*SH, (V(1) - V(9))*SV, 2
CALL PLOT(H(5) - H(9))*SH, (V(5) - V(9))*SV, 2
IF(MGN .EQ. 3) GO TO 2139
CALL PLOT(H(6) - H(9))*SH, (V(6) - V(9))*SV, 2
CALL PLOT(H(5) - H(9))*SH, (V(5) - V(9))*SV, 2
CALL PLOT(H(4) - H(9))*SH, (V(4) - V(9))*SV, 2
CALL PLOT(H(3) - H(9))*SH, (V(3) - V(9))*SV, 2
A-95

SUBROUTINE THREE2 (X, Y, Z, XP, H, V, KODE)
C FIND THE CORNERS OF THE ROTATED CUBE.
C
DIMENSION X(2), Y(2), Z(2), H(10), V(10), XP(16)
C
05C L = 0
07C DO 180 I = 1, 2
C
09C CG 170 J = 1, 2
C
11C DO 160 K = 1, 2
C
13C L = L + 1
14C CALL THREE4 (X(I), Y(J), Z(K), XP(L),
H(L), V(L), KODE )
16C CONTINUE
17C CONTINUE
18C CONTINUE
19C RETURN
20C END

SUBROUTINE THREE4 (X, Y, Z, XP, YP, ZP, KODE)
C FIND THE LOCATION OF A POINT IN THE ROTATED CUBE.
COMMON /THREE6/ ANGA, ANGB, HV, D, SH, SV
COMMON /THREE7/ SL, SH, SN, CX, CY, CZ, QX, QY, QZ, SD
SK = D / ( (X - CX) * SL + ( Y - CY ) * SM + ( Z - CZ ) * SN )
XP = CX + SK * ( X - CX )
YP = CY + SK * ( Y - CY )
ZP = CZ + SK * ( Z - CZ )
RETURN

SUBROUTINE THREE3 (X, Y, N, M, H, V, K, KODE)
C DRAW THE FIGURE.
COMMON /THREE6/ ANGA, ANGB, HV, D, SH, SV
COMMON /THREE7/ SL, SP, SN, CX, CY, CZ, QX, QY, QZ, SD
C
DIMENSION X(2), Y(2), H(10), V(10), A(151, 151)
COMMON /ARRAY/ A
INTEGER UP, DOWN, PEN, P, C
INTEGER P1, PO

ENC = 1.0 / 16.0
CAN USE 1 / 32 OR 1 / 64 FOR FINER INTERPOLATION

UP = 3
DOWN = 2
SH = HV / (H(10) - H(9))
SV = HV / (V(10) - V(9))
SH = SIGN(AMINI(SH,SV),SH)
SV = SIGN(SH,SV)
MP = M
NN = N

IF(K-1) 100,120,100
IF(K-3) 1110,120,1110

CRAWM LINES ALONG THE Y-AXIS
12C CONTINUE
L = 0
LD = 1
EC = 0.5 * LD

14C DO 1060 J = 1, M
Q = 0
YJ = J
16C DO 1030 I = 1, NN

L = L + LD
XI = L
CALL THREE5 ( XI,YJ,N,M,P,KODE)
PEN = UP
IF (P) 510, 520, 530
51C CONTINUE
IF (Q) 540, 550, 540
52C CONTINUE
IF (Q) 610, 1020, 610
53C CONTINUE
IF (Q) 540, 550, 540
54C CONTINUE
PEN = DOWN
GC TO 170
55C CONTINUE
IF (I .EQ. 1) GO TO 170
DI = CD
TC = L - LC
T = TC * DI
P1 = Q
560 IF (ABS(DI) .LT. END) GO TO 570
CALL THREE5 (T,YJ,N,M,P,KODE)
CI = DI * 0.5
IF (PO .EQ. C) GO TO 565
TC = T
P1 = PO
T = T - DI
GO TO 560
565 T = T + DI
GO TO 56C
57C CONTINUE
T = TO
IF ( P1 * P ) 170 , 170 , 580
58C CONTINUE
59C CONTINUE
ZP = A(L-0C,J)+T-L+LD(A(L,J)-A(L-LD,J))/LD
CALL THREE4(T,YJ,ZP,XP,HH,VV,KODE)
HH = ( (XP-QX)*SM - (HH-QY)*SL ) * SD
VV = ( VV - QZ ) * SD
HH = ( HH - H(9)) * SH
VV = ( VV - V(9)) * SV
CALL PLOT ( HH, VV, PEN )
60C PEN = 5 - PEN
GO TO 170
61C CONTINUE
PEN = DOWN
DI = OC
TO = L - LD
T = TO + DI
P1 = Q
62C IF ( ABS(DI) .LT. END ) GO TO 630
CALL THREE5 ( T,YJ,N,M,PO,KODE )
DI = DI * 0.5
IF ( PO .EQ. 0 ) GO TO 625
TC = T
P1 = PO
T = T + DI
GO TO 62C
625 T = T - DI
GO TO 62C
63C CONTINUE
T = TO
IF ( P1 * Q ) 600, 600, 590
17C CALL THREE4 ( XI, YJ, A(L,J), XP, HH, VV, KG(L) )
VV = ( VV - QZ ) * SD
HH = ( (XP-QX)*SM - (HH-QY)*SL ) * SD
19C HH = ( HH - H(9)) * SH
20C VV = ( VV - V(9)) * SV
CALL PLOT ( HH, VV, PEN )
102C Q = P
103C CONTINUE

C
L = L + LD
LC = -LD
DD = -DD

C
1060 CONTINUE
C
C
109C IF(K-3) 2060, 211C, 206C
DRAW LINES ALONG THE X-AXIS.

C

1110 CONTINUE

C

L = 0
LC = 1
CC = 0.5 * LD
XI = I
C = C
J = 1 + MM
L = L + LD
YJ = L
CALL THREE5 (XI, YJ, N, M, P, KODE)
PEN = UP
IF (P) 1510, 1520, 1530
1510 CONTINUE
IF (Q) 1540, 1550, 1540
1520 CONTINUE
IF (Q) 1610, 2010, 1610
1530 CONTINUE
IF (Q) 1540, 1550, 1540
1540 CONTINUE
PEN = DOWN
GO TO 1170
1550 CONTINUE
IF (J .EQ. 1) GO TO 1170
DI = DD
TC = L - LD
T = TO + DI
P1 = Q
1560 IF (ABS(DI) .LT. END) GO TO 1570
1561 CALL THREE5 (XI, T, N, M, PO, KODE)
DI = DI * 0.5
1562 IF (PO .EQ. 0) GO TO 1565
1563 TO = T
1564 P1 = PO
1565 T = T - DI
1566 GO TO 1560
1568 GO TO 1560
1570 CONTINUE
1571 T = TO
1572 IF (P1 * P) 1170, 1170, 1580
1580 CONTINUE
1590 CONTINUE
1595 ZP = A(I, L - LD) + (T - L + LD) * (A(I, L) - A(I, L - LD)) / LD
1600 CALL THREE4 (XI, T, ZP, XP, HH, VV, KODE)
HH = (XP-QX)*SM - (HH - QY)*SL)*SI
VV = (VV - GZ) * SD
HH = (HH - H(9)) * SH
VV = (VV - V(9)) * SV
1608 CALL PLOT (HH, VV, PEN)
1609 PEN = 5 - PEN
1610 CONTINUE
1611 PEN = DOWN
SUBROUTINE THREES (XI,YJ,M,N,P,KODE)
C SEE IF A POINT IS VISIBLE.
DIMENSION Z(151,151)
COMMON /THREE6/ ANGA, ANGB, HV, O, SH, SV
COMMON /THREE7/ SL, SM, SN, CX, CY, CZ, QX, CY, QZ, SD
COMMON /ARRAY/ Z
INTEGER CUM, CNT, P
REAL I, J, II, JJ
IF ( KODE .EQ. 1 ) GO TO 78
IR = XI
JC = YJ
ZB = Z ( IR + JC )
IF ( XI .EQ. IR ) GO TO 2
ZB = Z ( IR + JC ) + ( XI - IR ) * ( Z ( IR + 1 , JC ) - Z ( IR , JC ) )
GO TO 4
ZB = Z ( IR + JC ) + ( YJ - JC ) * ( Z ( IR, JC + 1 ) - Z ( IR, JC ) )
CONTINUE
XEND = COC
CX = 0.0
RETURN
END
SUBROUTINE THREES5 (XI,T,N,M,PO,KODE)
CALL THREE5 (XI,T,N,M,PO,KODE)
IF ( PC .EQ. 0 ) GO TO 1625
TOC = T
Pl = PC
T = T + 01
CONTINUE
TO = TO - 01
GO TO 1620
CONTINUE
C REPRODUCIBILITY OF THE ORIGINAL PAGE IS POOR.
YMULT = C.C
ZMULT = C.0

