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

Page

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.

Radiation Pattern Variables	Antenna Variables
I. Main Beams	I. Shape
A. Number	A. Linear
1. Single	1. Linear array
2. Multiple	2. Line source
B. Shape	B. Planar
1. Nominal	1. Planar array
2. Shape	2. Planar aperture
II. Side Lobes	II. Size
A. Nominal	
B. Low	
C. Complex	

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 simplier 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:

Continuous Aperture Sources	Arrays
Line source	Linear Array
Rectangular Aperture	Rectangular Array
Circular Aperture	Arbitrary Planar Array

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

2. Mathematical Modeling of Antennas

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

2.1 Equivalent Currents

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

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

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

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

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

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

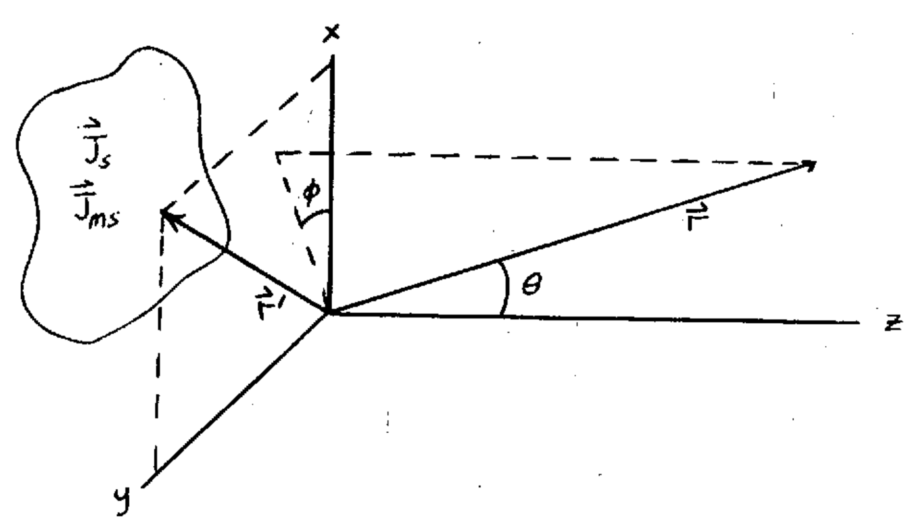
If the equivalent surface S is now a plane surface (closed at infinity such that actual sources are inside), calculations are simplified. Let S be the z = 0 plane and suppose the actual sources are on the left ($z < 0$). The surface normal is then $\hat{n} = \hat{z}$. The simplification arises from the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

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

$$\vec{A}(r) = \mu_o \frac{e^{-jkr}}{4\pi r} \iint \vec{J}_S(r') \, e^{jk\hat{r}\cdot\vec{r}'} \, dx'dy' \qquad (2\text{-}3)$$

$$\vec{A}_M(r) = \varepsilon_o \frac{e^{-jkr}}{4\pi r} \iint \vec{J}_{MS}(r') \, e^{jk\hat{r}\cdot\vec{r}'} \, dx'dy' \qquad (2\text{-}4)$$

where $\vec{r}' = x'\hat{x} + y'\hat{y}$ and the coordinate system is shown below



The currents are doubled for cases 2) and 3).

2.2 Representing Antennas as Finite Apertures

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

The approximate equivalent currents over the aperture are

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

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

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

$$E_\theta = -j\omega A_\theta - j\omega\eta_o A_{M\phi} \tag{2-7}$$

$$E_\phi = -j\omega A_\phi + j\eta_o A_{M\theta} \tag{2-8}$$

where $\eta_o = \sqrt{\mu_o/\varepsilon_o}$ and the magnetic fields are found using the plane wave relation

$$\vec{H} = \hat{r} \times \vec{E}/\eta_o \tag{2-9}$$

As an example suppose we use only a magnetic current and the aperture electric field is y-directed. Then

$$\vec{J}_{MS} = J_{MSx}\,\hat{x} = \hat{z} \times E_{ay}\,\hat{y} = E_{ay}\,\hat{x} \tag{2-10}$$

and

$$A_{Mx} = \frac{\varepsilon_o}{4\pi r}\, e^{-jkr} \iint_{\text{aperture}} 2\, E_{ay}\, e^{jk\hat{r}\cdot\vec{r}\,'}\, dx'dy' \tag{2-11}$$

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

$$A_{M\theta} = \cos\theta\,\cos\phi\, A_{Mx} \tag{2-12}$$

$$A_{M\phi} = -\sin\phi\, A_{Mx} \tag{2-13}$$

So

$$\begin{aligned} E_\theta &= -j\omega\eta_o\, A_{M\phi} \\ &= +j\omega\eta_o \sin\phi\, A_{Mx} \end{aligned} \tag{2-14}$$

$$\begin{aligned} E_\phi &= +j\eta_o\, A_{M\theta} \\ &= j\eta_o \cos\theta\,\cos\phi\, A_{Mx} \end{aligned} \tag{2-15}$$

