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>
<tr>
<td>6. Appendix: The ANTSYN Computer Program</td>
<td>A-1</td>
</tr>
<tr>
<td>6.1 Introduction.</td>
<td>A-1</td>
</tr>
<tr>
<td>6.2 Program Organization</td>
<td>A-2</td>
</tr>
<tr>
<td>6.3 User's Guide to ANTSYN</td>
<td>A-4</td>
</tr>
<tr>
<td>6.4 Program Variables.</td>
<td>A-8</td>
</tr>
<tr>
<td>6.5 Subroutine Descriptions</td>
<td>A-14</td>
</tr>
<tr>
<td>6.6 Statement Listing of ANTSYN.</td>
<td>A-24</td>
</tr>
<tr>
<td>7. Appendix: The ANTDATA Computer Program</td>
<td>A-56</td>
</tr>
<tr>
<td>7.1 Introduction.</td>
<td>A-56</td>
</tr>
<tr>
<td>7.2 Program Organization</td>
<td>A-57</td>
</tr>
<tr>
<td>7.3 User's Guide to ANTDATA</td>
<td>A-59</td>
</tr>
<tr>
<td>7.4 Program Variables.</td>
<td>A-62</td>
</tr>
<tr>
<td>7.5 Subroutine Descriptions</td>
<td>A-63</td>
</tr>
<tr>
<td>7.6 Statement Listing of ANDATA.</td>
<td>A-74</td>
</tr>
</tbody>
</table>
8. Appendix: Example of Input/Output Used With Computer-
Antenna Synthesis.

8.1 Input to ANTSYN
8.2 Output from ANTSYN
8.3 Input to ANTDATA
8.4 Output from ANTDATA
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:

$$\vec{J}_S = \hat{n} \times \vec{H}_1 \quad (2-1)$$

$$\vec{J}_{MS} = -\hat{n} \times \vec{E}_1 \quad (2-2)$$

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]

$$\vec{A}(r) = \mu_0 \frac{e^{-jkr}}{4\pi r} \iint \vec{J}_S(r') e^{jk(r-x)} \, dx'dy' \tag{2-3}$$
$$\vec{A}_M(r) = \varepsilon_0 \frac{e^{-jkr}}{4\pi r} \iint \vec{J}_MS(r') e^{jk(r-x)} \, dx'dy' \tag{2-4}$$

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

\[
E_\theta = -j\omega A_\theta - j\eta_0 A_{M\phi} \tag{2-7}
\]

\[
E_\phi = -j\omega A_\phi + j\eta_0 A_{M\theta} \tag{2-8}
\]

where \( \eta_0 = \sqrt{\mu_0 / \varepsilon_0} \) and the magnetic fields are found using the plane wave relation

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

\[ J_{MS} = J_{MSx} \hat{x} = z \times E_{ay} \hat{y} = E_{ay} \hat{x} \]  (2-10)

and

\[ A_{Mx} = \frac{\epsilon_0}{4\pi} e^{-jkr} \int_\text{aperture} \int 2 E_{ay} e^{jkr \cdot r'} dx'dy' \]  (2-11)

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

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

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

So

\[ E_\theta = -j\omega_0 A_{M\phi} = +j\omega_0 \sin \phi A_{Mx} \]  (2-14)

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

And

\[ A_{Mx} = \frac{\epsilon_0}{4\pi} e^{-jkr} \int_\text{aperture} \int E_{ay} e^{jkr \cdot (x' \sin \theta \cos \phi + y' \sin \theta \sin \phi)} dx'dy' \]  (2-16)

So

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

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

where

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

\[
E_{\theta} = \frac{jke^{-jkr}}{2\pi r} (F_y \sin \phi + F_x \cos \phi) \tag{2-20}
\]

\[
E_{\phi} = \frac{jke^{-jkr}}{2\pi r} \cos \theta (F_y \cos \phi - F_x \sin \phi) \tag{2-21}
\]

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

\[
E_{\theta} = \frac{jkn e^{-jkr}}{2\pi r} \cos \theta (F_y \cos \phi - F_x \sin \phi) \tag{2-22}
\]

\[
E_{\phi} = \frac{-jkn e^{-jkr}}{2\pi r} (F_y \sin \phi + F_x \cos \phi) \tag{2-23}
\]

where

\[
\vec{F} = \int \int \vec{H}_a (x', y') e^{jk(x' \sin \theta \cos \phi + y' \sin \theta \sin \phi)} \, dx' \, dy' \tag{2-24}
\]

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 $\mathbf{J}_S$ is an actual current and its Fourier transform $\mathbf{\hat{A}}$ 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

$$\mathbf{J}_S = \hat{x} J_{sx}(x') J_{sx}(y') + \hat{y} J_{sy}(x') J_{sy}(y')$$  (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 $\mathbf{E}_a$ and $\mathbf{H}_a$ 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

$$E(r) = \frac{jke^{-jkr}}{2\pi r}$$  (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

$$E_\theta(\theta,\phi) = E_\theta/E(r)$$  (2-30)

$$E_\phi(\theta,\phi) = E_\phi/E(r)$$

and

$$G_{\theta x} = \cos \phi \quad G_{\theta y} = \sin \phi$$

$$G_{\phi x} = -\cos \theta \sin \phi \quad G_{\theta y} = \cos \theta \cos \phi$$  (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
\[ F = \int \int F(x',y') e^{j\omega} dx' dy' \quad (2-37) \]

where we let

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

becomes

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

where

\[ \alpha_m = x_m' \sin \theta \cos \phi + y_m' \sin \theta \sin \phi \quad (2-40) \]

and \((x_m',y_m')\) are the element phase center locations. Similarly

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

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

\[
\begin{bmatrix}
\text{cos } \theta \text{ cos } \phi G_x \\
- \text{sin } \phi G_x \\
\text{cos } \theta \text{ sin } \phi G_y \\
- \text{cos } \phi G_y
\end{bmatrix}
\quad (2-42)
\]

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

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

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

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

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

Since the current is y-directed we have

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

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

\[ G_y = \sin \beta \]  

(2-47)

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

so

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

(2-48)

Now

\[ F_{ary} = \sum_{jk} \alpha_m e^{j k \lambda' y_m} \]  

(2-49)

where

\[ \alpha_m = x_m' \sin \theta \cos \phi \]  

(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:

\[ [V] = [Z][I] \] (2-51)

where

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

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

\[
[Z] = \begin{bmatrix}
Z_{11} & Z_{12} & \cdots \\
Z_{21} & \ddots & \vdots \\
\vdots & \ddots & \ddots
\end{bmatrix} = \text{mutual impedance matrix (known from calculations or experiment)}
\]
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

$$F(u,v) = \int_\text{aperture} \int E_a(x',y') e^{jk(x'u + y'v)} \, dx'\,dy'$$

(3-1)

where

$$u = \sin \theta \cos \phi$$

(3-2)

$$v = \sin \theta \sin \phi.$$  

(3-3)

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

$$s = x'/\lambda$$

(3-4)

$$t = y'/\lambda$$

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

\[ F(u,v) = \int \int f(s,t) e^{j2\pi(su + tv)} \, ds \, dt \]  
(3-6)

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

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

(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

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

(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 \((u_n^{(i)}, v_n^{(i)})\) 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 convergence. 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 patterns 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 \(g_n^{(i)}(s,t)\) related to it as follows:

\[
G(u-u_n^{(i)}, v-v_n^{(i)}) = \int_{\text{aperture}} \int g_n^{(i)}(s,t) e^{j(2\pi s u + 2\pi t v)} \, ds \, dt
\]  

(3-10)
The source distribution corresponding to the pattern of (3-7) is

\[ f(K)(s,t) = f(o)(s,t) + \sum_{i=1}^{K} \Delta f^{(i)}(s,t) \]  

(3-11)

where

\[ \Delta f^{(i)}(s,t) = \sum_{n} a_{n}^{(i)} g_{n}^{(i)}(s,t) \]  

(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

\[ f^{(K)}(s,t) = \sum_{\lambda m} I^{(K)}_{\lambda m} \delta(s-s_{\lambda}, t-t_{m}) \]  

(3-13)

where \( \delta \) is the dirac delta function and \( I^{(K)}_{\lambda m} \) are the currents for the \( \lambda m \) 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

\[ g_{n}^{(i)}(s,t) = \sum_{\lambda m} g_{n\lambda m}^{(i)} \delta(s-s_{\lambda}, t-t_{m}) \]  

(3-14)

for arrays. Then (3-10) becomes

\[ G(u-u_{n}^{(i)}, v-v_{n}^{(i)}) = \sum_{\lambda m} g_{n\lambda m}^{(i)} e^{j2\pi(s_{\lambda}u + t_{m}v)} \]  

(3-15)

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

\[ \Delta f^{(i)}(s,t) = \sum_{n} a_{n}^{(i)} \sum_{\lambda m} g_{n\lambda m}^{(i)} \delta(s-s_{\lambda}, t-t_{m}) \]  

(3-16)

and let

\[ \Delta f^{(1)}(s,t) = \sum_{\lambda m} I^{(1)}_{\lambda m} \delta(s-s_{\lambda}, t-t_{m}) \]  

(3-17)

So
\[ \Delta I_{\lambda m}(1) = \sum_n a_n g_n^{(1)} \]  
\[ I_{\lambda m}^{(K)} = I_{\lambda m}^{(o)} + \sum_{i=1}^{K} \Delta I_{\lambda m}^{(1)} \]  

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

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

(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

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

(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

\[ g_n(i)(t) = \begin{cases} \frac{1}{L_{y\lambda}} (1 - 2|t|/L_{y\lambda}) \exp(-j2\pi v_n(i)t) & |t| \leq L_{y\lambda}/2 \\ 0 & \text{elsewhere} \end{cases} \] (3-22)

The corresponding pattern found from (3-10) is

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

### 3.5.2 Linear Array

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

\[ g_{nm}(i) = \frac{1}{P} \exp(-j2\pi v_n(i)t_m) \] (3-24)

where \( t_m \) are the positions of the elements and equal \( md_{y\lambda} \) and \( P \) is the total number of elements. The corresponding pattern is

\[ G(u-u_n(i)) = \frac{\sin[P(v-v_n(i))\pi d_{y\lambda}]}{P \sin[(v-v_n(i))\pi d_{y\lambda}]} \] (3-25)

### 3.5.3 Rectangular Aperture

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

\[ g_n(i)(s,t) = \begin{cases} \frac{1}{L_{x\lambda}} \frac{1}{L_{y\lambda}} \exp(-j2\pi(u_n(i)s + v_n(i)t)) & |s| \leq L_{x\lambda}/2 \\\n0 & \text{elsewhere} \end{cases} \] (3-26)

The pattern is

\[ G(u-u_n(i), v-v_n(i)) = \frac{\sin[L_{x\lambda}(u-u_n(i))\pi]}{L_{x\lambda}(u-u_n(i))\pi} \frac{\sin[L_{y\lambda}(v-v_n(i))\pi]}{L_{y\lambda}(v-v_n(i))\pi} \] (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

\[
e_{n2m}^{(i)} = \frac{1}{P_x P_y} \exp \left[-j2\pi (u_n^{(i)} s_x + v_n^{(i)} t_m)\right] \tag{3-28}
\]

The pattern is

\[
G (u-u_n^{(i)}, v-v_n^{(i)}) = \frac{\sin \left[ P_x (u-u_n^{(i)}) d_{x\lambda} \right]}{P_x \sin \left[ (u-u_n^{(i)}) d_{x\lambda} \right]} \frac{\sin \left[ P_y (v-v_n^{(i)}) d_{y\lambda} \right]}{P_y \sin \left[ (v-v_n^{(i)}) d_{y\lambda} \right]} \tag{3-29}
\]

3.5.5 Circular Aperture

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

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

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

\[
e_n^{(i)}(\rho', \phi') = \frac{1}{\pi a_\lambda^2} \exp \left[-j2\pi \rho' \left(\cos \phi' u_n^{(i)} + \sin \phi' v_n^{(i)}\right)\right] \rho' \leq a_\lambda \tag{3-31}
\]

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

\[ = \frac{1}{\pi a_\lambda^2} \int_{0}^{2\pi} \int_{0}^{a_\lambda} \exp\left\{ j 2\pi \rho \left[ \cos (\alpha - \phi') \right] \rho \, d\rho \, d\phi' \right\} \]

where

\[ C = [(\mathbf{u} \cdot \mathbf{u}_m)^2 + (\mathbf{v} \cdot \mathbf{v}_m)^2]^{1/2} \]

\[ \alpha = \tan^{-1} \frac{\mathbf{u} \cdot \mathbf{u}_m}{\mathbf{v} \cdot \mathbf{v}_m} \]

Now (3-33) is easily integrated as

\[ G(\mathbf{u}, \mathbf{v}) = \frac{2\pi a_\lambda}{J_0(2\pi a_\lambda C)} \int_{0}^{\pi} J_0(2\pi a_\lambda C) \rho \, d\rho \]

\[ = 2 \frac{J_1(2\pi a_\lambda C)}{2\pi a_\lambda C} \]

If \( \mathbf{u} = \mathbf{v} = 0 \) we have

\[ G_n(\mathbf{u}, \mathbf{v}) = 2 \frac{J_1(2\pi a_\lambda \sin \theta)}{2\pi a_\lambda \sin \theta} \]

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

Also note that when \( \mathbf{u} = \mathbf{u}_m \) and \( \mathbf{v} = \mathbf{v}_m \), \( C = 0 \) and (3-38) becomes unity.

Thus \( (\mathbf{u}_m, \mathbf{v}_m) \) 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

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

(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

\[ \Omega_A = \int \int_{\Omega} |F(\theta, \phi)|^2 \, d\Omega \]  

(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 θ,φ space to the u,v plane using

\[ u = \sin \theta \cos \phi \]
\[ v = \sin \theta \sin \phi \] (3-44)

We are collapsing the spherical surface described by θ,φ onto a planar surface through its equator giving a circular disk. There is an ambiguity here because points on the upper hemisphere (θ > π/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

\[ \Omega_A = \int_0^{\pi/2} \int_0^\pi |F(\theta,\phi)|^2 d\Omega \] (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(θ,φ) is exact).

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

\[ du \, dv = \cos \theta \, d\Omega \] (3-46)

so

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

But from (3-44)

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

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

Thus (3-45) becomes

\[ \Omega_A = \int \int_{u^2 + v^2 < 1} |F(u,v)|^2 \frac{du \ dv}{\sqrt{1 - u^2 - v^2}} \]  \hspace{1cm} (3-50)

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

\[ \Omega_A = \int \int_{u^2 + v^2 < 1} |F(u,v)|^2 \sin \theta \ J \ du \ dv \]  \hspace{1cm} (3-51)

where J is the Jacobian given by

\[ J = \frac{\partial (\theta, \phi)}{\partial (u,v)} = \frac{1}{\frac{\partial (u,v)}{\partial (\theta, \phi)}} \]  \hspace{1cm} (3-52)

and

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

\[ = \begin{vmatrix} \cos \theta \cos \phi - \sin \theta \sin \phi \\ \cos \theta \sin \phi \sin \theta \cos \phi \end{vmatrix} \]

\[ = \cos \theta \sin \theta \]  \hspace{1cm} (3-53)

So

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

\[ = \frac{du \ dv}{\sqrt{1 - u^2 - v^2}} \]  \hspace{1cm} (3-54)

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)

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

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

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

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

where \( \alpha = 1 - r^2 \). 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., ..., -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 = $\infty$
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>v</td>
<td>0. dB</td>
<td>0.5 dB</td>
<td>- 0.5 dB</td>
</tr>
<tr>
<td></td>
<td>0.4 &lt;</td>
<td>v</td>
<td>&lt; 1.0</td>
</tr>
</tbody>
</table>

The desired pattern is then a square beam with no side lobes but - 40 dB side lobes will be tolerated. The original pattern is a Woodward-Lawson pattern with sample points at v = - 0.2, - 0.1, 0., 0.1, 0.2 and sample values of 1.0 at these points. This original pattern has excursions + 0.86 and - 0.25 dB over the main beam and a side lobe level of - 20. 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>
<thead>
<tr>
<th>Antenna Type - 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_yλ = 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 d_xλ = 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_yλ = 10</td>
<td>0.128</td>
<td>0.128</td>
<td>-26.6</td>
</tr>
<tr>
<td>Uniform rectangular aperture - 4</td>
<td>L_xλ = 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 d_xλ = 0.5</td>
<td>0.0443</td>
<td>0.0443</td>
<td>-13.3</td>
</tr>
<tr>
<td>Uniform circular aperture - 6</td>
<td>a_λ = 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. 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 $t_m = 0., +0.496, +0.983, +1.926, +2.372, +3.188, +3.545$. 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 -25dB 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:

<table>
<thead>
<tr>
<th>( u, v )</th>
<th>( F_d(u,v) )</th>
<th>( F_{II}(u,v) )</th>
<th>( F_r(u,v) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>(</td>
<td>u</td>
<td>&lt; 0.2,</td>
<td>u</td>
</tr>
<tr>
<td>( 0.36 &lt;</td>
<td>u</td>
<td>&lt; 0.50 )</td>
<td>unspecified</td>
</tr>
<tr>
<td>( 0.12 &lt;</td>
<td>v</td>
<td>&lt; 0.50 )</td>
<td>-20.0</td>
</tr>
</tbody>
</table>

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 = 51
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;v&lt;0.50</td>
<td></td>
<td></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 J₁(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 CURREN 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 NUPAT, NUMTRK, NUMSKP and IPASS are read off of the storage unit. NUPAT is the pattern number assigned to the previous job. The program adds one to NUPAT 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, FDBCNP, FDBPR, FORGPT, FORGCP, 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>Step</th>
<th>Description</th>
<th>Device</th>
<th>Location</th>
<th>Availability</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Job assignment</td>
<td>Auxiliary</td>
<td>MAIN</td>
<td>Standard</td>
</tr>
<tr>
<td>2.</td>
<td>Pattern parameters</td>
<td>Cards; use</td>
<td>MAIN</td>
<td>Standard</td>
</tr>
<tr>
<td></td>
<td>Namelist PARAM</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.</td>
<td>Output print control</td>
<td>Cards; use</td>
<td>MAIN</td>
<td>Standard</td>
</tr>
<tr>
<td></td>
<td>Namelist IPRINT</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.</td>
<td>Antenna parameters</td>
<td>Cards; use</td>
<td>INPUT</td>
<td>Standard</td>
</tr>
<tr>
<td></td>
<td>Namelist PATIN</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>If ITYPE &gt;7 also use subroutine SINPUT</td>
<td></td>
<td>SINPUT</td>
<td>Special</td>
</tr>
<tr>
<td>5.</td>
<td>Desired pattern</td>
<td>Program</td>
<td>DESPAT</td>
<td>Special</td>
</tr>
<tr>
<td></td>
<td>statements to load FDES, FU</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>or</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Cards; call subroutine READ to load FDES, FU, and FL</td>
<td></td>
<td>DESPAT</td>
<td>Special</td>
</tr>
<tr>
<td>6.</td>
<td>Original pattern and original source</td>
<td>Cards; read</td>
<td>ORGPAT</td>
<td>Standard</td>
</tr>
<tr>
<td></td>
<td>NORG, US, VS, CORG to generate original state using</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Woodward-Lawson method</td>
<td></td>
<td>ORGPAT</td>
<td>Special</td>
</tr>
<tr>
<td></td>
<td>or</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Cards or program statements; if Woodward-Lawson is not used, load F, CURR, and CURI, set NORG=0</td>
<td></td>
<td>ORGPAT</td>
<td>Special</td>
</tr>
<tr>
<td>7.</td>
<td>Special correction pattern function</td>
<td>Program</td>
<td>SPECPT</td>
<td>Special</td>
</tr>
<tr>
<td></td>
<td>statements to generate values of PAT, ITYPE &gt;7</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8.</td>
<td>Special correction source function</td>
<td>Program</td>
<td>SPSOR</td>
<td>Special</td>
</tr>
<tr>
<td></td>
<td>statements to generate companion to special pattern, ITYPE &gt;7</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9.</td>
<td>Special location</td>
<td>Program</td>
<td>SPLOC</td>
<td>Special</td>
</tr>
<tr>
<td></td>
<td>statements to generate source coordinates given source array subscripts, use when ITYPE &gt;7</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10.</td>
<td>Job storage</td>
<td>Programming</td>
<td>MAIN</td>
<td>Standard</td>
</tr>
<tr>
<td></td>
<td>to write job data onto storage unit</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Step 4. This step provides input concerning the particular antenna to be used. It is read in on cards under

MAP/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 print out 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>$a_n$</td>
<td>ARAD</td>
</tr>
<tr>
<td>$a_\lambda$</td>
<td>DISX</td>
</tr>
<tr>
<td>$d_x\lambda$</td>
<td>DISY</td>
</tr>
<tr>
<td>$d_y\lambda$</td>
<td></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 directivities 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 distribution profile or list; 0 None, 1 Profile (S and/or T axis) for continuous 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.

