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

1. Introduction. .................................................. 1
   1.1 The Need for New Approaches. ............................... 1
   1.2 A Practical Approach to Antenna Synthesis. ................. 2
   1.3 Scope of the Research. ...................................... 4

   2.1 Equivalent Currents. ........................................ 6
   2.2 Representing Antennas as Finite Apertures .................. 8
   2.3 Vector Radiation Fields. .................................... 11
   2.4 Antenna Hardware Parameter Control ......................... 15

3. The Iterative Sampling Method for Planar Sources .......... 17
   3.1 History of the Iterative Sampling Method .................... 17
   3.2 Pattern Evaluation - What is an Acceptable Pattern ? ....... 18
   3.3 The Integral Equation. ...................................... 19
   3.4 Mathematical Development of the Method ...................... 20
   3.5 Common Antenna Types ....................................... 24
      3.5.1 Line Sources. .......................................... 24
      3.5.2 Linear Array. ......................................... 25
      3.5.3 Rectangular Aperture. ................................ 25
      3.5.4 Rectangular Array ..................................... 26
      3.5.5 Circular Aperture ..................................... 26
      3.5.6 Arbitrary Planar Array. ................................ 27
   3.6 Calculation of Directivity ................................ 28

4. Examples of Computer Antenna Synthesis ..................... 32
   4.1 Common Antenna Types ....................................... 32
   4.2 Linear Antenna Synthesis ................................ 39
   4.3 Rectangular Antenna Synthesis. ............................. 47

5. Conclusions .................................................... 60

References. ....................................................... 62

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

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

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

1. **Introduction**

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

1.1 The Need for New Approaches

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

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

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

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

1.2 A Practical Approach to Antenna Synthesis

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

a. Ability to form the necessary number of main beams.
b. Isolation levels between beams.
c. Polarization control.
d. Power handling capability.
e. Center frequency of operation.
f. Bandwidth
g. Efficiency
h. Size
i. Weight
j. Reliability
k. Pattern control (scanning and beam reshaping for changing user needs).

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

The antenna design problem is then described in three stages:

1. Listing the antenna types which possibly can meet system specifications.

2. Determining the excitation of each antenna type required to meet the pattern requirements.

3. Singling out the one "best" antenna system.

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

1.3 Scope of the Research

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

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

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

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

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

### 2.1 Equivalent Currents

It is often convenient to use equivalent currents to obtain the radiation fields from an antenna. Suppose the source antenna is entirely enclosed by a closed surface $S$. Let $\mathbf{E}_1$ and $\mathbf{H}_1$ 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:

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

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

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

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

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

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

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

\[
\begin{align*}
\vec{J}_S &= \hat{z} \times \vec{H}_a \quad (2-5) \\
\vec{J}_{MS} &= -\hat{z} \times \vec{E}_a \quad (2-6)
\end{align*}
\]

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

\[
\begin{align*}
E_\theta &= -j\omega A_\theta - j\eta_0 A_{\phi}^M \\
E_\phi &= -j\omega A_\phi + j\eta_0 A_{\theta}^M \quad (2-7, 2-8)
\end{align*}
\]

where \( \eta_0 = \sqrt{\mu_0 / \epsilon_0} \) and the magnetic fields are found using the plane wave relation

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

$$\mathbf{J}_{MS} = \mathbf{J}_{MSx} \hat{x} = z \times E_y \hat{y} = E_{ay} \hat{x}$$  \hspace{1cm} (2-10)

and

$$A_{Mx} = \frac{\varepsilon_0}{4\pi r} e^{-jkr} \int \int_{\text{aperture}} 2 E_{ay} e^{jk\mathbf{r} \cdot \mathbf{r'}} \, dx' dy'$$  \hspace{1cm} (2-11)

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

$$A_{M\theta} = \cos \theta \cos \phi A_{Mx}$$  \hspace{1cm} (2-12)

$$A_{M\phi} = - \sin \phi A_{Mx}$$  \hspace{1cm} (2-13)

So

$$E_\theta = -j\omega_0 A_{M\phi}$$

$$= +j\omega_0 \sin \phi A_{Mx}$$  \hspace{1cm} (2-14)

$$E_\phi = +j\omega_0 A_{M\theta}$$

$$= j\omega_0 \cos \theta \cos \phi A_{Mx}$$  \hspace{1cm} (2-15)

And

$$A_{Mx} = \frac{\varepsilon_0}{2\pi r} e^{-jkr} \int \int_{\text{aperture}} E_{ay} e^{jk(x' \sin \theta \cos \phi + y' \sin \theta \sin \phi)} \, dx' \, dy'$$  \hspace{1cm} (2-16)

So

$$E_\theta = \frac{jke^{-jkr}}{2\pi r} \sin \phi F_y$$  \hspace{1cm} (2-17)

$$E_\phi = \frac{jke^{-jkr}}{2\pi r} \cos \theta \cos \phi F_y$$  \hspace{1cm} (2-18)

where

$$F_y = \int \int_{\text{aperture}} E_{ay} e^{jk(x' \sin \theta \cos \phi + y' \sin \theta \sin \phi)} \, dx' \, dy'$$  \hspace{1cm} (2-19)
If there is an \( x \)-directed component of the aperture electric field the far-field expressions become

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

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

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

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

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

where

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

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

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

Frequently aperture antennas are not used, but rather one wishes to relate directly to a source current. In this case \( \mathbf{J}_S \) is an actual current and its Fourier transform \( \hat{A} \) 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

\[
\mathbf{J}_S = \hat{x} J_{Sx}(x') J_{Sx}(y') + \hat{y} J_{Sy}(x') J_{Sy}(y') \tag{2-25}
\]

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

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

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

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

\[ G_{\phi x} = -\cos \theta \sin \phi \quad G_{\phi y} = \cos \theta \cos \phi \]

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):

\[
[F] = [G]^{-1}[E] \tag{2-33}
\]

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

\[
F_x(\theta,\phi) = \int \int E_{ax}(x',y')e^{jk(x' \sin \theta \cos \phi + y' \sin \theta \sin \phi)} dx'dy' \tag{2-34}
\]

\[
F_y(\theta,\phi) = \int \int E_{ay}(x',y')e^{jk(x' \sin \theta \cos \phi + y' \sin \theta \sin \phi)} dx'dy' \tag{2-35}
\]

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

\[
[G] = \begin{bmatrix}
\cos \theta \cos \phi & -\cos \theta \sin \phi \\
-\sin \phi & -\cos \phi
\end{bmatrix} \tag{2-36}
\]

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

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

where we let

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

becomes

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

where

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

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

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

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

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

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

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

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

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

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

Since the current is y-directed we have

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

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

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

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

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

Now

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

where

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

since \( y_m' = 0 \).

2.4 Antenna Hardware Parameter Control

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

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

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

where

\[
[V] = \begin{bmatrix}
V_1 \\
V_2 \\
\vdots
\end{bmatrix}
\hspace{1cm} = \text{terminal voltage matrix (computed from (2-51))}
\]

\[
[I] = \begin{bmatrix}
I_1 \\
I_2 \\
\vdots
\end{bmatrix}
\hspace{1cm} = \text{terminal current matrix (found by synthesis)}
\]

\[
[Z] = \begin{bmatrix}
Z_{11} & Z_{12} & \cdots \\
Z_{21} & Z_{22} & \cdots \\
\vdots & & \ddots
\end{bmatrix}
\hspace{1cm} = \text{mutual impedance matrix (known from calculations or experiment)}
\]
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

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

where

$$u = \sin \theta \cos \phi$$  \hspace{1cm} (3-2)$$

$$v = \sin \theta \sin \phi.$$  \hspace{1cm} (3-3)$$

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

$$s = x'/\lambda$$  \hspace{1cm} (3-4)$$

$$t = y'/\lambda$$

and source function
\[ f(s,t) = \begin{cases} \lambda^2 F_a(x',y') & \text{over the aperture} \\ 0 & \text{elsewhere} \end{cases} \] (3-5)

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

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

(3-7)

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

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

(3-8)

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

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

(3-9)

In other words, at the point \( (u_n^{(i)}, v_n^{(i)}) \) the amount \( a_n^{(i)} \) is added to the \( (i-1) \)th iteration pattern to obtain the desired pattern value at that point. The pattern is, of course, also changed at other points. If several corrections are applied in a given iteration of (3-8) the pattern will equal the desired pattern at the sample points only if the samples are uncorrelated. However, if the sample points are relatively far apart the correlation between samples can be very low. For a given iteration there are usually only a few corrections, frequently positioned to maintain symmetry. Thus if one abandons the idea that samples must be completely uncorrelated and replaces it with the concept that they should not be strongly correlated, the method is much more powerful and flexible. Also since the type of correction function is not based upon satisfying the property of being uncorrelated, the designer can choose one that is convenient.
For a given iteration then we have forced the pattern to be very close (exactly equal if only one correction is used) to the desired pattern at the sample points. The entire pattern is then recomputed and new corrections are evaluated using (3-9). It has been found that the position of the samples \((u_n^{(i)}, v_n^{(i)})\) 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:

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

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

(3-11)

where

\[ \Delta f(i)(s,t) = \sum_n a_n(1) g_n(s,t) \]  

(3-12)

The pattern \( F(\mathbf{K})(u,v) \) and source \( f(\mathbf{K})(s,t) \) 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

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

(3-13)

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

\[ g_n(i)(s,t) = \sum_{\ell} \sum_{m} g(\ell m) \delta(s-s_{\ell m}, t-t_{m}) \]  

(3-14)

for arrays. Then (3-10) becomes

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

(3-15)

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

\[ \Delta f(i)(s,t) = \sum_n a_n(i) \sum_{\ell} \sum_{m} g(\ell m) \delta(s-s_{\ell m}, t-t_{m}) \]  

(3-16)

and let

\[ \Delta f(i)(s,t) = \sum_{\ell} \sum_{m} I(\ell m) \delta(s-s_{\ell m}, t-t_{m}) \]  

(3-17)

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

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, \( v_n^{(i)} \). The source correction function is

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

(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

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

(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

$$g_n(i)(t) = \begin{cases} L_y^{-1} \left(1 - 2|t|/L_y\right) \exp(-j2\pi v_n(i)t) & |t| \leq L_y/2 \\ 0 & \text{elsewhere} \end{cases}$$

(3-22)

The corresponding pattern found from (3-10) is

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

(3-23)

3.5.2 Linear Array

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

$$g_{nm}(i) = \frac{i}{P} \exp(-j2\pi v_n(i)t_m)$$

(3-24)

where $t_m$ are the positions of the elements and equal $md_y$ and $P$ is the total number of elements. The corresponding pattern is

$$G(u-u_n(i)) = \frac{\sin[P(v-v_n(i))\pi d_y]}{P \sin[(v-v_n(i))\pi d_y]}$$

(3-25)

3.5.3 Rectangular Aperture

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

$$g_n(i)(s,t) = \begin{cases} L_x^{-1} L_y^{-1} \exp(-j2\pi(u_n(i)s + v_n(i)t)) & |s| \leq L_x/2 \\ 0 & |t| \leq L_y/2 \end{cases}$$

(3-26)

The pattern is

$$G(u-u_n(i), v-v_n(i)) = \frac{\sin[L_x(u-u_n(i))\pi]}{L_x(u-u_n(i))\pi} \frac{\sin[L_y(v-v_n(i))\pi]}{L_y(v-v_n(i))\pi}$$

(3-27)
3.5.4 Rectangular Array

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

\[
\ell_{n}\delta m \left( i \right) = \frac{1}{P_x P_y} \exp \left[-j2\pi(u_n(i)+v_n(i))\right] (3-28)
\]

The pattern is

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

3.5.5 Circular Aperture

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

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

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

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

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

(3-32)

where

\[
C = [(u-u_n (i))^2 + (v-v_n (i))^2]^{1/2} 
\]

(3-34)

\[
\alpha = \tan^{-1} \frac{u-u_n (i)}{v-v_n (i)} 
\]

(3-35)

Now (3-33) is easily integrated as

\[
G (u-u_n (i), v-v_n (i)) = \frac{2\pi}{\pi a_\lambda} \int_{0}^{a_\lambda} J_0 (2\pi \rho C) \rho'_\lambda \ d\rho'_\lambda \\
= \frac{J_1 (2\pi a_\lambda C)}{2\pi a_\lambda} 
\]

(3-37)

(3-38)

If \( u_n (i) = v_n (i) = 0 \) we have

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

(3-39)

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

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

3.5.6 Arbitrary Planar Array

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

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

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

3.6 Calculation of Directivity

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

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

\[ D = \frac{4\pi}{\Omega_A} \] (3-41)

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

\[ \Omega_A = \int_{0}^{2\pi} \int_{0}^{\pi} |F(\theta, \phi)|^2 \sin \theta \, d\theta \, d\phi \] (3-42)

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

\[ d\Omega = \sin \theta \, d\theta \, d\phi \] (3-43)
It is frequently convenient to transform from the $\theta, \phi$ space to the $u, v$ plane using

\[ u = \sin \theta \cos \phi \]
\[ v = \sin \theta \sin \phi \]  

(3-44)

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

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

(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 $F(\theta, \phi)$ is exact).

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

\[ du \ dv = \cos \theta \ d\Omega \]  

(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

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

Thus (3-45) becomes

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

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

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

where \( J \) is the Jacobian given by

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

and

\[ \begin{vmatrix} \frac{\partial u}{\partial \theta} & \frac{\partial u}{\partial \phi} \\ \frac{\partial v}{\partial \theta} & \frac{\partial v}{\partial \phi} \end{vmatrix} \]

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

\[ = \cos \theta \sin \theta \quad (3-53) \]

So

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

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

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)

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

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

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

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

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

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

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

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

4.1 Common Antenna Types

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

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

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

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

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

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

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

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

PATTERN = 56
Synthesis capability is provided for circular aperture by inclusion of an elementary pattern from such an aperture which is uniformly excited in amplitude. Figs. 4.9 and 4.10 show u and v axis profiles of the pattern from a uniform amplitude, uniform phase, five wavelength radius circular aperture. These plots are, of course, identical. A contour map of this pattern in the uv - plane for the whole visible region is shown in Fig. 4.11. The three dimensional view of the pattern is shown in Fig. 4.12.

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

4.2 Linear Antenna Synthesis

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

<table>
<thead>
<tr>
<th>v</th>
<th>F_d(v)</th>
<th>F_u(v)</th>
<th>F_L(v)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></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 4.1 Pattern Parameters for Elementary Correction Patterns

<table>
<thead>
<tr>
<th>Antenna Type - ITYPE</th>
<th>Source Dimensions (λ)</th>
<th>Beam Width</th>
<th>Side Lobe Level (dB)</th>
<th>Directivity (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Theory</td>
<td>Computer</td>
<td>Theory</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Computer</td>
<td>Theory</td>
<td>Computer</td>
</tr>
<tr>
<td>Uniform line</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>source - 1</td>
<td>$L_y \lambda = 10$</td>
<td>0.0886</td>
<td>0.0886</td>
<td>-13.3</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-13.3</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>13.1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>13.2</td>
</tr>
<tr>
<td>Uniform linear</td>
<td>$P_y = 21$, $d_x \lambda = 0.5$</td>
<td>0.0886</td>
<td>0.0886</td>
<td>-13.3</td>
</tr>
<tr>
<td>array - 2</td>
<td></td>
<td></td>
<td></td>
<td>-13.3</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>13.1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>13.3</td>
</tr>
<tr>
<td>Triangular line</td>
<td>$L_y \lambda = 10$</td>
<td>0.128</td>
<td>0.128</td>
<td>-26.6</td>
</tr>
<tr>
<td>source - 3</td>
<td></td>
<td></td>
<td></td>
<td>-26.6</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>11.8</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>12.7</td>
</tr>
<tr>
<td>Uniform rectangular</td>
<td>$L_x \lambda = 10$</td>
<td>0.0886</td>
<td>0.0886</td>
<td>-13.3</td>
</tr>
<tr>
<td>aperture - 4</td>
<td></td>
<td></td>
<td></td>
<td>-13.3</td>
</tr>
<tr>
<td></td>
<td>$L_y \lambda = 20$</td>
<td>0.0443</td>
<td>0.0443</td>
<td>-13.3</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-13.3</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>34.0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>34.0</td>
</tr>
<tr>
<td>Uniform rectangular</td>
<td>$P_x = 21$, $d_x \lambda = 0.5$</td>
<td>0.0886</td>
<td>0.0886</td>
<td>-13.3</td>
</tr>
<tr>
<td>array - 5</td>
<td>$P_y = 41$, $d_y \lambda = 0.5$</td>
<td>0.0443</td>
<td>0.0443</td>
<td>-13.3</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>34.0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>33.9</td>
</tr>
<tr>
<td>Uniform circular</td>
<td>$a_\lambda = 5$</td>
<td>0.102</td>
<td>0.102</td>
<td>-17.6</td>
</tr>
<tr>
<td>aperture - 6</td>
<td></td>
<td></td>
<td></td>
<td>-17.6</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>29.95</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>29.86</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 $t_m = 0., \pm 0.496, \pm 0.983, \pm 1.926, \pm 2.372, \pm 3.188, \pm 3.545$. The original pattern for this 13 element array was formed using the same Woodward-Lawson specifications as the first example of this section. The original pattern has an excursion of 4.51 dB above the desired level of 0. dB over the main (square) beam region and none below. The side lobe level is -9.2 dB. Thus, in this case, the original pattern is quite far from desired performance. The final pattern obtained from ANTSYN took 15 iterations to meet the specifications. In fact, all side lobes were below -22. dB. The pattern is shown in Fig. 4.17. This compares to a side lobe level of -18.6 dB from [14]. The element currents for the two methods are similar.

### 4.3 Rectangular Antenna Synthesis

The multibeam capability of this technique is displayed with the synthesis of a pattern with pencil beams positioned at (0.5, 0.5), (0.5, -0.5) (-0.5, -0.5), and (0.5, 0.5). The side lobe upper limit was specified to be -25 dB 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 -25 dB and no lower limit.

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

| $|v| \leq 0.26$ | $F_d(v)$ | $F_u(v)$ | $F_L(v)$ |
|----------------|----------|----------|----------|
| 0.00 dB +0.42 dB -0.42 dB | 0.00 dB +0.42 dB -0.42 dB |
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 $|v| < 1.0$ 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>-20.0</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 -20.0 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 -40.0 dB in 5.0 dB steps and the -35. and -40.0 dB contours are dotted. The three-dimensional view is shown in Figure 4.25.
Figure 4.18 A multiple beam radiation pattern profile in the $u$-direction for $v = 0.5$ synthesized using a 10 by 10 wavelength aperture antenna. The visible region is for $|u| \leq 0.866$.

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

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

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

<table>
<thead>
<tr>
<th>(u, v)</th>
<th>( F_d(u,v) )</th>
<th>( F_H(u,v) )</th>
<th>( F_L(u,v) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>(-0.2&lt;u&lt;0.2)</td>
<td>0 dB</td>
<td>0.5 dB</td>
<td>-0.5 dB</td>
</tr>
<tr>
<td>(-0.05&lt;v&lt;0.05)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(0.34&lt;u&lt;0.50)</td>
<td>( \infty )</td>
<td>-25 dB</td>
<td>unspecified</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 $J_1(x)$ routine could be written to avoid the general purpose IBM SSP routine.

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


6. Appendix: The ANTSYN Computer Program

6.1 Introduction

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

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

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

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

The subroutines such as ANTSYN, SEARCH, UPDATE, CHECK, and 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 NUMPAT, NUMTRK, NUMSKP and IPASS are read off of the storage unit. NUMPAT is the pattern number assigned to the previous job. The program adds one to NUMPAT to form the current job pattern number. NUMTRK is the track number on disk where the previous job data was stored. NUMSKP is an array whose subscripts correspond to disk storage track numbers. If this number is 0 or 1 there is not data stored on that track. This information is used in step 10 to write onto disk.

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

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

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

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

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

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

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

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

Step 5. The patterns FDES, FU, and FL are to be loaded in this step. This can be done in two ways. First programming can be provided in Subroutine DESPAT to give a value to these 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>( 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>( f^{(K)}(s,t) )</td>
<td>CURR(M,N) &amp; CURI(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 direc-
tivities DIRORG and DIRFNL; 0 No, 1 Yes - Default is 0. Original
pattern is to be of Woodward-Lawson type.

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

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

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

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

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

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

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 distribu-
tion profile or list; 0 None, 1 Profile (U and/or V axis), 2 List
(primarily for use with ITYPE = 7) - Default is 0.

ICURPT  Input variable controlling print out of a listing of initial cur-
rent 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 specifcifications 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.