IF ( XI .EQ. CX ) GO TO 10
YMULT = ( YJ - CY ) / ( XI - CX )
ZMULT = ( ZB - CZ ) / ( XI - CX )

CX = 1.0
XEND = N + 1
IF ( XI .LT. CX) GO TO 10
CX = -1.0
XEND = 0.0

10 CONTINUE
10 YEND = C.C
10 XMULT = 0.0
10 YMULT = 0.0

IF ( YJ .EQ. CY ) GO TO 20
XMULT = ( XI - CX ) / ( YJ - CY )

IF ( ZMULT .EQ. 0.0 ) ZMULT=(ZB - CZ ) / ( YJ - CY )
DY = 1.0
YEND = N + 1
IF ( YJ .LT. CY ) GO TO 20
DY = -1.0
YEND = C.C

20 CONTINUE
20 CX = 1.0
20 YEND = C.C

30 CONTINUE
30 II = AINT(XB)
30 JJ = AINT(YB)
30 XSTEP = DX
30 YSTEP = DY

IF ( XB .EQ. II ) GO TO 40
IF ( CX .LT. 0.0 ) XSTEP = 0.0
GO TO 45

40 IF ( YB .EQ. JJ ) GO TO 45
40 IF ( DY .LT. 0.0 ) YSTEP = 0.0

50 CONTINUE
50 I = II + XSTEP
50 J = JJ + YSTEP

IF ( I .EQ. XEND ) GO TO 80
IF ( J .EQ. YEND ) GO TO 80

55 IF ( XB .LT. I ) GO TO 50
55 IF ( YB .LT. J ) GO TO 70
55 IF ( YB .NE. J ) GO TO 70

5C XB = I
5C GC TO 65

55 IF ( XB .LT. I ) GO TO 50
55 IF ( YMULT .LT. CY ) GO TO 50
60 YB = J

65 CONTINUE
65 ZB = CZ + ZMULT * ( XB - CX )
65 IR = I
65 JC = J

60 IF ( YMULT .NE. J ) GO TO 70
60 ICX = I - CX
FUNCTION PAT(U,V,ITYPE)

C
C THIS SUBPROGRAM GIVES THE BASIC CORRECTION PATTERN F(U,V).
C
C ITYPE = 1 -- UNIFORM LINE SOURCE LOCATED AT S=0.
C ITYPE = 2 -- UNIFORM LINEAR ARRAY LOCATED AT S=0.
C ITYPE = 3 -- TRIANGULAR LINE SOURCE LOCATED AT S=0.
C ITYPE = 4 -- UNIFORM RECTANGULAR APERTURE.
C ITYPE = 5 -- UNIFORM RECTANGULAR ARRAY.
C ITYPE = 6 -- UNIFORM CIRCULAR APERTURE.
C ITYPE = 7 -- GENERAL ARRAY.
C
C ITYPE > 7 -- SPECIAL SOURCE (FUNCTION SPECPT(U,V,ITYPE) WILL BE CALLED.

VERSION 1 LEVEL 1


THIS WORK SUPPORTED BY NASA GRANT NGR 47-004-103

FOR FURTHER INFORMATION CONTACT:
W.L. STUTZMAN DEPT. OF ELEC. ENGR. 951-6624.
E.L. COFFEY DEPT. OF ELEC. ENGR. 951-5494

COMMON /PAT1/ P1,P2,P3,P4,P5,P6,P7,S(S100),IT(100),RR(100)
COMMON /PAT2/ 11,12,13,14,15

C

1134 ZS = Z( IR, JC ) - DX * ( XB - 1 ) * (Z(IDX,JC) - Z(IR,JC))
1135 GC TO 75
1136 70 JDY=J-DY
1137 ZS = Z( IR, JC ) - DY * ( YB-J ) * (Z( IR,JDY ) - Z( IR,JC )
1138 75 CONTINUE
1139 SGN = 1
1140 IF ( ZB .LT. ZS ) SGN = -1
1141 CUM = CUM + SGN
1142 CNT = CNT + 1
1143 IF ( IABS ( CUM ) .EQ. CNT ) GO TO 30
1144 GC TO 90
1145 78 P=1
1146 GO TO 95
1147 80 CONTINUE
1148 P = 1
1149 IF ( CUM ) 84, 86, 90
1150 84 P = -1
1151 GO TO 90
1152 86 CONTINUE
1153 IF ( ZB .LE. CZ ) GO TO 90.
1154 P = -1
1155 90 CONTINUE
1156 95 RETURN
1157 END

FUNCTION PAT(U,V,ITYPE)
1162 IF(ITYPE.GT.7) GO TO 990
1163 GO TO (1CC,2CC,300,400,500,600,700),ITYPE
1164 ITYPE .LT. 1
1165 WRITE(6,10) ITYPE
1166 FORMAT(1HC,5X,***ERROR*** ITYPE HAS THE VALUE ',',111,'::',2X,
1167 EXECUTION TERMINATED')
1168 STOP
1169 ITYPE = 1 -- UNIFORM LINE SOURCE.
1170 FLEN=PI
1171 CONTINUE
1172 PAT=1.0
1173 IF(V.NE.C.) PAT=SIN(PI*PI*V)/(PI*PI*V)
1174 GO TO 999
1175 ITYPE = 2 -- UNIFORM LINEAR ARRAY
1176 CONTINUE
1177 FLEN=PI
1178 PAT=1.0
1179 NELMT=II
1180 IF(V.NE.C.) PAT=SIN(PI*PI*V)/(II*SIN(PI*PI*V/I))
1181 GO TO 999
1182 ITYPE = 3 -- TRIANGULAR LINE SOURCE.
1183 CONTINUE
1184 FLEN=PI/2.
1185 PAT=1.0
1186 IF(V.NE.C.) PAT=(SIN(FLEN*PI*V)/(FLEN*PI*V))**2
1187 GO TO 999
1188 ITYPE = 4 -- UNIFORM RECTANGULAR APERTURE
1189 CONTINUE
1190 FLS=PI
1191 FLT=P2
1192 ARG1=PI*PI*U
1193 ARG2=PI*PI*V
1194 IF(ARG1) 4C1,4C1,4C1,4C1
1195 IF(ARG2) 4C3,4C4,4C4,4C4
1196 4C1 PAT=SIN(ARG1)/ARG1*SIN(ARG2)/ARG2
1197 GO TO 999
1198 4C2 PAT=SIN(ARG2)/ARG2
1199 C TO 999
1200 4C3 PAT=1.0
1201 GO TO 999
C
C ITYPE = 5 -- UNIFORM RECTANGULAR ARRAY
C
1193  500 CONTINUE
C
C NELS=II
C NELT=12
1194  ARG1=PI*P1*U
1195  ARG2=PI*P2*V
1196  IF(ARG1) 501,502,501
1197  501 IF(ARG2) 503,504,503
1198  503 PAT=SIN(ARG1)/(II*SIN(ARG1/II))*SIN(ARG2)/(12*SIN(ARG2/12))
1199  GO TO 999
1200  504 PAT=SIN(ARG1)/(II*SIN(ARG1/II))
1201  GO TO 999
1202  502 IF(ARG2) 505,506,505
1203  505 PAT=SIN(ARG2)/(II2*SIN(ARG2/12))
1204  GO TO 999
1205  506 PAT=1.0
1206  GO TO 999

C
C ITYPE = 6 -- UNIFORM CIRCULAR APERTURE.
C
1207  600 C=SQRT(U*U+V*V)
C
A=P1
1208  IF(C.EQ.0.) GO TO 601
1209  X=2.*PI*P1*C
1210  CALL BESJ(X,1,BJ,0.0001,IER)
1211  PAT=BJ/X*2.0
1212  GO TO 999
1213  601 PAT=I.C
1214  GO TO 999.

C
C ITYPE = 7 -- GENERAL ARRAY
C
1215  700 [IMAG=(0.0,1.0]
1216  NELMT=II*12
1217  TEMP=(0.0,0.0)
1218  DO 701 J=1,NELMT
1219  701 TEMP=TEMP*1.0*CEXP(IMAG*2.*PI*(U*SS(J)+V*TT(J))]
1220  CONTINUE
1221  PAT=REAL(TEMP)/NELMT
1222  GO TO 999
1223  990 PAT=SPECPT(U,V,ITYPE)
1224  999 RETURN
1225  ENC

1226  COMPLEX FUNCTION SOURCE(M,N,U,V,ITYPE)
C
C THIS SUBPROGRAM CALCULATES THE CURRENT AT POINT (M,N) DUE TO
C THE PATTERN AT POINT (U,V).
ITYPE = 1 -- UNIFORM LINE SOURCE LOCATED AT S=0.

ITYPE = 2 -- UNIFORM LINEAR ARRAY LOCATED AT S=0.