FDESPR  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 distribution 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 current distribution; 0 No, 1 Yes - Default is 0.

IDISK  Input variable controlling output of data to disk storage; 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</td>
<td>Uniform</td>
</tr>
<tr>
<td>2</td>
<td>Equally Spaced Linear Array</td>
<td>Uniform</td>
</tr>
<tr>
<td>3</td>
<td>Line Source</td>
<td>Triangular</td>
</tr>
<tr>
<td>4</td>
<td>Rectangular Aperture</td>
<td>Uniform</td>
</tr>
<tr>
<td>5</td>
<td>Rectangular Array</td>
<td>Uniform</td>
</tr>
<tr>
<td>6</td>
<td>Circular Aperture</td>
<td>Uniform</td>
</tr>
<tr>
<td>7</td>
<td>General Array</td>
<td>Uniform</td>
</tr>
<tr>
<td>GT7</td>
<td>SPECPT</td>
<td>SPSOR</td>
</tr>
</tbody>
</table>

MCENT  Input variable - First subscript of pattern array where pattern is to 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.

DELCN 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.
<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DELTAU</td>
<td>Input variable - Increment between comparison points in U direction. Also, Increment between pattern print out points.</td>
</tr>
<tr>
<td>DELTAV</td>
<td>Input variable - Increment between comparison points in V direction. Also, increment between pattern print out points.</td>
</tr>
<tr>
<td>DIRFNL</td>
<td>Directivity of final pattern.</td>
</tr>
<tr>
<td>DIRORG</td>
<td>Directivity of original pattern.</td>
</tr>
<tr>
<td>DISX</td>
<td>Input variable - Spacing between antenna array elements in X direction normalized to a wavelength.</td>
</tr>
<tr>
<td>DISY</td>
<td>Input variable - Spacing between antenna array elements in Y direction normalized to a wavelength.</td>
</tr>
<tr>
<td>F( , )</td>
<td>Current pattern value.</td>
</tr>
<tr>
<td>FDES( , )</td>
<td>Input variable - Desired pattern value.</td>
</tr>
<tr>
<td>FINALS</td>
<td>Input variable - Final point of current distribution print outs in S direction.</td>
</tr>
<tr>
<td>FINALT</td>
<td>Input variable - Final point of current distribution print outs in T direction.</td>
</tr>
<tr>
<td>FINALU</td>
<td>Input variable - Final point of pattern comparison and print outs in U direction.</td>
</tr>
<tr>
<td>FINALV</td>
<td>Input variable - Final point of pattern comparison and print outs in V direction.</td>
</tr>
<tr>
<td>FL( , )</td>
<td>Input variable - Lower limit on synthesized pattern.</td>
</tr>
<tr>
<td>FNORM</td>
<td>Factor by which pattern F( , ) is divided to normalize it to 0 dB at the point (MCENT, NCENT).</td>
</tr>
<tr>
<td>FU( , )</td>
<td>Input variable - Upper limit on synthesized pattern.</td>
</tr>
<tr>
<td>INITLS</td>
<td>Input variable - Initial point of current distribution print outs in S direction.</td>
</tr>
<tr>
<td>INITLT</td>
<td>Input variable - Initial point of current distribution print outs in T direction.</td>
</tr>
<tr>
<td>LX</td>
<td>Length of antenna in X direction in wavelengths - For continuous aperture sources this is an input variable.</td>
</tr>
<tr>
<td>LY</td>
<td>Length of antenna in Y direction in wavelengths - For continuous aperture sources this is an input variable.</td>
</tr>
<tr>
<td>S</td>
<td>Source coordinate X normalized to a wavelength.</td>
</tr>
<tr>
<td>SS( )</td>
<td>Antenna array element position in S direction - Input variable for ITYPE = 7.</td>
</tr>
</tbody>
</table>
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.*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 NORG 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, J1, 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 J1 of the pattern matrices. II and J1 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, J1, 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 J1 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:

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

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

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.
<table>
<thead>
<tr>
<th>Pattern</th>
<th>Variable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Original</td>
<td>FORGPR = 1</td>
</tr>
<tr>
<td>Final</td>
<td>FDBPR = 1</td>
</tr>
</tbody>
</table>

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

Subroutine CONTUR 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.

<table>
<thead>
<tr>
<th>Source</th>
<th>Variable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Original distribution</td>
<td>ICURCN = 1</td>
</tr>
<tr>
<td>Final distribution</td>
<td>FCURCN = 1</td>
</tr>
</tbody>
</table>

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 Namelist 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 JOB 51702, COFFEY
/*MAIN TIME=3, REGION=220K, LINES=10, CARDS=0
/*PIORITY PRIORITY
// EXEC FORTRAN,LIB2=55PLIB
//FORT,SYAIN UD *
/*
//GB.F122EDIT DD RSN=ANTDATA,UNIT=767,FORM=3330,YOL=SER=USEPK,DIS=SER
//GB,SYAIN 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:
  DIRCIV
  INPUT
  READ
  CRGPAT
  ANTSYN
  SEARCH
  CHECK
  UPDATE
  PAT
  SOURCE
  LOCSOR
  SPLOC
  SPECPT
  SPSOSR
  SINPUT
  CURREN
  PRINT
  PROFIL
  CONTUR
  PATCCN
  LIST
  DESPAT
  DATE
  STIME
  BESJ

...STANDARD FORTRAN LIBRARY SUBPROGRAMS...

INPUT/OUTPUT SUPPORT:
  FTC5FC01  (SYSIN)  --  CARD READER
  FT06F001  (SYSPRINT)  --  LINE PRINTER
  FT22FC01  (ANTDATA.A507C2)  --  AUXILIARY STORAGE

STORAGE REQUIREMENTS:  220K

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

INTEGER TITLE(20)
INTEGER NUMSKP(35), FDESPT, FDESN, FDESPR
INTEGER FORGPT, FORGCN, FORGPR, FDBPT, FDBCN, FDBPR, DIRECT
REAL FDES(51,51), FU(51,51), FL(51,51), F(51,51)
REAL DATA(401,2), DATA2(401,2)
REAL UORG(100), VORG(100)
REAL FDB(1), FDBCN(1), FDBPR(1)
REAL INITLS, INITLT
COMPLEX SOURCE
INTEGER FORGPT, FDBPT, FDBCN, FDBPR
COMMON /MPROG/ NCUR, ICUR
COMMON /START/ NORG, UORG, VORG, CURG
COMMON /PAT1/ P1, P2, P3, P4, P5, P6, PI, SS(400), IT(400), RR(400)
COMMON /PAT2/ II, I2, I3, I4, I5
COMMON /LOC/ ITYPE
NAMELIST /PARAM/ IDISK, ISYM, ITRMAX, DELTAU, DELTAV, STARTU, STARTV,
NAMELIST /IPRINT/ FDESPT, FDESPR, FDESN, FDBPT, FDBCN, FDBPR,
$FORGPT, FORGCN, FORGPR, FDBPT, FDBCN, FDBPR, 
$DIRECT

BEGIN PROCESSING

9999 CONTINUE
READ(22'1,6550) NUMPAT, NUMTRK, NUMSKP, IPASS

8550 FORMAT(75A4,11(200A4))
NUMPAT=NUMPAT+1
READ(22'1,6550) NUMPAT, NUMTRK, NUMSKP, IPASS

CALL DATE(11, J1, K1)
CALL STIME(IT)
IHR=IT/10000
IFR=IT-IHR*10000
FHR=IFR/10000
FM=FHR*60.
IMIN=FM
ISEC=(FM-IMIN)*60.
WRITE(6,1) I1, J1, K1, IHR, IMIN, ISEC, NUMPAT
1 FORMAT('1 ANTE', 'NN', 'SYNT', 'HESI', 'S PR', 'OGRA',
$' S', 'VERS', 'ION', '1', 'EVEL', '1', '73', '/164',
$' E', 'PI', 'E.E', ' DEP', 'T', '1')
DEFAULT PARAMETERS

34  IDISK=0
35  ISYM=0
36  ITRMAX=100
37  MMAX=1
38  NMAX=1
39  MCENT=1
40  NCENT=1
41  DELTAU=0.
42  DELTAV=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  FORCPPR=0
62  DIRECT=0
63  MCURM=1.
64  ISIC=0
65  DELTAS=C.
66  DELTAT=0.

INPUT

67  READ(5,PARAM)

WRITE(6,1521) IDISK,STARTU,MMAX,ISYM,STARTV,NMAX,ITRMAX,DELTAU,
MCENT,DELTAV,NCENT
1521 FORMAT(87X,'PROGRAM PARAMETERS'//35X,'IDISK = ',11,26X,'STARTU =
5,F6.3,26X,'MMAX = ',13/35X,'ISYM = ',11,26X,'STARTV = ',14,22X,'DELTAV = ',14,22X,
MCENT = ',13/35X,'DELTAV = ',14,22X,NCENT = ',13//)
READ(5,PRINT)
WRITE(6,1522) FDESPT,FCURPT,FDESPT,FCURPT,
FDESPT,FDESPT,FDESPT,FDESPT,
FDESPT,FDESPT,FDESPT,FDESPT,
FDESPT,FDESPT,FDESPT,FDESPT,
FDESPT,FDESPT,FDESPT,FDESPT,
FDESPT,FDESPT,FDESPT,FDESPT,
FDESPT,FDESPT,FDESPT,FDESPT,
FDESPT,FDESPT,FDESPT,FDESPT,
FDESPT,FDESPT,FDESPT,FDESPT,
FDESPT,FDESPT,FDESPT,FDESPT,
FDESPT,FDESPT,FDESPT,FDESPT,
FDESPT,FDESPT,FDESPT,FDESPT,
FDESPT,FDESPT,FDESPT,FDESPT,
FDESPT,FDESPT,FDESPT,FDESPT,
FDESPT,FDESPT,FDESPT,FDESPT,
FDESPT,FDESPT,FDESPT,FDESPT,
FDESPT,FDESPT,FDESPT,FDESPT,
FDESPT,FDESPT,FDESPT,FDESPT,
FDESPT,FDESPT,FDESPT,FDESPT,
FDESPT,FDESPT,FDESPT,FDESPT,
FDESPT,FDESPT,FDESPT,FDESPT,
FDESPT,FDESPT,FDESPT,FDESPT,
FDESPT,FDESPT,FDESPT,FDESPT,
FDESPT,FDESPT,FDESPT,FDESPT,
FDESPT,FDESPT,FDESPT,FDESPT,
FDESPT,FDESPT,FDESPT,FDESPT,
FDESPT,FDESPT,FDESPT,FDESPT,
FDESPT,FDESPT,FDESPT,FDESPT,
FDESPT,FDESPT,FDESPT,FDESPT,
FDESPT,FDESPT,FDESPT,FDESPT,
FDESPT,FDESPT,FDESPT,FDESPT,
FDESPT,FDESPT,FDESPT,FDESPT,
FDESPT,FDESPT,FDESPT,FDESPT,
FDESPT,FDESPT,FDESPT,FDESPT,
FDESPT,FDESPT,FDESPT,FDESPT,
FDESPT,FDESPT,FDESPT,FDESPT,
FDESPT,FDESPT,FDESPT,FDESPT,
FDESPT,FDESPT,FDESPT,FDESPT,
FDESPT,FDESPT,FDESPT,FDESPT,
FDESPT,FDESPT,FDESPT,FDESPT,
FDESPT,FDESPT,FDESPT,FDESPT,
FDESPT,FDESPT,FDESPT,FDESPT,
FDESPT,FDESPT,FDESPT,FDESPT,
FDESPT,FDESPT,FDESPT,FDESPT,
FDESPT,FDESPT,FDESPT,FDESPT,
FDESPT,FDESPT,FDESPT,FDESPT,
FDESPT,FDESPT,FDESPT,FDESPT,
FDESPT,FDESPT,FDESPT,FDESPT,
FDESPT,FDESPT,FDESPT,FDESPT,
FDESPT,FDESPT,FDESPT,FDESPT,
71 CALL INPUT
72 IF(ITYPE.EQ.7.AND.ICURPR.EQ.1) ICURPR =2
73 IF (ITYPE.EQ.7 .AND. FCURPR.EQ.1) FCURPR =2
74 CALL LOCSCR(1,1,INITLS,INITLT)
75 CALL LOCSOR(NCUR,NCUR,FINALS,FINALT)
76 IF(NCUR.NE.1) DELTAS=(FINALS-INITLS)/(NCUR-1)
77 IF(NCUR.NE.1) DELTAT=(FINALT-INITLT)/(NCUR-1)
78 CALL DESPAT(FDES,FL,MMAX,NMAX,STARTU,STARTV,$DELTAU,DELTAV)
79 IF(FDESPT) 300,300,301
80 WRITE(6,302)
81 FORMAT(1H1,1C1\$5X,'DESERVED PATTERN IN DB.\$')
82 CALL PRINT(FDES,MMAX,NMAX,STARTU,STARTV,DELTAU,DELTAV)
83 IF(FCURPN) 303,303,304
84 CONTINUE
85 CALL PATCON(FDES,MMAX,NMAX,-0.5,1.3,0.2,STARTU,STARTV,$DELTAU,DELTAV,NUMPAT,ISYM)
86 IF(FDESPT) 306,306,307
87 IF(MMAX.LE.1) GO TO 306
88 WRITE(6,31C) NUMPAT
89 FORMAT(1H1,1C1\$12X,'U-AXIS PROFILE OF DESERVED PATTERN ',14//
90 $12X,'U',16X,'V',15X,'FOES(U,V)',10X,'FU(U,V)',12X,'FL(U,V)'/
91 V=STARTV+(NCENT-1)*DELTAV
92 DO 309 I=1,MMAX
93 U=STARTU+(I-1)*DELTAU
94 WRITE(6,311) U,V,FDES(I,NCENT),FU(I,NCENT),FL(I,NCENT)
95 IF(NMAX.EQ.1) GO TO 306
96 WRITE(6,312) NUMPAT
97 FORMAT(1H1,10X,'V-AXIS PROFILE OF DESERVED PATTERN ',14//
98 $12X,'U',16X,'V',15X,'FOES(U,V)',10X,'FU(U,V)',12X,'FL(U,V)'/
99 U=STARTU+(MCENT-1)*DELTAV
100 DO 313 J=1,NMAX
101 V=STARTV+(J-1)*DELTAV
102 WRITE(6,311) U,V,FDES(MCENT,J),FU(MCENT,J),FL(MCENT,J)
103 CONTINUE
104 ENTER ORIGINAL PATTERN
105 CALL CRGCPAT(F,MMAX,NMAX,STARTU,STARTV,DELTAU,DELTAV,$CURR,CURR,MCUR,NCUR)
106 Output of Original Pattern
107 IF(FORCGP) 404,404
108 CALL PATCON(F,MMAX,NMAX,-0.5,1.3,0.2,STARTU,STARTV,DELTAU,$DELTAU,NUMPAT,ISYM)
109 CONTINUE
110 IF(FORCGP) 406,406,407
111 CONTINUE
DC 408 J=1,401
U=(J-1)*C,CO5-1.C
V=U
SUMU=0.
SUMV=C.
DO 409 K=1,NORG
SUMU=SUMU+CORG(K)*PAT(U-UORG(K),-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)

IF(NMAX.LE.1)
GO TO 406
WRITE(6,411)
FCRMAT(IH1,SX,'V-AXIS PROFILE OF INITIAL PATTERN')
CALL PROFIL(DATA2,401,NUMPAT)
CONTINUE

C CRITICAL EXCITATION

IF(ICURPT)
500,500,501
WRITE(6,502)
502 FORMATT(IH1,55X,'INITIAL CURRENT')
CALL PRINT(CURR,MCUR,NCUR,INITLS,INITLT,DELTAS,DELTAT)
WRITE(6,503)
503 IF(ICURPR-I) 506,503,504
WRITE(6,505)
505 FORMATT(IH1,10X,'INITIAL CURRENT')
CALL CONTUR(MCUR,NCUR,0.005,-0.04,0.04,0.0,CURI,NUMPAT)
WRITE(6,506)
506 FORMATT(IH1,10X,'INITIAL CURRENT')
CALL CONTUR(MCUR,NCUR,0.005,-0.04,0.04,0.0,CURI,NUMPAT)
507 IF(MCUR.LTE.1) GO TO 508
WRITE(6,510)
510 FORMATT(IH1,10X,'S AXI5 PROFIUE OF INI5UAL CUI5ENT'//
$13X,'S',17X,'T',18X,'REAL',12X,'IMAGINARY',10X,'MAGNITUDE',
$12X,'PHASE'/)
J=NCUR/2+1
DO 509 I=1,MCUR
CALL LOCGR(I,J,S,T,ITYPE)
AMAG=SQRT(CURR(I,J)**2+CURI(I,J)**2)
IF(AMAG.EQ.0.) APH=0.
IF(AMAG.EQ.0.) GO TO 509
APH=ATAN2(CURI(I,J),CURR(I,J))*57.2957795
WRITE(6,511) S,T,CURR(I,J),CURI(I,J),AMAG,APH
511 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'//
$13X,'S',17X,'T',18X,'REAL',12X,'IMAGINARY',10X,'MAGNITUDE',
$12X,'PHASE'/)
166 I=MCUR/2+1
167 DO 513 J=1,NCUR
168 CALL LCSOR(I,J,S,T,ITYPE)
169 CR=CURR(I,J)
170 CI=CURI(I,J)
171 AMAG=SQRT(CR*CR+CI*CI)
172 IF(AMAG.EQ.0.) APH=0.
173 IF(AMAG.EQ.0.) GO TO 513
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,
$'J',10X,'S',15X,'T',15X,'CURR',11X,'CURI')
179 CALL LIST(CURR,CURI,MCUR,NCUR)
180 506 CONTINUE
181 IC=0
182 WRITE(6,4747)
183 4747 FORMAT(1H1)
184 CALL ANTSYN(ISUC,MMAX,NMAX,FDES,FU,FL,ITRMAX,ISYMM,CORC(JF,IC,US
16$ VS,STARTU,DELTAV,STARTV,DELTAV,MCENT,NCENT,ITER,FNORM,F)
C
C PRINT OUT RESULTS
C
185 WRITE(6,391)
186 391 FORMAT(1H1///52X,'--- FINAL COEFFICIENTS ---'//45X,'J',7X,
$'US(J)',7X,'VS(J)',5X,'CORC(J)'//)
187 IF(IC.LE.0) WRITE(6,977)
188 977 FORMAT(10X,'NC ITERATIONS PERFORMED')
189 IF(IC.LE.0) GO TO 978
190 DO 498 J=1,IC
191 498 WRITE(6,35) J,US(J),VS(J),CORC(J)
192 35 FORMAT(44X,I3,5X,F7.4,5X,F7.4,5X,F7.4)
193 WRITE(6,497) ITER
194 497 FORMAT(1H0,9X,'NUMBER OF ITERATIONS = ',16)
195 WRITE(6,496) FNORM
196 496 FORMAT(1H0,9X,'FNORM = ',F10.5)
197 WRITE(6,976) NUMPAT
198 976 FORMAT(1H0,9X,'PATTERN NUMBER = NUMPAT = ','15)

C

C OUTPUT FINAL PATTERN IN DB
C
199 978 CONTINUE
200 DO 29 J=1,MMAX
201 DO 29 K=1,NMAX
202 IF(F(J,K)) 290,289,290
203 289 F(J,K)=-200.
GO TO 29
F(J,K)=20.*ALCG10(ABS(F(J,K)))
CONTINUE

IF(FDOPT) 600,600,601
WRITE(6,602)
FORMAT(1HI//////////////////////////////////////////////////////////55X,'FINAL PATTERN IN OR.')
CALL PRINT(F,MMAX,NMAX,STARTU,STARTV,DELTAU,DELTAV)

IF(FDOCN) 603,603,604
CONTINUE
CALL PATCON(F,MMAX,NMAX,2,-45.,0.0,5.0,STARTU,STARTV,DELTAV,NUMPAT,ISYMM)

IF(FDOPT) 606,606,607
CONTINUE
DO 608 J=1,401
U=(J-1)*C.005-1.0
V=U
SUMU=0.
SUMV=0.
DO 609 K=1,IC
SUMU=SUMU+CORCOF(K)*PAT(U-US(K),-VS(K),ITYPE)
SUMV=SUMV+CORCOF(K)*PAT(-US(K),V-VS(K),ITYPE)
CONTINUE
DATAI(J,1)=U
DATA2(J,1)=V
DATA1(J,2)=DATA1(J,2)+SUMU
DATA2(J,2)=DATA2(J,2)+SUMV
DATA1(J,2)=DATA1(J,2)*FNORM
DATA2(J,2)=DATA2(J,2)*FNORM
CONTINUE
IF(MMAX.LE.1) GO TO 2901
WRITE(6,610)
FORMAT(1HI,25X,'U-AXIS PROFILE OF FINAL PATTERN')
CALL PROFIL(DATA1,401,NUMPAT)