DELCON 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>Variable</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>
<tr>
<td>Variable</td>
<td>Description</td>
</tr>
<tr>
<td>----------</td>
<td>-------------</td>
</tr>
<tr>
<td>STARTU</td>
<td>Input variable - Starting point in U direction for pattern comparisons and print outs.</td>
</tr>
<tr>
<td>STARTV</td>
<td>Input variable - Starting point in V direction for pattern comparisons and print outs.</td>
</tr>
<tr>
<td>T</td>
<td>Source coordinate Y normalized to a wavelength.</td>
</tr>
<tr>
<td>TT( )</td>
<td>Antenna array element position in T direction - Input variable for ITYPE = 7.</td>
</tr>
<tr>
<td>U</td>
<td>Pattern coordinate.</td>
</tr>
<tr>
<td>UORG( )</td>
<td>Input variable - Positions of sample points for original pattern in U direction.</td>
</tr>
<tr>
<td>US( )</td>
<td>Positions of corrections (samples) in U direction.</td>
</tr>
<tr>
<td>V</td>
<td>Pattern coordinate.</td>
</tr>
<tr>
<td>VORG( )</td>
<td>Input variable - Positions of sample points for original pattern in V direction.</td>
</tr>
<tr>
<td>VS( )</td>
<td>Positions of corrections (samples) in V direction.</td>
</tr>
</tbody>
</table>
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 $20.\times\log_{10}(\ )$. 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, ISYM, 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 ISYM. 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, FNM, 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 U1 and V1 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>FORCPT = 1</td>
</tr>
<tr>
<td>Final pattern</td>
<td>FDBPT = 1</td>
</tr>
<tr>
<td>Original current</td>
<td>ICURPT = 1</td>
</tr>
<tr>
<td>Final current</td>
<td>FCURPT = 1</td>
</tr>
</tbody>
</table>

**SUBROUTINE PROFIL** (DATA1, NPT, NUMPAT)

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

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

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

To invoke subroutine PROFIL, code the following variables in Namelist IPRINT.
Pattern Variable
Original FORGPR = 1
Final FDBPR = 1