ITYPE = 3 -- TRIANGULAR LINE SOURCE LOCATED AT S=0.

ITYPE = 4 -- UNIFORM RECTANGULAR APERTURE.

ITYPE = 5 -- UNIFORM RECTANGULAR ARRAY.

ITYPE = 6 -- UNIFORM CIRCULAR APERTURE.

ITYPE > 7 -- SPECIAL SOURCE (FUNCTION SPSOR(M,N,U,V,ITYP))

VERSION 1 LEVEL C

DATE OF LAST REVISION: 73/166 JUNE 15, 1973

THIS WORK SUPPORTED BY NASA GRANT NGR 47-004-103.

FOR FURTHER INFORMATION CONTACT:
W. L. STUTZMAN DEPT. OF ELECTR. ENGR. 951-6624.
E. L. COFFEY DEPT. OF ELECTR. ENGR. 951-5494.

1227 COMPLEX TEMP, CEXP, IMAG, SPSOR
1228 COMMON /PAT1/ P1, P2, P3, P4, P5, P6, P1, SS(100), TT(100), RR(100)
1229 COMMON /PAT2/ 11, 12, 13, 14, 15
1230 IMAG=(0.0, 1.0)
1231 CALL LOCSOR(M, N, S, T)
1232 IF(ITYPE.GT.7) GO TO 990
1233 GO TO (100, 200, 300, 400, 500, 600, 700), ITYPE

1234 WRITE(6,10) ITYPE
1235 FORMAT(1HC0.5X, '***ERROR*** ITYPE HAS THE VALUE', ITYPE, 'X', *EXECUTION TERMINATED*)
1236 STOP

ITYPE = 1 -- UNIFORM LINE SOURCE

1237 CONTINUE
1238 SOURCE=CEXP(-IMAG*PI*2.*T*V)/P1
1239 GO TO 999

ITYPE = 2 -- UNIFORM LINEAR ARRAY

1240 CONTINUE
1241 SOURCE=CEXP(-IMAG*2.*PI*V*T)/P1
1242 GO TO 999

ITYPE = 3 -- TRIANGULAR LINE SOURCE

REPRODUCIBILITY OF THE ORIGINAL PAGE IS POOR
A-105

1243  CONTINUE
1244     CON=ABS(2.*T/P1)
1245     SOURCE=2.*P1*CEXP(-IMAG*2.*PI*T*V)/(1.-CON)
1246     IF(CON.GT.1) SOURCE=(0.0,0.0)
1247     GO TO 999
1248
1249     CON=ABS(2.*T/PI)
1250     SOURCE=CEXP(-IMAG*2.*PI*S*U+V*T)/(P1*P2)
1251     GO TO 999
1252
1253     CON=ABS(2.*T/PI)
1254     SOURCE=CEXP(-IMAG*2.*PI*S*U+V*T)/(P1*P2)
1255     GO TO 999
1256
1257
1258
1259
1260
1261
1262
1263

SUBROUTINE LOCSOR(M,N,S,T)
1264     INTEGER PX,PY
1265     REAL INITLS,INITLT
1266     COMMON /PAT1/ P1,P2,P3,P4,P5,P6,P1,S5(100),T5(100),R5(100)
1267     COMMON /PAT2/ 11,12,13,14,15
1268     COMMON /LOC/ ITYPE
1269
1270     IF(ITYPE.GT.7) GO TO 999
1271     GO TO (100,200,300,400,500,600,700), ITYPE
1272 WRITE(6,10) ITYPE
1273 10 FORMAT(1HC,5X,***ERROR***
*EXECUTION TERMINATED*)
1274    STOP
1275  100 CONTINUE
1276     INITLT=P1
1277     CELTAT=P3
1278     S=0
1279     T=P2+(N-1)*P3
1280     GO TO 999
1281  200 CONTINUE
1282     PY=11
1283     DISY=P2
1284     S=0
1285     T=(N-11/2-1)*P2
1286     IF(11/2*2.EQ.11) T=T+0.5*P2
1287     GO TO 999
1288  300 GO TO 100
1289  400 CONTINUE
1290     INITLS=P3
1291     INITLT=P4
1292     CELTAS=P5
1293     CELTAT=P6
1294     S=P3+(M-1)*P5
1295     T=P4+(N-1)*P6
1296     GO TO 999
1297  500 CONTINUE
1298     PX=I1
1299     PY=I2
1300     DISX=P3
1301     DISY=P4
1302     S=(N-11/2-1)*P3
1303     T=(N-12/2-1)*P4
1304     IF(11/2*2.EQ.11) S=S+0.5*P3
1305     IF(12/2*2.EQ.12) T=T+0.5*P4
1306     GO TO 999
1307  600 GO TO 400
1308  700 CONTINUE
1309     NELPT=(M-1)*I2*N
1310     S=SS(NELMT)
1311     T=TT(NELPT)
1312     GO TO 999
C
1361 990 CALL SPLOC(M,N,S,T)
1362 999 RETURN
1363 ENC

1364 COMPLEX FUNCTION SPSOR(M,N,U,V,ITYPE)
   C DUMMY SUBPROGRAM
1365   SPSOR=(0.0,0.0)
1366   RETURN
1367 ENC

1368 FUNCTION SPECPT(U,V,ITYPE)
   C DUMMY SUBPROGRAM
1369   SPECPT=0.
1370   RETURN
1371 ENC

1372 SUBROUTINE SPLOC(M,N,S,T)
   C DUMMY SUBROUTINE
1373   RETURN
1374 ENC
8. Appendix: Example of Input/Output Used With Computer

Antenna Synthesis

In this chapter one example will be used to illustrate the input and output of ANTSYN and ANTDATA. The pattern to be synthesized is a rectangular shaped beam of extent

\[ \begin{array}{c|c|c|c|}
(u,v) & F_d(u,v) & F_u(u,v) & F_L(u,v) \\
\hline
-0.2 \leq u \leq 0.2 & 0. \text{ dB} & 0.5 \text{ dB} & -0.5 \text{ dB} \\
-0.05 \leq v \leq 0.05 & & & \\
\end{array} \]

and a maximum sidelobe level of \(-25 \text{ dB}\). The source is a rectangular aperture \((\text{ITYPE}=4)\), \(10\lambda\) by \(20\lambda\).

8.1 Input to ANTSYN

Since a rectangular aperture is included in our types of patterns \((\text{ITYPE}=4)\), it is only necessary to include "dummy" subprograms for SINPUT, SPECPT, SPSOR, and SPLOC:

```fortran
SUBROUTINE SINPUT
RETURN
END

SUBROUTINE SPLOC
RETURN
END

FUNCTION SPECPT(U,V,ITYPE)
SPECPT=0.
RETURN
END

COMPLEX FUNCTION SPSOR(M,N,S,T,ITYPE)
SPSOR=(0.,0.)
RETURN
END
```

For this particular desired pattern, subroutine DESPAT is written as follows:
SUBROUTINE DESPAT(FDES,FU,FL,MMAX,NMAX,STARTU,STARTV,DELTU,$DELTAV)
DIMENSION FDES(51,51),FU(51,51),FL(51,51)
C
C READ MAINBEAM LIMITS ULIM AND VLIM
C
READ(5,1) ULIM,VLIM
1 FORMAT(8F10.0)
C
C READ TRANSITION REGION LIMITS UTRAN AND VTRAN
C
READ(5,1) UTRAN,VTRAN
C
DO 10 M=1,MMAX
U=STARTU+(M-1)*DELTU
DO 10 N=1,NMAX
V=STARTV+(N-1)*DELTAV
IF(U.LE.ULIM .AND. V.LE.VLIM) GO TO 20
IF(U.GT.UTRAN .OR. V.GT.VTRAN) GO TO 30
C TRANSITION REGION
FDES(M,N)=99.0
FU(M,N)=99.0
FL(M,N)=99.0
GO TO 10
20 CONTINUE
C MAIN BEAM REGION
FDES(M,N)=1.0
FU(M,N)=1.06
FL(M,N)=0.943
GO TO 10
30 CONTINUE
C SIDELOBE REGION
FDES(M,N)=0.
FU(M,N)=0.057
FL(M,N)=99.0
10 CONTINUE
C
RETURN
END

The value "99.0" in an array signals that a comparison is not to be made at that point (e.g., in the sidelobe region, FL( , ) = 99.0 since a lower bound is not specified).
The data cards for this example are:

Card Column

<table>
<thead>
<tr>
<th></th>
<th>11</th>
<th>21</th>
<th>31</th>
<th>41</th>
<th>51</th>
<th>61</th>
</tr>
</thead>
</table>
| 1 | &PARAM
| 2 | IDISK=1,ISYMM=3,DELTAU=0.02,DELTAV=0.01,MMAX=26,NMAX=51,
| 3 | &END
| 4 | &IPRINT
| 5 | FDESCN=1,FDBPR=1,FDBCN=1,FCURPR=1,
| 6 | &END
| 7 | &PATIN
| 8 | ITYPE=4,LX=10.,LY=20.,
| 9 | INITLS=-5.0,DELTAS=0.2,FINALS=5.0,
|10 | INITLT=-10.0,DELTAT=0.4,FINALT=10.0,
|11 | &END
|12 | 0.2 0.05
|13 | 0.34 0.12
|14 | 0.0 0.0 1.0
|15 | 0.0 0.05 1.0
|16 | 0.0 -0.05 1.0
|17 | 0.1 0.0 1.0
|18 | -0.1 0.0 1.0
|19 | 0.1 0.05 1.0
|20 | -0.1 -0.05 1.0
|21 | 0.1 0.0 1.0
|22 | 0.2 0.05 1.0
|23 | 0.2 -0.05 1.0
|24 | 0.2 0.0 1.0
|25 | -0.2 0.0 1.0
|26 | 0.0 0.0 1.0
|27 | 0.0 0.05 1.0
|28 | 0.0 -0.05 1.0
|29 | 0.0 0.0 1.0

Notice that it is not necessary to code all the namelist variables. For example, STARTU is not coded because its default value is 0. (which is what we want). In order to better understand why certain parameters were coded, refer to section 6.3 to steps 2 through 10 as they are discussed below.

Step 2. Cards 1 to 3: Pattern Parameters

i. Put output data onto unit 22 if the synthesis is successful (IDISK=1)

ii. Use quadrilateral symmetry (ISYMM=3)

iii. Have a maximum of 100 iterations (ITRMAX=100)

iv. STARTU=STARTV=0., DELTAU=0.02,DELTAV=0.01
v. Make the comparisons at 26 points in the u direction and 51 points in the v direction (MMAX=26, NMAX=51)
vii. Assure that F(1,1)=0 dB. at all times (MCENT=1, NCENT=1)

Note that F(MMAX, NMAX) corresponds to (u,v)=(0.5,0.5): only part of the (u,v) plane is considered.

Step 3. Cards 4 to 6: Output Switches
i. Profiles of the final pattern and final current (FDBPR=FCURPR=1)
ii. Contour maps of the desired pattern (FDESCN=1) and final current (FCURCN=1) are to be made

Step 4. Cards 7 to 11: Source Specifications
i. Rectangular aperture (ITYPE=4)
ii. Dimensions of 10\lambda by 20\lambda (LX=10., LY=20.)
iii. The value of current will be calculated at 51 x 51 points from 
    s = -5.0 to 5.0 by 0.2, and t = -10.0 to 10.0 by 0.4.
    (INITLS=-5., FINALS=5., DELTAS=0.2; INITLT=-10., FINALT=10., DELTAT=0.4)

Step 5. Cards 12 to 13: The Desired Pattern
For a more complete explanation, see the listing of subroutine DESPAT earlier in this section.

Step 6. Cards 14, 15 to 29: Initial Pattern
These are the number of (NORG) and the values of (UORG, VORG, CORG) the original correction coefficients.

Steps 7,8,9. See subroutines SINPUT, SPLOC, SPECPT, SPSOR.

8.2 Output from ANTSYN

This section is devoted to the actual output from the computer program ANTSYN with data as specified in Section 8.1. Due to page size limitations, some of the output has been edited. The omissions are indicated by an ellipsis(...).
ANTENNA SYNTHESIS PROGRAM  VERSION 3  LEVEL 1  VPI EE DEPT.

DATE = 09-25-73  TIME= 5.30.40  PATTERN 77

PROGRAM PARAMETERS

IDISK = 1  MMAX = 26
ISYMM = 3  NMAX = 51
ITRMAX = 200  MCENT = 1

STARTU = 0.0  DELTAU = 0.020
STARTV = 0.0  DELTAV = 0.010

DE.LTAU = 0.020  NCENT = 1
DELTAV = 0.010

ITYPE=4  -- UNIFORM RECTANGULAR APERTURE

DIMENSIONS = LX,LY = 10.0000, 20.0000

INITLS,DELTAS,FINALS: -5.0000  0.2000  5.0000
INITLT,DELTAT,FINALT: -10.0000  0.4000  10.0000

MCUR,NCUR: 51  51

REPRODUCIBILITY OF THE ORIGINAL PAGE IS POOR
CONTOUR PLOT OF THE DESIRED PATTERN.

<table>
<thead>
<tr>
<th>-0.5000</th>
<th>-0.4000</th>
<th>-0.3000</th>
<th>-0.2000</th>
<th>-0.1000</th>
<th>-0.0000</th>
<th>0.1000</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.5000</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>0.4900</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>0.4800</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>0.4700</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>0.4600</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>0.4500</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>0.4400</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>0.4300</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>0.4200</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>0.4100</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>0.4000</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>0.3900</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>0.3800</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>0.3700</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>0.3600</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>0.3500</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>0.3400</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>0.3300</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>0.3200</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>0.3100</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>0.3000</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>0.2900</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>0.2800</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>0.2700</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>0.2600</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>0.2500</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>0.2400</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>0.2300</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>0.2200</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>0.2100</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>0.2000</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>0.1900</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>0.1800</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>0.1700</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>0.1600</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>0.1500</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>0.1400</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>0.1300</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>0.1200</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>0.1100</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>0.1000</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>0.0900</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>0.0800</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>0.0700</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>0.0600</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>0.0500</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>0.0400</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>0.0300</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>0.0200</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>0.0100</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>0.0000</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>-0.0100</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>-0.0200</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>-0.0300</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>-0.0400</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>-0.0500</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
</tbody>
</table>
CONTOUR LEVEL KEY

0: -0.6000000E 00 TO -0.3999900E 00
1: -0.4000000E 00 TO -0.1999900E 00
2: -0.2000000E 00 TO 0.1000000E-04
3: 0.0 TO 0.2000099E 00
4: 0.2000000E 00 TO 0.4000099E 00
5: 0.4000000E 00 TO 0.6000099E 00
6: 0.5999998E 00 TO 0.8000099E 00
7: 0.7999997E 00 TO 0.99000010E 01
8: 0.9999995E 00 TO 0.1200006E 01
9: 0.1199999E 01 TO 0.1400005E 01
-= -0.9999999E 30 TO -0.6000000E 00
+*: 0.1400008E 01 TO 0.9999999E 30