IF(NMAX.LE.1) GO TO 606
WRITE(6,611)
FORMAT(1HI,25X,'V-AXIS PROFILE (IF FINAL PATTERN)')
CALL PROFIL(DATA2,4C1,NUMPAT)

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

IF(FCURPT) 700,700,701
WRITE(6,702)
FORMAT(1HI//////////////////////////////////////////////////////////55X,'FINAL CURR')
CALL PRINT(CURR,MCUR,NCUR,INITLS,INITLT,DELTAS,DELTAT)
WRITE(6,782)

IF(CURCN) 703,703,704
CALL PRINT(CURI,MCUR,NCUR,INITLS,INITLT,DELTAS,DELTAT)

IF(FCURCN) 704,704,705
WRITE(6,705)
252 705 FCRMAT(1H1///10X, 'FINAL CURR')
253 CALL CONTUR(MCUR, NCUR, C, COS, -0.04, 0.04, 0.0, CURR, NUMPAT)
254 WRITE(6,705)
255 785 FCRMAT(1H1///10X, 'FINAL CURR')
256 CALL CONTUR(MCUR, NCUR, C, COS, -0.04, 0.04, 0.0, CURR, NUMPAT)
257 703 IF(FCURPR-1) T06, 707, 711
258 707 IF(MCUR.LE.1) GO TO 708
259 WRITE(6,710)
260 710 FCRMAT(1H1///10X, 'FINAL CURR')
261 IF(NCUR.LE.1) GO TO 707
262 DO 709 I=1, MCUR
263 CALL LCSCR(I, J, S, T, ITYPE)
264 CR=CURR(I, J)
265 CI=CURI(I, J)
266 AMAG=SQRT(CR*CR+CI*CI)
267 IF(AMAG.EQ.0.) APH=0.
268 IF(AMAG.EQ.0.) GO TO 710
269 APH=ATAN2(CI, CR)*57.2957795
270 709 WRITE(6,511) S, T, CR, CI, AMAG, APH
271 708 IF(NCUR.LE.1) GO TO 706
272 WRITE(6,712)
273 712 FCRMAT(1H1///10X, 'FINAL CURR')
274 IF(NCUR.LE.1) GO TO 707
275 DO 713 J=1, NCUR
276 CALL LCSCR(I, J, S, T, ITYPE)
277 CR=CURR(I, J)
278 CI=CURI(I, J)
279 AMAG=SQRT(CR*CR+CI*CI)
280 IF(AMAG.EQ.0.) APH=0.
281 IF(AMAG.EQ.0.) GO TO 713
282 APH=ATAN2(CI, CR)*57.2957795
283 713 WRITE(6,511) S, T, CR, CI, AMAG, APH
284 GO TO 706
285 711 WRITE(6,714)
286 714 FCRMAT(1H1///10X, 'FINAL CURR')
287 CALL LIST(CURR, CURI, MCUR, NCUR)
288 706 CONTINUE

C

289 ICGOUNT=NUMPAT
290 WRITE(2?1,8850) ICGOUNT, NUMTRK, NUMSKP, IPASS
291 IF(IRECT.EQ.0.) GO TO 9998
292 CALL DIRCVICORG, UORG, VORG, NORG, US, VS, CORCORG, IC, MMAX, NMAX, $DIRCRG, DIRENL)
293 WRITE(6,789) DIORG, DIRENL
294 6789 FORMAT('1 DIRORG = ',F7.2, ' DB.'/
295 $0 S DIRENL = ',F7.2, ' DB.')
296 9998 CONTINUE
IF(IDISK.EQ.C) GO TO 9997
IF(IDISK.EQ.1 .AND. ISUC.NE. 1) GO TO 9997
C DISK OUTPUT
GO 7000 J=2,35
IF(NUMSKP(J) .EQ. 0) GO TO 7001
7000 CONTINUE
C WRITE(6,7002)
C 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, SYMM, ITER, ISUC, F,HORM, IDISK,
$NOR,M,IC,(UCRG(M),VORG(M),CCOF(M),M=1,NRG),
$(US(M),VS(M),CCRCOF(M),M=1,NCUR),ITYPE,P1,P2,P3,P4,P5,P6,
$PI,TT(M),M=1,400),11,12,13,14,15,MCUR,NCUR
WRITE(6,7003) NUMPAT, J
7003 FORMATT(0 PATPATERN NUMBER ', I4, ' HAS BEEN STORED ON RECORD',
$ ', I4, ' OF ANDATA.A507C2')
9997 GO TO 9999
END

SUBROUTINE DIRCTV(CORG,UORG,VORG,NRG,US,VS,CCOF,IC,MMAX,NMAX,
1 CIRG,DIRFNL)
C THIS SUBROUTINE CALCULATES THE DIRECTIVITY OF THE ORIGINAL PATTERN
C ,CORG, AND OF THE FINAL PATTERN, DIRFNL
C DIMENSION CORG(1CC),UORG(1CO),VORG(100),US(500),VS(500),CCOF(500
$)
COMMON /LCC/,ITYPE
FORGSC=C.
FSC=0.
FMAX1=0.
FMAX2=0.
DO 10 J=1,101
U=-1.0+(J-1)*C.02
10 CONTINUE
DO 20 K=1,101
V=-1.0+(K-1)*C.02
20 CONTINUE
UVSQ=L*L+V*V
F=0.
IF(UVSC.GE.1.0) GO TO 10.

IF(NRG.LE.0) GO TO 25
DO 20 L=1,NRG
F=F+CORG(L)*PAT(U-UORG(L),V-VORG(L),ITYPE)
FORGSC=FCRGSC+F**2/SQRT(1.0-UVSQ)
25 CONTINUE
FSC = FCGRSQ
FMAX2 = FMAX1
IF (IC .LE. 0) GO TO 10
CO 30 L = 1, IC
336 F = F + CCRCCF(L) * PAT(U-US(L), V- VS(L), ITYPE)
337 FSC = FSC + F**2 / SQRT(1.0 - UVSQ)
338 IF (ABS(F) .GT. FMAX2) FMAX2 = ABS(F)
339 IC CONTINUE
340 FORGSQ = FORGSQ * 0.0004 / FMAX1
341 FSC = FSC * CC04 / FMAX2
342 DIRCRG = 4.0 * 3.14159265 / FORGSQ
343 DIRFNL = 4.0 * 3.14159265 / FSQ
C
344 DIRCRG = 10. * ALOG10(DIRORG)
345 DIRFNL = 10. * ALOG10(DIRFNL)
346 RETURN
347 END

SUBROUTINE INPUT
C
INTEGER PX, PY
REAL LX, LY, INITLS, INITLT
COMMON /PAT1/ P1, P2, P3, P4, P5, P6, PI, SS(400), TT(400), RR(400)
COMMON /PAT2/ 11, 12, 13, 14, 15
COMMON /LOC/ ITYPE
COMMON /MPROG/ MCUR, NCUR
COMMON /SYN/ LX, LY
NAMELIST /PATIN/ LX, LY, PX, PY, DISX, DISY, INITLS, DELTAS, FINALS,
$INITLT, CELTAT, FINALT, NELMT, ARAU, ITYPE, MCUR, NCUR
WRITE(6, 10)
10 FORMAT(/55X, 'SOURCE SPECIFICATIONS'//)
PI = 3.14159265
READ(5, PATIN)
IF (ITYPE .GT. 7) GO TO 990
GO TO (100, 200, 300, 400, 500, 600, 700), ITYPE
WRITE(6, 2C) ITYPE
2C FORMAT(1HO, 5X, '***ERROR*** ITYPE HAS THE VALUE ', ITYPE,'**EXE
$EXECUTION TERMINATED')
STOP
100 PI = LY
101 P2 = INITLT
102 P3 = DELTAT
LX = 0.0
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)
GO TO 999

200 P1=LY
11=PY
LY=C.C
P2=DISY
NCUR=PY
MCUR=1
WRITE(6,201) LY,PY,DISY
201 FORMAT(1X,'ITYPE=2  -- UNIFORM LINEAR ARRAY'//$15X,'LY = ',F7.3//15X,'NUMBER OF ELEMENTS = ',I3//15X,'INTER-ELEMENT SPACING = ',F6.3)
GO TO 999

300 P1=LY
LX=0.0
P2=INITLT
P3=DELTAT
NCUR=(FINALT-INITLT)/DELTAT+1.5
MCUR=1
WRITE(6,301) LY,INITLT,FINALT,DELTAT,NCUR
301 FORMAT(1X,'ITYPE=3  -- TRIANGULAR LINE SOURCE'//$15X,'LY = ',F7.3//15X,'T VARIATES FROM ',F8.4,' TO ',F6.4,5X,$'DELTAT = ',F6.3//15X,'NUMBER OF SAMPLE POINTS = NCUR = ',I3)
GO TO 999

400 P1=LX
P2=LY
P3=INITS
P4=INITLT
P5=DELTAS
P6=DELTAT
MCUR=(FINALS-INITLS)/DELTAS+1.5
NCUR=(FINALT-INITLT)/DELTAT+1.5
WRITE(6,401) LX,LY,INITLS,DELTAS,FINALS,INITLT,DELTAT,FINALT,MCUR,NCUR
401 FORMAT(1X,'ITYPE=4  -- UNIFORM RECTANGULAR APERTURE'//$15X,'DIMENSIONS = LX,LY = ',F7.4,' , ',F7.4//$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

500 P1=LX
P2=LY
11=PX
A-36

407    I2=PY
408    P3=DISX
409    P4=DISY
410    MCUR=11
411    NCUR=12
412    WRITE(6,501) LX,LY,PX,PY,DISX,DISY
413    501 FORMAT(LUX,*ITYPE=5 -- UNIFORM RECTANGULAR ARRAY**/
414    $15X,*DIMENSIONS = LX,LY = ',F7.4,' , ',F7.4**/
415    $15X,*NUMBER OF ELEMENTS = PX,PY = ',I3,' , ',I3**/
416    $15X,*INTER-ELEMENT SPACING = DISX,DISY = ',F6.3,' , ',F6.3)
417    GO TO 999

C
418

C
419

C
420

C
421

C
422

C
423

C
424

C
425

C
426

C
427

C
428

C
429

C
430

C
431

C
432

C
433

C
434

C
435

C
436

C
437

C
438

C
439

C
440

C
441

C
442

C
443
SUBROUTINE READ (F, MMAX, NMAX)
DIMENSION F(51,51), I(6), VAL(6)
CO 100, J=1, MMAX
K2=0
2CC CONTINUE
READ (5,1) (I(L), VAL(L), L=1, 6)
1 FORMAT(6(I3,F10.0))
DC 20 L=1, 6
II=I(L)
IF (II.EQ.0) GO TO 100
K1=K2+1
K2=K1+II-1
DO 10 K=K1, K2
10 F(J, K)=VAL(L)
2C CONTINUE
IF (K2.LT.NMAX) GO TO 2CC
100 CONTINUE
RETURN
END

SUBROUTINE ORG PAT (F, MMAX, NMAX, STARTU, STARTV, DELTAU, DELTAV, CURR, $CUR, MCR, NCR)
REAL F(51,51), CURR(51,51), CUR(51,51)
REAL UCRG(100), VCRG(100), CORG(100)
COMPLEX SOURCE
COMPLEX TEMP
COMMON /START/ NCRG, UCRG, VCRG, CORG
COMMON /LOC/ ICRG

THIS ORG PAT WILL BE "WOODWARD-LAWSON" INPUT.

CO 10 N=1, MMAX
DC 10 N=1, NMAX
1C F(M,N)=0.

REPRODUCIBILITY OF THE ORIGINAL PAGE IS POOR.
DO 15 M=1,MCUR
C 15 N=1,NCUR
 Curr(m,n) = 0.
15 CURI(M,N) = 0.
WRITE(6,17)
17 FORMAT(1HL,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,NORG
2 READ(5,2) US,VS,CORCOF
2 FORMAT(I5)
 DO 30 M=1,MMAX
30 U=STARTU+(M-1)*DELTAU
 DU=U-US
30 DO 40 N=1,NCUR
40 U=STARTU+(M-1)*DELTAV
 DV=V-VS
40 DO 50 M=1,MCUR
50 DO 40 N=1,NCUR
 TEMP=SOURCE(M,N,US,VS,ITYPE)
40 CURR(M,N)=CURR(M,N)*CORCOF*REAL(TMP)
40 CURI(M,N)=CURI(M,N)*CORCOF*AIMAG(TMP)
50 WRITE(6,50) IC,US,VS,CORCOF
50 FORMAT(44X, T3,5XF7.4,5X,F7.4,5X,F7.4)
50 CONTINUE
5 CONTINUE
5 RETURN

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

REAL FDES(51,51),FU(51,51),FL(51,51),F(51,51)
 REAL US(500),VS(500),CORCOF(500) UG-GL,UL,VERB(1C,CURR(1C))
 REAL LX,LY,XY
 COMMON /FAT/ NORG,US,VS,STARTU,DELTAV,STARTV,DELTAV,MCENT,NCENT,ITER,FNORM,F

REAL LX,LY,LXY
 COMMON /SYN/ LX,LY
 LXY=1./AMAX1(LX,LY)
 ITER=0
27 ITER=ITER+1
C NORMALIZE...
 FBIG=F(VCENT,NCENT)
 DO 150 M=1,MMAX
 DO 150 N=1,NCENT
150 F(M,N)=F(M,N)/FBIG
 FNORM=FNORM/FBIG
 DO 151 I=1,NORG
151 CORG(I)=CORG(I)/FBIG
 IF(IC.LT.0) GO TO 153
 LG 152 I=1,IC
152 CORCOF(I)=CORCOF(I)/FBIG
153 CONTINUE
-- ITERATION PROCEDURE --

SET IF SPECS ARE MET.

DO 24 J=1,MMAX
U=STARTU+(J-1)*DELTAV
DO 24 K=1,MMAX
V=STARTV+(K-1)*DELTAV
UVSU=U*U+V*V
IF(UVSU.LT.1.0) GO TO 24
519 IF(FDES(J,K).LT.99.) GO TO 24
520 IF(FL(J,K).LE.0.00001 .AND. ABS(F(J,K)).LE.1.E-4) GO TO 24
521 XI=ABS(F(J,K))
522 IF(XI.GT.FU(J,K)) GO TO 25
523 IF(FL(J,K).LT.99.0) GO TO 24
524 IF(XI.LT.FL(J,K)) GO TO 25
525 24 CONTINUE
526 ISC=1
527 IC=IC+1

SPECS ARE MET -- PROCEED TO PRINTOUT.

GO TO 75C

SPECS ARE NOT MET AT POINT (J,K)

IC=IC+1

IF(ITER/ICC*ICC.EQ.ITER) WRITE(6,7117) ITER
532 7117 FORMAT(10X,6,' ITERATIONS COMPLETED')
533 IF(ITER-ITRMAX) 22,22,23
534 23 WRITE(6,34) ITRMAX
535 34 FORMAT(IHO,9X,'NUMBER OF ITERATIONS EXCEEDED', IS/
$10X,'PRINTOUT OF INTERMEDIATE RESULTS FOLLOWS.')
536 GO TO 750
537 22 CONTINUE

FIND RELATIVE MAXIMUM ERROR

CALL SEARCH(J,K, VAL, FDES, FU, FL, F, MMAX, NMAX, STARTU, STARTV, DELTAV,
$DELTAV)
539 IF(VAL.NE. 0.0) GC 248

VAL EQUALS ZERO

WRITE(6,100)
541 100 FORMAT(' ERROR IN SUBROUTINE SEARCH -- VAL=0. ')
542 GO TO 750

U1=(J-1)*DELTAV+STARTU
544 248 VI=(K-1)*DELTAV+STARTV
545 IF(ABS(U1).LE.0.1*DELTAV) U1=0.
546 IF(ABS(V1).LE.0.1*DELTAV) VI=0.
547 IF(LX.EQ.0.) GO TO 1000
548 IF(U1.NE.0.0 .AND. ABS(U1).LE.0.5/LX) VAL=VAL/2.
549 1000 IF(V1.NE.0.0 .AND. ABS(V1).LE.0.5/LY) VAL=VAL/2.
IF(ISYVN.NE.4) GC TO 1001
ITEMP=0
UV=ABS(ABS(U1)-ABS(V1))
IF(UV.EQ.0.) GC TO 1001
IF(UV*1.414.LE.LXY) VAL=VAL/2.
1001 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
IF(ISYMM-2) 261,260,260
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.LT.3) GO TO 27

QUADRILATERAL SYMMETRY ONLY -- ISYMM = 3,4

582 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.LT.4. OR ITEMP.EQ.1) GO TO 27

FOR QUADRILATERAL SYMMETRY ONLY -- ISYMM = 4

ITEMP=1
IF(UI.EQ.VI) GO TO 27
IC=IC+1
UTEMP=UI
VTEMP=VI
UI=VTEMP
VI=UTEMP
GO TO 1001
598 750 CONTINUE
599 IC=IC-1
600 ITER=ITER-1
601 RETURN
602 END
603 SUBROUTINE SEARCH(I1,J1,VAL,FDES,FU,FL,F,MMAX,NMAX,STARTU,$STARTV,DELTAU,DELTAV)
604 REAL FU(51,51),FL(51,51),F(51,51),FDES(51,51)
605 VAL=0.
606 EMAX=0.
607 12=11
608 J2=J1
609 CC 10 J=J2,MMAX
610 U=STARTU+(J-1)*DELTAU
611 CC 20 K=J2,NMAX
612 V=STARTV+(K-1)*DELTAV
613 UVSQ=U*U+V*V
614 IF(UVSQ.GT.1.0) GO TO 20
615 FITER=ABS(F(J,K))
616 IF(FITER.GT.FU(J,K)) GO TO 2000
617 IF(FITER.GT.FL(J,K)) GO TO 20
618 IF(FITER.LE.1.E-4) GO TO 20
619 IF(FITER.GT.FL(J,K)) GO TO 20
620 2CCC X=FDES(J,K)
621 ERROR = FITER-X
622 IF(ABS(ERROR)-ABS(EMAX)) 20,20,21
623 21 EMAX=ERROR
624 VAL=SIGN(ERROR,F(J,K)*(X-FITER))
625 II=J
626 J1=K
627 2C CONTINUE
628 1C CONTINUE
629 WRITE(6,'(10C)') I1,J1,VAL
630 100 FCMAT(5X,'**SEARCH**',I8,I8,5X,F7.4)
631 RETURN
632 ENC
634 SUBROUTINE CHECK(IC,VAL,US,VS,COORDOF,DELTAU,DELTAV)
635 REAL US(500),VS(500),COORDOF(500)
636 IF(IC.EQ.1) RETURN
637 CU=C.1*DELTAU
638 CV=C.1*DELTAV
639 IC1=IC-1
640 U=US(IC)
641 W=VS(IC)
A-42

DO 10 J=1,IC1
643 IF(ABS(U-US(J)).LE.UA AND ABS(V-VS(J)).LE.OV) GO TO 20
644 10 CONTINUE
5
RETURN
646 20 CORCOF(J)=CORCOF(J)+VAL
647 IC=IC-1
648 RETURN
649 ENC

SUBROUTINE UPDATE(IC,US,VS,CORCOF,F,MMAX,NMAX,FNORM,STARTU,STARTV,$DELTAU,DELTAV)
CIPENSICN
FUNCTION PAT(U,V,ITYPE)
C THIS SUBPROGRAM GIVES THE BASIC CORRECTION PATTERN F(U,V).
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:
K.L. STUTZMAN DEPT. OF ELEC. ENGR. 951-6624.
E.L. C/pFFEY DEPT. OF ELEC. ENGR. 951-5494

COMPLEX TEMP,CEXP,IMAG
COMMON /PAT1/ P1,P2,P3,P4,P5,P6,P1,SS(400),IT(400),RR(400)
COMMON /PAT2/ 11,12,13,14,15

ENC
IF (ITYPE.GT.7) GO TO 990
GO TO (100,200,300,400,500,600,700), ITYPE
C
ITYPE .LT. 1
C
WRITE (6,10) ITYPE
10 FORMAT (1HC,5X,"***ERROR*** ITYPE HAS THE VALUE ",II1,":",2X,
$"EXECUTION TERMINATED")
STOP
C
ITYPE = 1 -- UNIFORM LINE SOURCE.
C
ITYPE = 2 -- UNIFORM LINEAR ARRAY
C
ITYPE = 3 -- TRIANGULAR LINE SOURCE.
C
ITYPE = 4 -- UNIFORM RECTANGULAR APERTURE
C
ITYPE = 5 -- UNIFORM RECTANGULAR ARRAY
A-44

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) 503,504,501
704    503    PAT=SIN(ARG1)/(I1*SIN(ARG1/I1))*SIN(ARG2)/(I2*SIN(ARG2/I2))
8)
705    GO TO 999
706    504    PAT=SIN(ARG1)/(I1*SIN(ARG1/I1))
707    GO TO 999
708    502    IF(ARG2) 505,506,505
709    505    PAT=SIN(ARG2)/(I2*SIN(ARG2/I2))
710    GO TO 999
711    506    PAT=1.0
712    GO TO 999
C
C ITYPE = 6 -- UNIFORM CIRCULAR APERTURE.
C
713    600    C=SCRT(U*.U+V*V)
714    A=P1
715    IF(C.EQ.C.) GO TO 601
716    X=2.*PI*PI*C
717    CALL BESJ(X,1,BJ,0.0001,IER)
718    PAT=2.*BJ/X
719    GO TO 999
720    601    PAT=1.0
721    GO TO 999
C
C ITYPE = 7 -- GENERAL ARRAY
C
721    700    IMAG=(C.C,1.C)
722    NELMT=I1*I2
723    TEMP=(C.C,C.0)
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    GO TO 999
729    990    PAT=SPECPT(U,V,ITYPE)
730    999    RETURN
731    END
C
COMPLEX FUNCTION SOURCE(M,N,U,V,ITYPE)
C
THIS SUBPRGAM 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
ITYPE = 7 -- UNIFORM LINEAR ARRAY LOCATED AT S=0.
A-45

3 -- TRIANGULAR LINE SOURCE LOCATED AT S=0.
4 -- UNIFORM RECTANGULAR APERTURE.
5 -- UNIFORM RECTANGULAR ARRAY.
6 -- UNIFORM CIRCULAR APERTURE.
7 -- GENERAL ARRAY.