SUBROUTINE 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, DELTAV, 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>FDBCN = 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

```
//F066A57 JOB  SC762, GOFEEY
/#MAIN TIME=3, REGION=220K, LINES=10, CARDS=10
/#PRIORITY PRIORITY
// EXEC FORT766,LIB2=SSPLIB
//F0RTSYSIN UD *
/*
//60-F122F01 DD RSN=ANTSIN, RES=762, UNIT=3330, VOL=SER=USERPK, LISP=S/IKK
//60SYSIN 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
   SPSOR
   SINPUT
   CURREN
   PRINT
   PROFIL
   CONCUR
   PATCCN
   LIST
   DESPAT
   DATE
   STIME
   BESJ

...STANDARD FORTRAN LIBRARY SUBPROGRAMS...

INPUT/OUTPUT SUPPORT:

FTC5FC01 (SYSIN) -- CARD READER
FTO6FC01 (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, FDESCP, FDESPR
INTEGER FORGPT, FORGCN, FORGPR, FDBPT, FDBCN, FDBPR, DIRECT
REAL FDES(51,51), FU(51,51), FL(51,51), F(51,51)
REAL US(500), VS(500), CORCOF(500), CURR(51,51), CURI(51,51)
REAL UORG(100), VORG(100)
REAL INITLS, INITLT

INTEGER FORGPT, FORGCN, FORGPR
COMMON /MPROG/ ICUR, NCUR
COMMON /START/ NORG, UORG, VORG, CURG
COMMON /PAT1/ PI, P2, P3, P4, P5, P6, PI, SS(400), IT(400), RK(400)
COMMON /PAT2/ 11, 12, 13, 14, 15
COMMON /LOC/ IYPE
DATA TITLE / 'ANTE', 'NNA', 'SYNT', 'HESI', 'S PK', 'ODRA',
'S', 'VERS', 'ION', '3 L', 'EVEL', '1', '73', '/164',
'S', 'VPI', 'E.E.', 'DEP', 'T. ', /
NAMELIST /PARAM/ IDISK, ISYM, ITMAX, DELTAU, DELTAV, STARTU, STARTV,
NMAX, NMAT, MCENT, NCENT

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

BEGIN PROCESSING
9999 CONTINUE
READ(22,1) NUMPAT, NUMTRK, NUMSKP, IPASS
READ(22,50) NUMPAT = NUMPAT + 1
CALL DATF(I1, J1, K1)
CALL STIME(IT)
IHR = IT/10000
IFR = IT - IHR*10000
FHR = IFR/10000.
FM = FHR*60.
IMIN = FM
IMSEC = (FM - IMIN)*60.
WRITE(6,1) I1, J1, K1, IHR, IMIN, ISEC, NUMPAT
1 FORMAT('1 ANTE', 'NNA', 'SYNT', 'HESI', 'S PK', 'ODRA',
'S', 'VERS', 'ION', '3 L', 'EVEL', '1', '73', '/164',
'S', 'VPI', 'E.E.', 'DEP', 'T. ', /
A-26
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  FDESR=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  FORGPR=0
62  DIRECT=0

63  FCNRM=1.0
64  ISUBC=0
65  DELTAS=C.
66  DELTAT=0.

INPUT

67  READ(5,PARAM)

REPRODUCIBILITY OF THE ORIGINAL PAGE IS POOR
CALL INPUT
IF(ITYPE.EQ.7.AND.ICURPR.EQ.1) ICURPR = 2
IF (ITYPE.EQ.7 .AND. FCURPR.EQ.1) FCURPR = 2
CALL LOCSCR(11,INITLS,INITLT)
CALL LOCSDR(MCUR,MCUR,FINALS,FINALT)
IF(*CUR.NE.1) DELTAS = (FINALS-INITLS)/(MCUR-1)
IF(*CUR.NE.1) DELTAT = (FINALT-INITLT)/(MCUR-1)
CALL DESPAT(FDES,FU,FL,MMAX,NMAX,STARTU,STARTV,$DELTAU,$DELTAV)

CALL INPUT
IF(FDESPT) 300, 300, 301
300 WRITE(6,302)
302 FORMAT(1H1,10X,'DESIRED PATTERN IN DB'.")
301 CALL PRINT(FDES,MMAX,NMAX,STARTU,STARTV,$DELTAU,$DELTAV)

300 IF(FDESPT) 303, 303, 304
304 CONTINUE
CALL PATCON(FDES,MMAX,NMAX,0,-0.5,1.3,0.2,STARTU,STARTV,$DELTAU,$DELTAV,NUMPAT,ISYM)
303 IF(FDESPT) 306, 306, 307
307 IF(MMAX.LE.1) GO TO 308
308 WRITE(6,31C) NUMPAT
310 FORMAT(1H1,10X,'U-AXIS PROFILE OF DESIRED PATTERN',14//
$12X,'U','16X, 'V','15X,'FDES(U,V)',10X,'FU(U,V)',12X,'FL(U,V)'/
V=STARTV+(NCENT-1)*DELTAV
DO 309 I=1,MMAX
U=STARTU+(I-1)*DELTAV
309 WRITE(6,311) U,V,FDES(I,NCENT),FU(I,NCENT),FL(I,NCENT)
311 FORMAT(10X,F7.4,10X,F7.4,10X,3(F9.2,10X))
308 IF(NMAX.EQ.1) GO TO 306
306 CONTINUE
303 IF(FDESPT) 306, 306, 307
307 IF(MMAX.LE.1) GO TO 308
308 WRITE(6,31C) NUMPAT
312 FORMAT(1H1,10X,'V-AXIS PROFILE OF DESIRED PATTERN',14//
$12X,'U','16X, 'V','15X,'FDES(U,V)',10X,'FU(U,V)',12X,'FL(U,V)'/
U=STARTU+(MCENT-1)*DELTAV
DO 313 J=1,NMAX
V=STARTV+(J-1)*DELTAV
313 WRITE(6,311) U,V,FDES(MCENT,J),FU(MCENT,J),FL(MCENT,J)
312 CONTINUE

ENTER ORIGINAL PATTERN
CALL CRGPR(F,MMAX,NMAX,STARTU,STARTV,$DELTAU,$DELTAV,$CURR,$CURR,MCUR,MCUR)

OUTPUT OF ORIGINAL PATTERN

IF(FORGPT) 400, 400, 401
400 WRITE(6,402)
401 FORMAT(1H1,10X,'INITIAL PATTERN')
402 CALL PRINT(F,MMAX,NMAX,STARTU,STARTV,$DELTAU,$DELTAV)

IF(FORGPT) 403, 403, 404
404 CONTINUE
CALL PATCON(F,MMAX,NMAX,0,-0.5,1.3,0.2,STARTU,STARTV,$DELTAU,$DELTAV,NUMPAT,ISYM)
403 IF(FORGPT) 406, 406, 407
407 CONTINUE
DO 408 J=1,401
U=(J-1)*C.C05-1.C
V=U
SUMU=0.
SUMV=C.
DO 409 K=1,NORG
SUMU=SUMU+CORG(K)*PAT(U-UORG(K),-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(MMAX.LE.1) GO TO 2801
WRITE(6,410)
FORMAT(1HI,25X,'U-AXIS PROFILE OF INITIAL PATTERN*')
CALL PROFIL(DATA1,401,NUMPAT)
2801 IF(MMAX.LE.1) GO TO 406
WRITE(6,411)
FORMAT(1HI,25X,'V-AXIS PROFILE OF INITIAL PATTERN*')
CALL PROFIL(DATA2,401,NUMPAT)
CONTINUE
IF(ICURPT) 500,501,504
501 WRITE(6,502)
FORMAT(1HI,25X,'INITIAL CURR')
CALL PRINT(CURR,MCUR,NCUR,INITLS,INITLT,DELTAS,DELTAT)
WRITE(6,503)
FORMAT(1HI,25X,'INITIAL CURR')
CALL CONTUR(MCUR,NCUR,0.005,-0.04,0.04,0.0,CURI.,NUPAT)
WRITE(6,506)
FORMAT(1HI,25X,'INITIAL CURR')
CALL CONTUR(MCUR,NCUR,0.005,-0.04,0.04,0.0,CURI,NUMPAT)
503 IF(ICURPR-1) 506,507,514
504 WRITE(6,505)
FORMAT(1HI,25X,'INITIAL CURR')
CALL PRINT(CURR,MCUR,NCUR,INITLS,INITLT,UELTAS,UELTAT)
507 IF(MCUR.LT.1) GO TO 508
WRITE(6,510)
FORMAT(1HI,10X,'S AXIS PROFILE OF INITIAL CURRENT*//' $13X,'S',17X,'T',18X,'REAL',12X,'IMAGINARY',10X,'MAGNITUDE', $12X,'PHASE*/')
J=NCUR/2+1
DO 509 J=1,MCUR
CALL LOCSGR(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))*57.295795
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))
IF(NCUR.LE.1)
  GC
  GO TO 506
WRITE(6,512)
165
   IF(NCUR.LE.1)
   GC
   TO 506
WRITE(6,512)
FORMAT(IH1,1OX,'T AXIS PROFILE UF INITIAL CURRENT'//
$13X,'S',17X,'T',18X,'REAL',12X,'IMAGINARY',10X,'MAGNITUDE',
$12X,'PHASE'//)
I=MCUR/2+1
DC 513 J=1,NCUR
CALL LCSCOR(I,J,S,T,ITYPE)
CR=CURR(I,J)
CI=CURI(I,J)
AMAG=SQRT(CR*CR+CI*CI)
   IF(AMAG.EQ.0.) APH=0.
   IF(AMAG.EQ.0.) GO TO 513
APH=ATAN2(CI,CR)*57.2957795
513 WRITE(6,511) S,T,CR,CI,AMAG,APH
GO TO 506
WRITE(6,515)
515 FORMAT(IH1///10X,'INITIAL ELEMENT CURRENTS'//$13X,'S',15X,'T',15X,'CURRI,I X,ICURI')
CALL LIST(CURR,CURI,MCUR,NCUR)
CONTINUE
E=0
WRITE(6,4747)
4747 FORMAT(IH1)
CALL ANTSYN(ISUC,MMAX,NMAX,FDES,FU,FL,ITRMAX,ISYMM,CORC(JF,IC,US,
S,VS,STARTU,DELTAV,STARTV,DELTAV,MCENT,NCENT,ITER,FNORM,F)
PRINT OUT RESULTS
WRITE(6,391)
391 FORMAT(H1,5?X,'-- FINAL COEFFICIENTS --'//45X,'J',7X,
$'US(J)',7X,'VS(J)',5X,'CURRI,I X,ICURI')
IF(IE.EQ.0) WRITE(6,977)
977 FORMAT(10X,'NC ITERATIONS PERFORMED')
IF(IE.EQ.0) GO TO 978
498 WRITE(6,35) J,US(J),VS(J),CORC(J)
35 FORMAT(44X,I3,5X,F7.4,5X,F7.4,5X,F7.4)
WRITE(6,497) ITER
497 FORMAT(H0,9X,'NUMBER OF ITERATIONS = ',16)
WRITE(6,496) FNORM
496 FORMAT(H0,9X,'FNORM = ',F10.5)
WRITE(6,976) NUMPAT
976 FORMAT(H0,9X,'PATTERN NUMBER = NUMPAT = ',15)
OUTPUT FINAL PATTERN IN DB
978 CONTINUE
CO 29 J=1,MMAX
CO 29 K=1,NMAX
IF(F(J,K)) 290,289,290
289 F(J,K)=-2CC.
204 GO TO 29
205 290 F(J,K)=20.*ALCG10(ABS(F(J,K)))
206 29 CONTINUE

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

211 IF(FDBCN) 603,603,604
212 604 CONTINUE
213 CALL PATCON(F,MMAX,NMAX,2,-45.,0.0,5.0,STARTU,STARTV,
$DELTAU,DELTAV,NUMPAT,ISYMM)
214 603 IF(FORPR) 606,606,607

215 607 CONTINUE
216 DO 608 J=1,401
217 U=(J-1)*C.005-1.0
218 V=U
219 SUMU=0.
220 SUMV=0.
221 DO 609 K=1,IC
222 SUMU=SUMU+CORCOF(K)*PAT(U-US(K),-VS(K),ITYPE)
223 SUMV=SUMV+CORCOF(K)*PAT(-US(K),V-VS(K),ITYPE)
224 609 CONTINUE
225 DATA1(J,1)=U
226 DATA2(J,1)=V
227 DATA1(J,2)=DATA1(J,2)+SUMU
228 DATA2(J,2)=DATA2(J,2)+SUMV
229 DATA1(J,2)=DATA1(J,2)*FNORM
230 DATA2(J,2)=DATA2(J,2)*FNORM
231 608 CONTINUE
232 IF(NMAX.LE.1) GO TO 2901
233 WRITE(6,610)
234 610 FORMAT(1HI,25X,'U-AXIS PROFILE OF FINAL PATTERN')
235 CALL PROFIL(DATA1,401,NUMPAT)
236 2901 IF(NMAX.LE.1) GO TO 606
237 WRITE(6,611)
238 611 FORMAT(1HI,25X,'V-AXIS PROFILE OF FINAL PATTERN')
239 CALL PROFIL(DATA2,401,NUMPAT)
240 606 CONTINUE

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

243 IF(FCURPT) 700,700,701
244 701 WRITE(6,702)
245 702 FORMAT(1HI/////////////////////////////////////////////////////////55X,'FINAL CURR')
246 CALL PRINT(CURR,MCUR,NCUR,INITLS,INITLT,DELTAS,DELTAT)
247 WRITE(6,703)
248 703 FORMAT(1HI/////////////////////////////////////////////////////////55X,'FINAL CURI')
249 CALL PRINT(CURI,MCUR,NCUR,INITLS,INITLT,DELTAS,DELTAT)
250 704 WRITE(6,705)
A-32

252 705 FORMAT(1H1///10X,'FINAL CURR')
253 CALL CONTUR(MCUR,NCUR,C,COS,-0.04,0.04,0.0,CURR,NUMPAT)
254 WRITE(6,705)
255 785 FORMAT(1H1///10X,'FINAL CURR')
256 CALL CONTUR(MCUR,NCUR,C,COS,-0.04,0.04,0.0,CURR,NUMPAT)
257 703 IF(FCURPR-1) 706,707,711
258 707 IF(MCUR.LE.1) GO TO 708
259 WRITE(6,710)
260 710 FORMAT(1H1,10X,'S AXIS PROFILE OF FINAL CURRENT'//
$13X,'S',17X,'T',18X,'REAL',12X,'IMAGINARY',10X,'MAGNITUDE',
$12X,'PHASE')/
261 J=NCUR/2+1
262 DO 709 I=1,MCUR
263 CALL LCCSCR(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 709
269 APH=ATAN2(CI,CR)*57.2957795
270 709 WRITE(6,511) S,T,CR,CI,AMAG,APH
271
272 708 IF(NCUR.LE.1) GO TO 706
273 WRITE(6,712)
274 712 FORMAT(1H1,10X,'T AXIS PROFILE OF FINAL CURRENT'//
$13X,'S',17X,'T',18X,'REAL',12X,'IMAGINARY',10X,'MAGNITUDE',
$12X,'PHASE')/
275 1=MCUR/2+1
276 DC 713 J=1,NCUR
277 CALL LCCSCR(I,J,S,T,ITYPE)
278 CR=CURR(I,J)
279 CI=CURI(I,J)
280 AMAG=SQRT(CR*CR+CI*CI)
281 IF(AMAG.EQ.0.) APH=0.
282 IF(AMAG.EQ.0.) GO TO 713
283 APH=ATAN2(CI,CR)*57.2957795
284 713 WRITE(6,511) S,T,CR,CI,AMAG,APH
285 GO TO 706
286 711 WRITE(6,714)
287 714 FORMAT(1H1///10X,'FINAL ELEMENT CURRENTS'//5X,
$'S','J','10X','S','15X','T','15X','CURR','15X','CURI/')
288 CALL LIST(CURR,CURI,MCUR,NCUR)
289 706 CONTINUE

C

289 IGCOUNT=NUMPAT
290 WRITE(2,1850) IGCOUNT,NUMTRK,NUMSKP,IPASS
291 IF(IRECT.EQ.0) GO TO 9998
292 CALL DIRCTV(DIRG,UURG,VORG,NORG,US,VS,COUGF,IC,MMAX,NMAX,
$DIRCRG,DIRFNL)
293 WRITE(6,789) DIRORG,DIRFNL
294 6789 FORMAT('I DIRORG = ',F7.2,') D8.1/
$D0 DIRFNL = ',F7.2,') D8.1)
295 9998 CONTINUE
A-33

IF (IDISK.EQ.C) GO TO 9997
IF (IDISK.EQ.1 .AND. ISUC.NE. 1) GO TO 9997
C DISK OUTPUT
GO TO 7000 J=2, 29
IF (NUMSKP(J) .EQ. 0) GO TO 7001
C CONTINUE
WRITE(6, 7002)
C FORMAT('C NO DISK SPACE AVAILABLE -- DATA NOT STORED')
GO TO 9999
C CONTINUE
C SPACE IS AVAILABLE ON RECORD "J"
C
NUMSKP(J) = 1
WRITE(22, 8850) NUMPAT, J, NUMSKP, IPASS
C
WRITE(22, J, 8850) NUMPAT, TITLE, [SYM, MTER, ISUC, FHORN, IDISK,
$ NORG, IC, (UORG(M), VORG(M), CORG(M), M=1, NORG),
$ (US(M), VS(M), CORCOF(M), M=1, IC), ITYPE, P1, P2, P3, P4, P5, P6,
$ PT1, (SS(W), TT(M), M=1, 400), II, 12, 13, 14, 15, MCUR, NCUR
C
WRITE(6, 7003) NUMPAT, J
C FORMAT('O PATTERN NUMBER ', I4, ' HAS BEEN STORED ON RECORD',
$ I4, ' OF ANO DATA, A507C2')
C
9997 GO TO 9999
C
C
SUBROUTINE DIRCTV(CORG, UORG, VORG, NORG, US, VS, CORCOF, IC, MMAX, NMAX,
1 CIRG, CIRFNL)
C THIS SUBROUTINE CALCULATES THE DIRECTIVITY OF THE ORIGINAL PATTERN
C , DIRCRG, AND OF THE FINAL PATTERN, DIRFNL.
C DIMENSION CORG(100), UORG(100), VORG(100), US(500), VS(500), CORCOF(500)
$ C
COMMON /LCC/ ITYPE
C FORGSC=C.
C FSC=0.
C FMAX1=0.
C FMAX2=0.
DO 10 J=1, 101
U=-1.0+(J-1)*0.02
C 10 CONTINUE
DO 20 K=1, 101
V=-1.0+(K-1)*0.02
C 20 CONTINUE
UVSQ=L*L+V*V
F=0.
IF (ABS(UVSG.GE.1.0)) GO TO 10.
C
IF (NCRG.LE.0) GO TO 25
C DO 20 L=1, NCRG
C F=F+PAT(U-UORG(L), V-VORG(L), ITYPE)
C FORGSC=F+SCRT(U, V)
C FMAX1=ABS(F)
C FMAX1=FMAX1
C 25 CONTINUE
FSC = FCRGSQ;
FMAX2 = FMAX1
IF (IC .LE. 0) GO TO 10
CO 30 L = 1, IC
3C F = F + CCRCCF(L) * PAT(U-US(L), V-VS(L), ITYPE)
FSC = FSC + F**2 / SQRT(1.0 - UVSQ)
IF (ABS(F) .GT. FMAX2) FMAX2 = ABS(F)
1C CONTINUE
FORGSQ = FORGSQ * 0.0004 / FMAX1
FSC = FSC * C.CCC4 / FMAX2
DIRCRG = 4.0 * 3.14159265 / FORGSQ
DIRFNL = 4.0 * 3.14159265 / FSQ
C C
DIRCRG = IC * ALOGIO(DIRORG)
DIRFNL = IC * ALOG10(DIRFNL)
RETURN
END

SUBROUTINE INPUT
C INTEGER PX, PY
REAL LX, LY, INITLS, INITLT
C
COMMON /PAT1/ P1, P2, P3, P4, P5, P6, PI, SS(400), TT(400), RR(400)
COMMON /PAT2/ P1, P2, P3, P4, P5, P6, PI, SS(400), TT(400), RR(400)
COMMON /LOC/ ITYPE
COMMON /MPROG/ MCUR, NCUR
COMMON /SYN/ LX, LY
C NAMELIST /PATIN/ LX, LY, PX, PY, DISX, DISY, INITLS, DELTAS, FINALS, $INITLT, CELTAT, FINALT, NELMT, ARAU, ITYPE, MCUR, NCUR
C
WRITE(6, 10)
10 FORMAT(////55X, 'SOURCE SPECIFICATIONS'///)
PI = 3.14159265
READ(5, PATIN)
IF (ITYPE .GT. 7) GO TO 990
GO TO (106, 200, 300, 400, 500, 600, 700), ITYPE
WRITE(6, 2C) ITYPE
2C FORMAT(1HO, 5X, '***ERROR*** ITYPE HAS THE VALUE ', ITYPE, '***EXECUTION TERMINATED***')
STOP
C
10C PI = LY
20C P2 = INITLT
30C 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//
515X,'INITLT,FINALT,DELTAT: ',3(1X,F8.4)//15X,'NUMBER OF SAMPLE POINTS = NCUR = ',13)
GO TO 999

C

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 = ',13//15X,'INTER-ELEMENT SPACING = ',F6.3)
GO TO 999

C

3CC P1=LY
384 LX=0.0
386 P2=INITLT
387 P3=DELTAT
388 NCUR=(FINALT-INITLT)/DELTAT+1.5
389 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,
$8'DELTAT=',F6.3//15X,'NUMBER OF SAMPLE POINTS = NCUR = ',13)
GO TO 999

C

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

C

5CC P1=LX
405 P2=LY
406 II=PX
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'//
$15X,'DIMENSIONS = LX,LY = ',F7.4,','F7.4//
$15X,'NUMBER OF ELEMENTS = PX,PY = ',I3,',I3//
$15X,'INTER-ELEMENT SPACING = DISX,DISY = ',F6.3,','F6.3)
414  GO TO 999

C

415  601 P1=ARAD
416  P3=INITLS
417  P4=INITLT
418  P5=DELTAS
419  P6=DELTAT
420  LX=ARAD*2.
421  LY=LX
422  MCUR=(FINALS-INITLS)/DELTAS+1.5
423  NCUR=(FINALT-INITLT)/DELTAT+1.5
424  WRITE(6,601) ARAD,INITLS,DELTAS,FINALS,INITLT,DELTAT,MCUR,
$NCUR
425  601 FORMAT(ICX,'ITYPE=6 -- UNIFORM CIRCULAR APERTURE'//
$15X,'ARAD = ',F7.3//15X,'INITLS,DELTAS,FINALS: ',3(F8.4,1X)//
$15X,'INITLT,DELTAT,FINALT: ',3(F8.4,1X)//
$15X,'MCUR,NCUR: ',2(I3,2X))
426  GO TO 999

C

427  701 I1=MCUR
428  I2=NCUR
429  LX=1.0
430  LY=1.0
431  NELMT=I1*I2
432  WRITE(6,701)
433  701 FORMAT(I1X,'ITYPE=7 -- GENERAL ARRAY'//
$15X,'ELEMENT',7X,'SS(J)*,14X,'TT(J)')
434  CC 702 J=1,NELMT
435  READ(5,703) SS(J),TT(J)
436  703 FORMAT(F1IC,C)
437  WRITE(6,704) J,SS(J),TT(J)
438  704 FORMAT(17X,13,5X,3(El4.7,5X))
439  702 CONTINUE
440  GO TO 999
441  990 CALL SINPUT(PX,PY,OISX,OISY,INITLS,DELTAS,FINALS,INITLT,
$DELTAT,FINALT,NFLMT,ARAD,ITYPE)
442  999 RETURN
443  ENC
SUBROUTINE READ (F, MM, N)
DIMENSION F(51,51), I(6), VAL(6)
CC 100 J=1, MM
K2=0
2CC CONTINUE
READ (5,1) (I(L), VAL(L), L=1, 6)
1 FORMAT (6(I3, F10.0))
CC 20 L=1, 6
II=I(L)
IF (II.EQ.0) GO TO 100
K1=K2+1
K2=K1+II-1
DO 10 K=KI,K2
10 F(J,K)=VAL(L)
2CC CONTINUE
IF (K2.LT.N) GO TO 200
100 CONTINUE
RETURN
END

SUBROUTINE ORGPAT(F, MM, N, STARTU, STARTV, DELTAU, DELTAV, CURR, $CUR, MCUR, NCUR)
DIMENSION F(51,51), CURR(51,51), CUR(51,51)
REAL F(51,51), CURR(51,51), CUR(51,51)
REAL CURG(100), VORG(100), COR(100)
COMPLEX SOURCE
COMPLEX TEMP
COMMON /START/ NORG, VORG, COR
COMMON /LOC/ ITYPE

DO 10 N=1, MM
10 F(M,N)=0.

THIS ORGPAT WILL BE "WOODWARD-LAWSON" INPUT.

REPRODUCIBILITY OF THE ORIGINAL PAGE IS POOR.
DO 15 M = 1, MCUR
CC 15 N = 1, NCUR
75 CURR(M, N) = 0.
76 CURR(M, N) = 0.
15 WRITE(6, 17)
17 FORMAT(1HI, 50X, -- INITIAL COEFFICIENTS -- '/45X, 'J', 6X, '
$ UORG(J)', 5X, 'VORG(J)', 6X, ', CORG(J)'), /
19 REAC(5, 1) NORC
20 1 FORMAT(15)
21 DO 20 IC = 1, NCRG
22 READ(5, 2) US, VS, CORCOF
23 2 FORMAT(3F10.0)
44 UORG(1C) = US
45 VORG(1C) = VS
46 CORC(1C) = CORCOF
47 DO 30 M = 1, MMAX
48 U = STARTU + (M - 1) * DELTAV
49 DU = U - US
50 DO 30 N = 1, NMAX
51 V = STARTV + (N - 1) * DELTAV
52 DV = V - VS
53 30 F(M, N) = F(M, N) + CORCOF * PPROU(DU, DV, ITYPE)
49 CC 40 M = 1, NCUR
50 CC 40 N = 1, NCUR
51 TEMP = SOURCE(M, N, US, VS, ITYPE)
52 CURR(M, N) = CURR(M, N) + CORCOF * REAL(TEMP)
53 40 CURR(M, N) = CURR(M, N) + CORCOF * AIMAG(TEMP)
54 WRITE(6, 50) IC, US, VS, CORCOF
55 50 FORMAT(44X, T3, 5FX, F7.4, 5X, F7.4, 5X, F7.4)
56 SUBROUTINE ANTSYN(ISUC, MMAX, NMAX, FDES, FU, FL, ITRMAX, ISYMM, CORCOF, /
57 ICUS, VS, STARTU, DELTAV, MCENT, NCENT, ITER, F1NORM, F)
58 REAL FDES(51, 51), F1U(51, 51), F(51, 51), FL(51, 51)
59 REAL US(500), VS(500), CORCOF(500), GUC, G1U, YG1, YG2, Y3, Y4
60 COMMON /SYN/ LX, LY
61 COMMON /SYN/ LX, LY
62 LXY = 1., LXY = 1.
63 ITER = 0
64 27 ITER = ITER + 1
65 NORMALIZE...
66 FBIG = F(0, 0)
67 DO 150 M = 1, MMAX
68 DO 150 N = 1, NMAX
69 150 F(M, N) = F(M, N) / FBIG
70 FNORM = FNORM / FBIG
71 DO 151 1 = 1, NCRG
72 CURG(1) = CURG(1) / FBIG
73 IF(1C, LEX) GO TO 153
74 LG 152 1 = 1, 1C
75 157 CORCOF(1) = CORCOF(1) / FBIG
76 CONTINUE
-- ITERATION PROCEDURE --

SET IF SPECS ARE MET.

DO 24 J=1,NMAX
   U=STARTU*(J-1)*DELTAU
24 CONTINUE

DO 24 K=1,NMAX
   V=STARTV*(K-1)*DELTAV
24 CONTINUE

IF(UVSU.GT.1.0) GO TO 24

IF(FDES(J,K).LT.99.0) GO TO 24
IF(FL(J,K).LE.0.00001 .AND. ABS(F(J,K)).LE.1.E-4) GO TO 24
IF(XL.GT.FU(J,K)) GO TO 25
IF(XL.LT.FL(J,K)) GO TO 25
24 CONTINUE

ISUC=1
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)
7117 FORMAT(10X,6,' ITERATIONS COMPLETED')

IF(ITER-ITRMAX) 22,22,23
23 WRITE(6,34)
34 FORMAT(IHO,9X,'NUMBER OF ITERATIONS EXCEEDED', 15/
$10X,"PRINTOUT OF INTERMEDIATE RESULTS FOLLOWS.")
GO TO 75C

22 CONTINUE

FIND RELATIVE MAXIMUM ERROR

CALL SEARCH(J,K,VAL,FDES,FU,FL,F,NMAX,STARTU,STARTV,DELTAU,
$DELTAV)

IF(VAL.NE.0.0) GO TO 248
VAL EQUALS ZERO

WRITE(6,100)
100 FORMAT('ERROR IN SUBROUTINE SEARCH -- VAL=0.')
GO TO 75C

248 U1=(J-1)*DELTAU+STARTU
249 V1=(K-1)*DELTAV+STARTV
IF(ABS(U1).LT.0.1*DELTAU) U1=0.
IF(ABS(V1).LT.0.1*DELTAV) V1=0.
IF(LX.EQ.0.0) GO TO 1000
IF(U1.NE.0.0 .AND. ABS(U1).LE.0.5/LX) VAL=VAL/2.
19 ICCC IF(LY.EQ.0.0 .AND. ABS(V1).LE.0.5/LY) VAL=VAL/2.
IF(ISYVN.NE.4) GC TO 1001
ITEMP=C
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.EQ.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

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 rhiquadrilateral symmetry only -- ISYMM = 4

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

SUBROUTINE SEARCH(I1,J1,VAL,FDES,FL,F,FMAX,NMAX,STARTU,STARTV,DELTAV)
REAL FL(51,51),F(51,51),FDES(51,51)
VAL=0.
E1MAX=0.
12=11
J2=J1
CC 10 J=12,NMAX
U=STARTU+(J-1)*DELTAV
CC 20 K=J2,NMAX
V=STARTV+(K-1)*DELTAV
UVSQ=U*U+V*V
IF(UVSQ.GT.1.0) GO TO 20
FITER=ABS(F(J,K))
IF(FDES(J,K).EQ.99.0) GO TO 20
IF(FITER.GT.FU(J,K)) GO TO 2000
IF(FL(J,K).LE.1.E-4) GO TO 20
IF(FITER.GT.FL(J,K)) GO TO 20
X=FDES(J,K)
ERROR=FITER-X
IF(ABS(ERROR)-ABS(E1MAX)) 20,20,21
EMAX=ERROR
VAL=SIGN(ERROR,F(J,K)*(X-FITER))
II=J
J1=K
20 CONTINUE
WRITE(6,10C) II,J1,VAL
100 FCRMAT(5X,'**SEARCH**',I8,I8,5X,F7.4)
RETURN

SUBROUTINE CHECK(IC,VAL,US,VS,CCRCOF,DELTAV,DELTAV)
REAL US(5CO),VS(5CO),CCRCOF(500)
IF(IC.EQ.1) RETURN
CU=C.1*DELTAV
CV=C.1*DELTAV
IC=IC-1
U=US(IC)
W=VS(IC)
DO 10 J=1,IC1
IF(ABS(U-US(J)).LE.U.ANO.ABS(V-VS(J)).LE.OV) GO TO 20
10 CONTINUE
RETURN

CORCOF(J)=CORCOF(J)+VAL
IC=IC-1
RETURN

SUBROUTINE UPDATE(IC,US,VS,CORCOF,F,MMAX,NMAX,FNORM,STARTU,STARTV,
$DELTAU,DELTAV)
DIMENSION F(51,51),US(500),VS(500), CORCOF(500)
COMMON /LOC/ ITYPE
C=CORCOF(IC)
DO 10 J=1,MMAX
U=STARTU+(J-1)*DELTAU
DU=U-US(IC)
DO 10 K=1,NMAX
V=STARTV+(K-1)*DELTAV
DV=V-VS(IC)
10 F(J,K)=F(J,K)+C*PAT(DU,DV,ITYPE)
CORCOF(IC)=CORCOF(IC)/FNORM
RETURN
END

FUNCTION PAT(U,V,ITYPE)
C
C THIS SUBPROGRAM GIVES THE BASIC CORRECTION PATTERN F(U,V).
C
I TYPE = 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 ARRAY.
I TYPE > 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 ELECTRICAL ENGR. 951-6624.
E.L. COFFEY DEPT. OF ELECTRICAL ENGR. 951-5494

COMMON /PAT1/ P1,P2,P3,P4,P5,P6,P7,SS(400),IT(400),RR(400)
COMMON /PAT2/ 11,12,13,14,15
IF (ITYPE.GT.7) GO TO 990
GO TO (100, 200, 300, 400, 500, 600, 700, 800), ITYPE
ITYPE .LT. 1
WRITE(6,10) ITYPE
10 FORMAT(1HC,5X,'***ERROR*** ITYPE HAS THE VALUE ',I11,',',2X,$'EXECUTION TERMINATED')
STOP

ITYPE = 1 -- UNIFORM LINE SOURCE.

100 CONTINUE
FLEN=PI
PAT=1.0
IF(V.NE.0.) PAT = SIN((PI*PI*V)/(PI*PI*V))
GO TO 999

ITYPE = 2 -- UNIFORM LINEAR ARRAY

200 CONTINUE
FLEN=PI
NELPT=11
PAT=1.0
IF(V.NE.0.) PAT = SIN((PI*PI*V)/(11*SIN((PI*PI*V)/11)))
GO TO 999

ITYPE = 3 -- TRIANGULAR LINE SOURCE.

300 FLEN=PI/2.
PAT=1.0
IF(V.NE.0.) PAT = ((SIN(FLEN*PI*V)/(FLEN*PI*V)))**2
GO TO 999

ITYPE = 4 -- UNIFORM RECTANGULAR APERTURE

400 CONTINUE
FLS=PI
FLT=P2
ARG1=PI*PI*U
ARG2=PI*PI*V
401 IF(ARG1) 401,402,401
402 IF(ARG2) 403,404,404
403 PAT=SIN(ARG1)/ARG1*SIN(ARG2)/ARG2
GO TO 999
404 PAT=SIN(ARG1)/ARG1
GO TO 999
405 PAT=SIN(ARG2)/ARG2
GO TO 999
406 PAT=1.0
GO TO 999

ITYPE = 5 -- UNIFORM RECTANGULAR ARRAY
C
500 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) 503,504,505
704 503 PAT=SIN(ARG1)/(11*SIN(ARG1/I1))*SIN(ARG2)/(12*SIN(ARG2/I2))
705 GO TO 999
706 504 PAT=SIN(ARG1)/(11*SIN(ARG1/I1))
707 GO TO 999
708 502 IF(ARG2) 505,506,505
709 505 PAT=SIN(ARG2)/(12*SIN(ARG2/I2))
710 GO TO 999
711 506 PAT=1.0
712 GO TO 999

C
C ITYPE = 6  -- UNIFORM CIRCULAR APERTURE.
C
600 C=SCRT(U*U+V*V)
C
A=P1
601 IF(C.EQ.C.) GO TO 601
602 X=2.*PI*PI*C
603 CALL BESJ(X,1,BJ,0.0001,IER)
604 PAT=2.*BJ/X
605 GO TO 999
606 601 PAT=1.0
607 GO TO 999

C
C ITYPE = 7  -- GENERAL ARRAY
C
700 IMAG=(C,C,1,C)
701 NELMT=I1*I2
702 TEMP=(C,C,0,0)
703 DO 701 J=1,NELMT
704 TEMP=TEMP+1.0*CEXP(IMAG*2.*PI*(U*SS(J)+V*TT(J)))
705 CONTINUE
706 PAT=REAL(TEMP)/NELMT
707 GO TO 999
708 CALL SPECPT(U,V,ITYPE)
709 999 RETURN
710 END

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).
C
C ITYPE = 1  -- UNIFORM LINE SOURCE LOCATED AT S=0.
C
730 731 END
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) WILL BE CALLED.)

VERSION 1 LEVEL 1
DATE OF LAST REVISION: 73/166 JULY 12, 1973

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

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

 COMMON /PATI/ P1, P2, P3, P4, P5, P6, PI, SS(400), TT(400), RR(400)
 COMMON /PAT2/ I1, I2, I3, I4, I5

IMAG=(0.0, 1.0)
CALL LOCSOR(M, N, S, T)
IF(ITYPE.GT.7) GO TO 990
GO TO (100, 200, 300, 400, 500, 600, 700), ITYPE

ITYPE .LT. 1

WRITE(6,10) ITYPE
10 FORMAT(1HC, 5X, '***ERROR***', X, 'EXECUTING TERMINATED')
STCP

ITYPE = 1 -- UNIFORM LINE SOURCE

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

ITYPE = 2 -- UNIFORM LINEAR ARRAY

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

ITYPE = 3 -- TRIANGULAR LINE SOURCE

300 CONTINUE
FLN=P1
CON=ABS(2.*T/P1)
A-46

751  SOURCE=2./PI*CEXP(-IMAG*2.*PI*T*V)*(1.-CON)
752  IF(CON.GT.1) SOURCE=(0,0,0,0)
753  GO TO 999

C ITYPE = 4 -- UNIFORM RECTANGULAR APERTURE
C
754  4CC CONTINUE
C FLS=P1
C FLT=P2
755  SOURCE=CEXP(-IMAG*2.*PI*(S*U+V*T))/(PI*P2)
756  GO TO 999

C ITYPE = 5 -- UNIFORM RECTANGULAR ARRAY
C
757  500 CONTINUE
C FLS=P1
C FLT=P2
758  SOURCE=CEXP(-IMAG*2.*PI*(S*U+V*T))/(PI*P2)
759  GO TO 999

C ITYPE = 6 -- UNIFORM CIRCULAR APERTURE
C
760  600 RHC=SCRT(S*S+T*T)
C A=P1
761  SOURCE=(C,C,C,C)
762  IF(RHC.LE.P1) SOURCE=CEXP(-IMAG*2.*PI*(S*U+T*V))/(2.*PI*PI*PI)
763  GO TO 999

C ITYPE = 7 -- GENERAL ARRAY
C
764  700 CONTINUE
765  SOURCE=CEXP(-IMAG*2.*PI*(U*S+V*T))/(PI*P2)
766  GO TO 999
767  990 SOURCE=SPSOR(M,N,U,V,ITYPE)
768  999 RETURN

769  END

770  SUBROUTINE LCCSOR(M,N,S,T)
771  INTEGER PX,PY
772  REAL INITLS,INITLT
773  COMMON /PAT1/ P1,P2,P3,P4,P5,P6,PI,SS(400),TT(400),RR(400)
774  COMMON /PAT2/ I1,I2,I3,I4,I5
775  COMMON /LCC/ ITYPE
C
776  IF(ITYPE.GT.7) GO TO 990
777  GO TO (100,200,300,400,500,600,700), ITYPE
778  WRITE(6,10) ITYPE
779  10 FORMAT(1HC,5X,***ERROR*** ITYPE HAS THE VALUE 'Character*1',111,';',2X,
C EXECUTION TERMINATED*)
80  STOP
781 CONTINUE
782 INITL=I2
783 DELTAT=P3
784 S=C.
785 T=P2+(N-I)*P3
786 GO TO 999
787 CONTINUE
788 PY=I1
789 DISY=P2
790 S=0.
791 T=(N-I/2-1)*P2
792 IF(I1/2*2.EQ.I1) T=T+0.5*P2
793 GO TO 999
794 CONTINUE
795 PX=I1
796 PY=I2
797 DISX=P3
798 DISY=P4
799 S=(M-I/2-1)*P3
800 T=(N-I/2-1)*P4
801 IF(I1/2*2.EQ.I1) S=S+0.5*P3
802 IF(I2/2*2.EQ.I2) T=T+0.5*P4
803 GO TO 999
804 CONTINUE
805 NELMT=(M-I)*I2+N
806 S=SS(NELMT)
807 T=TT(NELMT)
808 GO TO 999
809 CALL SPLOC(M,N,S,T)
810 999 RETURN
811 END
SUBROUTINE CURRENT(CURR,CUR1,MCUR,NCUR,US,VS,CURCOF,IC)

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


COMPLEX SOURCE, TEMP
REAL CURR(51,51), CUR1(51,51), US(500), VS(500), CURCOF(500)
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
TEMP=SOURCEx(M,N),UORG(I),VORG(I),ITYPE)
CURR(M,N)=CURR(M,N)+(CURG(I)*REAL(TEMP)
CURR(M,N)=CURR(M,N)+(CURG(I)*AIMAG(TEMP)
CONTINUE
IF(IC.LT.7) RETURN
C
DO 300 M=1,MCUR
DO 300 N=1,NCUR
DO 300 I=1,NORG
TEMP=SOURCEx(M,N),US(I),VS(I),ITYPE)
CURR(M,N)=CURR(M,N)+CURCOF(I)*REAL(TEMP)
CURR(M,N)=CURR(M,N)+CURCOF(I)*AIMAG(TEMP)
CONTINUE
C
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 10 COLUMNS TO A PAGE. 72/266 VERSION 3
C
DIMENSION A(51,51),U(51),V(51)
WRITE(6,969)
969 FORMAT(1H1)
10 CC 10 J=1,51
29 CC 10 J=STARTU+(J-1)*DU
30 CC 10 J=STARTV+(J-1)*DV
31 N2=N/10.+0.99
32 M2=M/10.+0.99
A-49

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

C PRINT A HEADING

841 WRITE(6,20) (V(I),I=N3,N4)
842 2C FORMAT(1H1,16X,F6.3,9(4X,F6.3))
843 WRITE(6,30)

C PRINT A PAGE

844 K2=(M4-M3+1)*6
845 DO 400 J=1,K2
846 J2=J/6
847 IF(J2*6-J) 27,28,27
848 J3=J2*M3-1
849 WRITE(6,29) U(J3),A(J3,I),I=N3,N4
850 CC TO 4000
851 27 WRITE(6,31)
852 4000 CONTINUE
853 IF(N4.EC.N .AND. M4.EQ.M) GO TO 300
854 2CC CONTINUE
855 1CC CONTINUE
856 3CC RETURN
857 29 FORMAT(3X,F6.3,'+',5X,10(F9.4,1X))
858 3C FORMAT(1CX,1H+,10(1OH---------+))
859 31 FORMAT(1CX,'I')
860 END

SUBROUTINE PRCFIL(DAT1,NPT,NUMPAT)
861 INTEGR SF
862 INTEGER OUTPUT(101)
863 INTEGER BLANK,PLUS,SLASH,STAR
864 REAL DATA(401,2),BOUNO(101)
865 REAL DATA(401,2)
866 DATA BLANK,PLUS,SLASH,STAR / '*'+'+'+'|'+'+'+' /
867 CC 47 J=1,401
868 DATA(J,1)=DATA(J,1)
869 DATA(J,2)=DATA(J,2)
870 47 CONTINUE

C FIND THE RANGE OF DEPENDENT DATA AND SCALE IF NECESSARY

871 IF(NPT.GT.600) CC TO 999
872 BIG=-1.10
873 SMALL = 1.10
874 CC 1 J=1,NPT
IF(DATA(J,2) .LT. -60.0) DATA(J,2) = -60.0
IF(DATA(J,2) .LT. SMALL) SMALL = DATA(J,2)
IF(DATA(J,2) .GT. BIG) BIG = DATA(J,2)
1 CONTINUE
DIFF = ABS(BIG - SMALL)
SF = C
IF(DIFF .LT. 1.) GO TO 10
IF(DIFF .LT. 100.) GO TO 21
DO 2 J=1,1C
IF(DIFF .LT. 1C**(-J) .GT. 100.) GO TO 2
SF = J
GO TO 20
2 CONTINUE
4C CONTINUE
WRITE(6,1CC)
100 FORMAT(10 YOUR DATA IS TOO LARGE FOR THIS PROGRAM.)
RETURN
1C DO 3 J=1,1C
K = 11 - J
IF(DIFF .LT. 1C**K .GT. 100.) GO TO 3
SF = -K
GO TO 20
3 CONTINUE
GO TO 4C
2C DO 4 J=1,NPT
4C DATA(J,2) = DATA(J,2)*10.**(SF)
C CALCULATE BOUNDS
C
21 SCALE = DIFF / 100.
DO 5 J=1,1C
K = J - 1
5 CONTINUE
BOUND(J) = (BIG - K*SCALE)*10.**(SF)
C PRINT TITLE
C
WRITE(6,64C) NUMPAT
64C FORMAT(26X,'PATTERN NUMBER ',15X)
IF (SF .EQ. C) GO TO 200
WRITE(6,65C) SF
65C FORMAT(53X,'SCALE FACTOR IS 10**',12/)
WRITE(6,65C) (BOUND(J),J=1,101,20)
650 FORMAT(/X,5(F7.3,13X),F7.3,2X,'REAL',5X,'C9.*')
DO 6 J1=1,NPT
J = NPT + 1 - J1
6 CONTINUE
CUTPUT(1) = BLANK
IF((J1-1)/1C*10-((J1-1)) 62,61,62
CUTPUT(1) = PLUS
GO TO 97
62 CUTFUT(1) = SLASH
CUTPUT(1C1) = SLASH
SUBROUTINE CCNTUR(K,L,DCON,CLND,BMAX,CINT,NUMPAT)

C This subroutine 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 DCON=Delta[increment] between contours for contour subroutine
C CONLOW=Lowest contour level
C CONMAX=Highest contour level
C CONINT=Contour interval
C NUMPAT=Pattern number
C
DIMENSION A(51,51)
DIMENSION ALPHA(10)
DIMENSION CLND AT LEAST L
DIMENSION COL(101)
DATA ALPHA/1HC,1H1,1H2,1H3,1H4,1H5,1H6,1H7,1H8,1H9/
DATA BLANK,OUT/1H,1H,1H/

IF(K.LE.1.OR.L.LE.1) RETURN
CONINT=CINT
CONLOW=CLND
CONMAX=BMAX
WRITE(6,87) NUMPAT
87 FCPAT(1HC,'FOR THE PATTERN NUMBERED',I5)
IF(CONINT) 99,99,100
99 BIC=-1.E27
SMALL=1.E27
GO TO 98
98 CONTINUE
CCINT=(BIG-SMALL)/10.
DELCON=C5*CCINT
CCLOW=SMALL+DELCON
CONMAX=BIG-DELCON
WRITE(6,71) DELCON,CCLOW,CONMAX,CCINT
FORMAT(1HC,*DELCON=*,F10.5,3X,*CCLOW=*,F10.5,3X,*CONMAX=*,F10.5)

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

WRITE HEADING
CC32 J=1,101
COL(J)=BLANK
CONTINUE

IF(L.GT.51) GO TO 33
CC30 J=1,101,2
COL(J)=DOT
CONTINUE

GO TO 34
CONTINUE

CC35 J=1,101
COL(J)=DCT
CONTINUE

WRITE(6,200)
FORMAT(1H1)
N1=L*2
IF(N1.GT.101) N1=101
WRITE(6,101) (CCL(J),J=1,N1)
FORMAT(1HC,14X,101Al)
CONTINUE

J2=-1
CC31 J=1,L
J2=J2+2
ICCN=ICCN+1
CON=CCNMAX+CONINT
CC50 M=1,MCHAR
ICCN=ICCN+1
CON=CON-CONINT
IF(A(I,J).GT.(CON+DELCON)) GO TO 50
IF(A(I,J).LT.(CON-DELCON)) GO TO 50
NEW A(I,J) IS LT CON+DELCON AND GT CON-DELCON
IF(L.LE.51) CCL(J2)=ALPHA(ICON+1)
CONTINUE
GO TO 2
1012 CONTINUE
1013 CONTINUE
1014 WRITE(6,140) (COL(J1),J1=1,101)
1015 FORMAT(1H140,13X,'.',1CIA1)
1016 CONTINUE
1017 RETURN
1018 END
1019 SUBROUTINE PATCON(RDATA,MMAX,NMAX,ICODE,CONLOW,CONMAX,CONINT,$STARTU,STARTV,DELTU,DELTAV,NUMPAT,ISYMM)
1020 REAL RDATA(51,51),UAXIS(11),LOW(12),HIGH(12)
1021 INTEGER CUTPUT(101),LEVEL(12),BLANK
1022 DATA BLANK/' '/
1023 DATA LEVEL/'C',' ','21,'30,14,50,1,60,7,8,9,1,9,+'/
1024 CALL DATE(I,J,K)
1025 WRITE(6,10) I,J,K,NUMPAT
1026 FORMAT(1H1, 'PATTERN CONTOUR SUBPROGRAM',34X,'DATE = ',A2,'-',A2
$,'-',A2,3CX,'PATTERN NUMBER',I5//////)
1027 IF(ICODE.EQ.0) WRITE(6,11)
1028 IF(ICODE.EQ.1) WRITE(6,12)
1029 IF(ICODE.EQ.2) WRITE(6,13)
1030 CONTINUE
1031 FORMAT(42X,'CONTOUR PLOT OF THE DESIRED PATTERN'////)
1032 FORMAT(45X,'CONTOUR PLOT OF THE INITIAL PATTERN'////)
1033 FORMAT(45X,'CONTOUR PLOT OF THE FINAL PATTERN IN DB.'////)
1034 FINALU=STARTU+(NMAX-1)*DELTU
1035 FINALV=STARTV+(NMAX-1)*DELTAV
1036 U1=STARTU
1037 U2=FINALU
1038 V1=STARTV
1039 V2=FINALV
1040 NCCOUNT=NMAX
1041 IF(ISYMM-1) 70,30,20
1042 UBIG=AMAX1(AHS(STARTU),ABS(FINALU))
1043 U1=-UBIG
1044 U2=UBIG
1045 NCCOUNT=2*NCCOUNT-1
1046 IF(ISYMM.EQ.2) GO TO 70
1047 VBIG=AMAX1(AHS(STARTV),ABS(FINALV))
1048 V1=-VBIG
1049 V2=VBIG
1050 NCCOUNT=2*NCCOUNT-1
1051 CONTINUE
1052 ESTABLISH LOWER AND UPPER LIMITS
A- 54

NUMCON=(CONMAX-CONLOi)/CONINT+1.5
1053 CONLON=CONINT/2.
1054 DC 71 J=1,NUMCCN
1055 LOW(J)=CCNLOW+(J-1)*CONINT-DELCON
1056 HIGH(J)=LOW(J)+CONINT+0.0001
1057 71 CONTINUE
1058 LOW(11)=-1.E3C
1059 HIGH(12)=1.E30
1060 HIGH(11)=LOW(11)
1061 LOW(12)=HIGH(NUMCON)
1062 MSIP=100/(MCOUNT-1)
1063 NSKIP=100/(NCOUNT-1)
1064 CONTINUE
1065 CO 40 I=1,11
1066 40 UAIXS(I)=UI+(I-1)*DU
1067 WRITE(6,42) (UAIXS(I),I=1,11)
1068 42 FORMAT(13X,11(F7.4,3X)/16X,11(' ',9X))
1069 CONTINUE
1070 DU=(U2-U1)/100.
1071 CV=(V2-V1)/100.
1072 NC 50 N=1,101,NSkip
1073 V=V2-(N-1)*DV
1074 CC 51 K=1,101
1075 51 OUTPUT(K)=BLANK
1076 CC 60 M=1,101,MSIP
1077 U=U1+(M-1)*DU
1078 IF(U+V**.GT.1.0) GO TO 60
1079 C
1080 IF(IJ) 2CC,102,2CC
1081 102 IF(IK) 300,1000,300
1082 2CC IF(ISYM=-1) 6C,6C,2C1
1083 201 J=1.5-(U+STARTU)/DELTAV+1.5
1084 IF(IK) 1C00,300
1085 IF(IJ) 6C,202,60
1086 202 IF(IK) 300,1000,60
1087 IF(ISYM.EQ.0 .OR. ISYM.EQ.2) GO TO 60
1088 K=1.5-(V+STARTV)/DELTAV
1089 IF(IK) 1C00,60
1090 IF(IJ) 6C,202,60
1091 202 IF(IK) 300,1000,60
1092 IF(K + V**.GT.1.0) GO TO 60
1093 IF(K + U**.GT.1.0) GO TO 60
1094 IF(IK) 6C,1000,60
1095 IF(IJ) 6C,202,60
1096 1CCC IF=RDATA(J,K)
1097 IF(F.EQ.LGW(1)) GO TO 1001
1098 IF(F.GT.HIGH(NUMCON)) GO TO 1002
1099 C
DO 61 K = 1, NUMCON
   IF (F.GT.LOW(K) .AND. F.LE.HIGH(K)) GO TO 62
11C1  61 CONTINUE
11C2 1002 OUTPUT(K) = LEVEL(12)
11C3  GO TO 60
11C4 1001 OUTPUT(M) = LEVEL(11)
11C5  GO TO 60
11C6  62 OUTPUT(M) = LEVEL(K)
C
11C7  6C CONTINUE
11C8  WRITE(6, 64) V, (OUTPUT(K), K = 1, 101), V
11C9  64 FORMAT(7X, F7.4, 1X, ',', 101AI, ',', 1X, F7.4)
11C10 IF (N1.EQ.0) GO TO 50
11C11  GO 55 K = 1, N1
11C12  WRITE(6, 56)
11C13  56 FORMAT('')
11C14  55 CONTINUE
C
11C15  5C CONTINUE
11C16  WRITE(6, 43) (UAXIS(I), I = 1, 11)
11C17  43 FORMAT(16X, 11(' ', 9X)/13X, 11(F7.4, 3X))
C
11C18  WRITE(6, 44)
11C19  44 FORMAT(/56X, 'CONTOUR LEVEL KEY'//)
11C20  45 I = 1, 4
11C21  WRITE(6, 46) (LEVEL(J), LOW(J), HIGH(J), J = 1, 12, 4)
11C22  46 FORMAT(5X, 3(A1, ': ', EI4.7, ' TO ', EI4.7, 4X))
11C23  RETURN
11C24  ENC
C
SUBROUTINE LIST(CURR, CURI, MCUR, NCUR)
C
THIS SUBROUTINE LISTS ARRAY ELEMENT COORDINATES AND CURRENTS
C
C
C
DIMENSION CURR(51, 51), CURI(51, 51)
11C26  DO 10 M = 1, MCUR
11C27  DO 10 N = 1, NCUR
11C28  J = (M-1)*NCUR+N
11C29  CALL LOGCSR(M, N, S, T)
11C30  WRITE(6, 1CC) J, S, T, CURR(M, N), CURI(M, N)
11C31  10 CONTINUE
11C32  100 FORMAT(3X, 14, 5X, 4(E14.7, 2X))
11C33  RETURN
11C34  ENC
7. Appendix: The ANTDATA Computer Program

7.1 Introduction

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

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

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

7.2 Program Organization

Again a modular structure using several subroutines has been used to allow for modifications. The main program generates the pattern and current arrays and controls which plots are made. Fig. 7.1 shows a block diagram of the program organization. Subroutines PAT, SPECPT, SOURCE, SPSOR, LOCSON, 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 Pl, 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 OPTIU, OPTIV, OPT2, and OPT3 from a card under a 4I1 format. Use zeros for no plot and ones for plot.

Step 6. U profile location.

Read CONST from a card under a 8F10.0 format. This is the value of V where the profile is made. In other words, the profile is parallel to the U-axis with a value of V equal to CONST. If CONST is zero the profile is on the U-axis. Use only if OPTIU=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 OPTIV=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 OPTIU, OPTIV, OPT2, and OPT3 from a card under a 4I1 format. Use zeros for no plot and ones for plot. OPTIU and OPTIV 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 "+V" and "-V" and the value stored in CONST will be reproduced as "U=CONST." If CODE=2, the horizontal axis will be labeled "+U" and "-U" and CONST will be reproduced as "V=CONST."

**CONST** - Label constant.

**NUMPAT** - Pattern number.

Remarks:

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

2. 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 /PLTI, 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 PLOT1P

Purpose:

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

Usage:

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

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

Remarks:

i. 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=.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, D, 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, W, P, KODE)
Description of Parameters:

**XI** - Abscissa of the projected point.

**YJ** - Ordinate of the projected point.

**M** - Number of horizontal points.

**N** - Number of vertical points.

**P** - PLOT CODE; IF P = -1 INVISIBLE TO VISIBLE

1 VISIBLE TO INVISIBLE

0 VISIBLE TO VISIBLE OR INVISIBLE TO INVISIBLE.

**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
```