-- INITIAL COEFFICIENTS --

<table>
<thead>
<tr>
<th>J</th>
<th>UORG(J)</th>
<th>VORG(J)</th>
<th>CORG(J)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>1.0000</td>
</tr>
<tr>
<td>2</td>
<td>0.0</td>
<td>0.0500</td>
<td>1.0000</td>
</tr>
<tr>
<td>3</td>
<td>0.0</td>
<td>-0.0500</td>
<td>1.0000</td>
</tr>
<tr>
<td>4</td>
<td>0.1000</td>
<td>0.0</td>
<td>1.0000</td>
</tr>
<tr>
<td>5</td>
<td>-0.1000</td>
<td>0.0</td>
<td>1.0000</td>
</tr>
<tr>
<td>6</td>
<td>0.1000</td>
<td>0.0500</td>
<td>1.0000</td>
</tr>
<tr>
<td>7</td>
<td>0.1000</td>
<td>-0.0500</td>
<td>1.0000</td>
</tr>
<tr>
<td>8</td>
<td>-0.1000</td>
<td>0.0500</td>
<td>1.0000</td>
</tr>
<tr>
<td>9</td>
<td>-0.1000</td>
<td>-0.0500</td>
<td>1.0000</td>
</tr>
<tr>
<td>10</td>
<td>0.2000</td>
<td>0.0500</td>
<td>1.0000</td>
</tr>
<tr>
<td>11</td>
<td>0.2000</td>
<td>-0.0500</td>
<td>1.0000</td>
</tr>
<tr>
<td>12</td>
<td>-0.2000</td>
<td>0.0500</td>
<td>1.0000</td>
</tr>
<tr>
<td>13</td>
<td>-0.2000</td>
<td>-0.0500</td>
<td>1.0000</td>
</tr>
<tr>
<td>14</td>
<td>0.2000</td>
<td>0.0</td>
<td>1.0000</td>
</tr>
<tr>
<td>15</td>
<td>-0.2000</td>
<td>0.0</td>
<td>1.0000</td>
</tr>
<tr>
<td><strong>SEARCH</strong></td>
<td>9</td>
<td>4</td>
<td>-0.1842</td>
</tr>
<tr>
<td><strong>SEARCH</strong></td>
<td>9</td>
<td>14</td>
<td>0.1524</td>
</tr>
<tr>
<td><strong>SEARCH</strong></td>
<td>1</td>
<td>14</td>
<td>0.2088</td>
</tr>
<tr>
<td><strong>SEARCH</strong></td>
<td>11</td>
<td>6</td>
<td>0.1875</td>
</tr>
<tr>
<td><strong>SEARCH</strong></td>
<td>5</td>
<td>6</td>
<td>0.1962</td>
</tr>
<tr>
<td><strong>SEARCH</strong></td>
<td>9</td>
<td>18</td>
<td>-0.1572</td>
</tr>
<tr>
<td><strong>SEARCH</strong></td>
<td>1</td>
<td>18</td>
<td>-0.2146</td>
</tr>
<tr>
<td><strong>SEARCH</strong></td>
<td>1</td>
<td>22</td>
<td>0.1065</td>
</tr>
<tr>
<td><strong>SEARCH</strong></td>
<td>19</td>
<td>6</td>
<td>0.1304</td>
</tr>
<tr>
<td><strong>SEARCH</strong></td>
<td>8</td>
<td>14</td>
<td>0.1241</td>
</tr>
<tr>
<td><strong>SEARCH</strong></td>
<td>8</td>
<td>22</td>
<td>0.1394</td>
</tr>
<tr>
<td><strong>SEARCH</strong></td>
<td>1</td>
<td>14</td>
<td>0.1209</td>
</tr>
<tr>
<td><strong>SEARCH</strong></td>
<td>11</td>
<td>6</td>
<td>0.1322</td>
</tr>
<tr>
<td><strong>SEARCH</strong></td>
<td>4</td>
<td>6</td>
<td>0.1436</td>
</tr>
<tr>
<td><strong>SEARCH</strong></td>
<td>23</td>
<td>6</td>
<td>-0.1201</td>
</tr>
<tr>
<td><strong>SEARCH</strong></td>
<td>7</td>
<td>18</td>
<td>-0.1188</td>
</tr>
<tr>
<td><strong>SEARCH</strong></td>
<td>1</td>
<td>4</td>
<td>-0.0716</td>
</tr>
<tr>
<td><strong>SEARCH</strong></td>
<td>9</td>
<td>5</td>
<td>-0.1592</td>
</tr>
<tr>
<td><strong>SEARCH</strong></td>
<td>1</td>
<td>5</td>
<td>-0.1155</td>
</tr>
<tr>
<td><strong>SEARCH</strong></td>
<td>9</td>
<td>5</td>
<td>-0.0856</td>
</tr>
<tr>
<td><strong>SEARCH</strong></td>
<td>1</td>
<td>5</td>
<td>-0.0900</td>
</tr>
<tr>
<td><strong>SEARCH</strong></td>
<td>1</td>
<td>4</td>
<td>-0.0643</td>
</tr>
<tr>
<td><strong>SEARCH</strong></td>
<td>9</td>
<td>5</td>
<td>-0.1467</td>
</tr>
<tr>
<td><strong>SEARCH</strong></td>
<td>1</td>
<td>5</td>
<td>-0.1023</td>
</tr>
<tr>
<td><strong>SEARCH</strong></td>
<td>9</td>
<td>5</td>
<td>-0.0757</td>
</tr>
<tr>
<td><strong>SEARCH</strong></td>
<td>7</td>
<td>16</td>
<td>0.1005</td>
</tr>
<tr>
<td><strong>SEARCH</strong></td>
<td>11</td>
<td>6</td>
<td>0.0889</td>
</tr>
<tr>
<td><strong>SEARCH</strong></td>
<td>14</td>
<td>14</td>
<td>0.1093</td>
</tr>
<tr>
<td><strong>SEARCH</strong></td>
<td>1</td>
<td>5</td>
<td>-0.0670</td>
</tr>
<tr>
<td><strong>SEARCH</strong></td>
<td>9</td>
<td>5</td>
<td>-0.0946</td>
</tr>
<tr>
<td><strong>SEARCH</strong></td>
<td>11</td>
<td>1</td>
<td>0.1536</td>
</tr>
<tr>
<td><strong>SEARCH</strong></td>
<td>19</td>
<td>1</td>
<td>0.1523</td>
</tr>
<tr>
<td><strong>SEARCH</strong></td>
<td>1</td>
<td>5</td>
<td>-0.0784</td>
</tr>
<tr>
<td><strong>SEARCH</strong></td>
<td>23</td>
<td>1</td>
<td>-0.1235</td>
</tr>
<tr>
<td><strong>SEARCH</strong></td>
<td>9</td>
<td>4</td>
<td>-0.1375</td>
</tr>
<tr>
<td><strong>SEARCH</strong></td>
<td>11</td>
<td>1</td>
<td>0.1179</td>
</tr>
<tr>
<td><strong>SEARCH</strong></td>
<td>6</td>
<td>1</td>
<td>0.1176</td>
</tr>
<tr>
<td><strong>SEARCH</strong></td>
<td>6</td>
<td>6</td>
<td>0.0956</td>
</tr>
<tr>
<td><strong>SEARCH</strong></td>
<td>9</td>
<td>4</td>
<td>-0.0875</td>
</tr>
<tr>
<td><strong>SEARCH</strong></td>
<td>11</td>
<td>6</td>
<td>0.1438</td>
</tr>
<tr>
<td><strong>SEARCH</strong></td>
<td>11</td>
<td>1</td>
<td>0.0898</td>
</tr>
<tr>
<td><strong>SEARCH</strong></td>
<td>5</td>
<td>1</td>
<td>0.0873</td>
</tr>
<tr>
<td><strong>SEARCH</strong></td>
<td>5</td>
<td>6</td>
<td>0.1050</td>
</tr>
<tr>
<td><strong>SEARCH</strong></td>
<td>9</td>
<td>4</td>
<td>-0.0652</td>
</tr>
<tr>
<td><strong>SEARCH</strong></td>
<td>6</td>
<td>1</td>
<td>0.1144</td>
</tr>
<tr>
<td><strong>SEARCH</strong></td>
<td>11</td>
<td>1</td>
<td>0.1130</td>
</tr>
<tr>
<td><strong>SEARCH</strong></td>
<td>11</td>
<td>6</td>
<td>0.0960</td>
</tr>
<tr>
<td><strong>SEARCH</strong></td>
<td>9</td>
<td>4</td>
<td>-0.0996</td>
</tr>
<tr>
<td><strong>SEARCH</strong></td>
<td>6</td>
<td>6</td>
<td>0.1263</td>
</tr>
<tr>
<td><strong>SEARCH</strong></td>
<td>11</td>
<td>1</td>
<td>0.1018</td>
</tr>
<tr>
<td><strong>SEARCH</strong></td>
<td>6</td>
<td>1</td>
<td>0.0968</td>
</tr>
<tr>
<td><strong>SEARCH</strong></td>
<td>9</td>
<td>4</td>
<td>-0.0924</td>
</tr>
<tr>
<td><strong>SEARCH</strong></td>
<td>11</td>
<td>6</td>
<td>0.1390</td>
</tr>
<tr>
<td><strong>SEARCH</strong></td>
<td>5</td>
<td>6</td>
<td>0.0783</td>
</tr>
<tr>
<td><strong>SEARCH</strong></td>
<td>11</td>
<td>1</td>
<td>0.0947</td>
</tr>
<tr>
<td><strong>SEARCH</strong></td>
<td>9</td>
<td>5</td>
<td>-0.0938</td>
</tr>
<tr>
<td><strong>SEARCH</strong></td>
<td>11</td>
<td>6</td>
<td>0.0748</td>
</tr>
<tr>
<td><strong>SEARCH</strong></td>
<td>1</td>
<td>4</td>
<td>-0.0624</td>
</tr>
<tr>
<td><strong>SEARCH</strong></td>
<td>10</td>
<td>5</td>
<td>-0.1378</td>
</tr>
<tr>
<td><strong>SEARCH</strong></td>
<td>1</td>
<td>5</td>
<td>-0.0909</td>
</tr>
<tr>
<td><strong>SEARCH</strong></td>
<td>9</td>
<td>5</td>
<td>-0.0783</td>
</tr>
<tr>
<td><strong>SEARCH</strong></td>
<td>1</td>
<td>5</td>
<td>-0.0715</td>
</tr>
</tbody>
</table>
## FINAL COEFFICIENTS