ITYPE > 7 -- SPECIAL SOURCE (FUNCTION SPSOR(M,N,U,V,ITYPE) WILLL 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,P7,SS(400),TT(400),RR(400)
COMMON /PAT2/ 11,12,13,14,15

IMAG=(0.0,1.0)
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 ',I11,':',X,'EXECUTION TERMINATED')
STOP

ITYPE = 1 -- UNIFORM LINE SOURCE

1CC CONTINUE
FLEN=P1
SOURCE=CEXP(-IMAG*PI*2.*T*V)/P1
GO TO 999

ITYPE = 2 -- UNIFORM LINEAR ARRAY

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

ITYPE = 3 -- TRIANGULAR LINE SOURCE

3CC CONTINUE
FLEN=P1
CON=ABS(2.*T/P1)
751 SOURCE=2./P1*CEXP(-IMAG*2.*PI*T*V)*(1.-CON)
752 IF(CON.GT.1) SOURCE=(0.0,0.0)
753 GO TO 999

    ITYPE = 4 -- UNIFORM RECTANGULAR APERTURE

754 CONTINUE
    FLS=P1
    FLT=P2
755 SOURCE=CEXP(-IMAG*2.*PI*(S*U+V*T))/(P1*P2)
756 GO TO 999

    ITYPE = 5 -- UNIFORM RECTANGULAR ARRAY

757 CONTINUE
    FLS=P1
    FLT=P2
758 SOURCE=CEXP(-IMAG*2.*PI*(S*U+V*T))/(P1*P2)
759 GO TO 999

    ITYPE = 6 -- UNIFORM CIRCULAR APERTURE

760 CONTINUE
    RHC=SCRT(S*S+T*T)
    A=P1
761 SOURCE=(C.C,C.C,C.C)
762 IF(RHC.LE.P1) SOURCE=CEXP(-IMAG*2.*PI*(S*U+T*V))/(2.*PI*P1*P1)
763 GO TO 999

    ITYPE = 7 -- GENERAL ARRAY

764 CONTINUE
765 SOURCE=CEXP(-IMAG*2.*PI*(U*S+V*T))/(11*I2)
766 GO TO 999
767 SOURCE=SPSOR(M,N,U,V,ITYPE)
768 999 RETURN
769 END

770 SUBROUTINE LCCSOR(M,N,U,T)
771 INTEGER PX,PY
772 REAL INITLS,INITLT
773 COMMON /PAT1/ P1,P2,P3,P4,P5,P6,PI,SS(400),TT(400),RX(400)
774 COMMON /PAT2/ I1,I2,I3,I4,I5
775 COMMON /LCC/ ITYPE

776 IF(ITYPE.GT.7) GO TO 990
777 GC TO (100,200,300,400,500,600,700), ITYPE
778 WRITE(6,10) ITYPE
779 10 FORMAT(1HC,5X,'***ERROR***',111,':',2X,'EXECUTION TERMINATED')
780 STOP

C
781 1CO CONTINUE
782 INITL1=P2
783 DELTAT=P3
784 S=C.
785 T=P2+(N-L)*P3.
786 GO TO 999

787 2CO CONTINUE
788 PY=I1
789 DISY=P2
790 S=0.
791 T=(N-I1/2-1)*P2
792 IF(I1/2*2.EQ.I1) T=T+0.5*P2
793 GO TO 999

794 3CO GO TO 100

795 4CO CONTINUE
796 INITLS=P3
797 INITLT=P4
798 DELTAS=P5
799 DELTAT=P6
800 S=P3+(M-1)*P5
801 T=P4+(N-1)*P6.
802 GO TO 999

803 5CO CONTINUE
804 PX=I1
805 PY=I2
806 DISX=P3
807 DISY=P4
808 S=(M-I1/2-1)*P3
809 T=(N-I2/2-1)*P4
810 IF(I1/2*2.EQ.I1) S=S+0.5*P3
811 IF(I2/2*2.EQ.I2) T=T+0.5*P4
812 GO TO 999

813 6CO GO TO 400

814 7CO CONTINUE
815 NELMT=(M-1)*I2+N
816 S=SS(NELMT)
817 T=TT(NELMT)
818 GO TO 999

819 99C CALL SPLOC(M,N,S,T)
820 999 RETURN
821 END
SUBROUTINE CURREN(CURK,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 CURK(51,51), CURI(51,51), US(550), VS(550), CURCOF(550)
REAL UORG(100), VORG(100), CORG(100)
COMMON /START/, NORG, UORG, VORG, CORG
COMMON /LOC/, ITYPE
DO 100 M=1,MCUR
DO 100 N=1,NCUR
CURR(M,N)=0.
100 CURR(M,N)=C.
DO 200 M=1,MCUR
DO 200 N=1,NCUR
DO 200 I=1,NORG

C(100) CONTINUE
IF(IC.LT.0) RETURN

C(100) CONTINUE
DO 10 M=1,MCUR
DO 10 N=1,NCUR
DO 10 I=1,1C
THP=SOURCE(M,N,US(I),VS(I),ITYPE)
CRR(M,N)=CRR(M,N)+CURCOF(I)*REAL(THP)
CURI(M,N)=CURI(M,N)+CURCOF(I)*AIMAG(THP)
10 CONTINUE
RETURN
END

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 1C COLUMNS TO A PAGE.  72/266 VERSION 3

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
631 N2=N/10.+0.99
632 M2=M/10.+0.99
A-49

DC 100 J1=1,M2
DC 200 K=1,N2
M3=1+(J1-1)*10
M4=M3+9
IF(M4.GT.M) M4=M
N3=1+(K-1)*10
N4=N3+9
IF(N4.GT.N) N4=N

PRINT CUT 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 4000 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)
GO TO 4000
27 WRITE(6,31)
4000 CONTINUE
IF(N4.GE.N .AND. M4.EQ.M) GO TO 300
CONTINUE
CONTINUE
CONTINUE
CONTINUE
RETURN

SUBROUTINE PRCFIL(DATA1,NPT,NUMPAT)
INTEGER SF
INTEGER OUTPUT(101)
INTEGER BLANK,PLUS,SLASH,STAR
REAL DATA(401,2),BOUND(111)
REAL DATA1(401,2)
DATA BLANK,PLUS,SLASH,STAR / ' ', '+', '*', |', '* '/
47 CONTINUE

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

47 CONTINUE
A-50

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 CONTINUE
880 DIFF=ABS(BIG-SMALL)
881 SF = C
882 IF(DIFF.LT.1.) GO TO 10
883 IF(DIFF.LT.100.) GO TO 21
884 DO 2 J=1,1C
885 IF(DIFF*1C**(-J).GT.100.) GO TO 2
886 SF=-J
887 GO TO 20
888 2 CONTINUE
889 40 GO TO 20
890 100 FORMAT(10 YOUR DATA IS TOO LARGE FOR THIS PROGRAM.)
891 RETURN
892 1C DO 3 J=1,1C
893 K=11-J
894 IF(DIFF*1C**K.GT.100.) GO TO 3
895 SF=-K
896 GO TO 20
897 3 CONTINUE
898 GO TO 400
899 2C DO 4 J=1,NPT
900 4 DATA(J,2) = DATA(J,2)*10**(-SF)
C
C CALCULATE BOUNDS
C
901 21 SCALE=DIFF/100.
902 20 DO 5 J=1,1C
903 K=J-1
904 5 BCUND(J)=(BIG-K*SCALE)*10**(-SF)
C
C PRINT TITLE
C
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/)
910 200 WRITE(6,65C) (BOUND(J),J=1,101,20)
911 65C FORMAT(1X,F7.3,13X),F7.3,2X,'REAL',5X,'C3',5X)
912 DO 6 J1=1,NPT
913 J=NPT+1-J1
914 6 CC 50 K=1,101
915 50 CUTPUT(K)=BLANK
916 IF((J-1)/IC*10-(J-1)) 62,61,62
917 61 CC 40 K=1,101,10
918 40 CUTPUT(K)=PLUS
919 GO TO 87
920 62 CUTPUT(1)=SLASH
921 CUTPUT(1C1) = SLASH
SUBROUTINE CCNTUR(K, L, DCON, CLow, CMax, CINT, A, NUMPAT)

C**************************************************************************
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 CCLOW=LOWEST CONTOUR LEVEL
C CCONMAX=HIGHEST CONTOUR LEVEL
C CINT=CONTOUR INTERVAL
C NUMPAT=PATTERN NUMBER
C**************************************************************************

DIMENSION A(51,51)
DIMENSION ALPHA(10)
DIMENSION CCL(101)
DATA ALPHA/IHC, IH1, IH2, IH3, IH4, IH5, IH6, IH7, IH8, IH9/
DATA BLANK, OUT/1H, 1H/*

IF(K.LE.1.OR.L.LE.1) RETURN
CINT=CINT
CCLOW=CLow
CCONMAX=CMax
WRITE(6,87) NUMPAT

87 FCPRAT(1HC,'FCR THE PATTERN NUMBERED', I5)
IF(CINT) 99,99,100

99 BIG=-1.E27
SMALL=1.E27
GO 98 1=1,K
GO 98 J=1,L
IF(A(I,J) .GT. BIG) BIG=A(I,J)
IF(A(I,J) .LT. SMALL) SMALL = A(I,J)

98 CONTINUE
A-52

962 CONINT=(BIG-SMALL)/10.
963 DELCON=C.5*CONINT
964 CCLOW=SMALL+DELCON
965 CONMAX=BIG-DELCON
966 WRITE(6,71) DELCON,CCLOW,CONMAX,CONINT
967 FORMAT(1HO, *DELCON=",F10.5,3X,*CCLOW=",F10.5,3X,*CONMAX=",F10.5)
968 71 OELCON=C.5*CCNINT
969 CCLOW=SPALL+DELCON
970 CONMAX=BIG-DELCON
971 WRITE(6,72) CCNLOW,CONMAX,CONINT
972 FORMAT(1HO, 'CONLOW=',F10.5,3X,'CONMAX=',F10.5,3X,'CONINT=',F10.5)
973 PRINT LEVEL DESIGNATIONS
974 NCHAR=ABS((CCNMAX-CONLO)/CONINT+1.1)
975 CON=CONMAX+CONINT
976 ICON=M-1
977 ICON=CON-CONINT
978 WRITE(6,72) ICON,CON
979 FORMAT(1HC,'CONTOM LEVEL ','12,=*',F10.5)
980 CONTINUE
981 WRITE(6,200)
982 EXIT
983 GO TO 34
984 CONTINUE
985 GO TO 34
986 CONTINUE
987 CONTINUE
988 CONTINUE
989 WRITE(6,200)
990 FORMAT(1HC)
991 N1=L*2
992 IF(N1.GT.101) N1=101
993 WRITE(6,101) (CCL(JI),J1=1,N1)
994 101 FORMAT(/1HC,14X,101Al)
995 CO 1 I=1, K
996 CC 31 J=1,101
997 CCL(J)=BLANK
998 CONTINUE
999 J2=1
1000 GO 2 J=1, L
1001 J2=J2+2
1002 ICEN=-1
1003 CON=CONMAX+CONINT
1004 CO 50 M=1,MCHAR
1005 ICEN=ICON+1
1006 CON=CON-CONINT
1007 IF(A(I,J).GT.(CON+DELCON)) GO TO 50
1008 IF(A(I,J).LT.(CON-DELCON)) GO TO 50
1009 GO TO 2
SUBROUTINE PATCON(RDATA, MMAX, NMAX, ICODE, CONLOW, CONMAX, CONINT, \$ STARTU, STARTV, DELTAU, DELTAV, NUMPAT, ISYMM)

REAL RDATA(51, 51), UAXIS(11), LOW(12), HIGH(12)
INTEGER CUTPUT(101), LEVEL(12), BLANK

DATA BLANK/' '/
DATA LEVEL/"C", "1", "2", "3", "4", "5", "6", "7", "8", "9", "+"/

CALL DATE(I, J, K)
WRITE(6, 10) I, J, K, NUMPAT
FORMAT(1H1, 'PATTERN CONTOUR SUBPROGRAM', 34X, 'DATE = ', A2, '-', A2 
$, '-', A2, 3CX, 'PATTERN NUMBER', 15////)
IF(ICCDE.EQ.0) WRITE(6, 11)
IF(ICCDE.EQ.1) WRITE(6, 12)
IF(ICCDE.EQ.2) WRITE(6, 13)

10 FORMAT(42X, 'CONTOUR PLOT OF THE DESIRED PATTERN'////)
11 FORMAT(46X, 'CONTOUR PLOT OF THE INITIAL PATTERN'////)
12 FORMAT(45X, 'CONTOUR PLOT OF THE FINAL PATTERN IN DB.'////)

FINALU = STARTU + (MMAX - 1) * DELTAU,
FINALV = STARTV + (NMAX - 1) * DELTAV
U1 = STARTU
U2 = FINALU
V1 = STARTV
V2 = FINALV
MCOUNT = MMAX
NCOUNT = NMAX

IF(ISYMM-1) 70, 30, 20
2C UBIG = AMAX1(ABS(STARTU), ABS(FINALU))
U1 = -UBIG
U2 = UBIG
MCOUNT = 2 * MCOUNT - 1
IF(ISYMM.EQ.2) GO TO 70

30 VBIG = AMAX1(ABS(STARTV), ABS(FINALV))
V1 = -VBIG
V2 = VBIG
NCOUNT = 2 * NCOUNT - 1

70 CONTINUE

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

DO 71 J=1,NUMCON
    LOW(J)=CCNL+(J-1)*CONINT-DELCON
    HIGH(J)=LOW(J)+CONINT+0.0001
    71 CONTINUE

LOW(11)=-1.0E30
HIGH(12)=1.0E30
HIGH(11)=LOW(1)
LOW(12)=HIGH(NUMCON)
MSIP=100/(MCOUNT-1)
NSIP=100/(NCOUNT-1)

CU=(U2-U1)/10.
DO 40 I=1,11
    UAXIS(I)=U1+(I-1)*DU
    WRITE(6,42) (UAXIS(I),I=1,11)
42 FORMATT(13X,11(F7.4,3X)/16X,11(‘,’9X))

DU=(U2-U1)/100.
CV=(V2-V1)/100.
N1=NSKIP-1
DO 50 N=1,101,NSKIP
    V=V2-(N-1)*CV
    DO 51 K=1,101
51 OUTPUT(K)=BLANK
    DC 60 M=1,101,NSKIP
    U=U1+(K-1)*DU
    IF(U*U+V*V.GT.1.0) GO TO 60
FIND F(U,V)

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

IF(IJ) 2CC,1CC,2CC
201 IF(IK) 3CC,1CC,3CC

2CC IF(ASYM.EQ.1) 202,2CC
202 IF(IK) 3CC,1CC,300

300 IF(ISYMM.EQ.0 .OR. ISYMM.EQ.2) GO TO 60
K=1.5-(V-STARTV)/DELTAV
IF(K.GE.1 .AND. K.LE.NMAX) IK=0
IF(IK) 60,1CC,60

3CC F=RDATA(J,K)
IF(F.LE.LOW(1)) GO TO 1001
IF(F.GT.HIGH(NUMCON)) GO TO 1002
DO 61 K=1,NUMCON
IF(F.GT.L(W(K)) .AND. F.LE.HIGH(K)) GO TO 62
61 CONTINUE
1002 OUTPUT(K)=LEVEL(12)
1003 GO TO 60
1001 OUTPUT(K)=LEVEL(11)
1004 GO TO 60
1005 62 OUTPUT(M)=LEVEL(K)
1006 CONTINUE
WRITE(6,64) (OUTPUT(K),K=1,101)
64 FORMAT(7X,F7.4,1X,' ',101AI,' ',1X,F7.4)
IF(N1.EQ.0) GO TO 50
50 CONTINUE
WRITE(6,56)
56 FORMAT(' ')
55 CONTINUE
WRITE(6,56) (UAXIS(I),I=1,1L)
56 FORMAT(///5X,'CONTOUR LEVEL KEY'//)
10 CONTINUE
WRITE(6,43) (LEVEL(J),LOW(J),HIGH(J),J=1,12,4)
43 FORMAT(5X,3(A1,':',E14.7,12X,'TO',E14.7,4X))
RETURN
ENC

SUBROUTINE LIST(CURR,CURI,MCUR,NCUR)
C
C THIS SUBROUTINE LISTS ARRAY ELEMENT COORDINATES AND CURRENTS
C
C
DIMENSION CURR(51,51),CURI(51,51)
10 CONTINUE
CALL LCCSCR(M,N,S,T)
WRITE(6,1CC) J,S,T,CURR(M,N),CURI(M,N)
1CC CONTINUE
WRITE(6,44) (LEVEL(J),LOW(J),HIGH(J),J=1,12,4)
44 FORMAT(///56X,'CONTOUR LEVEL KEY'//)
10 CONTINUE
WRITE(6,44) (UAXIS(I),I=1,1L)
44 FORMAT(///56X,'CONTOUR LEVEL KEY'//)
10 CONTINUE
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, FPECPT, SOURCE, SPSOR, LOCOR, 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, MCUR, 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 4Il 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 4Il 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 OPlU, 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, LOCSOR, and SPLOC 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 "+U" and "-U" 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:

i. 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.

ii. 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 PLOTIP

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 PLOTIP (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:

i. 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.
COMMON Blocks Required: COMMON /ARRAY/ X

Subroutine and Function Subprogram Required: None.

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, U, SH, SV
COMMON /THREE7/ SL, SM, SN, CX, CY, CZ, QX, QY, QZ, SD
Subroutine and Function Subprograms Required: THREE2, 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, M, P, KODE)
Description of Parameters:

\( \text{XI} \) - Abscissa of the projected point.

\( \text{YJ} \) - Ordinate of the projected point.

\( \text{M} \) - Number of horizontal points.

\( \text{N} \) - Number of vertical points.

\( \text{P} \) - PLOT CODE; IF \( P = -1 \) INVISIBLE TO VISIBLE
1 VISIBLE TO INVISIBLE
0 VISIBLE TO VISIBLE OR
INVISIBLE TO INVISIBLE.

\( \text{KODE} \) - 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

Subprograms Purpose

\text{DATE} \quad \text{To return the current month, day, and year.}

\text{STIME} \quad \text{To return the time of day in ten thousandths of an hour (Integer Format)}

\text{TIMEON} \quad \text{To set the interval timer to zero}

\text{TIMECK} \quad \text{To return the amount of CPU time used in hundredths of seconds since the last call to TIMEON.}

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

VPI PLOTTER SUBROUTINES

Subroutine Purpose

\text{AXIS} \quad \text{To draw a labeled axis of a desired length with annotated tic marks every inch.}
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 alphanumeric 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=CALCAMP
// EXEC FORTGCG,PARM.GO='PAPER=39,PTIME=119',EP=MAIN
//FORT.SYSLIN DD *
/*
//GO.SYSLIB DD
// DD DSN=VPI.PLOTLIB,DISP=SHR
// DD DSN=VPI.SSPLIB,DISP=SHR
//GO.SYSLIN DD
// DD *
/*
//GO.FT22FOO DD DSN=ANTDATA.A507C2,UNIT=3330,VOL=SER=tJSERPK,DISP=SHR
//GO.FT06FOO DD SYSOUT=A,DCB=(BLKSIZE=133,RECFM=F)
//GO.SYSLIN 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 O.D.), 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 (PLOT).

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 A CHARACTER FORMAT.