Subroutines and Function Subprograms Required: None.

### 7.5.2 Virginia Tech Subroutines

**VPI UTILITY SUBPROGRAMS**

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

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

**VPI PLOTTER SUBROUTINES**

<table>
<thead>
<tr>
<th>Subroutine</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>AXIS</td>
<td>To draw a labeled axis of a desired length with annotated tic marks every inch.</td>
</tr>
</tbody>
</table>
FACTOR To scale the plot in both the X and Y directions.

NUMBER To draw a floating point number.

PLOT To move the pen from one point to another, to draw a line between points, to establish a new origin, and to signal the end of a plot.

SYMBOL To plot a string of 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.

//0063PL3 JOB 507C2, COFFEY
/*MAIN TIME=19, LINES=3, REGION=250K, CARDS=C
/*PRIORITY PRIORITY
/*FORMAT PL, FORMS=PFGURAGV, PEN=XXFIAE, DDNAME=CALCUMP
// EXEC FORTGCG, PARM.GO='PAPER=39, PTIME=119', EP=MAIN
// FORT.SYSIN DD *
/*
//GO.SYSLIB DD
// DD DSN=VPI.PLOTLIB, DISP=SHR
// DD DSN=VPI.SSPLIB, DISP=SHR
//GO.SYSLIN DD
// DD *
/*
//GO.FT22FOO DD DSN=ANTDATA.A507C2, UNIT=3330, VOL=SER=tJSERPK, DISP=SHR
//GO.FT06FOO DD SYSOUT=A, DCB=(BLKSIZE=133, RECFM=F)
//GO.SYSIN DD *
/*
//

7.6.2 Source Listing

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

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

VERSION 1 73 - 094 -- APRIL 5, 1973

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

UNDER NASA GRANT: 47-004-103

ADDITIONAL SUBPROGRAMS REQUIRED:

ICVT -- ASSEMBLER LANGUAGE SUBPROGRAM TO CONVERT AN INTEGER
to A2 CHARACTER FORMAT.