<table>
<thead>
<tr>
<th>J</th>
<th>US(J)</th>
<th>VS(J)</th>
<th>CORCOF(J)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.1600</td>
<td>0.0300</td>
<td>-0.6602</td>
</tr>
<tr>
<td>2</td>
<td>-0.1600</td>
<td>0.0300</td>
<td>-0.6602</td>
</tr>
<tr>
<td>3</td>
<td>0.1600</td>
<td>-0.0300</td>
<td>-0.6602</td>
</tr>
<tr>
<td>4</td>
<td>-0.1600</td>
<td>-0.0300</td>
<td>-0.6602</td>
</tr>
<tr>
<td>5</td>
<td>0.1600</td>
<td>0.1300</td>
<td>0.1689</td>
</tr>
<tr>
<td>6</td>
<td>-0.1600</td>
<td>0.1300</td>
<td>0.1689</td>
</tr>
<tr>
<td>7</td>
<td>0.1600</td>
<td>-0.1300</td>
<td>0.1689</td>
</tr>
<tr>
<td>8</td>
<td>-0.1600</td>
<td>-0.1300</td>
<td>0.1689</td>
</tr>
<tr>
<td>9</td>
<td>0.0</td>
<td>0.1300</td>
<td>0.3694</td>
</tr>
<tr>
<td>10</td>
<td>0.0</td>
<td>-0.1300</td>
<td>0.3694</td>
</tr>
<tr>
<td>80</td>
<td>-0.0800</td>
<td>0.0</td>
<td>0.0840</td>
</tr>
<tr>
<td>81</td>
<td>0.1800</td>
<td>0.0400</td>
<td>-0.1449</td>
</tr>
<tr>
<td>82</td>
<td>-0.1800</td>
<td>0.0400</td>
<td>-0.1449</td>
</tr>
<tr>
<td>83</td>
<td>0.1800</td>
<td>-0.0400</td>
<td>-0.1449</td>
</tr>
<tr>
<td>84</td>
<td>-0.1800</td>
<td>-0.0400</td>
<td>-0.1449</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>J</th>
<th>UORG(J)</th>
<th>VORG(J)</th>
<th>CORG(J)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>1.0352</td>
</tr>
<tr>
<td>2</td>
<td>0.0</td>
<td>0.0500</td>
<td>1.0352</td>
</tr>
<tr>
<td>3</td>
<td>0.0</td>
<td>-0.0500</td>
<td>1.0352</td>
</tr>
<tr>
<td>4</td>
<td>0.1000</td>
<td>0.0</td>
<td>1.0352</td>
</tr>
<tr>
<td>5</td>
<td>-0.1000</td>
<td>0.0</td>
<td>1.0352</td>
</tr>
<tr>
<td>6</td>
<td>0.1000</td>
<td>0.0500</td>
<td>1.0352</td>
</tr>
<tr>
<td>7</td>
<td>0.1000</td>
<td>-0.0500</td>
<td>1.0352</td>
</tr>
<tr>
<td>8</td>
<td>-0.1000</td>
<td>0.0500</td>
<td>1.0352</td>
</tr>
<tr>
<td>9</td>
<td>-0.1000</td>
<td>-0.0500</td>
<td>1.0352</td>
</tr>
<tr>
<td>10</td>
<td>0.2000</td>
<td>0.0500</td>
<td>1.0352</td>
</tr>
<tr>
<td>11</td>
<td>0.2000</td>
<td>-0.0500</td>
<td>1.0352</td>
</tr>
<tr>
<td>12</td>
<td>-0.2000</td>
<td>0.0500</td>
<td>1.0352</td>
</tr>
<tr>
<td>13</td>
<td>-0.2000</td>
<td>-0.0500</td>
<td>1.0352</td>
</tr>
<tr>
<td>14</td>
<td>0.2000</td>
<td>0.0</td>
<td>1.0352</td>
</tr>
<tr>
<td>15</td>
<td>-0.2000</td>
<td>0.0</td>
<td>1.0352</td>
</tr>
</tbody>
</table>

**NUMBER OF ITERATIONS = 62**

**FNORM = 1.03517**

**PATTERN NUMBER = NUMPAT = 77**
<table>
<thead>
<tr>
<th>U-AXIS PROFILE OF FINAL PATTERN -- V = 0.0</th>
<th>PATTERN NUMBER 77</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.017</td>
<td>0.153</td>
</tr>
<tr>
<td>1.000 *</td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>*</td>
</tr>
<tr>
<td>0.950 *</td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>*</td>
</tr>
<tr>
<td>0.900 *</td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>*</td>
</tr>
<tr>
<td>0.850 *</td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>*</td>
</tr>
<tr>
<td>0.800 *</td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>*</td>
</tr>
<tr>
<td>0.750 *</td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>*</td>
</tr>
<tr>
<td>0.250</td>
<td>0.666</td>
</tr>
<tr>
<td>-------</td>
<td>-------</td>
</tr>
<tr>
<td>0.200</td>
<td>0.707</td>
</tr>
<tr>
<td>0.150</td>
<td>0.747</td>
</tr>
<tr>
<td>0.100</td>
<td>0.784</td>
</tr>
<tr>
<td>0.050</td>
<td>0.819</td>
</tr>
<tr>
<td>0.000</td>
<td>0.852</td>
</tr>
<tr>
<td>-0.050</td>
<td>0.882</td>
</tr>
<tr>
<td>-0.100</td>
<td>0.909</td>
</tr>
<tr>
<td>-0.150</td>
<td>0.933</td>
</tr>
<tr>
<td>-0.200</td>
<td>0.954</td>
</tr>
<tr>
<td>-0.250</td>
<td>0.972</td>
</tr>
<tr>
<td>-0.300</td>
<td>0.986</td>
</tr>
<tr>
<td>-0.350</td>
<td>0.998</td>
</tr>
<tr>
<td>-0.400</td>
<td>1.007</td>
</tr>
<tr>
<td>-0.450</td>
<td>1.013</td>
</tr>
<tr>
<td>-0.500</td>
<td>1.016</td>
</tr>
<tr>
<td>-0.550</td>
<td>1.017</td>
</tr>
<tr>
<td>-0.600</td>
<td>1.016</td>
</tr>
<tr>
<td>-0.650</td>
<td>1.013</td>
</tr>
<tr>
<td>-0.700</td>
<td>1.009</td>
</tr>
<tr>
<td>-0.750</td>
<td>1.003</td>
</tr>
<tr>
<td>-0.800</td>
<td>0.996</td>
</tr>
<tr>
<td>-0.850</td>
<td>0.988</td>
</tr>
<tr>
<td>-0.900</td>
<td>0.980</td>
</tr>
<tr>
<td>-0.950</td>
<td>0.972</td>
</tr>
<tr>
<td>-1.000</td>
<td>0.964</td>
</tr>
<tr>
<td>-1.050</td>
<td>0.957</td>
</tr>
<tr>
<td>-1.100</td>
<td>0.950</td>
</tr>
<tr>
<td>-1.150</td>
<td>0.943</td>
</tr>
<tr>
<td>-1.200</td>
<td>0.938</td>
</tr>
<tr>
<td>-1.250</td>
<td>0.934</td>
</tr>
<tr>
<td>-1.300</td>
<td>0.931</td>
</tr>
<tr>
<td>-1.350</td>
<td>0.929</td>
</tr>
<tr>
<td>-1.400</td>
<td>0.926</td>
</tr>
<tr>
<td>-1.450</td>
<td>0.923</td>
</tr>
<tr>
<td>-1.500</td>
<td>0.920</td>
</tr>
<tr>
<td>-1.550</td>
<td>0.916</td>
</tr>
<tr>
<td>-1.600</td>
<td>0.912</td>
</tr>
<tr>
<td>-1.650</td>
<td>0.909</td>
</tr>
<tr>
<td>-1.700</td>
<td>0.905</td>
</tr>
<tr>
<td>-1.750</td>
<td>0.901</td>
</tr>
<tr>
<td>-1.800</td>
<td>0.897</td>
</tr>
<tr>
<td>-1.850</td>
<td>0.894</td>
</tr>
<tr>
<td>-1.900</td>
<td>0.890</td>
</tr>
<tr>
<td>-1.950</td>
<td>0.887</td>
</tr>
<tr>
<td>-2.000</td>
<td>0.883</td>
</tr>
</tbody>
</table>

Diagram showing a graph with various data points marked with asterisks.
### S Axis Profile of Final Current