DEFINE FILE 22(35,9100,E,NREC)
DIMENSION A(151,151),PTS(4001),US(500),VS(500),Cовор(500)
DIMENSION UREG(100),VREG(100),CREG(100)
DIMENSION AU(151),AV(151)
INTEGER TITLE(20)
REAL NITLS,NITLT
REAL LOWCON
INTEGER OPT1V,OPT2,OPT3,PX,PY
COMPLEX CTEMP,CI
COMMON /PLT/ PTS
COMMON /ARRAY/ A
COMMON /PAT1/ P1,P2,P3,P4,P5,P6,P7,SS(400),TT(400),RR(400)
COMMON /PAT2/ I1,I2,I3,I4,I5
COMMON /LOC/ ITYPE
IPAGE=0
PI=3.14159265
CI=CMPLX(0.0,1.0)
CALL STIME(ITIME)
TIME=0.
9999 CONTINUE
A-76

CALL TIMECN

IPAGE=IPAGE+1

CALL DATE(J1,JI,K1)

CALL STIME(1T)

IHR=IT/10000

IFR=IT-IHR*10000

FHR=IFR/10000.

FM=FHR*60.

ISEC=(FM-I1MIN)*60

IHR=ICVT(IHR)

IMIN=ICVT(IMIN)

ISEC=ICVT(ISEC)

IPG=ICVT(IPAGE)

WRITE(6,1) II,J1,K1,IHR,IMIN,ISEC,IPG

1 FORMAT(1H1,2X,"ANTDATA I VERSION 1 LEVEL 2", 28X,"VPI EE DEPT. ",5X,"DATE = ",2A2,",",A2","",",A2,",",",A2, 
25X,"TIME = ",2A2,",",A2,",",",A2,",",",A2,9X,",PAGE 00",2A2, 

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",I5,

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

47 IF(NUM.EQ.NUMPAT) GO TO 50

30 CONTINUE

NUMPAT IS NOT ON DISK

WRITE(6,60) NUMPAT

60 FORMAT(" PATTERN NUMBER",I5," WAS NOT LOCATED -- PROGRAM HALT")

GO TO 999

NUMPAT FOUND ON UNEXPECTED TRACK

50 WRITE(6,60) NUMPAT,NUMTRK,I

60 FORMAT(" PATTERN NUMBER",I5," WAS NOT FOUND ON TRACK",I2,

" BUT WAS LOCATED ON TRACK",I2)

GO TO 999

NUMTRK=I

51 CONTINUE

C RESTART PROCESSING

READ(5,10) NMAX,NMAX

READ(22*NUMTRK,7C) ITEMPT,ITEMPL

70 CONTINUE

TO FORMAT(104X,2A4)
A-77

59 READ(22,NUMTRK,101) NUMPAT,TITLE,ISYMM,ITER,ISUC,FNORM,IVISK, 
$NCRG, IC, (UGR(J),VRG(J),CORG(J),J=1,ITEMP), 
$(US(J),VS(J),CRCOF(J),J=1,ITEMP1),ITYPE,P1,P2,P3,P4,P5,P6, 
$PI, (SS(J),IT(J),J=1,400), 11,12,13,14,15,MCUR,NCUR

60 FORMAT(75A4,11(200A4))
C
C READ OPTIONS FOR PATTERN MAGNITUDE
C
61 READ(5,29) OPTIU,OPTIV,OPT2,OPT3
62 29 FORMAT(411)
63 IF(OPTIU-1) 80,81,60
64 81 CONTINUE
65 READ(5,31) CONST
66 IF(MMAX.LE.1) GO TO 80
67 CO 90 J=1,4001
68 U=(J-1)*0.0005-1.0
69 SUM=0.
70 CO 91 K=1,NORC
71 SUM=SUM+CORG(K)*PAT(U-UORG(K),CONST-VORG(K),ITYPE)
72 IF(UC.LE.C) GO TO 90
73 CO 91 K=1,IC
74 SUM=SUM+CRCOF(K)*PAT(U-US(K),CONST-VS(K),ITYPE)
75 91 CONTINUE
76 PTS(J)=2C.*ALOG10(ABS(SUM))
77 WRITE(6,92) CONST
78 92 FORMAT('CU-AXIS PROFILE PLOT REQUESTED -- V = ',F6.3)
79 CALL PLOT1(-1.0,1.0,4001.2,CONST,NUMPAT)
80 IF(OPTIV-1) 82,83,62
81 83 CONTINUE
82 READ(5,31) CONST
83 IF(MMAX.LE.1) GO TO 82
84 CO 900 J=1,4001
85 V=(J-1)*0.0005-1.0
86 SUM=0.
87 CO 909 K=1,NORC
88 SUM=SUM+CORG(K)*PAT(CONST-UORG(K),V-VORG(K),ITYPE)
89 IF(VC.LE.0) GO TO 900
90 CO 910 K=1,IC
91 SUM=SUM+CRCOF(K)*PAT(CONST-US(K),V-VS(K),ITYPE)
92 CO 910 K=1,IC
93 WRITE(6,93) CONST
94 93 FORMAT('CV-AXIS PROFILE PLOT REQUESTED -- U = ',F6.3)
95 CALL PLOT1(-1.0,1.0,4001.2,CONST,NUMPAT)
96 82 IF(OPT2+OPT3) 85,85,84
97 84 CONTINUE
C
C GENERATE PATTERN ARRAY
C
98 READ(5,31) LOWCON,DASH
99 IF(MMAX.LE.1 OR NMAX.LE.1) GO TO 239
100 DELTAG=2.0/(MMAX-1)
101 DELTAV=2.0/(NMAX-1)
102 WRITE(6,701) LOWCON,LOWCON
103 701 FORMAT('OPATTERN IS NOW BEING GENERATED. IF PATTERN < ',F7.2,
104 90 PATTERN = ',F7.2)
105 IF(ITYPE.GT.5) GO TO 5000
C LOAD UP AU AND AV

C
DO 2000 I=1,NMAX
U=(I-1)*DELTAU
C 2000 AU(I)=PAT(U,0.,ITYPE)
DO 2010 J=1,NMAX
V=(J-1)*DELTAV
C 2010 AV(J)=PAT(C-,V,ITYPE)
C BEGIN
C
U=-1.0-DELTAU
DO 2040 M=1,NMAX
U=U+DELTAU
V=-1.0-DELTAV
DO 242 I=1,NORG
U=-1.0-(M-1)*DELTAU
DO 201 N=1,NMAX
V=-1.0+(N-1)*DELTAV
TEMP=0.
DO 242 TEMP=TEMP+CORG(I)*AU(I)*AV(J)
IF(1C.LE.0) GO TO 2020
DO 2040 K=1,NORG
I=ABS(U-UORG(K))/DELTAU+1.5
J=ABS(V-VORG(K))/DELTAV+1.5
2030 TEMP=TEMP+CORG(K)*AU(I)*AV(J)
IF(1C.LE.0) GO TO 2020
DO 242 I=1,NORG
242 TEMP=TEMP+CORG(K)*AU(I)*AV(J)
CONTINUE
DO 2040 M=1,NMAX
U=-1.0+(M-1)*DELTAU
DO 201 N=1,NMAX
V=-1.0+(N-1)*DELTAV
TEMP=0.
DO 242 TEMP=TEMP+CORG(K)*AU(I)*AV(J)
IF(1C.LE.0) GO TO 2020
DO 202 I=1,IC
TEMP=TEMP+CORCOF(K)*AU(I)*AV(J)
CONTINUE
DO 202 A(M,N)=20.*ALOG10(ABS(TEMP))
GO TO 239
CONTINUE
DO 2000 M=1,NMAX
U=-1.0+(M-1)*DELTAU
DO 201 N=1,NMAX
V=-1.0+(N-1)*DELTAV
TEMP=0.
DO 242 TEMP=TEMP+CORG(I)*PAT(U-UORG(I),V-VORG(I),ITYPE)
IF(1C.LE.0) GO TO 2021
DO 202 I=1,IC
202 TEMP=TEMP+CORCOF(I)*PAT(U-US(I),V-VS(I),ITYPE)
CONTINUE
A(M,N)=20.*ALOG10(ABS(TEMP))
CONTINUE
CONTINUE
IF(IUNIT .GE. 210) WRITE(6,210)
210 FORMAT(6FI0.0)
IF(IUNIT .GE. 210) 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
220 FORMAT('CONTOUR PLOT OF PATTERN REQUESTED')
   $1  ' HIGHEST CONTOUR = ',F7.2/
   $1  ' CONTOUR INTERVAL = ',F7.2/
   $1

   CALL PLOT2(MMAX,NMAX,CONLOW,CONMAX,CONINT,NUMPAT,DASH)

6   231 IF(OPT3) 230,230,231
157   231 WRITE(6,24C)
158   240 FORMAT(1HO,'THREE-DIMENSIONAL PLOT OF PATTERN REQUESTED')
159   CALL PLOT3(PMAX,NMAX,NUMPAT)
160   23C CONTINUE
161   23C IF(MMAX.LE.1.OR.NMAX.LE.1) WRITE(6,23)
162   23C FORMAT('TWO AND THREE DIM. PLOTS CANCELLED SINCE SOURCE IS
163   85 CONTINUE

C END OF PATTERN

C

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

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=1) 404,403,404
177 404 IF(ITYPE=3) 405,403,405
178 403 CONTINUE

C ITYPE=1 OR 3
179 INITIAL=0.
180 DELTAS=0.
181 FINALS=0.
182 INITLT=P2
183 FINALT=P2+P1
184 P3=P1/4CC0.
185 DELTAT=P3
186 GO TO 410
187 405 IF(ITYPE=4) 407,406,407
188 406 CONTINUE

C ITYPE=4
189 INITIAL=P3
190 FINALS=P3+P1
191 INITLT=P4
192 FINALT=P4+P2
193 P5=P2/4CC0.
194 P6=P2/4000.
195 DELTAT=P6
196 DELTAS=P5
197 GO TO 410
198 407 CONTINUE
199 IF(ITYPE=6) 410,409,410
READ(5,29) OPTIU,OPTIV,OPT2,OPT3
CONTINUE

IF(OPTIU .EQ. 1) 302,301,302
CONTINUE

READ(5,31) CONST
IF(IA .EQ. 1) GO TO 3000
IF(IA .EQ. 1) GO TO 302
J=1

IF(DELTAT .NE. 0.0) J=1.5+ (CONST - INITLT) / DELTAT
DO 303 I=1,4001
CTEMP= (0.0,0.0)
IF(NORG .LE. 0) GO TO 304
DO 305 K=1,NORG
305 CTEMP = CTEMP + CORG(K) * SOURCE(I,J,UORG(K),VORG(K),ITYPE)
304 CONTINUE

IF(NCRG .LE. 0) GO TO 303
303 IF(INV .LE. 0) GO TO 300
300 CONTINUE

CONTINUE

IF(OPTIV .EQ. 1) 311,310,311
CONTINUE

READ(5,31) CONST
IF(IA .EQ. 1) GO TO 3000
IF(INV .LE. 0) GO TO 322
I=1

IF(DELTAS .NE. 0.0) I=1.5+ (CONST - INITLS) / DELTAS
DO 313 J=1,4001
CTEMP= (0.0,0.0)
IF(NORG .LE. 0) GO TO 314
DO 315 K=1,NORG
315 CTEMP = CTEMP + CORG(K) * SOURCE(I,J,UORG(K),VORG(K),ITYPE)
314 CONTINUE

IF(NCRG .LE. 0) GO TO 313
313 IF(INV .LE. 0) GO TO 322
322 CONTINUE

CONTINUE
A-81

257 IF(OPT2+OPT3) 320,320,321
258 321 CONTINUE
259 READ(5,31) LOWCON,DASH
260 IF(IA.EQ.1) GO TO 333

C GENERATE CURRENT MAGNITUDE ARRAY

261 DO 333 M=1,MCUR
262 DO 331 N=1,NCUR
263 CALL LOCSOR(M,N,S,T)
264 CTEMP=0.
265 DO 339 K=1,NORG
266 CTEMP=CTEMP+CORG(K)*SOURCE(M,N,UORG(K),VORG(K),ITYPE)
267 DO 332 K=1,1C
268 CTEMP=CTEMP+CORCOF(K)*SOURCE(M,N,US(K),VS(K),ITYPE)
269 A(K,N)=CABS(CTEMP)
270 331 CONTINUE
271 333 CONTINUE
272 333 CONTINUE
273 IF(OPT2) 350,350,351
274 351 READ(5,31) CONLOW,CONMAX,CONINT
275 IF(IA.EQ.1) GO TO 360
276 IF(VMAX.LE.1.OR.NMAX.LE.1) GO TO 360
277 WRITE(6,340) CONLOW,CONMAX,CONINT
278 34C FORMAT('CURRENT MAGNITUDE REQUESTED'/
$' LOWEST CONTOUR = ',F7.4/
$' HIGHEST CONTOUR = ',F7.4/
$' CONTOUR INTERVAL = ',F7.4)
279 CALL PLOT2 (MCUR,NCUR,CONLOW,CONMAX,CONINT,NUMPAT,DASH)
280 350 IF(OPT3) 360,360,361
281 361 IF(IA.EQ.1) GO TO 360
282 WRITE(6,355)
283 355 FORMAT('THREE DIMENSION PLOT OF CURRENT MAGNITUDE REQUESTED')
284 CALL PLOT3 (MCUR,NCUR,NUMPAT)
285 36C CONTINUE
286 IF(VMAX.LE.1.OR.NMAX.LE.1) WRITE(6,23)
287 32C CONTINUE

C END OF CURRENT MAGNITUDE

C READ OPTIONS FOR CURRENT PHASE

288 READ(5,29) CPT1U,OPTIV,OPT2,OPT3
289 IF(OPT2+OPT3) 520,520,521
290 521 CONTINUE
291 IF(IA.EQ.1) GO TO 533
292 READ(5,31) LOWCON,DASH

C GENERATE CURRENT PHASE

293 DO 530 M=1,MCUR
294 DO 531 N=1,NCUR
295 CALL LOCSOR(M,N,S,T)
296 CTEMP=0.
297 DO 549 K=1,NORG
298 CTEMP=CTEMP+CORG(K)*SOURCE(M,N,UORG(K),VORG(K),ITYPE)
CONTINUE
DO 532 K=1,IC
CTEMP = CTEMP + CORCOF(K) * SOURCE(M, N, US(K), VS(K), ITYPE)
CREAL = REAL(CTEMP)
CIMAG = AIMAG(CTEMP)
A(M, N) = ATAN2(CIMAG, CREAL) * 180. / PI
CONTINUE
CONTINUE
CONTINUE
IF (OPT2) 550, 550, 551
READ(5, 31) CONLOW, CONMAX, CONINT
IF (IA.EQ.1) GO TO 560
IF (PMAX .LE. 1.0 OR NMAX .LE. 1.0) GO TO 560
WRITE(6, 560) CONLOW, CONMAX, CONINT
CONTINUE
54C FORMAT (' CONTOUR PLOT OF CURRENT PHASE REQUESTED /
', ' ' 'LOWEST CONTOUR = ', F7.2/
', ' ' 'HIGHEST CONTOUR = ', F7.2/
', ' ' 'CONTOUR INTERVAL = ', F7.2)
CALL PLOT2 (MCUR, NCUR, CONLOW, CONMAX, CONINT, NUMPAT, DASH)
55C IF (OPT3) 560, 560, 561
561 IF (IA.EQ.1) GO TO 560
WRITE(6, 555)
555 FORMAT (' THREE DIMENSION PLOT OF CURRENT PHASE REQUESTED')
CALL PLOT3 (MCUR, NCUR, NUMPAT)
CONTINUE
IF (PMAX .LE. 1.0 OR NMAX .LE. 1.0) WRITE(6, 23)
52C CONTINUE
CALL TIMECK(ISEC)
FMIN = ISEC / 6000.
WRITE(6, 897) FMIN
897 FORMAT (' EXECUTION TIME: ', F7.2, ' MINUTES.')
TIME = TIME + FMIN
GO TO 9999
999 WRITE(6, 600)
60C FORMAT (1HI, '*** END OF EXECUTION ***')
CALL PLOT(0.0, 0.0, 0, -4)
WRITE(6, 898) TIME
898 FORMAT (' TOTAL EXECUTION TIME: ', F7.2, ' MINUTES.')
CALL STIME(JTIME)
IT = JTIME - ITIME
FMIN = IT / 10000. * 60.
WRITE(6, 899) FMIN
899 FORMAT (' TOTAL ELAPSED TIME: ', F7.2, ' MINUTES.')
STCP
END

SUBROUTINE PLOT1 (PSTRT, PEND, IP, CODE, CONST, NUMPAT)

SUBROUTINE PLOT1

WRITTEN BY: S. R. KAUFMAN

DATE: 73-113 APRIL 23, 1973
INPUT:

PSTRT. -- 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.

342 INTEGER NAME(2),CODE
343 DIMENSION PTS(4001)
344 COMMON /PLT1/ PTS
345 CALL FACTOR(0.5)
346 CALL PLOT(8.91.,-3)
347 IF(CODE.GT.0) GO TO 3
348 CALL SYMBOL(-1.2,-6,2,8,8HTHETA = ,0.,8)
349 CALL NUMBER(-.3,-8,2,CONST,0.,3)
GO TO 6
350 3 IF(CODE.GT.1) GO TO 4
351 CALL SYMBOL(-1.7,-8,2,1HU,0.,1)
352 CALL SYMBOL(-.9,-8,2,3H = ,0.,3)
353 CALL NUMBER(-.2,-8,2,CONST,0.,3)
354 CALL SYMBOL(-2.6,-4,2,2H-V,0.,2)
355 CALL SYMBOL(2.4,-4,2,2H+V,0.,2)
356 CALL SYMBOL(-1.2,-6,2,1HU,0.,1)
357 GO TO 6
358 4 IF(CODE.GT.2) GO TO 5
359 CALL SYMBOL(-1.7,-8,2,1HU,0.,1)
360 CALL SYMBOL(-.9,-8,2,3H = ,0.,3)
361 CALL NUMBER(-.2,-8,2,CONST,0.,3)
362 CALL SYMBOL(-2.6,-4,2,2H-U,0.,2)
363 CALL SYMBOL(2.4,-4,2,2H+U,0.,2)
6 CONTINUE
365 PDEL=(PSTRT-PEND)/IP
366 PTIC=(ABS(PSTRT-PEND))/10.
367 CALL AXIS(-5.00,1H *1.4*0.,PSTRT,PTIC)
368 PSTRE=PSTRT*(6.9*PTIC)+.00001
369 PTIC2=PTIC*0.00001
370 CALL AXIS(1.0,1H *1.4*0.,PSTRE,PTIC2)
371 CALL PLOT(-1.0,0.,3)
372 CALL PLOT(1.0,0.,2)
373 CALL PLOT(0.,0.,3)
374 CALL PLOT(0.,0.,2)
375 CALL PLOT(0.,0.,3)
376 CALL SYMBOL(-.05,-.4,2,1HU,0.,1)
377 X=0.05
378 CC 10 J=1,6
379 Y=0.5*(J-1)*1.0
380 CALL PLOT(-X,Y,3)
381 10 CALL PLOT(X,Y,2)
382 CALL PLOT(6.,0.,0.,3)
383 IF(PTS(1).LE.-50.) PTS(1)=-50.
384 FS=((PTS(1))/10.)*5.5
SUBROUTINE PLOTIC(PSTRT,PEND,IP,CODE,CONST,NUPPAT)

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 = ';
        IF CODE=1: LABEL IS 'S = '; IF CODE=2: LABEL IS 'T = '
CONST -- CONSTANT PARAMETER FOR LABEL.
NUPPAT -- PATTERN NUMBER FOR LABEL.