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

DIMENSION A(151,151),PTS(4001),US(500),VS(500),CORCOF(500)
DIMENSION UORG(100),VORG(100),CORG(100)
DIMENSION AV(151),AV(151)
INTEGER TITLE(20)
REAL INITLS,INITLT
REAL LOWCON
INTEGER OPT1V,OPT2,OPT3,OPT4,OPT5,OPT6
COMPLEX CTEMP,CI
COMMON /PLT1/ PTS
COMMON /PLT2/ A
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
IPAGE=0
PI=3.14159265
CI=CMPLX(0.0,1.0)
CALL STIME(ITIME)
TIME=0.
9999 CONTINUE
CALL TIMECN
IPAGE=IPAGE+1
CALL DATE(I,J,K)
CALL STIME(IT)
IHR=IT/10000
IFR=IT-IHR*10000
FHR=IFR/10000.
FM=FHR*60.
IHR=ICVT(IHR)
IFR=ICVT(IFR)
ISEC=ICVT(ISEC)
IPG=ICVT(IPG)
WRITE(6,1) I,J,K,IHR,IFR,FHR,FM,IP,IPG
1 FORMAT(1H1,2X,'ANTDATA I VERSION 1 LEVEL 2',
$8X,4X,'EE DEPT.',5X,'DATE = ',A2,'-',A2,'-',A2,'-',A2,'-',A2,'-',A2,'-',A2,
$5X,'TIME = ',A2,'-',A2,'-',A2,'-',A2,'-',A2,'-',A2,'-',A2,'-',A2,'-',A2,'-',A2,'-',A2',')
READ PATTERN NUMBER AND TRACK -- VERIFY THAT PATTERN IS ACTUALLY STORED
READ(5,10,END=999) NUMPAT,NUMTRK
10 FORMAT(14I5)
IF(NUMPAT.EQ.0) GO TO 999
WRITE(6,704) NUMPAT
704 FORMAT(' PLOT OUTPUT FOR PATTERN\',I5,:')
READ(22,NUMTRK,20) NUM
IF(NUMPAT.NE.NUMPAT) GO TO 50
30 CONTINUE
NUMPAT IS NOT ON DISK
WRITE(6,60) NUMPAT
60 FORMAT(1H0,'PATTERN NUMBER\',I5,\' WAS NOT LOCATED -- PROGRAM HALT\')
GO TO 999
NUMPAT FOUND ON UNEXPECTED TRACK
WRITE(6,60) NUMPAT,NUMTRK,I
60 FORMAT(1H0,'PATTERN NUMBER\',I5,\' WAS NOT FOUND ON TRACK\',I2,
\$ BUT WAS LOCATED ON TRACK\',I2) NUMTRK=I
51 CONTINUE
REGAIN PROCESSING
READ(5,10) NMAX,NMAX
READ(22,NUMTRK,7C) ITEMP,ITEMP1
70 FORMAT(104X,2A4)
READ(22,NUMTRK,101) NUMPAT,TITLE,ISYM,ITER,ISUC,FNORM,IVISK,
$NCRG,IC,(UORG(J),VORG(J),CORG(J),J=1,ITEMP),
$(US(J),VS(J),CORGOF(J),J=1,ITEMP1),ITYPE,P1,P2,P3,P4,P5,P6,
P1I,(SS(J),IT(J),J=1,400),11,12,13,14,15,MCUR,NCUR
101 FORMAT(75A4,11(200A4))

C READ OPTIONS FOR PATTERN MAGNITUDE
C
READ(5,29) OPTIU,OPTIV,OPT2,OPT3
29 FORMAT(4I1)
3 IF(OPTIU-1) 80,81,80
4 81 CONTINUE
5 READ(5,31) CONST
6 IF(MMAX.LE.1) GO TO 80
7 DO 90 J=1,4001
8 U=(J-1)*0.0005-1.0
9 SUM=0.
10 DO 91 K=1,NCRG
11 SUM=SUM+CORCH(K)*PAT(U-UORG(K),CONST-VORG(K),ITYPE)
12 IF(IC.LT.0) GO TO 90
13 DO 93 K=1,1C
14 SUM=SUM+CORCHOF(K)*PAT(U-US(K),CONST-VS(K),ITYPE)
15 91 CONTINUE
16 PTS(J)=2C.*ALOG10(ABS(SUM))
17 WRITE(6,92) CONST
18 92 FORMAT(*CU-AXIS PROFILE PLOT REQUESTED -- V = ,F6.3)
19 CALL PLOT1(-1.0,1.0,4001,2,CONST,NUMPAT)
20 IF(OPTIV-1) 82,83,82
21 82 CONTINUE
22 READ(5,31) CONST
23 IF(NMAX.LE.1) GO TO 82
24 DO 90 J=1,4001
25 V=(J-1)*0.0005-1.0
26 SUM=0.
27 DO 91 K=1,NCRG
28 SUM=SUM+CORCHOF(K)*PAT(CONST-UORG(K),V-VORG(K),ITYPE)
29 IF(IC.LT.0) GO TO 90
30 DO 93 K=1,1C
31 SUM=SUM+CORCHOF(K)*PAT(CONST-US(K),V-VS(K),ITYPE)
32 91 CONTINUE
33 PTS(J)=2C.*ALOG10(ABS(SUM))
34 WRITE(6,93) CONST
35 93 FORMAT(*CV-AXIS PROFILE PLOT REQUESTED -- U = ,F6.3)
36 CALL PLOT1(-1.0,1.0,4001,1,CONST,NUMPAT)
37 IF(OPT2+OPT3) 85,85,84
38 84 CONTINUE
C GENERATE PATTERN ARRAY
C
READ(5,31) LOWCON,DASH
29 IF(MMAX.LE.1 .OR. NMAX.LE.1) GO TO 239
30 DELTAU=2.0/(MMAX-1)
31 DELTAV=2.0/(NMAX-1)
32 WRITE(6,761) LOWCON,LOWCON
33 701 FORMAT(*PATTERN IS NOW BEING GENERATED. IF PATTERN < ,F7.2,
34 * PATTERN = ,F7.2)
35 IF(ITYPE.GT.5) GO TO 5000
A-78

C  LOAD UP Au AND AV.

105 C  GO 2CCG  I=1,MMAX
106 U=(I-1)*CELTAU
107 2CCG  AU(I)=PAT(U,O.,ITYPE)
108 GO 2010  J=1,NMAX
109 V=(J-1)*DELTAV
110 201C  AV(J)=PAT(C-,V,ITYPE)

C  BEGIN

111 C  U=-1.0-DELTAU
112 GO 2020  M=1,MMAX
113 U=U+DELTAU
114 V=-1.0-DELTAV
115 GO 2020  N=1,NMAX
116 V=V+DELTAV
117 TEMP=0.
118 GO 2030  K=1,NORG
119 I=ABS(U-UORG(K))/DELTAU1.5
120 J=ABS(V-VORG(K))/DELTAV1.5
121 203C  TEMP=TEMP+COR(K)*AU(I)*AV(J)
122 IF(1C.LE.O) GO TO 2020
123 GO 2040  K=1,IC
124 I=ABS(U-US(K))/DELTAU1.5
125 J=ABS(V-VS(K))/DELTAV1.5
126 204C  TEMP=TEMP+CORCOF(K)*AU(I)*AV(J)
127 202C  A(M,N)=20.*ALOG10(ABS(TEMP))
128 GO TO 239
129 5C00 CONTINUE
130 GO 200  M=1,MMAX
131 U=-1.0+(M-1)*DELTAU
132 GO 201 N=1,NMAX
133 V=-1.0+(N-1)*DELTAV
134 TEMP=0.
135 GO 242 I=1,NORG
136 242 TEMP=TEMP+COR(K)*PAT(U-UORG(K),V-VORG(K),ITYPE)
137 IF(1C.LE.O) GO TO 2021
138 GO 202  I=1,IC
139 202 TEMP=TEMP+CORCOF(I)*PAT(U-US(I),V-VS(I),ITYPE)
140 202C CONTINUE
141 A(M,N)=20.*ALOG10(ABS(TEMP))
142 201 CONTINUE
143 2CC CONTINUE
144 239 CONTINUE
145 IF(OPT2) 210,210,211
146 211 READ(5,31) CONLOW,CONMAX,CONINT
147 31 FORMAT(6FI0.0)
148 IF(MMAX.LE.1 OR NMAX.LE.1) GO TO 230
149 GO 257 N=1,NMAX
150 GO 257 M=1,MMAX
151 IF(A(M,N).LT. LOWCON) A(M,N)=LOWCON
152 257 CONTINUE
153 WRITE(6,220) CONLOW,CONMAX,CONINT
220 FORMAT(*CONTOUR PLOT OF PATTERN REQUESTED*)
   $1.  LOWEST CONTOUR  = ',F7.2/
   $1.  HIGHEST CONTOUR  = ',F7.2/
   $1.  CONTOUR INTERVAL  = ',F7.2)

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

IF(OPT3) 230,230,231

CALL PLOT3(MMAX,NMAX,NUMPAT)

CONTINUE

IF (MMAX.LE.1. OR NMAX.LE.1) WRITE(6,23)

FOCRMAT(IHO,THREE-DIMENSIONAL PLOT OF PATTERN REQUESTED)

CALL PLOT3(MMAX,NMAX,NUMPAT)

CONTINUE

END OF PATTERN

IA=0

IF(I TYPE.EQ.1) GO TO 401
IF(I TYPE.EQ.3) GO TO 401
IF(I TYPE.EQ.4) GO TO 401
IF(I TYPE.EQ.6) GO TO 401

WRITE(6,4C2)

WRITE(6,4C2)

IF(I TYPE-1) 404,403,404

CONTINUE

IF(I TYPE=4)

CONTINUE

IF(I TYPE=4)

CONTINUE

CONTINUE

CONTINUE

CONTINUE

CONTINUE

CONTINUE
409 INITLS=P3
201 FINALS=P3+2.*P1
202 INITLT=P4
203 FINALT=P4+2.*P1
204 P5=P1/2000.
206 DELTAT=P6
207 DELTAS=P5
410 CONTINUE
208 READ(5,29) OPTIU,OPTIV,OPT2,OPT3
209 IF(OPTIU-1) 302,301,302
210 CONTINUE
211 READ(5,31) CONST
212 IF(IA.EQ.1) GO TO 3000
213 IF(INMAX.LE.1) GO TO 302
214 J=1
215 IF(DELTAT.NE.0.) J=1.5+(CONST-INITLT)/DELTAT
216 DO 303 I=1,4001
217 CTEMP=(0.0,0.0)
218 IF(NORG.LE.0) GO TO 304
219 DO 305 K=1,NORG
220 305 CTEMP=CTEMP+CORG(K)*SOURCE(I,J,UORG(K),VORG(K),ITYPE)
221 304 IC.LE.0) GO TO 303
222 DO 306 K=1,IC
223 306 CTEMP=CTEMP+CORCOF(K)*SOURCE(I,J,US(K),VS(K),ITYPE)
224 303 PTS(I)=CABS(CTEMP)
225 WRITE(6,307) CONST
226 FORMAT('OS-AXIS PROFILE PLOT REQUESTED -- T = ',F6.3)
227 CALL PLOTIC(INITLS,FINALT,4001,2,CONST,NUMPAT)
228 CONTINUE
229 IF(CPTIV-1) 311,310,311
230 CONTINUE
231 READ(5,31) CONST
232 IF(IA.EQ.1) GO TO 3000
233 IF(INMAX.LE.1) GO TO 322
234 I=1
235 IF(DELTAS.NE.0.) I=1.5+(CONST-INITLS)/DELTAS
236 DO 313 J=1,4001
237 CTEMP=(0.0,0.0)
238 IF(NORG.LE.0) GO TO 314
239 DO 315 K=1,NORG
240 315 CTEMP=CTEMP+CORG(K)*SOURCE(I,J,UORG(K),VORG(K),ITYPE)
241 314 IC.LE.0) GO TO 313
242 DO 316 K=1,IC
243 316 CTEMP=CTEMP+CORCOF(K)*SOURCE(I,J,US(K),VS(K),ITYPE)
244 313 PTS(J)=CABS(CTEMP)
245 WRITE(6,317) CONST
246 FORMAT('OS-AXIS PROFILE PLOT REQUESTED -- S = ',F6.3)
247 CALL PLOTIC(INITLT,FINALT,4001,1,CONST,NUMPAT)
248 CONTINUE
249 CONTINUE
250 CONTINUE
251 CONTINUE
252 P3=P3TEMP
253 P5=P5TEMP
254 P6=P6TEMP
255 MCUR=51
256 NCUR=51
IF(DOPI2+CPT3) 320, 320, 321
CONTINUE
READ(5,31) LOWCON,DASH
IF(IA.EQ.1) GO TO 333

C GENERATE CURRENT MAGNITUDE ARRAY
C
DO 330 M = 1, MCUR
DO 331 N = 1, NCUR
CALL LOCSOR(M, N, S, T)
CTEMP = 0.
DC 339 K = 1, NORG
CTEMP = CTEMP + CORG(K) * SOURCE(M, N, UORG(K), VORG(K), ITYPE)
DO 332 K = 1, IC
CTEMP = CTEMP + CORCOF(K) * SOURCE(M, N, US(K), VS(K), ITYPE)
A(P, N) = CABS(CTEMP)
CONTINUE
CONTINUE
CONTINUE
IF(OPT2) 350, 350, 351
READ(5,31) CONLOW, CONMAX, CONINT
IF(IA.EQ.1) GO TO 360
IF(VMAX.LE.1.OR.NMAX.LE.1) GO TO 360
WRITE(6, 340) CONLOW, CONMAX, CONINT
C
C CONTOUR PLOT OF CURRENT MAGNITUDE REQUESTED
C
C LOWEST CONTOUR = 1, F7.4
C HIGHEST CONTOUR = 9, F7.4
C CONTOUR INTERVAL = 1, F7.4
CALL PLOT2(MCUR, NCUR, CONLOW, CONMAX, CONINT, NUMPAT, DASH)
IF(OPT3) 360, 360, 361
READ(5, 31) CONLOW, DASH
IF(IA.EQ.1) GO TO 360
WRITE(6, 355)
FORMAT(1HO, 'THREE DIMENSION PLOT OF CURRENT MAGNITUDE REQUESTED')
CALL PLOT3(MCUR, NCUR, NUMPAT)
CONTINUE
IF(VMAX.LE.1.OR.NMAX.LE.1) WRITE(6, 23)
36C CONTINUE
C END OF CURRENT MAGNITUDE
C
C READ OPTIONS FOR CURRENT PHASE
C
READ(5,29) OPT1U, OPT1V, OPT2, OPT3
IF(OPT2+CPT3) 520, 520, 521
CONTINUE
IF(IA.EQ.1) GO TO 533
READ(5,31) LOWCON,DASH
C GENERATE CURRENT PHASE
C
DO 530 M = 1, MCUR
DO 531 N = 1, NCUR
CALL LOCSOR(M, N, S, T)
CTEMP = 0.
DO 549 K = 1, NORG
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
CONTINUE
IF(OPT2) 550,550,551
READ(5,31) CONLOW,CONMAX,CONINT
IF(IA.EQ.1) GO TO 560
IF(PMAX.LE.1.OR.NMAX.LE.1) GO TO 560
WRITE(6,560) CONLOW,CONMAX,CONINT
540 FORMAT('CONTOUR PLOT OF CURRENT PHASE REQUESTED '/
$\quad$LOWEST CONTOUR = ',F7.2/
$\quad$HIGHEST CONTOUR = ',F7.2/
$\quad$CONTOUR INTERVAL = ',F7.2)
CALL PLOT2 (MCUR,NCUR,CONLOW,CONMAX,CONINT,NUMPAT,DASH)
550 IF(OPT3) 560,560,561
561 IF(IA.EQ.1) GO TO 560
WRITE(6,555)
555 FORMAT('OTWO DIMENSION PLOT OF CURRENT PHASE REQUESTED')
CALL PLOT3(MCUR,NCUR,NUMPAT)
560 CONTINUE
IF(PMAX.LE.1.OR.NMAX.LE.1) WRITE(6,23)
520 CONTINUE
CALL TIMECK(ISEC)
FMIN=ISEC/6000.
WRITE(6,897) FMIN
897 FORMAT('EXECUTION TIME: ',F6.2,' MINUTES.'),
TIME=TIME+FMIN
GO TO 9999
999 WRITE(6,600)
600 FORMAT('** END OF EXECUTION **'//)
CALL PLOT3(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.'),
STOP
END

SUBROUTINE PLOT1(PSTRT,PEND,IP,CODE,CONST,NUMPAT)
SUBROUTINE PLOT1
WRITTEN BY: S. R. KAUFMAN
DATE: 73-113 APRIL 23, 1973
INTEGER NAME(2), CODE
DIMENSION PTS(4001)
COMMON /PLT/ PTS
CALL FACTOR(0.5)
CALL PLOT(8.91.,-3)
IF(CODE.GT.0) GO TO 3
CALL SYMBOL(-1.2,-6,2,8,HTHETA = 0.,8)
CALL NUMBER(.3,-8.2,CONST,0.,3)
GO TO 6
3 IF(CODE.GT.1) GO TO 4
CALL SYMBOL(-1.2,-8.2,1HV,0.,1)
CALL SYMBOL(-9.2,-2,3H = 0.,3)
CALL NUMBER(-2.2,-8.2,CONST,0.,3)
CALL SYMBOL(-2.6,-4,2,2HV=0.,2)
CALL SYMBOL(2.4,-4,2,2HV=0.,2)
GO TO 6
4 IF(CODE.GT.2) GO TO 5
CALL SYMBOL(-1.2,-8.2,1HV,0.,1)
CALL SYMBOL(-9.2,-2,3H = 0.,3)
CALL NUMBER(-2.2,-8.2,CONST,0.,3)
CALL SYMBOL(-2.6,-4,2,2HV=0.,2)
CALL SYMBOL(2.4,-4,2,2HV=0.,2)
6 CONTINUE
PDEL=(PSTRT-PEND)/IP
PTIC=(ABS(PSTRT-PEND))/10.
CALL AXIS(-5.0,0.1H,1.4,0.0,PSTRE,PTIC)
PSTRE=PSTRT*(6.*PTIC)+.00001
PTIC2=PTIC*.00001
CALL AXIS(1.0,0.1H,1.4,0.0,PSTRE,PTIC2)
CALL PLOT(-1.0,0.,3)
CALL PLOT(-1.0,0.,2)
CALL PLOT(0.0,0.3)
CALL PLOT(0.0,0.3)
CALL PLOT(0.0,0.3)
CALL SYMBOL(-.05,-.4,-2,1HV=0.,1)
X=0.05
CC 10 J=1,6
Y=0.5*(J-1)*1.0
CALL PLOT(-X,Y,3)
10 CALL PLOT(X,Y,2)
CALL PLOT(0.0,0.3)
IF(PTS(1).LE.-50.) PTS(1)=-50.
FS=((PTS(1))/10.)*5.5
SUBROUTINE PLOTIC

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.
NUPAT -- PATTERN NUMBER FOR LABEL.