<table>
<thead>
<tr>
<th>S</th>
<th>T</th>
<th>Real</th>
<th>Imaginary</th>
</tr>
</thead>
<tbody>
<tr>
<td>-5.0000</td>
<td>-0.0000</td>
<td>0.5700823E-02</td>
<td>0.0</td>
</tr>
<tr>
<td>-4.8000</td>
<td>-0.0000</td>
<td>0.7003162E-02</td>
<td>0.0</td>
</tr>
<tr>
<td>-4.6000</td>
<td>-0.0000</td>
<td>0.6889808E-02</td>
<td>0.2095046E-08</td>
</tr>
<tr>
<td>-4.4000</td>
<td>-0.0000</td>
<td>0.5990162E-02</td>
<td>0.3026799E-08</td>
</tr>
<tr>
<td>-4.2000</td>
<td>-0.0000</td>
<td>0.4723225E-02</td>
<td>0.0</td>
</tr>
<tr>
<td>-4.0000</td>
<td>-0.0000</td>
<td>0.3161762E-02</td>
<td>0.0</td>
</tr>
<tr>
<td>-3.8000</td>
<td>-0.0000</td>
<td>0.1145484E-02</td>
<td>0.0</td>
</tr>
<tr>
<td>-3.6000</td>
<td>-0.0000</td>
<td>-0.1570189E-02</td>
<td>0.0</td>
</tr>
<tr>
<td>-3.4000</td>
<td>-0.0000</td>
<td>-0.5123518E-02</td>
<td>0.0</td>
</tr>
<tr>
<td>-3.2000</td>
<td>-0.0000</td>
<td>-0.9384781E-02</td>
<td>0.0</td>
</tr>
<tr>
<td>-3.0000</td>
<td>-0.0000</td>
<td>-0.1387722E-01</td>
<td>0.0</td>
</tr>
<tr>
<td>-2.8000</td>
<td>-0.0000</td>
<td>-0.1780687E-01</td>
<td>0.0</td>
</tr>
<tr>
<td>-2.6000</td>
<td>-0.0000</td>
<td>-0.2019734E-01</td>
<td>0.0</td>
</tr>
<tr>
<td>-2.4000</td>
<td>-0.0000</td>
<td>-0.2009303E-01</td>
<td>0.0</td>
</tr>
<tr>
<td>-2.2000</td>
<td>-0.0000</td>
<td>-0.1677621E-01</td>
<td>0.0</td>
</tr>
<tr>
<td>-2.0000</td>
<td>-0.0000</td>
<td>-0.9939682E-02</td>
<td>0.2095476E-08</td>
</tr>
<tr>
<td>-1.8000</td>
<td>-0.0000</td>
<td>0.2297403E-03</td>
<td>0.0</td>
</tr>
<tr>
<td>-1.6000</td>
<td>-0.0000</td>
<td>0.1307478E-01</td>
<td>0.0</td>
</tr>
<tr>
<td>-1.4000</td>
<td>-0.0000</td>
<td>0.2758893E-01</td>
<td>0.0</td>
</tr>
<tr>
<td>-1.2000</td>
<td>-0.0000</td>
<td>0.4259761E-01</td>
<td>0.0</td>
</tr>
<tr>
<td>-1.0000</td>
<td>-0.0000</td>
<td>0.5695011E-01</td>
<td>0.0</td>
</tr>
<tr>
<td>-0.8000</td>
<td>-0.0000</td>
<td>0.6967366E-01</td>
<td>0.0</td>
</tr>
<tr>
<td>-0.6000</td>
<td>-0.0000</td>
<td>0.8006346E-01</td>
<td>0.0</td>
</tr>
<tr>
<td>-0.4000</td>
<td>-0.0000</td>
<td>0.8768129E-01</td>
<td>0.0</td>
</tr>
<tr>
<td>-0.2000</td>
<td>-0.0000</td>
<td>0.9230632E-01</td>
<td>0.0</td>
</tr>
<tr>
<td>0.0000</td>
<td>-0.0000</td>
<td>0.9385431E-01</td>
<td>0.7105427E-14</td>
</tr>
<tr>
<td>0.2000</td>
<td>-0.0000</td>
<td>0.9230632E-01</td>
<td>0.0</td>
</tr>
<tr>
<td>0.4000</td>
<td>-0.0000</td>
<td>0.8768129E-01</td>
<td>0.2095476E-08</td>
</tr>
<tr>
<td>0.6000</td>
<td>-0.0000</td>
<td>0.8006346E-01</td>
<td>0.2793969E-08</td>
</tr>
<tr>
<td>0.8000</td>
<td>-0.0000</td>
<td>0.6967366E-01</td>
<td>0.3725290E-08</td>
</tr>
<tr>
<td>1.0000</td>
<td>-0.0000</td>
<td>0.5695021E-01</td>
<td>0.3725290E-08</td>
</tr>
<tr>
<td>1.2000</td>
<td>-0.0000</td>
<td>0.4259773E-01</td>
<td>0.0</td>
</tr>
<tr>
<td>1.4000</td>
<td>-0.0000</td>
<td>0.2758904E-01</td>
<td>0.0</td>
</tr>
<tr>
<td>1.6000</td>
<td>-0.0000</td>
<td>0.1307483E-01</td>
<td>0.0</td>
</tr>
<tr>
<td>1.8000</td>
<td>-0.0000</td>
<td>0.2297729E-03</td>
<td>0.0</td>
</tr>
<tr>
<td>2.0000</td>
<td>-0.0000</td>
<td>0.9939581E-02</td>
<td>0.0</td>
</tr>
<tr>
<td>2.2000</td>
<td>-0.0000</td>
<td>0.1677619E-01</td>
<td>0.0</td>
</tr>
<tr>
<td>2.4000</td>
<td>-0.0000</td>
<td>0.2099503E-01</td>
<td>0.0</td>
</tr>
<tr>
<td>2.6000</td>
<td>-0.0000</td>
<td>0.2019734E-01</td>
<td>0.0</td>
</tr>
<tr>
<td>2.8000</td>
<td>-0.0000</td>
<td>0.1780687E-01</td>
<td>0.0</td>
</tr>
<tr>
<td>3.0000</td>
<td>-0.0000</td>
<td>0.1387723E-01</td>
<td>0.1396984E-08</td>
</tr>
<tr>
<td>3.2000</td>
<td>-0.0000</td>
<td>-0.9384889E-02</td>
<td>0.0</td>
</tr>
<tr>
<td>3.4000</td>
<td>-0.0000</td>
<td>-0.5123563E-02</td>
<td>0.0</td>
</tr>
<tr>
<td>3.6000</td>
<td>-0.0000</td>
<td>-0.1570206E-02</td>
<td>0.0</td>
</tr>
<tr>
<td>3.8000</td>
<td>-0.0000</td>
<td>-0.1145449E-02</td>
<td>0.0</td>
</tr>
<tr>
<td>4.0000</td>
<td>-0.0000</td>
<td>-0.3161710E-02</td>
<td>0.0</td>
</tr>
<tr>
<td>4.2000</td>
<td>-0.0000</td>
<td>-0.4723225E-02</td>
<td>0.0</td>
</tr>
<tr>
<td>4.4000</td>
<td>-0.0000</td>
<td>-0.5990129E-02</td>
<td>0.0</td>
</tr>
<tr>
<td>4.6000</td>
<td>-0.0000</td>
<td>-0.6880980E-02</td>
<td>0.2561137E-08</td>
</tr>
<tr>
<td>4.8000</td>
<td>-0.0000</td>
<td>-0.7003162E-02</td>
<td>0.0</td>
</tr>
<tr>
<td>5.0000</td>
<td>-0.0000</td>
<td>-0.5700838E-02</td>
<td>0.0</td>
</tr>
<tr>
<td>S</td>
<td>T</td>
<td>REAL</td>
<td>IMAGINARY</td>
</tr>
<tr>
<td>-----</td>
<td>-----</td>
<td>-----------------</td>
<td>------------------</td>
</tr>
<tr>
<td>-0.0000</td>
<td>-10.0000</td>
<td>-0.2455123E-01</td>
<td>0.4423782E-08</td>
</tr>
<tr>
<td>-0.0000</td>
<td>-9.6000</td>
<td>-0.1836829E-01</td>
<td>0.2793968E-08</td>
</tr>
<tr>
<td>-0.0000</td>
<td>-9.2000</td>
<td>-0.1313753E-01</td>
<td>0.3725290E-08</td>
</tr>
<tr>
<td>-0.0000</td>
<td>-8.8000</td>
<td>-0.9560350E-02</td>
<td>0.5122274E-08</td>
</tr>
<tr>
<td>-0.0000</td>
<td>-8.4000</td>
<td>-0.7850584E-02</td>
<td>0.1396984E-08</td>
</tr>
<tr>
<td>-0.0000</td>
<td>-8.0000</td>
<td>-0.7672068E-02</td>
<td>0.1396984E-08</td>
</tr>
<tr>
<td>-0.0000</td>
<td>-7.6000</td>
<td>-0.3307122E-02</td>
<td>0.1396984E-08</td>
</tr>
<tr>
<td>-0.0000</td>
<td>-7.2000</td>
<td>-0.8702484E-02</td>
<td>0.3492460E-08</td>
</tr>
<tr>
<td>-0.0000</td>
<td>-6.8000</td>
<td>-0.8532621E-02</td>
<td>0.3026798E-08</td>
</tr>
<tr>
<td>-0.0000</td>
<td>-6.4000</td>
<td>-0.6708888E-02</td>
<td>0.3026798E-08</td>
</tr>
<tr>
<td>-0.0000</td>
<td>-6.0000</td>
<td>-0.3115309E-02</td>
<td>0.1164153E-08</td>
</tr>
<tr>
<td>-0.0000</td>
<td>-5.6000</td>
<td>0.2147641E-02</td>
<td>0.1164153E-08</td>
</tr>
<tr>
<td>-0.0000</td>
<td>-5.2000</td>
<td>0.8754689E-02</td>
<td>0.1164153E-08</td>
</tr>
<tr>
<td>-0.0000</td>
<td>-4.8000</td>
<td>0.1624599E-01</td>
<td>0.9313226E-09</td>
</tr>
<tr>
<td>-0.0000</td>
<td>-4.4000</td>
<td>0.2417203E-01</td>
<td>0.1862645E-08</td>
</tr>
<tr>
<td>-0.0000</td>
<td>-4.0000</td>
<td>0.3219855E-01</td>
<td>0.2326306E-08</td>
</tr>
<tr>
<td>-0.0000</td>
<td>-3.6000</td>
<td>0.4015272E-01</td>
<td>0.2793968E-08</td>
</tr>
<tr>
<td>-0.0000</td>
<td>-3.2000</td>
<td>0.4800258E-01</td>
<td>0.2328306E-09</td>
</tr>
<tr>
<td>-0.0000</td>
<td>-2.8000</td>
<td>0.5578430E-01</td>
<td>0.1629815E-09</td>
</tr>
<tr>
<td>-0.0000</td>
<td>-2.4000</td>
<td>0.6350774E-01</td>
<td>0.1629815E-09</td>
</tr>
<tr>
<td>-0.0000</td>
<td>-2.0000</td>
<td>0.7107019E-01</td>
<td>0.1629815E-09</td>
</tr>
<tr>
<td>-0.0000</td>
<td>-1.6000</td>
<td>0.7821018E-01</td>
<td>0.1629815E-09</td>
</tr>
<tr>
<td>-0.0000</td>
<td>-1.2000</td>
<td>0.8452117E-01</td>
<td>0.5820766E-10</td>
</tr>
<tr>
<td>-0.0000</td>
<td>-0.8000</td>
<td>0.8951896E-01</td>
<td>0.1600711E-09</td>
</tr>
<tr>
<td>-0.0000</td>
<td>-0.4000</td>
<td>0.9274071E-01</td>
<td>0.4656613E-09</td>
</tr>
<tr>
<td>-0.0000</td>
<td>0.0000</td>
<td>0.9385431E-01</td>
<td>0.7105427E-14</td>
</tr>
<tr>
<td>-0.0000</td>
<td>0.4000</td>
<td>0.9274071E-01</td>
<td>0.7103048E-09</td>
</tr>
<tr>
<td>-0.0000</td>
<td>0.8000</td>
<td>0.8951896E-01</td>
<td>0.8731149E-10</td>
</tr>
<tr>
<td>-0.0000</td>
<td>1.2000</td>
<td>0.8452106E-01</td>
<td>0.8731149E-10</td>
</tr>
<tr>
<td>-0.0000</td>
<td>1.6000</td>
<td>0.7821023E-01</td>
<td>0.0</td>
</tr>
<tr>
<td>-0.0000</td>
<td>2.0000</td>
<td>0.7107037E-01</td>
<td>0.2095476E-08</td>
</tr>
<tr>
<td>-0.0000</td>
<td>2.4000</td>
<td>0.6350780E-01</td>
<td>0.6984919E-09</td>
</tr>
<tr>
<td>-0.0000</td>
<td>2.8000</td>
<td>0.5578437E-01</td>
<td>0.1396984E-08</td>
</tr>
<tr>
<td>-0.0000</td>
<td>3.2000</td>
<td>0.4800263E-01</td>
<td>0.1396984E-08</td>
</tr>
<tr>
<td>-0.0000</td>
<td>3.6000</td>
<td>0.4015278E-01</td>
<td>0.3026798E-08</td>
</tr>
<tr>
<td>-0.0000</td>
<td>4.0000</td>
<td>0.3219860E-01</td>
<td>0.1862645E-08</td>
</tr>
<tr>
<td>-0.0000</td>
<td>4.4000</td>
<td>0.2417206E-01</td>
<td>0.1862645E-08</td>
</tr>
<tr>
<td>-0.0000</td>
<td>4.8000</td>
<td>0.1624606E-01</td>
<td>0.2328306E-09</td>
</tr>
<tr>
<td>-0.0000</td>
<td>5.2000</td>
<td>0.8754738E-02</td>
<td>0.1396984E-08</td>
</tr>
<tr>
<td>-0.0000</td>
<td>5.6000</td>
<td>0.2147641E-02</td>
<td>0.1164153E-08</td>
</tr>
<tr>
<td>-0.0000</td>
<td>6.0000</td>
<td>-0.3115296E-02</td>
<td>0.1862645E-08</td>
</tr>
<tr>
<td>-0.0000</td>
<td>6.4000</td>
<td>-0.6700821E-02</td>
<td>0.3026798E-08</td>
</tr>
<tr>
<td>-0.0000</td>
<td>6.8000</td>
<td>-0.8532599E-02</td>
<td>0.3026798E-08</td>
</tr>
<tr>
<td>-0.0000</td>
<td>7.2000</td>
<td>-0.8870188E-02</td>
<td>0.2561137E-08</td>
</tr>
<tr>
<td>-0.0000</td>
<td>7.6000</td>
<td>-0.8307122E-02</td>
<td>0.2328306E-09</td>
</tr>
<tr>
<td>-0.0000</td>
<td>8.0000</td>
<td>-0.7672135E-02</td>
<td>0.3492460E-08</td>
</tr>
<tr>
<td>-0.0000</td>
<td>8.4000</td>
<td>-0.7850584E-02</td>
<td>0.2328306E-09</td>
</tr>
<tr>
<td>-0.0000</td>
<td>8.8000</td>
<td>-0.9560246E-02</td>
<td>0.4423782E-08</td>
</tr>
<tr>
<td>-0.0000</td>
<td>9.2000</td>
<td>-0.1313743E-01</td>
<td>0.3725290E-05</td>
</tr>
<tr>
<td>-0.0000</td>
<td>9.6000</td>
<td>-0.1838810E-01</td>
<td>0.4423782E-08</td>
</tr>
<tr>
<td>-0.0000</td>
<td>10.0000</td>
<td>-0.2455102E-01</td>
<td>0.1862645E-08</td>
</tr>
</tbody>
</table>