INTEGER NAME(2), CODE
CALL FACTOR(0.5)
CALL PLOT(8.0, 1.0, -3)
DIMENSION PTS(4001)

CCMCMN /PLTI/ PTS

I=C
IF(CODE.GT.0) GO TO 3
CALL SYMBOL(-1.2,-0.6,0.2,8H\THETA = ,0.0,8)
CALL NUMBER(C.3,-0.8,0.2,CONST,0.0,3)
GO TO 6

3 IF(CODE.GT.1) GO TO 4
CALL SYMBOL(-1.0,-0.8,0.2,1H\S,0.0,1)
CALL SYMBOL(-0.9,-0.8,0.2,3H ,0.0,3)
CALL NUMBER(-0.2,-0.8,0.2,CONST,0.0,3)
CALL SYMBOL(-2.6,-0.4,0.2,2H\T,0.0,2)
CALL SYMBOL(2.4,-0.4,0.2,2H+T,0.0,2)
A-85

GO TO 6

416 IF(CODE.GT.2) GO TO 5

417 CALL SYMFCL(-1.0,-0.8,0.2,1H,0.0,1)

418 CALL SYMBOL(-0.9,-0.8,0.2,3H = ,0.0,3)

419 CALL NUMBER(-0.2,-0.8,0.2,CONST,0.0,3)

420 CALL SYMBOL(-2.6,-0.4,0.2,2H=5,0.0,2)

421 CALL SYMBOL(-2.4,-0.4,0.2,2H=S,0.0,2)

422 CONTINUE

423 PCEL=(PSTRT-PEND)/IP

424 PTIC=ABS(PSTRT-PEND)/10.0

425 CALL AXIS(-5.0C,C.0,IH,C,E,4.0,0.0,PSTRT,PTIC)

426 PSTRE=PSTRT+(6.0*PTIC)+0.00001

427 PTIC2=PTIC+0.00001

428 CALL AXIS(1.0,C.0,1H,C,E,1,4.0,0.0,PSTRE,PTIC2)

429 CALL PLOT(-1.0,0.0,0.0,3)

430 CALL PLOT(1.0,0.0,0.0,2)

431 CALL PLOT(0.0,0.0,3)

432 CALL PLOT(0.0,5.8,3)

433 CALL PLOT(0.0,5.8,2)

434 CALL PLOT(-1.0,0.0,3)

435 CALL SYMBOL(-0.50,0.5,0.2,1H,0.0,1)

436 X=0.05

437 CALL PLOT(X,Y,3)

438 CALL PLOT(-X,Y,3)

439 10 CALL PLOT(X,Y,2)

440 CALL PLOT(C.0,0.0,3)

441 PSTRS=5.0*C*PSTRT

442 GMAX=0.0

443 DO 11 IW=1,IP

444 IF(Pts(IW).GT.GMAX) GMAX=Pts(IW)

445 1 CONTINUE

446 IF(GMAX.GT.0.5) ASCLE=1.0

447 IF(GMAX.GT.0.5) ASCLE=0.5

448 IF(GMAX.GT.0.5) ASCLE=0.2

449 IF(GMAX.GT.0.5) ASCLE=0.1

450 IF(GMAX.LE.0.5) ASCLE=0.05

451 IF(GMAX.LE.0.5) ASCLE=0.02

452 APTS=((PTS(IW))/ASCLE)*5.+0.5

453 CALL PLOT(-5.0,PTS,3)

454 DO 7 IW=1,IP

455 THETA=(PSTRT-(IW1*PDEL))

456 THETS=(THETA-(ABS(PSTRT)))*5.

457 APTS=((PTS(IW1))/ASCLE)*5.+0.5

458 CALL PLOT(THETS,APTS,2)

459 7 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.GT.0.5) ATIC=0.04+0.0001

463 IF(GMAX.GT.0.5) ATIC=0.02+0.0001

464 IF(GMAX.GT.0.5) ATIC=0.01+0.0001

465 CALL AXIS(-5.5,0.5,16H SOURCE MAGNITUDE,16,5.0,90.0,0.0,ATIC)

466 CALL SYMBOL(-5.0,-0.8,0.2,125,10HPATTERN = ,0.,10)

467 FNUM=FLOAT(NUMPAT)

468 CALL NUMBER(-3.5,-0.8,C.125,FNUM,0.,-1)

469 CALL PLOT(6.-1.,-3)

470 RETURN

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 PLOT(8., 1., -3)
IF(CODE.GT.0) GO TO 3
CALL SYMBOL(-1.2, -6., 2, '8H\theta = , 0., 8)
CALL NUMBER(.3, -.8, 2, CONST, 0., 3)
GO TO 6
3 IF(CODE.GT.1) GO TO 4
CALL SYMBOL(-1.0, -.8, .2, '1HS\theta, 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-T, 0., 2)
CALL SYMBOL(2.4, -.4, 2, '2H+T, 0., 2)
GO TO 6
4 IF(CODE.GT.2) GO TO 5
CALL SYMBOL(-1.0, -.8, .2, '1HT, 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-5, 0., 2)
CALL SYMBOL(2.4, -.4, 2, '2H+S, 0., 2)
GO TO 6
5 PDEL=(PSTRT-PEND)/IP
PTIC=(ABS(PSTRT-PEND))/10.0
CALL AXIS(-5.0, 0.0, 1H .1, 4.0, 0.0, PSTRT, PTIC)
PSTRE=PSTRT+(6.0*PTIC)+.0001
PTIC2=PTIC+.00001
CALL AXIS(1.0, 0.0, 1H .1, 4.0, 0.0, PSTRE, PTIC2)
CALL PLOT(-1.0, .0, 3)
CALL PLOT(-1.0, .0, 2)
CALL PLOT(.0, .0, 3)
CALL PLOT(.0, 5.8, 2)
CALL PLOT(.0, 0.0, 3)
CALL SYMBOL(-.05,-.4,.2,lHO,0.,1)
507 CALL PLCT(0.0,0.0,3)
508 X=C.05
509 GO TO J=1,9
510 Y=0.5+(J-1)*1.0
511 CALL PLCT(-X,Y,3)
512 CALL PLCT(X,Y,2)
513 CALL PLCT(0.0,0.0,3)
514 DC 1 IW1=1,IP
515 THETA=(PSTRT-(IW1*PDEL))
516 THET=THETA/(ABS(PSTRT))#5
517 PANGS=PSTRT(IW1)/180.*#4.55
518 IF(IW1.EQ.1)CALL PLOT(THET,PANGS,3)
519 IF(IW1.EQ.1)GO TO 1
520 CALL PLOT(THET,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=0.10)
524 FNLM=FLOAT(NUMPAT)
525 CALL NUMBER(-3.5,-0.8,0.125,FNUM,0.,-1)
526 CALL PLCT(8.,-1.,-3)
527 5 RETURN
528 END

SUBROUTINE PLOT2(N,M,CONLOW,CONMAX,CONINT,NUMPAT,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
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)1,3
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)4,5
549 4 CALL CNLAL(N,M,CNTRL0,CMAX,CNTRAL,0)
550 GC TO 7
551 5  CENTRAL=CNTRINT
552  IF(CNTRMAX.EQ.CNLOW)GO TO 6
553  CMAX=CNTRMAX
554  CNTRL=CNTRLO
555  GO TO 7
556 6  CALL CNLAL(N,M,CNTRLO,CMAX,CNTRAL,1)
557 7  CONTINUE
558  CONLOW=CNTRLO
559  CONMAX=CMAX
560  CNINT=CENTRAL
561  CALL PLOTL(SS,SY,SYS,0.,SYS,SCALE)
562  CALL PLOTL(0.,0.,SS,0.,SYS,SCALE)
563  CALL PLOTL(SS,0.,SYS,0.,0.,SCALE)
564  CALL PLOTL(1.0,0.25,3)
565  CALL PLOTL(0.60,0.25,2)
566  CALL PLOTL(0.60,0.25,2)
567  CALL PLOTL(1.00,0.25,2)
568  CALL PLOTL(1.00,0.25,2)
569  CALL SYMBOL(0.88,0.45,0.12,10,FNUM=NUMPAT)
570  CALL NUMBER(0.88,2.075,0.12,FNUM,90.,-1)
571  YCCNA=1.0/SMAX
572  DELTAX=SX/FLOAT(N-1)
573  X(1)=0.0
574  Y(1)=0.0
575  RB(1)=A(1,1)
576  CC 27 J=2,N
577  RB(J)=A(J,1)
578  X(J)=X(J-1)+DELTAX
579  DELTAY=SY/FLOAT(M-1)
580  Y(J)=Y(J-1)+DELTAY
581  CC 28 J=2,M
582  RA(J)=RB(J)
583  X(J)=X(J-1)
584  Y(J)=Y(J-1)
585  IF(RL-RA(J-1))41,40
586  RL=RA(J-1)
587  XL=X(J-1)
588  YL=Y(J-1)
589  GO TO 40
590  RL=RR
591  XL=XX
592  YL=YY
593  IF(RL-RA(J-1))41,40
594  RL=RA(J-1)
595  XL=X(J-1)
596  41  RL=RA(J-1)
597  XL=X(J-1)
598  YL=Y(J-1)
599  GO TO 40
600  GO TO 50
601  RS=RR
602  XL=XX
603  YL=YY
604  GO TO 50
605  XS=XX
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 = XP/SMAX
640   XCL = XL/SMAX
641   RC = CNTRL0
642   80 IF (RC.GT.CMAX) GO TO 110
643   IF (RC .NE. RM) GO TO 91
644   81 IF (RM .NE. RS) GO TO 91
645   82 IF (RL .EQ. 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) > 103, 103, 110
652   95 C = 0.0
653   GO TO 97
654   96 Q = (RC - RS)/(RM - RS)
655   97 XPA = XCS - 0.5(XCS - XCM)
656   YPA = YCS - 0.5(YCS - YCM)
657   99 Q = (RC - RS)/(RL - RS)
658   XPB = XCS - 0.5(XCS - XCL)
659   YPB = YCS - 0.5(YCS - YCL)
660   IF(RC - DASH) > 10115, 10115, 10116
661   10115 XPH = C.5(XPA*XPB)
662   YPB = 0.5*(YPA*YPB)
663   IF(ABS(XPA - XPB) > 0.01) 5001, 5002, 5002
664   5001 IF(ABS(YPA - YPB) > 0.01) 100, 5002, 5002
A-90

5002 CALL PLOT(SCALE*XPA+2.,SCALE*YPA+0.25,3)
5003 CALL PLOT(SCALE*XPB+2.,SCALE*YPB+0.25,2)
5004 CALL PLOT(SCALE*XPA+2.,SCALE*YPA+0.25,3)
5005 CALL PLOT(SCALE*XPB+2.,SCALE*YPB+0.25,2)
5006 RC = RC + CNTRAL
5103 XPA = XCL
5104 YPA = YCL
5105 GO TO 99
5106 C=(RC-RM)/(RL-RM)
5107 XPA=XCM-O*(XCM-XCL)
5108 YPA=YCM-O*(YCM-YCL)
5109 GO TO 99
5110 GC TO L,(112,118)
5112 ASSIGN 118 TO L
5113 RR = RB(J-1)
5114 XX = X (J-1)
5115 YY = Y (K)
5116 GO TO 37
5118 CONTINUE
5119 CALL PLOT(SCALE+6.,0.,-3)
5120 RFTURN
5121 END

SUBROUTINE CNLAL(NM,CNTRL,CMAX,CNTRAL,NC)

DIMENSION X(151,151)
COMMON /ARRAY/ X
5130 XMAX=X(1,1)
5131 XMIN=X(1,1)
5132 GO 10 J=1,N
5133 GO 10 I=1,N
5134 XMAX=AMAX1(XMAX,X(I,J))
5135 XMIN=AMIN1(XMIN,X(I,J))
5136 IF(NC.EQ.1) GO TO 40
5137 IF(XMAX.EQ.0.) GO TO 20
5138 SN=XMIN/XMAX
5139 IF(SN)20,2,30
5140 20 XCON=ABS(XMAX)
5141 IF(ABS(XMIN).GT.ABS(XMAX))XCON=ABS(XMIN)
5142 CNTRL=CNTRAL/10.
5143 CMAX=XMAX
5144 CNTRAL=CNTRAL*INT(XMIN/CNTRAL)
5145 RETURN
5146 30 XCON=ABS(XMAX-XMIN)
5147 CNTRL=CNTRAL/10.
5148 CNTRAL=XMIN
5150 CMAX=XMAX
5151 RETURN
5152 40 CMAX=CNTRAL*INT(XMAX/CNTRAL)
5153 CNTRAL=CNTRAL*INT(XMIN/CNTRAL)
5154 RETURN
5155 END
SUBROUTINE PLOT1(X1,Y1,X2,Y2,S)
DIMENSION X(2),Y(2)
X(1) = S *X1+2.
X(2) = S *X2+2.
Y(1) = S *Y1+0.25
Y(2) = S *Y2+0.25
CALL PLOT(X(1),Y(1),3)
CALL PLOT(X(2),Y(2),2)
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.
DISTS - DISTANCE FROM SURFACE TO EYE WHEN PERSPECTIVE IS
   CALCULATED -- SDISTS > 6 USUALLY WONT' 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

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,D,SH,SY
COMMON /THREE7/ SL,SM,SN,CX,CY,CZ,QX,QY,QZ,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)

****A= N BY M MATRIX OF DATA POINTS
****BORDS= PLOT LABELING
****ONCHAR= NUMBER OF CHARACTERS IN THE PLOT LABEL+SPACES

COMMON /ARRAY/ A
COMMON /THREE6/ ANGA,ANGB,HV,D,SH,SY
COMMON /THREE7/ SL,SM,SN,CX,CY,CZ,QX,QY,QZ,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.
KOCE=0
MGN=0
SCALE=1.
CALL FACTOR(1.1)
CALL PLCT(8., 0., -2)
CALL PLCT(0., 8., 2)
CALL PLCT(0., 0., 2)
CALL SYMBOL(0.3, 1.0, 0.12, 10)
FHNUM=FLOAT(NUMPAT)
CALL NUMBER(0.3, 2.1, 90, 12, FNUM, 90., 1.)
CALL PLOT(1.5, -0.2, -3)

ANGA = (YAW + 270.) * .0174532
ANGB = PITCH * .0174532
H-V = SIZE

C DIRECTION COMPONENTS TO THE EYE.
SL = -COS( ANGA ) * COS( ANGB )
SM = -SIN( ANGA ) * COS( ANGB )
SN = -SIN( ANGB )
IF (ABS(SN).NE.1.0) GO TO 10
WRITE(6, 20)
20 FORMAT ('1', 20X, 201**), / '0', 'YOU ARE ATTEMPTING TO LOOK STRAIGHT DOWN (OR UP) AT THE SURFACE'
GO TO 2150

10 CONTINUE
SD = 1.0 / SCRT( 1.0 - SN ** 2 )
X(1) = 1
X(2) = N
Y(1) = 1
Y(2) = M
T = MAXC(M, N)

C FIND THE DIAGONAL OF THE "CUBE".
D = M ** 2 + N ** 2 + T ** 2
D = SQRT( D )
SCL = SOISTS * D

C COORDINATES OF YOUR EYE.
CX = -SL * SCL
CY = -SM * SCL
CZ = -SN * SCL

C COORDINATES OF THE PROJECTION PLANE.
QX = CX + D * SL
QY = CY + D * SM
QZ = CZ + D * SN

C
GL TC 2060
WRITE(6, 100) CX, CY, CZ
WRITE(6, 100) QX, QY, QZ

100 FORMAT(1X, 3F15.3)
2060 Z(2) = A(I, 1)
A-94

781  \[ Z(1) = A(1,1) \]
782  IC 1000 J=1,N
783  IC 1000 K=1,M
784  Z(1) = AMIN1(Z(1), A(J,K))
785  Z(2) = AMAX1(Z(2), A(J,K))
786  IC 1000 CONTINUE
787  RANGE = \{ (Z[2] - Z[1]) \}
788  DCL = 1.O
789  IF(Scale .NE. 0) DOL = T/RANGE*SCALE
790  C SCALE THE SURFACE TO MAKE A "CUBE*
791  DO 30 I = 1, N
792  DO 30 J = 1, M
793  A(I,J) = \{ A(I,J) - Z(1) \} * DOL
794  30 CONTINUE
795  Z(1) = C.C
796  Z(2) = T
797  20 DO CALL THREEZ ( X, Y, Z, XP, H, V, KODE)
798  DO 2130 I = 1, 8
799  H(I) = ( (XP(I) - OX) * SM - ( H(1) - QY ) * SL ) * SD
800  V(I) = ( V(1) - QZ ) * SD
801  2130 CONTINUE
802  2100 H(10) = H(1)
803  H(9) = H(1)
804  H(9) = AMIN1(H(9), H(J))
805  H(10) = AMAX1(H(10), H(J))
806  IC 1001 CONTINUE
807  2120 H(9) = V(1)
808  V(10) = V(1)
809  DO 1002 J = 1, 8
810  V(J) = AMIN1(V(J), V(J))
811  V(10) = AMAX1(V(10), V(J))
812  1002 CONTINUE
813  IF(MGN .EQ. 0) GO TO 2140
814  S = H
815  IF(MGN .EQ. 1) S = 1.5
816  SH = S/ (H(10) - H(9))
817  SV = S/ (V(10) - V(9))
818  SH = SIGN( AMIN1(SH, SV), SH )
819  SV = SIGN(SH, SV)
820  IF(MGN .EQ. 1) CALL PLOT (0.0, 2., -3)
821  CALL SYMBOL1(H(1) - H(9))*SH, (V(1) - V(9))*SV, 14, "0", 0., 1)
822  CALL SYMBOL1(H(3) - H(9))*SH, (V(3) - V(9))*SV, 14, "M", 0., 1)
823  CALL SYMBOL1(H(2) - H(9))*SH, (V(2) - V(9))*SV, 14, "Z", 0., 1)
824  CALL SYMBOL1(H(5) - H(9))*SH, (V(5) - V(9))*SV, 14, "N", 0., 1)
825  CALL PLOT (0.0, 0.5, -3)
826  CALL PLOT (H(1) - H(9))*SH, (V(1) - V(9))*SV, 3)
827  CALL PLOT (H(2) - H(9))*SH, (V(2) - V(9))*SV, 2)
828  CALL PLOT (H(3) - H(9))*SH, (V(3) - V(9))*SV, 2)
829  CALL PLOT (H(1) - H(9))*SH, (V(1) - V(9))*SV, 2)
830  CALL PLOT (H(5) - H(9))*SH, (V(5) - V(9))*SV, 2)
831  IF(MGN .EQ. 3) GO TO 2139
832  CALL PLOT (H(6) - H(9))*SH, (V(6) - V(9))*SV, 2)
833  CALL PLOT (H(2) - H(9))*SH, (V(2) - V(9))*SV, 2)
834  CALL PLOT (H(4) - H(9))*SH, (V(4) - V(9))*SV, 2)
835  CALL PLOT (H(7) - H(9))*SH, (V(7) - V(9))*SV, 2)
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(8)

DO 180 J = 1, 2

CONTINUE

CONTINUE

CONTINUE

RETURN

END

SUBROUTINE THREE3 (X, Y, N, M, H, V, K, KODE)
C DRAW THE FIGURE.

DIMENSION X(2), Y(2), H(10), V(10), A(151, 151)
COMMON  ARRAY  A
INTEGER UP, DOWN, PEN, P, G
INTEGER P1, PO

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

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

IF(K-1) 100,120,100
T100 IF(K-3) 1110,120,1110
DRAW LINES ALONG THE Y-AXIS
12C CONTINUE
L = 0
LD = 1
EC = 0.5 * LD

14C DO 1060 J = 1, M
1050 C = 0
1060 YJ = J
1070 16C DO 1030 I = 1, NN
1080 L = L + LD
1090 XI = L
1100 CALL THREE5 ( XI, YJ, N, M, P, KODE)
1110 PEN = UP
1120 IF (.P ) 510, 520, 530
1130 510 CONTINUE
1140 IF ( .Q ) 540, 550, 540
1150 520 CONTINUE
1160 IF ( .Q ) 610, 610, 610
1170 530 CONTINUE
1180 IF ( .Q ) 540, 550, 540
1190 540 CONTINUE
1200 PEN = DOWN
1210 GC TO 170
1220 550 CONTINUE
1230 IF ( I .EQ. 1 ) GO TO 170
1240 DI = CD
1250 TC = L - LC
1260 T = TO + DI
1270 P1 = Q
1280 560 IF ( ABS( DI ) .LT. END ) GO TO 570
1290 CALL THREE5 ( T, YJ, N, M, PO, KODE)
1300 CI = DI * 0.5
1310 IF ( PO .EQ. C ) GO TO 565
TC = T
P1 = PO
T = T - DI
GO TO 560
T = T + DI
GO TO 56C
57C CONTINUE
T = TO
IF ( P1 * P ) 170, 170, 580
58C CONTINUE
59C CONTINUE
ZP = A(L-LC,J)+(T-L+LD)*(A(L,J)-A(L-LD,J))/LD
CALL THREE4(T,YJ,ZP,XP,HH,VV,KODE)
HP = ((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)
600 PEN = 5 - PEN
GO TO 170
61C CONTINUE
PEN = DOWN
DI = DC
TO = L - LD
T = TO + DI
P1 = Q
620 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 620
T = T - DI
GO TO 62C
63C CONTINUE
T = TC
IF ( P1 * Q ) 600, 600, 590
17C CALL THREE4 (XI, YJ, A(L, J), XP, HH, VV, K6(L)
VV = (VV - QZ)*SD
HH = ( (XP-QX)*SM-(HH-QY)*SL)*SD
19C HH = (HH - H(9))*SH
200 VV = (VV - V(9))*SV
21C CALL PLOT (HH, VV, PEN)
102C Q = P
1030 CONTINUE
C
L = L + LD
LC = -LD
DD = -DD
1060 CONTINUE
C
C
C109C IF(K-3) 2060, 111C, 206C
C DRAW LINES ALONG THE X-AXIS.