INTEGER NAME(2), CODE
CALL FACTOR(0.5)
CALL PLOT1(-5., 0.3)
COMMON /PLTI/ PTS

I=C
IF (CODE.GT.0) GO TO 3
CALL SYMBOL(-1.2, -0.6, 0.2, 0.8, 0.8, 0.8, 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, 0.8, 0.8, 0.8, 0.0, 8)
CALL SYMBOL(-0.9, -0.8, 0.2, 0.8, 0.8, 0.8, 0.0, 8)
CALL NUMBER(-0.2, -0.8, 0.2, CONST, 0.0, 3)
CALL SYMBOL(-2.6, -0.4, 0.2, 0.0, 0.0, 0.2)
CALL SYMBOL(2.4, -0.4, 0.2, 0.0, 0.0, 0.2)
416 GO TO 6
417 IF(CODE.GT.2) GO TO 5
418 CALL SYMBOL(-1.0,-0.8,0.2,1HT,0.0,1)
419 CALL SYMBOL(-0.9,-0.8,0.2,3H = ,0.0,3)
420 CALL NUMBER(-0.2,-0.8,0.2,CONST,0.0,3)
421 CALL SYMBOL(-2.6,-0.4,0.2,2H-5,0.0,2)
422 CALL SYMBOL(2.4,-0.4,0.2,2H+S,0.0,2)
423 6 CONTINUE
424 PDEL=(PSTRT-PEND)/IP
425 PTIC=(ABS(PSTRT-PEND))/10.0)
426 CALL AXIS(-5.0C,C.0,IH 1,,4.0,O.O,PSTRT,PTIC)
427 PSTRE=PSTRT+(6.0*PTIC)+0.00001
428 PTIC2=PTIC+0.00001
429 CALL AXIS(1.0,0.0,1H ,1,4.0,0.0,PSTRE,PTIC2)
430 CALL PLOT(-1.0,0.0,3)
431 CALL PLOT(1.0,0.0,2)
432 CALL PLOT(0.0,0.0,3)
433 CALL PLOT(0.0,5.0,2)
434 CALL PLOT(0.0,0.0,3)
435 CALL SYMBOL(-0.5,-0.4,0.2,H=PSTRE)
436 X=0.05
437 UC 10 J=1,6
438 Y=0.5+(J-1)*1.0
439 CALL PLOT(-X,Y,3)
440 CALL PLOT(X,Y,3)
441 CALL PLOT(1.0,0.0,3)
442 PSTRE=5.0*C*PSTRT
443 GMAX=0.0
444 DO 11W1=1,IP
445 IF(PTS(IW1).GT.GMAX) GMAX=PTS(IW1)
446 1 CONTINUE
447 IF(GMAX.GT.0.5) ASCLE=1.
448 IF(GMAX.LE.0.5) ASCLE=0.5
449 IF(GMAX.LE.0.2) ASCLE=0.2
450 IF(GMAX.LE.0.1) ASCLE=0.1
451 IF(GMAX.LE.0.05) ASCLE=0.05
452 PTS=(PTS(IW1)/ASCLE)*5.+0.5
453 CALL PLOT(-5.0,PTS,3)
454 DO 7 IW1=1,IP
455 THE=THETA=ABS(PSTRT-(IW1*PDEL))
456 THETS=(THE/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.LE.0.5) ATIC=0.1+0.0001
462 IF(GMAX.LE.0.2) ATIC=0.04+0.0001
463 IF(GMAX.LE.0.1) ATIC=0.02+0.0001
464 IF(GMAX.LE.0.05) ATIC=0.01+0.0001
465 CALL AXIS(-5.5,-0.5,16HSOURCE MAGNITUDE,16,5.0,90.0,0.0,ATIC)
466 CALL SYMBOL(-5.0,-0.8,0.2,10HPATTERN = ,0. ,10)
467 FNIM=FLOAT(NUMPAT)
468 CALL NUMBER(-3.5,-0.8,0.125,FNUM,0. ,1)
469 CALL PLOT(6.0,-1.0,3)
470 5 RETURN
471 ENC
SUBROUTINE PLOT1P(PSTRT, PEND, IP, CODE, CONST, NUMPAT)

SUBROUTINE PLOT1P

WRITTEN BY: S. R. KAUFMAN

DATE: 73-113 APRIL 24, 1973

INPUT:

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

INTEGER NAME(2), CODE
DIMENSION PTS(4001)
COMMON /PLTL/PTS
CALL FACTOR(0.5)
CALL PLCT(8., 1., -3)
IF(CODE.GT.0) GO TO 3
CALL SYMBOL(-1.2, -6., 2, 8, 'THETA = ', 0., 8)
CALL NUMBER(-.3, -.8, .2, CONST, 0., 3)
GO TO 6
3 IF(CODE.GT.1) GO TO 4
CALL SYMBOL(-1., -.8, 2, 1, 'HS', 0., 1)
CALL SYMBOL(-.9, -.8, 2, 3, 'H = ', 0., 3)
CALL NUMBER(-.2, -.8, 2, CONST, 0., 3)
CALL SYMBOL(-2.6, -.4, 2, 2, 'H + T', 0., 2)
CALL SYMBOL(2.4, -.4, 2, 2, 'H + T', 0., 2)
GO TO 6
4 IF(CODE.GT.2) GO TO 5
CALL SYMBOL(-1., -.8, 2, 1, 'HT', 0., 1)
CALL SYMBOL(-.9, -.8, 2, 3, 'H = ', 0., 3)
CALL NUMBER(-.2, -.8, 2, CONST, 0., 3)
CALL SYMBOL(-2.6, -.4, 2, 2, 'H + S', 0., 2)
CALL SYMBOL(2.4, -.4, 2, 2, 'H + S', 0., 2)
GO TO 6
5 PCEL = (PSTRT - PEND) / IP
PTIC = (ABS(PSTRT - PEND)) / 10.0
CALL AXIS(-5.0, 0.0, .1H, 1.0, .0, .0, PSTRT, PTIC)
PSTRE = PSTRT + (6.04*PTIC) + 0.00001
PTIC2 = PTIC + 0.00001
CALL AXIS(1., 0., 1H, 1.0, .0, .0, PSTRE, PTIC2)
CALL PLOT(1., 0., 3)
CALL PLOT(-1., 0., 3)
CALL PLOT(C., 0., 3)
CALL PLOT(0., 5.8, 2)
CALL PLOT(C., 0., 3)
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

DIMENSION A(151,151),RA(151),RB(151),X(151),Y(151)
COMMON /ARRAY/ A
CALL PLOT(0.,0.,-3)
CALL FACTOR(0.7)
MS=M
NS=N
RATIO=MS/NS
SCALE=10.
ANM=AMAX0(N-1,M-1)
IF(RATIO-1.0)1,1,2
SX=ANM
SY=RATIO*ANM
GO TO 3
2 SX=1./RATIO*ANM
SY=ANM
3 SMAX=AMAX1(SX,SY)
SS=SX/SMAX
SYS=SY/SMAX
IF(CONINT)4,4,5
4 CALL CNLAL(N,H,CNTRL0,CMAX,CNTRAL,0)
GO TO 7
551  \texttt{CNTRAL=CNINT}
552  \texttt{IF(CNMAX.EQ.CNLOW)GO TO 6}
553  \texttt{CMAX=CNMAX}
554  \texttt{CNTRLO=CNLOW}
555  \texttt{GO TO 7}
556  \texttt{6 CALL CNLAL(N,M,CNTRLO,CMAX,CNTRAL,1)}
557  \texttt{7 CONTINUE}
558  \texttt{CNLOW=CNTRLO}
559  \texttt{CONMAX=CMAX}
560  \texttt{CONINT=CNTRAL}
561  \texttt{CALL PLOTL(SS,SYS,0.,SYS,SSCALE)}
562  \texttt{CALL PLOTL(0.,0.,SS,0.,SSCALE)}
563  \texttt{CALL PLCTL(SS,0.,SS,SYS,SSCALE)}
564  \texttt{CALL PLOTL(0.,SYS,0.,0.,SSCALE)}
565  \texttt{CALL PLOT(1.00,0.25,3)}
566  \texttt{CALL PLOT(0.60,0.25,2)}
567  \texttt{CALL PLOT(0.60,0.25,2)}
568  \texttt{CALL PLOT(1.00,0.25,2)}
569  \texttt{CALL PLOT(1.00,0.25,2)}
570  \texttt{CALL SYMBOL(0.88,0.45,0.12,10HPATTERN=,90.,10)}
571  \texttt{FNUM=NUMPAT}
572  \texttt{CALL NUMBER(0.88,2.075,0.12,FNUM,90.,-1)}
573  \texttt{1125 YCCNA=1.0/SMAX}
574  \texttt{DELTAX=SS/FLOAT(N-1)}
575  \texttt{X(1)=0.0}
576  \texttt{Y(1)=0.0}
577  \texttt{RB(1)=A(1,1)}
578  \texttt{DO 27 J=2,N}
579  \texttt{RB(J)=A(J,1)}
580  \texttt{DO 27 X(J)=X(J-1)+DELTAX}
581  \texttt{DELTAY=SS/FLOAT(M-1)}
582  \texttt{DO 28 J=2,M}
583  \texttt{DO 28 Y(J)=Y(J-1)+DELTAY}
584  \texttt{DO 118 K=2,M}
585  \texttt{RA(J)=RB(J)}
586  \texttt{DO 118 RA(J)=A(J,K)}
587  \texttt{DO 118 J=2,N}
588  \texttt{ASSIGN 112 TO L}
589  \texttt{RR=RA(J)}
590  \texttt{XX=X(J)}
591  \texttt{YY=Y(K-1)}
592  \texttt{RL=RR}
593  \texttt{XL=XX}
594  \texttt{YL=YY}
595  \texttt{IF(RL-RA(J-1)) 41,40,40}
596  \texttt{41 RL=RA(J-1)}
597  \texttt{40 XL=X(J-1)}
598  \texttt{41 RL=RA(J-1)}
599  \texttt{GO TO 40}
600  \texttt{42 RL=RB(J)}
601  \texttt{GO TO 50}
602  \texttt{42 RL=RB(J)}
603  \texttt{YL=Y(K)}
604  \texttt{GO TO 50}
605  \texttt{50 RS=RR}
606  \texttt{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 \ 51 \ 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).LT.100, 95, 92
647 \ 92 \ IF (RC-RM).LT.96, 93, 94
648 \ 93 \ XPA = XCM
649 \ YPA = YCM
650 \ GO TO 99
651 \ 94 \ IF (RC-RL).LT.100, 103, 110
652 \ 95 \ C = 0.0
653 \ GO TO 97
654 \ 96 \ Q = (RC-RS)/(RM-RS)
655 \ 97 \ XPA = XCS-Q*(XCS-XCM)
656 \ YPA = YCS-Q*(YCS-YCM)
657 \ 99 \ Q = (RC-RS)/(RL-RS)
658 \ XPB = XCS-Q*(XCS-XCL)
659 \ YPB = YCS-Q*(YCS-YCL)
660 \ IF (RC-DASH).LT.10115, 10115, 10116
661 \ 10115 \ XPH1=C.50*XPA+XPB
662 \ YPB1=0.50*YPA+YPB
663 \ IF (ABS (XPA-XPB1).LT.001) 5001, 5002, 5002
664 \ 5001 \ IF (ABS (YPA-YPB1).LT.001) 100, 5002, 5002
5002 CALL PLCT(SCALE*XPA+2.,SCALE*YPA+0.25,3)
5003 CALL PLCT(SCALE*XPBL+2.,SCALE*YPBL+0.25,2)
GO TO 100
10116 IF(ABS(XPA-XPB)-.001)5003,5004,5004
5003 IF(ABS(YPA-YPB)-.001)100,5004,5004
5004 CALL PLCT(SCALE*XPA+2.,SCALE*YPA+0.25,3)
5005 CALL PLCT(SCALE*XPB+2.,SCALE*YPB+0.25,2)
100 RC = RC + CNTRAL
GO TO 80
103 XPA = XCL
104 YPA = YCL
GO TO 99
106 C=(RC-RM)/(RL-RM)
XPB=XCM-Q*(XCM-XCL)
YPB=YCM-Q*(YCM-YCL)
GO TO 99
110 GC TO L,(112,118)
112 ASSIGN 118 TO L
RR =RB(J-1)
XX =X (J-1)
YY =Y (K)
GO TO 37
118 CONTINUE
688 CALL PLCT(SCALE+6.,0.,-3)
RETURN
END
A-91

**SUBROUTINE PLOTL(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 PLOTL(X(1),Y(1),3)**

**CALL PLOTL(X(2),Y(2),2)**

RETURN

**ENC**

**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.

**DIST** - DISTANCE FROM SURFACE TO EYE WHEN PERSPECTIVE IS CALCULATED -- SDISTS > 6 USUALLY WON'T SHOW ANY DISTORTION DUE TO PARALLAX.

**YAW** - (IN DEGREES) HOW FAR THE OBJECT IS TURNED AWAY FROM THE VIEWER.

**PITCH** - (IN DEGREES) HOW THE SURFACE IS LOWERED OR RAISED AT THE FRONT EDGE. (POSITIVE PITCH TENDS TO EXPOSE THE TOP OF THE FIGURE).

**SIZE** - (IN INCHES) THE SIZE OF THE CUBE THAT ENCLOSES THE Figure.

**KODE** - "HIDDEN LINE" SWITCH. IF KODE=0 DO NOT DRAW HIDDEN LINES...IF KODE=1, ALL HIDDEN LINES ARE PLOTTED.


**SCALE** - HOW TALL TO MAKE THE SURFACE RELATIVE TO THE HEIGHT OF THE CUBE. SCALE=0: DO NOT SCALE THE DATA AT ALL
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,D,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.

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

SUBROUTINE PLOT3INM(N,M,NUMPAT)

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

COMMON /ARRAY/ A
COMMON /THREE6/ ANGA,ANGB,HV,D,SH,SV
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
*GN=0
SCALE=1.

CALL FACTOR(1.1)
CALL PLOT(8.,-2.-3)
CALL PLOT(.4,.0,.2)
CALL PLOT(.4,.8,.2)
CALL PLOT(0.,8.,.2)
CALL PLOT(0.,0.,.2)
CALL PLOT(0.,.8,.2)
CALL SYMBOL(0.3,1.0,.12,.10,PATTERN = .90,.10)
FNLM=FLOAT(NUMPAT)
CALL NUMBER(.3,2.1,.30,.90,.12,FNUM,.90.,-1)
CALL PLOT(1.5,-.2,-3)

C**** *****

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

FV = 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 )

C CONTINUE
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 = SCRT ( 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 PEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE

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

100 FORMAT(1x,3F15.3)
2060 X(2)=A(1,1)
Z(1) = A(1, 1)
C.C 1000 J = 1, M
Z(1) = AMIN1(Z(1), A(J, K))
Z(2) = AMAX1(Z(2), A(J, K))
100 C CONTINUE
RANGE = (Z(2) - Z(1))
DCL = 1.0
IF (SCALE .NE. 0) DCL = T/RANGE*SCALE
C SCALE THE SURFACE TO MAKE A *CUBE*.
DO 30 I = 1, N
DO 30 J = 1, M
A(I, J) = (A(I, J) - Z(1)) * DCL
30 CONTINUE
Z(1) = C.C
Z(2) = T
2000 CALL THREE2 ( X, Y, Z, XP, H, V, KODE)
DO 2130 I = 1, 8
H(I) = ( (XP(I) - OX) * SM - ( H(1) - OY ) * SL ) * SD
V(I) = ( V(1) - QZ ) * SD
2130 CONTINUE
H(10) = H(11)
H(9) = H(1)
DO 1001 J = 1, 8
H(J) = AMIN1(H(9), H(J))
H(10) = AMAX1(H(10), H(J))
1001 CONTINUE
V(9) = V(1)
V(10) = V(1)
DO 1002 J = 1, 8
V(J) = AMIN1(V(9), V(J))
V(10) = AMAX1(V(10), V(J))
1002 CONTINUE
IF (MGN .NE. 0) GO TO 2140
S = H
IF (MGN .EQ. 1) S = 1.5
SH = S/ (H(10) - H(9) )
SV = S/ (V(10) - V(9) )
SH = SIGN1 AMIN1(SH, SV), SH
SV = SIGN1(SH, SV)
IF (MGN .EQ. 1) CALL PLOT (0.0, 2.0, -3)
CALL SYMBOL(H(1) - H(9))SH, (V(1) - V(9))SV, 14, *O*, 0, 1
CALL SYMBOL(H(3) - H(9))SH, (V(3) - V(9))SV, 14, *M*, 0, 1
CALL SYMBOL(H(2) - H(9))SH, (V(2) - V(9))SV, 14, *Z*, 0, 1
CALL SYMBOL(H(5) - H(9))SH, (V(5) - V(9))SV, 14, *N*, 0, 1
CALL PLOT(0.0, 0.05, -3)
CALL PLOT ( (H(1) - H(9))SH, (V(1) - V(9))SV, 3)
CALL PLOT ( (H(2) - H(9))SH, (V(2) - V(9))SV, 2)
CALL PLOT ( (H(1) - H(9))SH, (V(1) - V(9))SV, 2)
CALL PLOT ( (H(3) - H(9))SH, (V(3) - V(9))SV, 2)
CALL PLOT ( (H(1) - H(9))SH, (V(1) - V(9))SV, 2)
CALL PLOT ( (H(5) - H(9))SH, (V(5) - V(9))SV, 2)
IF (MGN .EQ. 3) GO TO 2139
CALL PLOT ( (H(6) - H(9))SH, (V(6) - V(9))SV, 2)
CALL PLOT ( (H(2) - H(9))SH, (V(2) - V(9))SV, 2)
CALL PLOT ( (H(4) - H(9))SH, (V(4) - V(9))SV, 2)
CALL PLOT ( (H(7) - H(9))SH, (V(3) - V(9))SV, 2)
A-95

837  CALL PLOT ( (H(7)-H(9))*SH, (V(7)-V(9))*SV,2)
838  CALL PLOT ( (H(5)-H(9))*SH, (V(5)-V(9))*SV,2)
839  CALL PLOT ( (H(6)-H(9))*SH, (V(6)-V(9))*SV,2)
840  CALL PLOT ( (H(8)-H(9))*SH, (V(8)-V(9))*SV,2)
841  CALL PLOT ( (H(4)-H(9))*SH, (V(4)-V(9))*SV,2)
842  CALL PLOT ( (H(8)-H(9))*SH, (V(8)-V(9))*SV,2)
843  CALL PLOT ( (H(7)-H(9))*SH, (V(7)-V(9))*SV,2)
844  2139 IF(MGN .NE. 1) GO TO 2140
845  CALL PLOT (INT((H(10)-H(9))*SH+2.),-2.05,-3)
846  2140 CALL THREE3(X,Y,Z,H,V,KODE)
847  215C CONTINUE
848  CALL PLOT(16.,1.5,-3)
849  RETURN
850  END

851  SUBROUTINE THREE2 ( X, Y, Z, XP, H, V, KODE)
852     C FIND THE CORNERS OF THE ROTATED CUBE.
853     DIMENSION X(2),Y(2),Z(2),H(10),V(10),XP(6)
854  05C   L = 0
855  07C   DO 180 I = 1, 2
856  09C   CG 170 J = 1, 2
857  11C   DO 160 K = 1, 2
858  13C   L = L + 1
859  14C   CALL THREE4 ( X(I), Y(J), Z(K), XP(L),
860     H(L), V(L), KODE)
861  160  CONTINUE
862  170  CONTINUE
863  180 CONTINUE
864  190 RETURN
865  END

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

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

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

UP = 3
DOWN = 2
SH = HV / ( H (10) - H (9) )
SV = HV / ( V (10) - V (9) )
SH = SIGN(AMIN(SH,SV),SH)
SV = SIGN(SH,SV)
MP = M
NN = N
C
IIF(K-1) 100,120,100
C
IIF(K-3) 1110,120,1110
C
C DRAW LINES ALONG THE Y-AXIS

12C CONTINUE
L = 0
LD = 1
EC = 0.5 * LD
C
14C DO 1060 J = 1, M
C
16C DO 1030 I = 1, NN
C
L = L + LD
XI = L
CALL THREE5 ( XI, YJ, N, M, P, KODE)
PEN = UP
IF ( P ) 510, 520, 530
51C CONTINUE
IF ( Q ) 540, 550, 540
52C CONTINUE
IF ( Q ) 610, 1020, 610
53C CONTINUE
IF ( Q ) 540, 550, 540
54C CONTINUE
PEN = DOWN
GC TO 170
55C CONTINUE
IF ( PO EQ 0 ) GO TO 170
DI = CD
TC = L - LC
T = TC * DI
P1 = Q
560 IF ( ABS(DI) LT. ENO ) GO TO 570
CALL THREE5 (TC, YJ, N, M, PO, KODE)
C
C
IF ( PO EQ 0 ) GO TO 565
A-97

921     TC = T
922     P1 = PO
923     T = T - DI
924     GO TO 560
925     565 T = T + DI
926     GO TO 56C
927     57C CONTINUE
928     T = TO
929     IF ( P1 * P ) 170, 170, 580
930     58C CONTINUE
931
932     ZP = A(L-LC,J)+(T-L+LD)*(A(L,J)-A(L-LD,J))/LD
933     CALL THREE4(T,YJ,ZP,XP,HH,VV,KODE)
934     HH = ( ( XP-QX)*SM - (HH-QY)*SL ) * SD
935     VV = ( VV - QZ ) * SD
936     HH = ( HH - H(N) ) * SH
937     VV = ( VV - V(N) ) * SV
938     CALL PLOT ( HH , VV , PEN )
939     60C PEN = 5 - PEN
940     GO TO 170
941     61C CONTINUE
942     PEN = DOWN
943     DI = DC
944     TO = L - LD
945     T = TO + DI
946     P1 = Q
947     620 IF ( ABS(DI) LT. END ) GO TO 630
948     CALL THREE5 ( T, YJ, N, M, PO, KODE )
949     DI = DI * 0.5
950     IF ( PO .EQ. 0 ) GO TO 625
951     TO = T
952     P1 = PO
953     T = T + DI
954     GO TO 620
955     625 T = T - DI
956     GO TO 62C
957     63C CONTINUE
958     T = TC
959     IF ( P1 * Q ) 600, 600, 590
960     17C CALL THREE4 ( XI , YJ , A(L, J) , XP , HH , VV , K6(L) )
961     VV = ( VV - QZ ) * SD
962     HH = ( ( XP-QX)*SM - (HH-QY)*SL ) * SD
963     19C HH = ( HH - H(N) ) * SH
964     20C VV = ( VV - V(N) ) * SV
965     CALL PLOT ( HH , VV , PEN )
966     102C Q = P
967     1030 CONTINUE
968
969     L = L + LD
970     LC = -LD
971     DD = -DD
972
973     1060 CONTINUE
974
975     C
976
977     C
978     CI09C IF(K-3) 2660,111C,2060
C
C DRAW LINES ALONG THE X-AXIS.
972 1110 CONTINUE
C
973 1440 XI = 1
974 1110 I = 1, N
975 1140 DO 2040 J = 1, MM
976 2040 YJ = L
977 1160 CALL THREE5 (XI,YJ,N,M,P,KODE)
978 1170 PEN = UP
979 1510 IF ( P ) 1510 1520 1530
980 1510 CALL THREE5 (XI,YJ,N,M,P,KODE)
981 1520 IF ( Q ) 1540 1550 1540
982 1540 IF ( Q ) 1540 1550 1540
983 1540 CONTINUE
984 1550 IF ( J .EQ. 1 ) GO TO 1170
985 1550 P1 = 0
986 1550 TC = L-LD
987 1550 T = TO + DI
988 1550 P1 = Q
989 1560 IF ( ABS( DI ) .LT. END ) GO TO 1570
990 1570 CALL THREE5 (XI,T,ZP,XP,HH,VV,KODE)
991 1570 IF ( PO .EQ. 0 ) GO TO 1565
992 1570 TO = T
993 1570 P1 = PO
994 1570 T = T - DI
995 1570 GO TO 1560
996 1570 T = T + DI
997 1570 PEN = DOWN
998 1570 GO TO 1170
999 1570 CONTINUE
1000 1580 IF ( P1 * P ) 1170 1170 1580
1001 1580 CONTINUE
1002 1590 ZP = A(I,L-LD) + (T-L+LD) * (A(I,L) - A(I,L-LD))/LD
1003 1590 CALL THREE4 (XI,T,ZP,XP,HH,VV,KODE)
1004 1590 HH = ( (XP-QX)*SM - (HH - QY)*SL ) * SI
1005 1590 VV = (VV - GZ) * SD
1006 1590 HH = (HH - H(9)) * SH
1007 1590 VV = (VV - V(9)) * SV
1008 1590 CALL PLOT (HH,VV,PEN)
1009 1590 PEN = 5 - PEN
1010 1590 GO TO 1170
1011 1590 CONTINUE
1012 1590 PEN = DOWN
1013 1590 CONTINUE