**Pattern Number 77 Has Been Stored On Record 20 Of Antdata.a507c2**
8.3 Input to ANTDATA

Referring to Section 7.3, the following cards were punched.

Card Column

<table>
<thead>
<tr>
<th></th>
<th>11</th>
<th>21</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0007700020</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>0015100151</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>1111</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>0.0</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>0.0</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>-35.0</td>
<td>-35.0</td>
</tr>
<tr>
<td>7</td>
<td>-30.0</td>
<td>0.0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Step</th>
<th>Card</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>NUMPAT=77, NUMTRR=20</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>Array dimensions are 151 x 151</td>
</tr>
<tr>
<td>5</td>
<td>3</td>
<td>All options for pattern magnitude are specified</td>
</tr>
<tr>
<td>6</td>
<td>4</td>
<td>U-profile location is 0. (V=0)</td>
</tr>
<tr>
<td>7</td>
<td>5</td>
<td>V-profile location is 0. (U=0)</td>
</tr>
<tr>
<td>8</td>
<td>6</td>
<td>LOWCON=-35.0, DASH=-35.0</td>
</tr>
<tr>
<td>9</td>
<td>7</td>
<td>CONLOW=-40.0, COMMAX=0.0, CONINT=5.0</td>
</tr>
<tr>
<td>10</td>
<td>8</td>
<td>No options for current magnitude are specified</td>
</tr>
<tr>
<td>15</td>
<td>9</td>
<td>No options for current phase are specified</td>
</tr>
</tbody>
</table>

8.4 Output from ANTDATA

The following is the printout from computer program ANTDATA.

PLOT OUTPUT FOR PATTERN 77:

U-AXIS PROFILE PLOT REQUESTED -- V=0.

V-AXIS PROFILE PLOT REQUESTED -- U=0.

CONTOUR PLOT OF PATTERN REQUESTED.

LOWEST CONTOUR = -30.0

HIGHEST CONTOUR = 0.0

CONTOUR INTERVAL = 5.0

PATTERN IS NOW BEING GENERATED. IF PATTERN < -35.00 PATTERN = -35.00

THREE-DIMENSIONAL PLOT OF PATTERN REQUESTED

EXECUTION TIME: 27.57 MINUTES.
Refer to Chapter 4 for plotter output.

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.26</td>
<td>U-axis profile</td>
</tr>
<tr>
<td>4.27</td>
<td>V-axis profile</td>
</tr>
<tr>
<td>4.28</td>
<td>Contour plot</td>
</tr>
<tr>
<td>4.29</td>
<td>Three-dimensional plot</td>
</tr>
</tbody>
</table>