972  1110 CONTINUE

C
973    L = 0
974    LC = 1
975    CC = 0.5 * LD
976  1140 DO 2040 I = 1, N
977     XI = I
978    Q = C
979  1160    DO 2020 J = 1, MM
980       L = L + LD
981     YJ = L
982    CALL THREE5 (XI, YJ, N, M, P, KODE)
983    PEN = UP
984  1510 IF (P) 1510, 1520, 1530
985  1510 CONTINUE
986    IF (Q) 1540, 1550, 1540
987  1520 CONTINUE
988    IF (Q) 1610, 2010, 1610
989  1530 CONTINUE
990    IF (Q) 1540, 1550, 1540
991  1540 CONTINUE
992    PEN = DOWN
993   GO TO 1170
994  1550 CONTINUE
995    IF (J .EQ. 1) GO TO 1170
996     DI = DD
997    TC = L - LD
998   T = TO + DI
999     P1 = Q
1000  1560 IF (ABS(DI) .LT. END) GO TO 1570
1001    CALL THREE5 (XI, T, N, M, P, KODE)
1002     DI = DI * 0.5
1003    IF (P0 .EQ. 0) GO TO 1565
1004     TO = T
1005     P1 = PO
1006    T = T - DI
1007   GO TO 1560
1008  1565 T = T + DI
1009  GO TO 1560
1010  1570 CONTINUE
1011    T = TO
1012    IF (P1 * P) 1170, 1170, 1580
1013  1580 CONTINUE
1014  1590 CONTINUE
1016    CALL THREE4 (XI, T, ZP, XP, HH, VV, KODE)
1017    HH = (XP - QX) * SM - (HH - QY) * SL * SI
1018     VV = (VV - GZ) * SD
1019    HH = (HH - H(9)) * SH
1020     VV = (VV - V(9)) * SV
1021    CALL PLOT (HH, VV, PEN)
1022  1600 PEN = 5 - PEN
1023  GO TO 1170
1024  1610 CONTINUE
1025    PEN = DOWN
A-99

1026       DI = DD
1027       TO = L - LC
1028       T = TO + DI
1029       P1 = C
1030 1620 IF ( ABS(DI) .LT. END ) GO TO 1630
1031       CALL THREE5 (XI,T,N,M,PO,KODE)
1032       DI = DI * 0.5
1033       IF ( PC .EQ. 0 ) GO TO 1625
1034       TO = T
1035       P1 = PO
1036       T = T + DI
1037       GO TO 1620
1038 1625 T = T - DI
1039       GO TO 1620
1040       1630 CONTINUE
1041       T = TO
1042       IF ( P1 * C ) 1600 = 1600 , 1590
1043       CALL THREE4 ( XI, YJ, A(I,L), XP, HH ,VV ,KODE)
1044       HH = ( ( XP-QX)*SM - (HH - QY)*SL ) * SD
1045       VV = ( VV - GZ ) * SD
1046       HH = ( HH - H(9) ) * SH
1047       VV = ( VV - V(9) ) * SV
1048       CALL PLOT ( HH , VV , PEN )
1049 2010 C = P
1050 2020 CONTINUE
1051       L = L + LD
1052       LC = - LD
1053       DD = -DO
1054 2040 CONTINUE
1055       C
1056 2060 CONTINUE
1057       C
1058 2130 RETURN
1059       END

1058 SUBROUTINE THREE5 (XI,YJ,M,N,P,KODE)
1059       C SEE IF A POINT IS VISIBLE.
1060       DIMENSION Z(151,151)
1061       COMMON /THREE6/ ANGA , ANGB , HV , D, SH,SV
1062       COMMON /THREE7/ SL,SM,SN,CX,CY,CZ,QX,QY,QZ,SD
1063       COMMON /ARRAY/ Z
1064       INTEGER CUM , CNT , P
1065       REAL I , J , II , JJ
1066       IF( KODE .EQ. 1 ) GO TO 78
1067       IR = XI
1068       JC = YJ
1069       ZB = Z ( IR + JC )
1070       IF ( XI .EQ. IR ) GO TO 2
1070       ZB = Z(IN,JC) + ( XI - IR) * (Z(IR + 1 , JC) - Z(IR ,JC) )
1071       GO TO 4
1072       2 IF ( YJ .EQ. JC ) GO TO 4
1073       ZB = Z(IR , JC) + (YJ-JC) * (Z(IR , JC+1) - Z(IR , JC) )
1074       4 CONTINUE
1075       XEND = C*C
1076       CX = 0.0

REPRODUCIBILITY OF THE
ORIGINAL PAGE IS POOR
A-100

1077 YMULT = C.C
1078 ZMULT = C.C
1079 IF (XI .EQ. CX ) GO TO 10
1080 YMULT = (YJ - CY ) / (XI - CX )
1081 ZMULT = (ZB - CZ ) / ( XI - CX )
1082 CX = 1.0
1083 XEND = N + 1
1084 IF ( XI .LT. CX ) GO TO 10
1085 CX = -1.0
1086 XEND = 0.0
1087 CONTINUE
1088 YEND = C.C
1089 DY = 0.0
1090 XMULT = 0.0
1091 IF ( YJ .EQ. CY ) GO TO 20
1092 XMULT = ( XI - CX ) / (YJ - CY )
1093 IF ( ZMULT .EQ. 0.0 ) ZMULT=(ZB - CZ ) / ( YJ - CY )
1094 DY = 1.0
1095 YEND = N + 1
1096 IF ( YJ .LT. CY ) GO TO 20
1097 DY = -1.0
1098 YEND = C.C
1099 CONTINUE
1100 CXP = 0
1101 CNT = 0
1102 P = 0
1103 XB = XI
1104 YB = YJ
1105 CONTINUE
1106 II = AINT( XB )
1107 JJ = AINT( YB )
1108 XSTEP = DX
1109 YSTEP = DY
1110 IF ( XB .EQ. II ) GO TO 40
1111 IF ( CX .LT. 0.0 ) XSTEP = 0.0
1112 GO TO 45
1113 IF ( YB .EQ. JJ ) GO TO 45
1114 IF ( DY .LT. 0.0 ) YSTEP = 0.0
1115 CONTINUE
1116 I = II + XSTEP
1117 J = JJ + YSTEP
1118 IF ( I .EQ. XEND ) GO TO 80
1119 IF ( J .EQ. YEND ) GO TO 80
1120 XB = CX + XMULT * ( J - CY )
1121 YB = CY + YMULT * ( I - CX )
1122 IF ( DX .LT. 0.0 ) GO TO 55
1123 IF ( XB .LT. I ) GO TO 60
1124 XB = I
1125 GO TO 65
1126 IF ( XB .LT. I ) GO TO 50
1127 YB = J
1128 CONTINUE
1129 ZB = CZ + ZMULT * ( XB - CX )
1130 IR = I
1131 JC = J
1132 IF ( YB .NE. J ) GO TO 70
1133 ICX = I - CX
FUNCTION PAT(I,U,V,ITYPE)

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
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,PI,SS(100),IT(100),RR(100)
COMMON /PAT2/ 11,12,13,14,15
A-102

IF(ITYPE.GT.7) GO TO 990
GO TO (1CC,2CC,300,400,500,600,700),ITYPE

ITYPE .LT. 1

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

ITYPE = 1 -- UNIFORM LINE SOURCE.

FLEN=PI
100 CONTINUE
PAT=1.0
IF(V.NE.C.) PAT=SIN(P1*P1*V)/(P1*P1*V)
GO TO 999

ITYPE = 2 -- UNIFORM LINEAR ARRAY

FLEN=PI
NELMT=II
PAT=1.0
IF(V.NE.C.) PAT=SIN(P1*P1*V)/(II*SIN(P1*P1*V/I1))
GO TO 999

ITYPE = 3 -- TRIANGULAR LINE SOURCE.

FLEN=PI/2.
PAT=1.0
IF(V.NE.C.) PAT=(SIN(FLEN*PI*V)/(FLEN*PI*V))**2
GO TO 999

ITYPE = 4 -- UNIFORM RECTANGULAR APERTURE

FLS=PI
FLT=P2
ARG1=PI*P1*U
ARG2=PI*P2*V
IF(ARG1) 4C1,4C2,401
IF(ARG2) 4C3,401,402,403
401 PAT=SIN(ARG1)/ARG1*SIN(ARG2)/ARG2
GO TO 999
402 PAT=SIN(ARG1)/ARG1
GO TO 999

192 GO TO 999
A-103

ITYPE = 5 -- UNIFORM RECTANGULAR ARRAY

1193 500 CONTINUE
C 500 1194 ARG1=PI*P1*U
C 1195 ARG2=PI*P2*V
C 1196 IF(ARG1) 501,502,501
C 1197 501 IF(ARG2) 503,504,503
C 1198 503 PAT=SIN(ARG1)/(I1*SIN(ARG1/I1))*SIN(ARG2)/(12*SIN(ARG2/12))
C 1199 GO TO 999
C 1200 504 PAT=SIN(ARG1)/(I1*SIN(ARG1/I1))
C 1201 GO TO 999
C 1202 502 IF(ARG2) 505,506,505
C 1203 505 PAT=SIN(ARG2)/(I2*SIN(ARG2/12))
C 1204 GO TO 999
C 1205 506 PAT=1.0
C 1206 GO TO 999

ITYPE = 6 -- UNIFORM CIRCULAR APERTURE.

1207 600 C=SQRT(U*U+V*V)
C 600 1208 A=P1
C 1209 IF(C.EQ.0.) GO TO 601
C 1210 X=2.*PI*P1*C
C 1211 CALL BESJ(X,1,8J,0.0001,IER)
C 1212 PAT=BJ/X*2.0
C 1213 GO TO 999
C 1214 GO TO 999.

ITYPE = 7 -- GENERAL ARRAY

1215 700 IMAG=(O.0,1.O)
C 1216 NELMT=I1*12
C 1217 TEMP=(0.0,0.0)
C 1218 DC 701 J=1,NELMT
C 1219 TEMP=TEMP+1.0*CEXP(IMAG*2.*PI*(U*SS(J)+V*TT(J)))
C 1220 701 CONTINUE
C 1221 PAT=REAL(TEMP)/NELMT
C 1222 GO TO 999
C 1223 999 PAT=SPECPT(UV,ITYPE)
C 1224 999 RETURN
C 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.
2 -- UNIFORM LINEAR ARRAY LOCATED AT S=0.
3 -- TRIANGULAR LINE SOURCE LOCATED AT S=0.
4 -- UNIFORM RECTANGULAR APERTURE.
5 -- UNIFORM RECTANGULAR ARRAY.
6 -- UNIFORM CIRCULAR APERTURE.
7 -- GENERAL (3-D) ARRAY.

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

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 ELECTRICAL ENGR. 951-6624.
E. L. COFFEY DEPT. OF ELECTRICAL ENGR. 951-5494.

1227 COMPLEX TEMP, CEXP, IMAG, SPSOR
1228 COMMON /PAT1/ P1, P2, P3, P4, P5, P6, PI, SS(100), TT(100), RR(100)
1229 COMMON /PAT2/ I1, I2, I3, I4, I5

1230 IMAG = (0.0, 1.0)
1231 CALL LOCOSOR(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 10 FORMAT (IHC, 5X, '***ERROR*** ITYPE HAS THE VALUE ', ITYPE, IX, 
$'EXECUTION TERMINATED')

1236

1237 IF (ITYPE .LT. 1)
1238 C CONTINUE

1239 FLEN = P1
1240 SOURCE = CEXP(-IMAG * P1 * T * V) / P1

1241 GO TO 999

1242

1243 ITYPE = 2 -- UNIFORM LINEAR ARRAY

1244 C CONTINUE

1245 FLEN = P1
1246 SOURCE = CEXP(-IMAG * P1 * T * V) / P1

1247 GO TO 999

1248

1249 ITYPE = 3 -- TRIANGULAR LINE SOURCE

REPRODUCIBILITY OF THE ORIGINAL PAGE IS POOR
CONTINUE

CON=ABS(2.*T/P1)

SOURCE=2.*PI*CEXP(-IMAG*2.*PI*T*V)/(1.-CON)

IF(CON.GT.1) SOURCE=(0.0,0.0)

GO TO 999

ITYPE = 4 -- UNIFORM RECTANGULAR APERTURE

CONTINUE

FLS=P1

FLT=P2

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

GO TO 999

ITYPE = 5 -- UNIFORM RECTANGULAR ARRAY

CONTINUE

FLS=P1

FLT=P2

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

GO TO 999

ITYPE = 6 -- UNIFORM CIRCULAR APERTURE

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

A=PI

SOURCE=(0.0,0.0)

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

GO TO 999

ITYPE = 7 -- GENERAL ARRAY

CONTINUE

SOURCE=CEXP(-IMAG*2.*PI*(U*S+V*T))/(I*L*12)

GO TO 999

SOURCE=SPSOR(M,N,U,V,ITYPE)

999 RETURN

END

SUBROUTINE LOCSOR(M,N,S,T)

INTEGER PX,PY

REAL INITLS,INITLT

COMMON /PAT1/ P1,P2,P3,P4,P5,P6,PI,SS(100),IT(100),RR(100)

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

COMMON /LOC/ ITYPE

IF(ITYPE.GT.7) GO TO 999

GO TO (100,200,300,400,500,600,700), ITYPE
A-106

1272 WRITE(6,10) ITYPE
1273 10 FORMAT(1HC,5X,***ERROR***
1274     **EXECUTION TERMINATED**)
1275 STCP
1276
1277 100 CONTINUE
1278     INITLT=P1
1279     CELTAT=P3
1280     S=C.
1281     T=P2+(N-1)*P3
1282     GO TO 999
1283
1284 200 CONTINUE
1285     PY=I1
1286     DISY=P2
1287     S=C.
1288     T=(N-11/2-1)*P2
1289     IF(11/2*2.EQ.11) T=T+0.5*P2
1290     GO TO 999
1291
1292 300 GC TO 100
1293
1294 400 CONTINUE
1295     INITLS=P3
1296     INITLT=P4
1297     CELTAS=P5
1298     DELTAT=P6
1299     S=P3+(M-1)*P5
1300     T=P4+(N-1)*P6
1301     GO TO 999
1302
1303 500 CONTINUE
1304     PX=I1
1305     PY=I2
1306     DISX=P3
1307     DISY=P4
1308     S=(N-11/2-1)*P3
1309     T=(N-12/2-1)*P4
1310     IF(12/2*2.EQ.12) T=T+0.5*P4
1311     GO TO 999
1312
1313 600 GC TO 400
1314
1315 700 CONTINUE
1316     NELPT=(M-1)*I2+N
1317     S=SS(NELMT)
1318     T=TT(NELPT)
1319     GO TO 999
1320
C 1361      CALL SPLOC(M,N,S,T)
1362      RETURN
1363      END

1364      COMPLEX FUNCTION SPSCR(M,N,U,V,ITYPE)
1365          DUMMY SUBPROGRAM
1366      SPSCR=(0.0,0.0)
1367      RETURN
1368      END

1369      SUBROUTINE SPLOC(M,N,S,T)
1370          DUMMY SUBROUTINE
1371      RETURN
1372      END
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

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

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

8.1 Input to ANTSYN

Since a rectangular aperture is included in our types of patterns \((ITYPE=4)\) 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, DELTAU, DELTAV)
DIMENSION FDES(51,51), FU(51,51), FL(51,51)

READ MAINBEAM LIMITS ULIM AND VLIM
READ(5,1) ULIM, VLIM
1 FORMAT(8F10.0)

READ TRANSITION REGION LIMITS UTRAN AND VTRAN
READ(5,1) UTRAN, VTRAN

DO 10 M=1, MMAX
U=STARTU+(M-1)*DELTAU
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

TRANSITION REGION
FDES(M,N)=99.0
FU(M,N)=99.0
FL(M,N)=99.0
GO TO 10
20 CONTINUE

MAIN BEAM REGION
FDES(M,N)=1.0
FU(M,N)=1.06
FL(M,N)=0.943
GO TO 10
30 CONTINUE

SIDELOBE REGION
FDES(M,N)=0.
FU(M,N)=0.057
FL(M,N)=99.0
10 CONTINUE

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

<p>| | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>11</td>
<td>21</td>
<td>31</td>
<td>41</td>
<td>51</td>
<td>61</td>
</tr>
</tbody>
</table>

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>&amp;PARAM</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IDISK=1, ISYMM=3, DELTAU=0.02, DELTAV=0.01, MMAX=26, NMAX=51,</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>&amp;END</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>&amp;IPRINT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FDESCN=1, FDBPR=1, FDBCN=1, FCURPR=1,</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>&amp;END</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>&amp;PATIN</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ITYPE=4, LX=10., LY=20.,</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INITS=-5.0, DELTAS=0.2, FINALS=5.0,</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INITLT=-10.0, DELTAT=0.4, FINALT=10.0,</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>&amp;END</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.2</td>
<td>0.05</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.34</td>
<td>0.12</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>00015</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>1.0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0</td>
<td>0.05</td>
<td>1.0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0</td>
<td>-0.05</td>
<td>1.0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.1</td>
<td>0.0</td>
<td>1.0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-0.1</td>
<td>0.0</td>
<td>1.0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.1</td>
<td>0.05</td>
<td>1.0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.1</td>
<td>-0.05</td>
<td>1.0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-0.1</td>
<td>0.05</td>
<td>1.0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.2</td>
<td>0.05</td>
<td>1.0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.2</td>
<td>-0.05</td>
<td>1.0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-0.2</td>
<td>0.05</td>
<td>1.0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-0.2</td>
<td>0.0</td>
<td>1.0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-0.2</td>
<td>0.0</td>
<td>1.0</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

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 (\( M_{\text{MAX}}=26, N_{\text{MAX}}=51 \))

vi. Assure that \( F(1,1)=0 \, \text{dB} \) at all times (\( M_{\text{CENT}}=1, N_{\text{CENT}}=1 \))

Note that \( F(M_{\text{MAX}},N_{\text{MAX}}) \) 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.
\( \text{INITLS}=-5., \text{FINALS}=5., \text{DELTAS}=0.2; \text{INITLT}=-10., \text{FINALT}=10., \text{DELTAT}=0.4 \)

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 \( (N_{\text{ORG}}) \) and the values of \( (U_{\text{ORG}},V_{\text{ORG}},C_{\text{ORG}}) \) 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

FDESPT = 0  ICURPT = 0
FDESgn = 1  ICURCN = 0
FDESpr = 0  ICURPR = 0

FGRESPT = 0  FCURPT = 0
FGRSCN = 0  FCURCN = 0
FGRSPR = 0  FCURPR = 1

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
**A-113**

**CONTOUR PLOT OF THE DESIRED PATTERN.**

<table>
<thead>
<tr>
<th></th>
<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>
<td></td>
</tr>
<tr>
<td>0.4900</td>
<td></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>
<td></td>
</tr>
<tr>
<td>0.4700</td>
<td></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>
<td></td>
</tr>
<tr>
<td>0.4500</td>
<td></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>
<td></td>
</tr>
<tr>
<td>0.4300</td>
<td></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>
<td></td>
</tr>
<tr>
<td>0.4100</td>
<td></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>
<td></td>
</tr>
<tr>
<td>0.3900</td>
<td></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>
<td></td>
</tr>
<tr>
<td>0.3700</td>
<td></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>
<td></td>
</tr>
<tr>
<td>0.3500</td>
<td></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>
<td></td>
</tr>
<tr>
<td>0.3300</td>
<td></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>
<td></td>
</tr>
<tr>
<td>0.3100</td>
<td></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>
<td></td>
</tr>
<tr>
<td>0.2900</td>
<td></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>
<td></td>
</tr>
<tr>
<td>0.2700</td>
<td></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>
<td></td>
</tr>
<tr>
<td>0.2500</td>
<td></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>
<td></td>
</tr>
<tr>
<td>0.2300</td>
<td></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>
<td></td>
</tr>
<tr>
<td>0.2100</td>
<td></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>
<td></td>
</tr>
<tr>
<td>0.1900</td>
<td></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>
<td></td>
</tr>
<tr>
<td>0.1700</td>
<td></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>
<td></td>
</tr>
<tr>
<td>0.1500</td>
<td></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>
<td></td>
</tr>
<tr>
<td>0.1300</td>
<td></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>
<td></td>
</tr>
<tr>
<td>0.1100</td>
<td></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>
<td></td>
</tr>
<tr>
<td>0.0900</td>
<td></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>
<td></td>
</tr>
<tr>
<td>0.0700</td>
<td></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>
<td></td>
</tr>
<tr>
<td>0.0500</td>
<td></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>
<td></td>
</tr>
<tr>
<td>0.0300</td>
<td></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>
<td></td>
</tr>
<tr>
<td>0.0100</td>
<td></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>
<td></td>
</tr>
<tr>
<td>-0.0100</td>
<td></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>
<td></td>
</tr>
<tr>
<td>-0.0300</td>
<td></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>
<td></td>
</tr>
<tr>
<td>-0.0500</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Contour Level Key