A-99

1026        DO   = 0.0
1027        TO  = L - LC
1028        T = TO + CI
1029        PI = C
1030  IF ( ABS( DI ) .LT. 1590 ) GO TO 1630
1031        CALL THREE5 ( XI, T, N, M, PC, KODE )
1032        DI = DI * 0.5
1033        IF ( PC .EQ. 0 ) GO TO 1625
1034        T = T + CI
1035        PI = PC
1036        T = T + CI
1037        GO TO 1620
1038        T = T - CI
1039        GO TO 1620
1040        CONTINUE
1041        T = TO
1042        IF ( PI * C ) 1600, 1600, 1590
1043        CALL THREE4 ( XI, YJ, AL, L, XP, HH, VV, KODE )
1044        HH = ( ( XP - QX) * SM - (HH - QY) * SL ) * SD
1045        VV = ( VV - QZ ) * SD
1046        HH = ( HH - H(9)) * SH
1047        VV = ( VV - V(9)) * SV
1048        CALL PLOT ( HH, VV, PEN )
1049        CONTINUE
1050        IF ( KODE .EQ. 1 ) GO TO 78
1051        IR = XI
1052        JC = YJ
1053        ZB = Z ( IR + JC )
1054        IF ( XI .EQ. IR ) GO TO 2
1055        ZB = Z ( IR + JC ) + ( XI - IR ) * ( Z ( IR + 1, JC ) - Z ( IR, JC ) )
1056        GO TO 4
1057        IF ( YJ .EQ. JC ) GO TO 4
1058        ZB = Z ( IR + JC ) + ( YJ - JC ) * ( Z ( IR, JC + 1 ) - Z ( IR, JC ) )
1059        CONTINUE
1060        XEND = CO
1061        OX = 0.0

SUBROUTINE THREE5 ( XI, YJ, M, N, P, KODE )
C SEE IF A POINT IS VISIBLE.
DIMENSION Z(151,151)
COMMON /THREE6/ ANGA, ANGB, HV, D, SH, SV
COMMON /THREE7/ SM, SN, CX, CY, CZ, QX, QY, QZ, SD
COMMON /ARRAY/ Z
INTEGER CUM, CNT, P
REAL I, J, II, JJ
IF ( KODE .EQ. 1 ) GO TO 78
IR = XI
JC = YJ
ZB = Z ( IR + JC )
IF ( XI .EQ. IR ) GO TO 2
ZB = Z ( IR + JC ) + ( XI - IR ) * ( Z ( IR + 1, JC ) - Z ( IR, JC ) )
GO TO 4
ZB = Z ( IR + JC ) + ( YJ - JC ) * ( Z ( IR, JC + 1 ) - Z ( IR, JC ) )
CONTINUE
XEND = CO
OX = 0.0
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   CY = 0.0
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 ( DX .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

A-100
FUNCTION PAT(U,V,ITYPE)

C
C THIS SUBPROGRAM GIVES THE BASIC CORRECTION PATTERN F(U,V).
C
C ITYPE = 1 -- UNIFORM LINE SOURCE LOCATED AT S=0.
C 2 -- UNIFORM LINEAR ARRAY LOCATED AT S=L.
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
C BE CALLED.
C
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

COMPLEX TEMP,CEXP,IMAG
COMMON /PAT1/ P1,P2,P3,P4,P5,P6,P1,SS(100),IT(100),RR(100)
COMMON /PAT2/ 11,12,13,14,15
1162 IF (ITYPE.GT.7) GO TO 990
1163 GO TO (1CC,2CC,300,400,500,600,700),ITYPE
1164 C     ITYPE .LT. 1
1165 WRITE(6,10) ITYPE
1166 C     INSTRUCTIONS FOR ITYPE HAS THE VALUE ",111,"::",2X, 
1167 C     "$*EXECUTION TERMINATED*"
1168 C     STOP
1169 C     ITYPE = 1 -- UNIFORM LINE SOURCE.
1170 C     FLEN=PI
1171 CONTINUE
1172 C     PAT=1.0
1173 IF(V.NE.C.) PAT=SIN(PI*PI*V)/(PI*PI*V)
1174 GO TO 999
1175 C     ITYPE = 2 -- UNIFORM LINEAR ARRAY
1176 CONTINUE
1177 C     FLEN=PI
1178 C     NELMT=II
1179 C     PAT=1.0
1180 IF(V.NE.C.) PAT=SIN(PI*PI*V)/(II*SIN(PI*PI*V/II))
1181 GO TO 999
1182 C     ITYPE = 3 -- TRIANGULAR LINE SOURCE.
1183 C     FLEN=PI/2.
1184 CONTINUE
1185 C     PAT=1.0
1186 IF(V.NE.C.) PAT=(SIN(FLEN*PI*V)/(FLEN*PI*V))**2
1187 GO TO 999
1188 C     ITYPE = 4 -- UNIFORM RECTANGULAR APERTURE
1189 CONTINUE
1190 C     FLS=PI
1191 C     FLT=P2
1192 C     ORG=PI*PI*U
1193 C     ARG2=PI*P2*V
1194 IF(ARG1) GO TO 401,400
1195 IF(ARG2) GO TO 402
1196 IF(ARG1) GO TO 403,402
1197 IF(ARG2) GO TO 404
1198 PAT=SIN(ARG1)/ARG1*SIN(ARG2)/ARG2
1199 GO TO 999
1200 PAT=SIN(ARG1)/ARG1
1201 CC TO 999
1202 IF(ARG2) GO TO 405
1203 PAT=SIN(ARG2)/ARG2
1204 GO TO 999
1205 PAT=1.0
1206 GO TO 999
ITYPE = 5 -- UNIFORM RECTANGULAR ARRAY

C 500 CONTINUE C FLS=P1 C FLT=P2 C NELS=I1 C NELT=I2

ARG1=PI*P1*U
ARG2=PI*P2*V

IF(ARG1) 501,502,501
501 IF(ARG2) 503,504,503
503 PAT=SIN(ARG1)/I1*SIN(ARG1/I1)*SIN(ARG2)/I2*SIN(ARG2/I2)
GO TO 999
504 PAT=SIN(ARG1)/I1*SIN(ARG1/I1)
GO TO 999
505 PAT=SIN(ARG2)/I2*SIN(ARG2/I2)
GO TO 999
506 PAT=1.0
GO TO 999

C

ITYPE = 6 -- UNIFORM CIRCULAR APERTURE.

C 600 C=SQR(T(U*U+V*V)) C A=PI

IF(C.EQ.0.) GO TO 601
X=2.*PI*PI*C
CALL BESJ(X,1,BJ,0.0001,IER)
PAT=B/J/X*2.0
GO TO 999
601 PAT=I.C
GO TO 999.

C

ITYPE = 7 -- GENERAL ARRAY

C 700 IMAG=(0.0,1.0) NELMT=I1*I2
TEMP=(0.0,0.0)
DC 701 J=1,NELMT
701 CONTINUE
TEMP=TEMP*1.0*EXP(IMAG*2.*PI*(U*SS(J)+V*TT(J))
CONTINUE
701 CONTINUE
PAT=REAL(TEMP)/NELMT
GO TO 999
999 RETURN

C

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

C

THIS SUBPROGRAM CALCULATES THE CURRENT AT POINT (M,N) DUE TO THE PATTERN AT POINT (U,V).
ITYPE = 1 -- UNIFORM LINE SOURCE LOCATED AT S=0.
ITYPE = 2 -- UNIFORM LINEAR ARRAY LOCATED AT S=0.
ITYPE = 3 -- TRIANGULAR LINE SOURCE LOCATED AT S=0.
ITYPE = 4 -- UNIFORM RECTANGULAR APERTURE.
ITYPE = 5 -- UNIFORM RECTANGULAR ARRAY.
ITYPE = 6 -- UNIFORM CIRCULAR APERTURE.
ITYPE = 7 -- GENERAL (3-D) ARRAY.

ITYPE > 7 -- SPECIAL SOURCE (FUNCTION SPSSOR(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.

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

IMAG=(0.0, 1.0)
CALL LOCSOR(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(I10, 5X, '***ERROR*** ITYPE HAS THE VALUE ', I11, ':', 2X, 'EXE

STEP

ITYPE = 1 -- UNIFORM LINE SOURCE

CONTINUE

FLEN=P1
SOURCE=CEXP(-IMAG*PI*2.0*T*V)/P1
GO TO 999

ITYPE = 2 -- UNIFORM LINEAR ARRAY

CONTINUE

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

ITYPE = 3 -- TRIANGULAR LINE SOURCE
A-105

1243 3CC CONTINUE
1244   FLEN=P1
1245   CON=ABS(2.*F/P1)
1246   SOURCE=2.*P1*CEXP(-IMAG*2.*P1*T*V)*(1.-CON)
1247   IF(CON.GT.1) SOURCE=(0.0,0.0)
1248   GO TO 999
1249   ITYPE = 4  -- UNIFORM RECTANGULAR APERTURE
1250 4CC CONTINUE
1251   FLS=P1
1252   FLT=P2
1253   SOURCE=CEXP(-IMAG*2.*PI*(S*U+V*T))/(P1*P2)
1254   GO TO 999
1255   ITYPE = 5  -- UNIFORM RECTANGULAR ARRAY
1256 5CC CONTINUE
1257   FLS=P1
1258   FLT=P2
1259   SOURCE=CEXP(-IMAG*2.*PI*(S*U+V*T))/(P1*P2)
1260   GO TO 999
1261   ITYPE = 6  -- UNIFORM CIRCULAR APERTURE
1262 6CC CONTINUE
1263   RHC=SCRT(S*S+T*T)
1264   A=PI
1265   SOURCE=(0.0,0.0)
1266   IF(RHC.LE.P1) SOURCE=CEXP(-IMAG*2.*PI*(S*U+T*V))/(2.*P1*P2)
1267   GO TO 999
1268   ITYPE = 7  -- GENERAL ARRAY
1269 7CC CONTINUE
1270   SOURCE=CEXP(-IMAG*2.*PI*(U*S+V*T))/(11*12)
1271   GO TO 999
1272   990 990 SOURCE=SPSOR(M,N,U,V,ITYPE)
1273   999 RETURN
1274   END

1275 SUBROUTINE LOCSOR(M,N,S,T)
1276   INTEGER PX, PY
1277   REAL INITLS,INITLT
1278   COMMON /PAT1/ P1, P2, P3, P4, P5, P6, PI, SS(100), TT(100), RR(100)
1279   COMMON /PAT2/ 11, 12, 13, 14, 15
1280   COMMON /LOC/ ITYPE
1281   IF(ITYPE.GT.7) GO TO 99C
1282   GO TO (100,700,300,400,500,600,700), ITYPE
WRITE(6,10) ITYPE

1C FORMAT(1HC5X,***ERROR***
*EXECUTION TERMINATED*)

STCP

1C CONTINUE
C INITLT=P1
C CELTAT=P3
S=G.
T=P2+(N-1)*P3
GO TO 999

2C CONTINUE
C PY=I1
C DISY=P2
S=G.
T=(N-I1/2-1)*P2
IF(I1/2.EQ.11) T=T+0.5*P2
GO TO 999

30C GO TO 100

4C CONTINUE
C INITLS=P3
C INITLT=P4
C CELTAS=P5
C DELTAT=P6
S=P3+(M-1)*P5
T=P4+(N-I)*P6
GO TO 999

5C CONTINUE
C PX=I1
C PY=I2
C DISX=P3
C DISY=P4
S=(N-I1/2-1)*P3
T=(N-I2/2-1)*P4
IF(I1/2.EQ.11) S=S+0.5*P3
IF(I2/2.EQ.12) T=T+0.5*P4
GO TO 999

60C GO TO 400

7C CONTINUE
C NELPT=(M-1)*I2+N
S=SS(NELPT)
T=TT(NELPT)
GO TO 999
C
99C CALL SPLOC(M,N,S,T)
999 RETURN
END

COMPLEX FUNCTION SPSOR(M,N,U,V,ITYPE)
    DUMMY SUBPROGRAM
SPSOR=(0.0,0.0)
RETURN
END

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

SUBROUTINE SPLOC(M,N,S,T)
    DUMMY SUBROUTINE
RETURN
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

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

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)

C C READ MAINBEAM LIMITS ULIM AND VLIM C
C
READ(5,1) ULIM, VLIM
1 FORMAT(8F10.0)
C C READ TRANSITION REGION LIMITS UTRAN AND VTRAN C
C
READ(5,1) UTRAN, VTRAN
C
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
C TRANSITION REGION
FDES(M,N)=99.0
FU(M,N)=99.0
FL(M,N)=99.0
GO TO 10
20 CONTINUE
C MAIN BEAM REGION
FDES(M,N)=1.0
FU(M,N)=1.06
FL(M,N)=0.943
GO TO 10
30 CONTINUE
C SIDELOBE REGION
FDES(M,N)=0.
FU(M,N)=0.057
FL(M,N)=99.0
10 CONTINUE
C
RETURN
END

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

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

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

Step 2. Cards 1 to 3: Pattern Parameters
i. Put output data onto unit 22 if the synthesis is successful (IDISK=1)
ii. Use quadrilateral symmetry (ISYMM=3)
iii. Have a maximum of 100 iterations (ITRMAX=100)
iv. STARTU=STARTV=0., DELTAU=0.02, DELTAV=0.01
v. Make the comparisons at 26 points in the u direction and 51 points in the v direction (MMAX=26,NMAX=51)

vi. Assure that F(1,1)=0 dB. at all times (MCENT=1,NCENT=1)

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

Step 3. Cards 4 to 6: Output Switches

i. Profiles of the final pattern and final current (FDBPR=FCURPR=1)

ii. Contour maps of the desired pattern (FDESCN=1) and final current (FCURCN=1) are to be made

Step 4. Cards 7 to 11: Source Specifications

i. Rectangular aperture (ITYPE=4)

ii. Dimensions of 10\(\lambda\) by 20\(\lambda\) (LX=10.,LY=20.)

iii. The value of current will be calculated at 51 x 51 points from 
\(s = -5.0\) to 5.0 by 0.2, and \(t = -10.0\) to 10.0 by 0.4.

(INITLS=-5.,FINALS=5.,DELTAS=0.2;INITLT=-10.,FINALT=10.,DELTAT=0.4)

Step 5. Cards 12 to 13: The Desired Pattern

For a more complete explanation, see the listing of subroutine DESPAT earlier in this section.

Step 6. Cards 14, 15 to 29: Initial Pattern

These are the number of (NORG) and the values of (UORG,VORG,CORG) the original correction coefficients.

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

8.2 Output from ANTSYN

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

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

PROGRAM PARAMETERS

<table>
<thead>
<tr>
<th>IDISK</th>
<th>1</th>
<th>STARTU = 0.0</th>
<th>MMAX = 26</th>
</tr>
</thead>
<tbody>
<tr>
<td>ISYMM</td>
<td>3</td>
<td>STARTV = 0.0</td>
<td>NMAX = 51</td>
</tr>
<tr>
<td>ITRMAX</td>
<td>200</td>
<td>DELTAU = 0.020</td>
<td>MCENT = 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DELTAV = 0.010</td>
<td>NCENT = 1</td>
</tr>
<tr>
<td>FDFSPT</td>
<td>0</td>
<td>ICURPT = 0</td>
<td>FCURPT = 0</td>
</tr>
<tr>
<td>FDESCN</td>
<td>1</td>
<td>ICURCN = 0</td>
<td>FCURCN = 0</td>
</tr>
<tr>
<td>FDESPR</td>
<td>0</td>
<td>ICURPR = 0</td>
<td>FCURPR = 1</td>
</tr>
<tr>
<td>FORGPT</td>
<td>0</td>
<td>FDBPT = 0</td>
<td></td>
</tr>
<tr>
<td>FORGPN</td>
<td>0</td>
<td>FDBCN = 1</td>
<td></td>
</tr>
<tr>
<td>FORGPR</td>
<td>0</td>
<td>FDBPR = 1</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>ITYPE=4 -- UNIFORM RECTANGULAR APERTURE</td>
</tr>
</tbody>
</table>

DIMENSIONS = LX,LY = 10.0000, 20.0000
INITLS,DELTAS,FINALS: -5.0000, 0.2000, 5.0000
INITLT,DELTAT,FINALT: -10.0000, 0.4000, 10.0000
MCUR,NCUR: 51 51

REPRODUCIBILITY OF THE ORIGINAL PAGE IS POOR
### CONTOUR LEVEL KEY

<table>
<thead>
<tr>
<th>Level</th>
<th>Lower Value</th>
<th>Upper Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>-0.600000E 00</td>
<td>-0.399999E 00</td>
</tr>
<tr>
<td>1</td>
<td>-0.400000E 00</td>
<td>-0.199999E 00</td>
</tr>
<tr>
<td>2</td>
<td>-0.200000E 00</td>
<td>0.100000E-04</td>
</tr>
<tr>
<td>3</td>
<td>0.0</td>
<td>0.200000E 00</td>
</tr>
<tr>
<td>4</td>
<td>0.200000E 00</td>
<td>0.400000E 00</td>
</tr>
<tr>
<td>5</td>
<td>0.400000E 00</td>
<td>0.600000E 00</td>
</tr>
<tr>
<td>6</td>
<td>0.599999E 00</td>
<td>0.800000E 00</td>
</tr>
<tr>
<td>7</td>
<td>0.799997E 00</td>
<td>0.900010E 01</td>
</tr>
<tr>
<td>8</td>
<td>0.999995E 00</td>
<td>0.120000E 01</td>
</tr>
<tr>
<td>9</td>
<td>0.119999E 01</td>
<td>0.140000E 01</td>
</tr>
<tr>
<td>-</td>
<td>-0.999999E 01</td>
<td>-0.600000E 00</td>
</tr>
<tr>
<td>+</td>
<td>0.140000E 01</td>
<td>0.999999E 30</td>
</tr>
</tbody>
</table>

--- INITIAL COEFFICIENTS ---

<table>
<thead>
<tr>
<th>J</th>
<th>JORG(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>
## FINAL COEFFICIENTS

<table>
<thead>
<tr>
<th>J</th>
<th>US(J)</th>
<th>VS(J)</th>
<th>CORGF(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**
<table>
<thead>
<tr>
<th>Z</th>
<th>0.006</th>
<th>0.153</th>
<th>-0.063</th>
<th>0.000</th>
<th>-0.005</th>
<th>-0.008</th>
<th>-0.010</th>
<th>-0.012</th>
<th>-0.014</th>
<th>-0.016</th>
<th>-0.017</th>
<th>-0.018</th>
<th>-0.018</th>
<th>-0.017</th>
<th>-0.015</th>
<th>-0.013</th>
<th>-0.011</th>
<th>-0.009</th>
<th>-0.006</th>
<th>-0.003</th>
<th>-0.000</th>
<th>-0.000</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.017</td>
<td>+</td>
<td>+</td>
<td>*</td>
<td>+</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>0.003</td>
</tr>
<tr>
<td>0.950</td>
<td>+</td>
<td>+</td>
<td>*</td>
<td>+</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>0.010</td>
</tr>
<tr>
<td>0.900</td>
<td>+</td>
<td>+</td>
<td>*</td>
<td>+</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>0.014</td>
</tr>
<tr>
<td>0.850</td>
<td>+</td>
<td>+</td>
<td>*</td>
<td>+</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>0.017</td>
</tr>
<tr>
<td>0.800</td>
<td>+</td>
<td>+</td>
<td>*</td>
<td>+</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>0.018</td>
</tr>
<tr>
<td>0.750</td>
<td>+</td>
<td>+</td>
<td>*</td>
<td>+</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>0.020</td>
</tr>
<tr>
<td></td>
<td>+</td>
<td>+</td>
<td>*</td>
<td>+</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>0.021</td>
</tr>
<tr>
<td></td>
<td>+</td>
<td>+</td>
<td>*</td>
<td>+</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>0.023</td>
</tr>
<tr>
<td></td>
<td>+</td>
<td>+</td>
<td>*</td>
<td>+</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>0.025</td>
</tr>
<tr>
<td></td>
<td>+</td>
<td>+</td>
<td>*</td>
<td>+</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>0.026</td>
</tr>
<tr>
<td></td>
<td>+</td>
<td>+</td>
<td>*</td>
<td>+</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>0.031</td>
</tr>
<tr>
<td></td>
<td>+</td>
<td>+</td>
<td>*</td>
<td>+</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>0.033</td>
</tr>
<tr>
<td></td>
<td>+</td>
<td>+</td>
<td>*</td>
<td>+</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>0.033</td>
</tr>
</tbody>
</table>

Notes:
- The table represents a 4-axis profile of a pattern with coordinates for different Z values.
- The Z values range from 0.750 to 1.017.
- The profile is defined by Z values and their corresponding X, Y, and Z coordinates.
- The table includes decimal values for X, Y, and Z to approximate the coordinates.
- The pattern number is 77.
- The data is labeled as PATTERN NUMBER 77.
<table>
<thead>
<tr>
<th></th>
<th>0.666</th>
<th>-3.53</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.707</td>
<td>-3.01</td>
<td></td>
</tr>
<tr>
<td>0.747</td>
<td>-2.54</td>
<td></td>
</tr>
<tr>
<td>0.78</td>
<td>-2.11</td>
<td></td>
</tr>
<tr>
<td>0.819</td>
<td>-1.73</td>
<td></td>
</tr>
<tr>
<td>0.852</td>
<td>-1.39</td>
<td></td>
</tr>
<tr>
<td>0.882</td>
<td>-1.09</td>
<td></td>
</tr>
<tr>
<td>0.909</td>
<td>-0.83</td>
<td></td>
</tr>
<tr>
<td>0.933</td>
<td>-0.60</td>
<td></td>
</tr>
<tr>
<td>0.954</td>
<td>-0.41</td>
<td></td>
</tr>
<tr>
<td>0.972</td>
<td>-0.25</td>
<td></td>
</tr>
<tr>
<td>0.986</td>
<td>-0.12</td>
<td></td>
</tr>
<tr>
<td>0.998</td>
<td>-0.02</td>
<td></td>
</tr>
<tr>
<td>1.007</td>
<td>0.06</td>
<td></td>
</tr>
<tr>
<td>1.013</td>
<td>0.11</td>
<td></td>
</tr>
<tr>
<td>1.016</td>
<td>0.14</td>
<td></td>
</tr>
<tr>
<td>1.017</td>
<td>0.15</td>
<td></td>
</tr>
<tr>
<td>1.016</td>
<td>0.14</td>
<td></td>
</tr>
<tr>
<td>1.013</td>
<td>0.11</td>
<td></td>
</tr>
<tr>
<td>1.009</td>
<td>0.07</td>
<td></td>
</tr>
<tr>
<td>1.003</td>
<td>0.02</td>
<td></td>
</tr>
<tr>
<td>0.996</td>
<td>-0.04</td>
<td></td>
</tr>
<tr>
<td>0.988</td>
<td>-0.10</td>
<td></td>
</tr>
<tr>
<td>0.980</td>
<td>-0.17</td>
<td></td>
</tr>
<tr>
<td>0.972</td>
<td>-0.24</td>
<td></td>
</tr>
<tr>
<td>0.964</td>
<td>-0.32</td>
<td></td>
</tr>
<tr>
<td>0.957</td>
<td>-0.38</td>
<td></td>
</tr>
<tr>
<td>0.950</td>
<td>-0.45</td>
<td></td>
</tr>
<tr>
<td>0.943</td>
<td>-0.51</td>
<td></td>
</tr>
<tr>
<td>0.938</td>
<td>-0.55</td>
<td></td>
</tr>
<tr>
<td>0.934</td>
<td>-0.59</td>
<td></td>
</tr>
<tr>
<td>0.931</td>
<td>-0.62</td>
<td></td>
</tr>
<tr>
<td>0.929</td>
<td>-0.64</td>
<td></td>
</tr>
<tr>
<td>0.920</td>
<td>-0.66</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>-0.011</th>
<th>-35.04</th>
</tr>
</thead>
<tbody>
<tr>
<td>-0.018</td>
<td>-35.14</td>
<td></td>
</tr>
<tr>
<td>-0.018</td>
<td>-34.87</td>
<td></td>
</tr>
<tr>
<td>-0.018</td>
<td>-34.83</td>
<td></td>
</tr>
<tr>
<td>-0.018</td>
<td>-35.00</td>
<td></td>
</tr>
<tr>
<td>-0.017</td>
<td>-35.39</td>
<td></td>
</tr>
<tr>
<td>-0.016</td>
<td>-36.01</td>
<td></td>
</tr>
<tr>
<td>-0.014</td>
<td>-36.91</td>
<td></td>
</tr>
<tr>
<td>-0.012</td>
<td>-38.13</td>
<td></td>
</tr>
<tr>
<td>-0.010</td>
<td>-39.79</td>
<td></td>
</tr>
<tr>
<td>-0.008</td>
<td>-42.08</td>
<td></td>
</tr>
<tr>
<td>-0.005</td>
<td>-45.46</td>
<td></td>
</tr>
<tr>
<td>-0.003</td>
<td>-51.37</td>
<td></td>
</tr>
</tbody>
</table>
V-AXIS PROFILE OF FINAL PATTERN -- U = 0.0
PATTERN NUMBER 77

<table>
<thead>
<tr>
<th>Pattern Number</th>
<th>Real</th>
<th>DB</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.000</td>
<td>1.056</td>
<td>0.163</td>
</tr>
<tr>
<td>1.000</td>
<td>0.002</td>
<td>-55.08</td>
</tr>
<tr>
<td>0.950</td>
<td>0.003</td>
<td>-54.16</td>
</tr>
<tr>
<td>0.900</td>
<td>0.002</td>
<td>-53.19</td>
</tr>
<tr>
<td>0.850</td>
<td>0.004</td>
<td>-52.15</td>
</tr>
<tr>
<td>0.800</td>
<td>0.007</td>
<td>-51.10</td>
</tr>
<tr>
<td>1.000</td>
<td>0.009</td>
<td>-50.07</td>
</tr>
<tr>
<td>S</td>
<td>T</td>
<td>REAL</td>
</tr>
<tr>
<td>-----</td>
<td>-----</td>
<td>-----------------</td>
</tr>
<tr>
<td>-5.0000</td>
<td>-0.0000</td>
<td>0.5700823E-02</td>
</tr>
<tr>
<td>-4.8000</td>
<td>-0.0000</td>
<td>0.7003162E-02</td>
</tr>
<tr>
<td>-4.6000</td>
<td>-0.0000</td>
<td>0.6889808E-02</td>
</tr>
<tr>
<td>-4.4000</td>
<td>-0.0000</td>
<td>0.5990162E-02</td>
</tr>
<tr>
<td>-4.2000</td>
<td>-0.0000</td>
<td>0.4723225E-02</td>
</tr>
<tr>
<td>-4.0000</td>
<td>-0.0000</td>
<td>0.3161762E-02</td>
</tr>
<tr>
<td>-3.8000</td>
<td>-0.0000</td>
<td>0.1145484E-02</td>
</tr>
<tr>
<td>-3.6000</td>
<td>-0.0000</td>
<td>-0.1570189E-02</td>
</tr>
<tr>
<td>-3.4000</td>
<td>-0.0000</td>
<td>-0.5123518E-02</td>
</tr>
<tr>
<td>-3.2000</td>
<td>-0.0000</td>
<td>-0.9384781E-02</td>
</tr>
<tr>
<td>-3.0000</td>
<td>-0.0000</td>
<td>-0.1387722E-01</td>
</tr>
<tr>
<td>-2.8000</td>
<td>-0.0000</td>
<td>-0.1780687E-01</td>
</tr>
<tr>
<td>-2.6000</td>
<td>-0.0000</td>
<td>-0.2019734E-01</td>
</tr>
<tr>
<td>-2.4000</td>
<td>-0.0000</td>
<td>-0.2009303E-01</td>
</tr>
<tr>
<td>-2.2000</td>
<td>-0.0000</td>
<td>-0.1677621E-01</td>
</tr>
<tr>
<td>-2.0000</td>
<td>-0.0000</td>
<td>-0.9939682E-02</td>
</tr>
<tr>
<td>-1.8000</td>
<td>-0.0000</td>
<td>0.2297403E-03</td>
</tr>
<tr>
<td>-1.6000</td>
<td>-0.0000</td>
<td>0.1307478E-01</td>
</tr>
<tr>
<td>-1.4000</td>
<td>-0.0000</td>
<td>0.2758893E-01</td>
</tr>
<tr>
<td>-1.2000</td>
<td>-0.0000</td>
<td>0.4259761E-01</td>
</tr>
<tr>
<td>-1.0000</td>
<td>-0.0000</td>
<td>0.5695011E-01</td>
</tr>
<tr>
<td>-0.8000</td>
<td>-0.0000</td>
<td>0.6967366E-01</td>
</tr>
<tr>
<td>-0.6000</td>
<td>-0.0000</td>
<td>0.8006346E-01</td>
</tr>
<tr>
<td>-0.4000</td>
<td>-0.0000</td>
<td>0.8768129E-01</td>
</tr>
<tr>
<td>-0.2000</td>
<td>-0.0000</td>
<td>0.9230632E-01</td>
</tr>
<tr>
<td>-0.0000</td>
<td>-0.0000</td>
<td>0.9385431E-01</td>
</tr>
<tr>
<td>0.2000</td>
<td>-0.0000</td>
<td>0.9230632E-01</td>
</tr>
<tr>
<td>0.4000</td>
<td>-0.0000</td>
<td>0.8768129E-01</td>
</tr>
<tr>
<td>0.6000</td>
<td>-0.0000</td>
<td>0.8006346E-01</td>
</tr>
<tr>
<td>0.8000</td>
<td>-0.0000</td>
<td>0.6967366E-01</td>
</tr>
<tr>
<td>1.0000</td>
<td>-0.0000</td>
<td>0.5695021E-01</td>
</tr>
<tr>
<td>1.2000</td>
<td>-0.0000</td>
<td>0.4259773E-01</td>
</tr>
<tr>
<td>1.4000</td>
<td>-0.0000</td>
<td>0.2758904E-01</td>
</tr>
<tr>
<td>1.6000</td>
<td>-0.0000</td>
<td>0.1307483E-01</td>
</tr>
<tr>
<td>1.8000</td>
<td>-0.0000</td>
<td>0.2297729E-03</td>
</tr>
<tr>
<td>2.0000</td>
<td>-0.0000</td>
<td>-0.9939581E-02</td>
</tr>
<tr>
<td>2.2000</td>
<td>-0.0000</td>
<td>-0.1677619E-01</td>
</tr>
<tr>
<td>2.4000</td>
<td>-0.0000</td>
<td>-0.2009305E-01</td>
</tr>
<tr>
<td>2.6000</td>
<td>-0.0000</td>
<td>-0.2019734E-01</td>
</tr>
<tr>
<td>2.8000</td>
<td>-0.0000</td>
<td>-0.1780686E-01</td>
</tr>
<tr>
<td>3.0000</td>
<td>-0.0000</td>
<td>-0.1387723E-01</td>
</tr>
<tr>
<td>3.2000</td>
<td>-0.0000</td>
<td>-0.9384889E-02</td>
</tr>
<tr>
<td>3.4000</td>
<td>-0.0000</td>
<td>-0.5123563E-02</td>
</tr>
<tr>
<td>3.6000</td>
<td>-0.0000</td>
<td>-0.1570206E-02</td>
</tr>
<tr>
<td>3.8000</td>
<td>-0.0000</td>
<td>0.1145449E-02</td>
</tr>
<tr>
<td>4.0000</td>
<td>-0.0000</td>
<td>0.3161710E-02</td>
</tr>
<tr>
<td>4.2000</td>
<td>-0.0000</td>
<td>0.4723225E-02</td>
</tr>
<tr>
<td>4.4000</td>
<td>-0.0000</td>
<td>0.5990129E-02</td>
</tr>
<tr>
<td>4.6000</td>
<td>-0.0000</td>
<td>0.6880980E-02</td>
</tr>
<tr>
<td>4.8000</td>
<td>-0.0000</td>
<td>0.7003162E-02</td>
</tr>
<tr>
<td>5.0000</td>
<td>-0.0000</td>
<td>0.5700838E-02</td>
</tr>
<tr>
<td>S</td>
<td>T</td>
<td>REAL</td>
</tr>
<tr>
<td>-----</td>
<td>-----</td>
<td>----------------</td>
</tr>
<tr>
<td>-0.0000</td>
<td>-10.0000</td>
<td>-0.2455123E-01</td>
</tr>
<tr>
<td>-0.0000</td>
<td>-9.6000</td>
<td>-0.1838829E-01</td>
</tr>
<tr>
<td>-0.0000</td>
<td>-9.2000</td>
<td>-0.1313753E-01</td>
</tr>
<tr>
<td>-0.0000</td>
<td>-8.8000</td>
<td>-0.9560350E-02</td>
</tr>
<tr>
<td>-0.0000</td>
<td>-8.4000</td>
<td>-0.7850584E-02</td>
</tr>
<tr>
<td>-0.0000</td>
<td>-8.0000</td>
<td>-0.7672068E-02</td>
</tr>
<tr>
<td>-0.0000</td>
<td>-7.6000</td>
<td>-0.3307122E-02</td>
</tr>
<tr>
<td>-0.0000</td>
<td>-7.2000</td>
<td>-0.8702486E-02</td>
</tr>
<tr>
<td>-0.0000</td>
<td>-6.8000</td>
<td>-0.8532612E-02</td>
</tr>
<tr>
<td>-0.0000</td>
<td>-6.4000</td>
<td>-0.6700886E-02</td>
</tr>
<tr>
<td>-0.0000</td>
<td>-6.0000</td>
<td>-0.3113753E-02</td>
</tr>
<tr>
<td>-0.0000</td>
<td>-5.6000</td>
<td>0.2147641E-02</td>
</tr>
<tr>
<td>-0.0000</td>
<td>-5.2000</td>
<td>0.8754689E-02</td>
</tr>
<tr>
<td>-0.0000</td>
<td>-4.8000</td>
<td>0.1624599E-01</td>
</tr>
<tr>
<td>-0.0000</td>
<td>-4.4000</td>
<td>0.2417203E-01</td>
</tr>
<tr>
<td>-0.0000</td>
<td>-4.0000</td>
<td>0.3219855E-01</td>
</tr>
<tr>
<td>-0.0000</td>
<td>-3.6000</td>
<td>0.4015272E-01</td>
</tr>
<tr>
<td>-0.0000</td>
<td>-3.2000</td>
<td>0.4800258E-01</td>
</tr>
<tr>
<td>-0.0000</td>
<td>-2.8000</td>
<td>0.5578430E-01</td>
</tr>
<tr>
<td>-0.0000</td>
<td>-2.4000</td>
<td>0.6350774E-01</td>
</tr>
<tr>
<td>-0.0000</td>
<td>-2.0000</td>
<td>0.7107019E-01</td>
</tr>
<tr>
<td>-0.0000</td>
<td>-1.6000</td>
<td>0.7821018E-01</td>
</tr>
<tr>
<td>-0.0000</td>
<td>-1.2000</td>
<td>0.8452117E-01</td>
</tr>
<tr>
<td>-0.0000</td>
<td>-0.8000</td>
<td>0.8951896E-01</td>
</tr>
<tr>
<td>-0.0000</td>
<td>-0.4000</td>
<td>0.9274071E-01</td>
</tr>
<tr>
<td>-0.0000</td>
<td>0.0000</td>
<td>0.9385431E-01</td>
</tr>
<tr>
<td>-0.0000</td>
<td>0.4000</td>
<td>0.9274071E-01</td>
</tr>
<tr>
<td>-0.0000</td>
<td>0.8000</td>
<td>0.8951896E-01</td>
</tr>
<tr>
<td>-0.0000</td>
<td>1.2000</td>
<td>0.8452106E-01</td>
</tr>
<tr>
<td>-0.0000</td>
<td>1.6000</td>
<td>0.7821023E-01</td>
</tr>
<tr>
<td>-0.0000</td>
<td>2.0000</td>
<td>0.7107037E-01</td>
</tr>
<tr>
<td>-0.0000</td>
<td>2.4000</td>
<td>0.6350780E-01</td>
</tr>
<tr>
<td>-0.0000</td>
<td>2.8000</td>
<td>0.5578437E-01</td>
</tr>
<tr>
<td>-0.0000</td>
<td>3.2000</td>
<td>0.4800263E-01</td>
</tr>
<tr>
<td>-0.0000</td>
<td>3.6000</td>
<td>0.4015278E-01</td>
</tr>
<tr>
<td>-0.0000</td>
<td>4.0000</td>
<td>0.3219860E-01</td>
</tr>
<tr>
<td>-0.0000</td>
<td>4.4000</td>
<td>0.2417206E-01</td>
</tr>
<tr>
<td>-0.0000</td>
<td>4.8000</td>
<td>0.1624606E-01</td>
</tr>
<tr>
<td>-0.0000</td>
<td>5.2000</td>
<td>0.8754738E-02</td>
</tr>
<tr>
<td>-0.0000</td>
<td>5.6000</td>
<td>0.2147641E-02</td>
</tr>
<tr>
<td>-0.0000</td>
<td>6.0000</td>
<td>-0.3115309E-02</td>
</tr>
<tr>
<td>-0.0000</td>
<td>6.4000</td>
<td>-0.6700821E-02</td>
</tr>
<tr>
<td>-0.0000</td>
<td>6.8000</td>
<td>-0.8532569E-02</td>
</tr>
<tr>
<td>-0.0000</td>
<td>7.2000</td>
<td>-0.8870188E-02</td>
</tr>
<tr>
<td>-0.0000</td>
<td>7.6000</td>
<td>-0.3071222E-02</td>
</tr>
<tr>
<td>-0.0000</td>
<td>8.0000</td>
<td>-0.7672135E-02</td>
</tr>
<tr>
<td>-0.0000</td>
<td>8.4000</td>
<td>-0.7850584E-02</td>
</tr>
<tr>
<td>-0.0000</td>
<td>8.8000</td>
<td>-0.9560246E-02</td>
</tr>
<tr>
<td>-0.0000</td>
<td>9.2000</td>
<td>-0.1313743E-01</td>
</tr>
<tr>
<td>-0.0000</td>
<td>9.6000</td>
<td>-0.1838810E-01</td>
</tr>
<tr>
<td>-0.0000</td>
<td>10.0000</td>
<td>-0.2455102E-01</td>
</tr>
</tbody>
</table>

PATTERN NUMBER 77 HAS BEEN STORED ON RECORD 20 OF ANTDATA.A507C2
8.3 Input to ANTDATA

Referring to Section 7.3, the following cards were punched.

Card Column

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>11</th>
<th>21</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0007700020</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>0015100151</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>1111</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>0.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>0.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>-35.0</td>
<td>-35.0</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>-30.0</td>
<td>0.0</td>
<td>5.0</td>
</tr>
<tr>
<td>8</td>
<td>0000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>0000</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

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

8.4 Output from ANTDATA

The following is the printout from computer program ANTDATA.

PLOT OUTPUT FOR PATTERN 77:

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

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

CONTOUR PLOT OF PATTERN REQUESTED.

LOWEST CONTOUR = -30.0

HIGHEST CONTOUR = 0.0

CONTOUR INTERVAL = 5.0

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

THREE-DIMENSIONAL PLOT OF PATTERN REQUESTED

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

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