<table>
<thead>
<tr>
<th>Number</th>
<th>Value Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>0:</td>
<td>$-0.6000000E+00$ to $-0.3999900E+00$</td>
</tr>
<tr>
<td>1:</td>
<td>$-0.4000000E+00$ to $-0.1999900E+00$</td>
</tr>
<tr>
<td>2:</td>
<td>$-0.2000000E+00$ to $0.1000000E-04$</td>
</tr>
<tr>
<td>3:</td>
<td>$0.0$ to $0.2000099E+00$</td>
</tr>
<tr>
<td>4:</td>
<td>$0.2000000E+00$ to $0.4000099E+00$</td>
</tr>
<tr>
<td>5:</td>
<td>$0.4000000E+00$ to $0.6000099E+00$</td>
</tr>
<tr>
<td>6:</td>
<td>$0.5999998E+00$ to $0.8000098E+00$</td>
</tr>
<tr>
<td>7:</td>
<td>$0.7999997E+00$ to $1.0000100E+01$</td>
</tr>
<tr>
<td>8:</td>
<td>$0.9999995E+00$ to $0.1200008E+01$</td>
</tr>
<tr>
<td>9:</td>
<td>$0.1199999E+01$ to $0.1400008E+01$</td>
</tr>
<tr>
<td>-:</td>
<td>$-0.9999999E+30$ to $-0.6000000E+00$</td>
</tr>
<tr>
<td>+:</td>
<td>$0.1400008E+01$ to $0.9999999E+30$</td>
</tr>
</tbody>
</table>

### 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>
</tbody>
</table>
**SEARCH**

9 4 -0.1842
9 14 0.1524
1 14 0.2088
11 6 0.1875
5 6 0.1962
9 18 -0.1572
1 18 -0.2146
1 22 0.1065
19 6 0.1304
8 14 0.1241
8 22 0.1394
1 14 0.1209
11 6 0.1322
4 6 0.1436
23 6 -0.1201
7 18 -0.1188
1 4 -0.0716
9 5 -0.1592
1 5 -0.1155
9 5 -0.0856
1 5 -0.0900
9 5 -0.1467
1 5 -0.1023
9 5 -0.0757
7 16 0.1005
11 6 0.0889
14 14 0.1093
1 5 -0.0670
9 5 -0.0946
11 1 0.1536
19 1 0.1523
1 5 -0.0784
23 1 -0.1235
9 4 -0.1375
11 1 0.1179
6 1 0.1176
6 6 0.0956
9 4 -0.0875
11 6 0.1438
11 1 0.0898
5 1 0.0873
5 6 0.1050
9 4 -0.0652
6 1 0.1144
11 1 0.1130
11 6 0.0960
9 4 -0.0996
6 6 0.1263
11 1 0.1018
6 1 0.0968
9 4 -0.0924
11 6 0.1390
5 6 0.0783
11 1 0.0947
9 5 -0.0938
11 6 0.0748
1 4 -0.0624
10 5 -0.1378
1 5 -0.0909
9 5 -0.0783
1 5 -0.0715
## 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>
</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**
| Pattern Number | A-118 | U-Axis Profile of Final Pattern | V = 0.0 | Pattern Number | 77 |

<table>
<thead>
<tr>
<th>1.017</th>
<th>0.153</th>
<th>-0.063</th>
<th>REAL</th>
<th>DR.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.000</td>
<td>+</td>
<td>+</td>
<td>*</td>
<td>+</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.950</td>
<td>+</td>
<td>+</td>
<td>*</td>
<td>+</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.900</td>
<td>+</td>
<td>+</td>
<td>*</td>
<td>+</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.850</td>
<td>+</td>
<td>+</td>
<td>*</td>
<td>+</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.800</td>
<td>+</td>
<td>+</td>
<td>*</td>
<td>+</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>x</td>
<td>y</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-----</td>
<td>-------</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.250</td>
<td>+</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.200</td>
<td>*</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.150</td>
<td>**</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.100</td>
<td>+</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.050</td>
<td>+</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.000</td>
<td>+</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>y</th>
<th>x</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.666</td>
<td>-3.53</td>
</tr>
<tr>
<td>0.707</td>
<td>-3.01</td>
</tr>
<tr>
<td>0.747</td>
<td>-2.54</td>
</tr>
<tr>
<td>0.784</td>
<td>-2.11</td>
</tr>
<tr>
<td>0.819</td>
<td>-1.73</td>
</tr>
<tr>
<td>0.852</td>
<td>-1.39</td>
</tr>
<tr>
<td>0.882</td>
<td>-1.09</td>
</tr>
<tr>
<td>0.909</td>
<td>-0.83</td>
</tr>
<tr>
<td>0.932</td>
<td>-0.60</td>
</tr>
<tr>
<td>0.954</td>
<td>-0.41</td>
</tr>
<tr>
<td>0.972</td>
<td>-0.25</td>
</tr>
<tr>
<td>0.986</td>
<td>-0.12</td>
</tr>
<tr>
<td>0.998</td>
<td>-0.02</td>
</tr>
<tr>
<td>1.007</td>
<td>0.06</td>
</tr>
<tr>
<td>1.013</td>
<td>0.11</td>
</tr>
<tr>
<td>1.016</td>
<td>0.14</td>
</tr>
<tr>
<td>1.017</td>
<td>0.15</td>
</tr>
<tr>
<td>1.016</td>
<td>0.14</td>
</tr>
<tr>
<td>1.013</td>
<td>0.11</td>
</tr>
<tr>
<td>1.009</td>
<td>0.07</td>
</tr>
<tr>
<td>1.003</td>
<td>0.02</td>
</tr>
<tr>
<td>0.996</td>
<td>-0.04</td>
</tr>
<tr>
<td>0.988</td>
<td>-0.10</td>
</tr>
<tr>
<td>0.980</td>
<td>-0.17</td>
</tr>
<tr>
<td>0.972</td>
<td>-0.24</td>
</tr>
<tr>
<td>0.964</td>
<td>-0.32</td>
</tr>
<tr>
<td>0.957</td>
<td>-0.38</td>
</tr>
<tr>
<td>0.950</td>
<td>-0.45</td>
</tr>
<tr>
<td>0.943</td>
<td>-0.51</td>
</tr>
<tr>
<td>0.938</td>
<td>-0.55</td>
</tr>
<tr>
<td>0.934</td>
<td>-0.59</td>
</tr>
<tr>
<td>0.931</td>
<td>-0.62</td>
</tr>
<tr>
<td>0.929</td>
<td>-0.64</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>y</th>
<th>x</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.017</td>
<td>0.152</td>
</tr>
<tr>
<td>0.100</td>
<td>+</td>
</tr>
<tr>
<td>0.050</td>
<td>+</td>
</tr>
<tr>
<td>0.000</td>
<td>+</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>y</th>
<th>x</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.017</td>
<td>0.043</td>
</tr>
<tr>
<td>0.005</td>
<td>0.003</td>
</tr>
<tr>
<td>-0.008</td>
<td>0.005</td>
</tr>
<tr>
<td>-0.010</td>
<td>0.010</td>
</tr>
<tr>
<td>-0.012</td>
<td>0.012</td>
</tr>
<tr>
<td>-0.014</td>
<td>0.014</td>
</tr>
<tr>
<td>-0.016</td>
<td>0.016</td>
</tr>
<tr>
<td>-0.018</td>
<td>0.018</td>
</tr>
<tr>
<td>-0.018</td>
<td>0.018</td>
</tr>
<tr>
<td>-0.018</td>
<td>0.018</td>
</tr>
<tr>
<td>-0.017</td>
<td>0.017</td>
</tr>
<tr>
<td>-0.016</td>
<td>0.016</td>
</tr>
<tr>
<td>-0.014</td>
<td>0.014</td>
</tr>
<tr>
<td>-0.012</td>
<td>0.012</td>
</tr>
<tr>
<td>-0.010</td>
<td>0.010</td>
</tr>
<tr>
<td>-0.008</td>
<td>0.008</td>
</tr>
<tr>
<td>-0.005</td>
<td>0.005</td>
</tr>
<tr>
<td>-0.003</td>
<td>0.003</td>
</tr>
</tbody>
</table>
A-120

V-AXIS PROFILE OF FINAL PATTERN  --  U= 0.0
PATTERN NUMBER  77

<table>
<thead>
<tr>
<th>Pattern Number</th>
<th>0.163</th>
<th>-0.061</th>
<th>REAL</th>
<th>DB.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.000</td>
<td>+</td>
<td>+</td>
<td>-0.002</td>
<td>-55.08</td>
</tr>
<tr>
<td>0.950</td>
<td>+</td>
<td>+</td>
<td>-0.004</td>
<td>-48.45</td>
</tr>
<tr>
<td>0.900</td>
<td>+</td>
<td>+</td>
<td>-0.004</td>
<td>-48.45</td>
</tr>
<tr>
<td>0.850</td>
<td>+</td>
<td>+</td>
<td>-0.004</td>
<td>-48.45</td>
</tr>
<tr>
<td>0.800</td>
<td>+</td>
<td>+</td>
<td>-0.004</td>
<td>-48.45</td>
</tr>
<tr>
<td>1.000</td>
<td>+</td>
<td>+</td>
<td>-0.003</td>
<td>-51.06</td>
</tr>
<tr>
<td>1.056</td>
<td>+</td>
<td>+</td>
<td>-0.003</td>
<td>-51.06</td>
</tr>
</tbody>
</table>

1.056  0.163  -0.061
### 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>0.6880980E-02</td>
<td>-0.2095476E-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>-0.2099303E-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>-0.3492460E-08</td>
</tr>
<tr>
<td>-0.8000</td>
<td>0.0000</td>
<td>0.6967366E-01</td>
<td>-0.3492460E-08</td>
</tr>
<tr>
<td>-0.6000</td>
<td>0.0000</td>
<td>0.8006346E-01</td>
<td>-0.2793968E-08</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>-0.7105427E-14</td>
</tr>
<tr>
<td>0.0000</td>
<td>0.0000</td>
<td>0.9385431E-01</td>
<td>0.0</td>
</tr>
<tr>
<td>0.2000</td>
<td>0.0000</td>
<td>0.2297403E-03</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>0.6000</td>
<td>0.0000</td>
<td>0.8006346E-01</td>
<td>0.2793968E-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>0.5695011E-01</td>
<td>0.3725290E-08</td>
</tr>
<tr>
<td>1.2000</td>
<td>0.0000</td>
<td>0.4259761E-01</td>
<td>0.3725290E-08</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>0.1307478E-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>0.0000</td>
<td>0.0000</td>
<td>0.9399381E-01</td>
<td>0.0</td>
</tr>
<tr>
<td>0.2000</td>
<td>0.0000</td>
<td>-0.1677619E-01</td>
<td>0.0</td>
</tr>
<tr>
<td>0.4000</td>
<td>0.0000</td>
<td>-0.2099303E-01</td>
<td>0.0</td>
</tr>
<tr>
<td>0.6000</td>
<td>0.0000</td>
<td>-0.2019734E-01</td>
<td>0.0</td>
</tr>
<tr>
<td>0.8000</td>
<td>0.0000</td>
<td>-0.1780687E-01</td>
<td>0.0</td>
</tr>
<tr>
<td>1.0000</td>
<td>0.0000</td>
<td>-0.1387723E-01</td>
<td>-0.1396984E-08</td>
</tr>
<tr>
<td>1.2000</td>
<td>0.0000</td>
<td>-0.9384889E-02</td>
<td>0.0</td>
</tr>
<tr>
<td>1.4000</td>
<td>0.0000</td>
<td>-0.5123563E-02</td>
<td>0.0</td>
</tr>
<tr>
<td>1.6000</td>
<td>0.0000</td>
<td>-0.1570206E-02</td>
<td>0.0</td>
</tr>
<tr>
<td>1.8000</td>
<td>0.0000</td>
<td>-0.1145449E-02</td>
<td>0.0</td>
</tr>
<tr>
<td>2.0000</td>
<td>0.0000</td>
<td>0.3161710E-02</td>
<td>0.0</td>
</tr>
<tr>
<td>2.2000</td>
<td>0.0000</td>
<td>0.4723225E-02</td>
<td>0.0</td>
</tr>
<tr>
<td>2.4000</td>
<td>0.0000</td>
<td>0.5990129E-02</td>
<td>-0.2561137E-08</td>
</tr>
<tr>
<td>2.6000</td>
<td>0.0000</td>
<td>0.6880980E-02</td>
<td>-0.2095476E-08</td>
</tr>
<tr>
<td>2.8000</td>
<td>0.0000</td>
<td>0.7003162E-02</td>
<td>0.0</td>
</tr>
<tr>
<td>3.0000</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>-0.1838829E+01</td>
<td>0.2793968E+08</td>
</tr>
<tr>
<td>-0.0000</td>
<td>-9.2000</td>
<td>-0.1313793E+01</td>
<td>0.3725290E+08</td>
</tr>
<tr>
<td>-0.0000</td>
<td>-8.8000</td>
<td>-0.756035E+00</td>
<td>0.5122274E+08</td>
</tr>
<tr>
<td>-0.0000</td>
<td>-8.4000</td>
<td>-0.4923782E+00</td>
<td>0.1396984E+08</td>
</tr>
<tr>
<td>-0.0000</td>
<td>-8.0000</td>
<td>-0.872068E+00</td>
<td>0.1396984E+08</td>
</tr>
<tr>
<td>-0.0000</td>
<td>-7.6000</td>
<td>-0.3307122E+00</td>
<td>0.3492460E+08</td>
</tr>
<tr>
<td>-0.0000</td>
<td>-7.2000</td>
<td>-0.8870248E+00</td>
<td>0.3026798E+08</td>
</tr>
<tr>
<td>-0.0000</td>
<td>-6.8000</td>
<td>-0.8532621E+00</td>
<td>0.3026798E+08</td>
</tr>
<tr>
<td>-0.0000</td>
<td>-6.4000</td>
<td>-0.6700888E+00</td>
<td>0.1396984E+08</td>
</tr>
<tr>
<td>-0.0000</td>
<td>-6.0000</td>
<td>-0.3113743E+00</td>
<td>0.3725290E+08</td>
</tr>
<tr>
<td>-0.0000</td>
<td>-5.6000</td>
<td>-0.9560350E+00</td>
<td>0.4423782E+08</td>
</tr>
<tr>
<td>-0.0000</td>
<td>-5.2000</td>
<td>-0.1313743E+00</td>
<td>0.3725290E+08</td>
</tr>
<tr>
<td>-0.0000</td>
<td>-4.8000</td>
<td>-0.1624599E+00</td>
<td>0.1396984E+08</td>
</tr>
<tr>
<td>-0.0000</td>
<td>-4.4000</td>
<td>-0.2417203E+00</td>
<td>0.9313226E+08</td>
</tr>
<tr>
<td>-0.0000</td>
<td>-4.0000</td>
<td>-0.3219855E+00</td>
<td>0.1862645E+08</td>
</tr>
<tr>
<td>-0.0000</td>
<td>-3.6000</td>
<td>-0.4015272E+00</td>
<td>0.2328306E+08</td>
</tr>
<tr>
<td>-0.0000</td>
<td>-3.2000</td>
<td>-0.4800258E+00</td>
<td>0.2793968E+08</td>
</tr>
<tr>
<td>-0.0000</td>
<td>-2.8000</td>
<td>-0.5578430E+00</td>
<td>0.2328306E+08</td>
</tr>
<tr>
<td>-0.0000</td>
<td>-2.4000</td>
<td>-0.6350774E+00</td>
<td>0.1629815E+08</td>
</tr>
<tr>
<td>-0.0000</td>
<td>-2.0000</td>
<td>-0.7107019E+00</td>
<td>0.2328306E+08</td>
</tr>
<tr>
<td>-0.0000</td>
<td>-1.6000</td>
<td>-0.7821018E+00</td>
<td>0.1629815E+08</td>
</tr>
<tr>
<td>-0.0000</td>
<td>-1.2000</td>
<td>-0.8452117E+00</td>
<td>0.5820766E+08</td>
</tr>
<tr>
<td>-0.0000</td>
<td>-0.8000</td>
<td>-0.8951896E+00</td>
<td>0.1600711E+08</td>
</tr>
<tr>
<td>-0.0000</td>
<td>-0.4000</td>
<td>-0.9276407E+00</td>
<td>0.4656613E+08</td>
</tr>
<tr>
<td>-0.0000</td>
<td>0.0000</td>
<td>-0.9385431E+00</td>
<td>0.7105427E+14</td>
</tr>
<tr>
<td>-0.0000</td>
<td>0.4000</td>
<td>-0.9274071E+00</td>
<td>0.7130434E+09</td>
</tr>
<tr>
<td>-0.0000</td>
<td>0.8000</td>
<td>-0.8951896E+00</td>
<td>0.8731149E+10</td>
</tr>
<tr>
<td>-0.0000</td>
<td>1.2000</td>
<td>-0.8452106E+00</td>
<td>0.8731149E+10</td>
</tr>
<tr>
<td>-0.0000</td>
<td>1.6000</td>
<td>-0.7821023E+00</td>
<td>0.0000000000</td>
</tr>
<tr>
<td>-0.0000</td>
<td>2.0000</td>
<td>-0.7107037E+00</td>
<td>0.2095476E+08</td>
</tr>
<tr>
<td>-0.0000</td>
<td>2.4000</td>
<td>-0.6350780E+00</td>
<td>0.6984919E+09</td>
</tr>
<tr>
<td>-0.0000</td>
<td>2.8000</td>
<td>-0.5578437E+00</td>
<td>0.1396984E+08</td>
</tr>
<tr>
<td>-0.0000</td>
<td>3.2000</td>
<td>-0.4800263E+00</td>
<td>0.1396984E+08</td>
</tr>
<tr>
<td>-0.0000</td>
<td>3.6000</td>
<td>-0.4015278E+00</td>
<td>0.3026798E+08</td>
</tr>
<tr>
<td>-0.0000</td>
<td>4.0000</td>
<td>-0.3219860E+00</td>
<td>0.1862645E+08</td>
</tr>
<tr>
<td>-0.0000</td>
<td>4.4000</td>
<td>-0.2417206E+00</td>
<td>0.1862645E+08</td>
</tr>
<tr>
<td>-0.0000</td>
<td>4.8000</td>
<td>-0.1624606E+00</td>
<td>0.1862645E+08</td>
</tr>
<tr>
<td>-0.0000</td>
<td>5.2000</td>
<td>-0.8754738E+00</td>
<td>0.1862645E+08</td>
</tr>
<tr>
<td>-0.0000</td>
<td>5.6000</td>
<td>-0.2147641E+00</td>
<td>0.1862645E+08</td>
</tr>
<tr>
<td>-0.0000</td>
<td>6.0000</td>
<td>-0.3113743E+00</td>
<td>0.1862645E+08</td>
</tr>
<tr>
<td>-0.0000</td>
<td>6.4000</td>
<td>-0.6700821E+00</td>
<td>0.1862645E+08</td>
</tr>
<tr>
<td>-0.0000</td>
<td>6.8000</td>
<td>-0.8532569E+00</td>
<td>0.1862645E+08</td>
</tr>
<tr>
<td>-0.0000</td>
<td>7.2000</td>
<td>-0.8870188E+00</td>
<td>0.1862645E+08</td>
</tr>
<tr>
<td>-0.0000</td>
<td>7.6000</td>
<td>-0.3071222E+00</td>
<td>0.2561137E+08</td>
</tr>
<tr>
<td>-0.0000</td>
<td>8.0000</td>
<td>-0.7621353E+00</td>
<td>0.3492460E+08</td>
</tr>
<tr>
<td>-0.0000</td>
<td>8.4000</td>
<td>-0.7850584E+00</td>
<td>0.2328306E+08</td>
</tr>
<tr>
<td>-0.0000</td>
<td>8.8000</td>
<td>-0.9560246E+00</td>
<td>0.4423782E+08</td>
</tr>
<tr>
<td>-0.0000</td>
<td>9.2000</td>
<td>-0.1313743E+00</td>
<td>0.3725290E+08</td>
</tr>
<tr>
<td>-0.0000</td>
<td>9.6000</td>
<td>-0.1838810E+00</td>
<td>0.4423782E+08</td>
</tr>
<tr>
<td>-0.0000</td>
<td>10.0000</td>
<td>-0.2455102E+00</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.

<table>
<thead>
<tr>
<th>Card Column</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 11 21</td>
</tr>
<tr>
<td>-------------</td>
</tr>
<tr>
<td>1 0007700020</td>
</tr>
<tr>
<td>2 0015100151</td>
</tr>
<tr>
<td>3 1111</td>
</tr>
<tr>
<td>4 0.0</td>
</tr>
<tr>
<td>5 0.0</td>
</tr>
<tr>
<td>6 -35.0</td>
</tr>
<tr>
<td>7 -30.0</td>
</tr>
<tr>
<td>8 0000</td>
</tr>
<tr>
<td>9 0000</td>
</tr>
</tbody>
</table>

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

8.4 Output from ANTDATA

The following is the printout from computer program ANTDATA.

PLOT OUTPUT FOR PATTERN 77:

U-AXISPROFILE 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 BEINGgenerated. 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>