And

$$A_{Mx} = \frac{\varepsilon_o}{2\pi r}\, e^{-jkr} \iint_{\text{aperture}} E_{ay}\, e^{jk(x'\sin\theta\cos\phi + y'\sin\theta\sin\phi)} dx'dy' \tag{2-16}$$

So

$$E_\theta = \frac{jke^{-jkr}}{2\pi r} \sin\phi\, F_y \tag{2-17}$$

$$E_\phi = \frac{jke^{-jkr}}{2\pi r} \cos\theta\,\cos\phi\, F_y \tag{2-18}$$

where

$$F_y = \iint_{\text{aperture}} E_{ay}\, e^{jk(x'\sin\theta\cos\phi + y'\sin\theta\sin\phi)} dx'dy' \tag{2-19}$$

If there is an x-directed component of the aperture electric field the far-field expressions become

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

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

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

$$E_\theta = \frac{jk\eta_o e^{-jkr}}{2\pi r} \cos \theta \; (F_y \cos \phi - F_x \sin \phi) \tag{2-22}$$

$$E_\phi = \frac{-jk\eta_o e^{-jkr}}{2\pi r} (F_y \sin \phi + F_x \cos \phi) \tag{2-23}$$

where

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

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

These solutions are exact if a complete knowledge of the fields over the entire aperture plane is available. This is usually never possible. In fact, some assumption about the aperture fields is made in addition to the assumption that the fields are zero outside the aperture. If the aperture is connected to an infinite plane perfect conductor the formulation using magnetic current only is exact within the limits of a knowledge of the tangential electric field over the aperture. This is true because the tangential electric field is zero over the perfect conductor and thus the equivalent magnetic current is also.

As a first approximation to the aperture fields frequently the so-called physical optics approximation is used. It assumes that the aperture fields are those incident upon it from the actual source. For example, the physical-optics fields in the aperture (or mouth) of a horn antenna are those of the waveguide feed [4].

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

$$\vec{J}_S = \hat{x}\, J_{Sx}\,(x')\; J_{Sx}\,(y') + \hat{y}\, J_{Sy}\,(x')\; J_{Sy}\,(y') \tag{2-25}$$

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 $\vec{E}_a$ and $\vec{H}_a$ are usually separate and render the two-dimensional integrals of (2-19) and (2-24) a product of one-dimensional integrals.

2.3 Vector Radiation Fields

In the radiation field (or far-field) the waves are locally plane and may be completely described by θ and ϕ 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

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

$$E_\phi = E(r)\ [-\cos\theta\ \sin\phi\ F_x + \cos\theta\ \cos\phi\ F_y] \tag{2-27}$$

where

$$E(r) = \frac{jke^{-jkr}}{2\pi r} \tag{2-28}$$

This can be cast in a matrix form

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

where

$$\begin{aligned} E_\theta(\theta,\phi) &= E_\theta/E(r) \\ E_\phi(\theta,\phi) &= E_\phi/E(r) \end{aligned} \tag{2-30}$$

and

$$\begin{aligned} G_{\theta x} &= \cos\phi & G_{\theta y} &= \sin\phi \\ G_{\phi x} &= -\cos\theta\ \sin\phi & G_{\theta y} &= \cos\theta\ \cos\phi \end{aligned} \tag{2-31}$$

In still more compact form (2-29) becomes

$$[E] = [G][F] \tag{2-32}$$

This formulation is particularly convenient for synthesis problems. If a certain desired electric feild 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 θ. 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

$$F_x = \int \int I_x \, (x',y') \, e^{jk\alpha} \, dx'dy' \tag{2-37}$$

where we let

$$\alpha = x' \sin\theta \cos\phi + y' \sin\theta \sin\phi \tag{2-38}$$

becomes

$$F_x = G_x \sum_{m=1}^{P} I_{xm} \, e^{jk\alpha_m} = G_x \, F_{arx} \tag{2-39}$$

where

$$\alpha_m = x_m' \sin\theta \cos\phi + y_m' \sin\theta \sin\phi \tag{2-40}$$

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

$$F_y = G_y \sum_{m=1}^{P} I_{ym} \, e^{jk\alpha_m} = G_y \, F_{ary} \tag{2-41}$$

These element factors may be combined into [G] giving

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

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

$$[E] = [G_{ar}] \, [F_{ar}] \tag{2-43}$$

where the $[F_{ar}]$ entries are the array factors

$$F_{arx} = \sum_{m=1}^{P} I_{xm} \, e^{jk\alpha_m} \qquad F_{ary} = \sum_{m=1}^{P} I_{ym} \, e^{jk\alpha_m} \tag{2-44}$$

<u>Example</u> - A linear array of parallel short dipoles along the x-axis.

Since the current is y-directed we have

$$F_x = 0 \tag{2-45}$$

$$F_y = G_y \, F_{ary} \tag{2-46}$$

The short dipole pattern is

$$G_y = \sin \beta \tag{2-47}$$

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

$$G_y = \sqrt{1 - (\sin \theta \sin \phi)^2} \tag{2-48}$$

Now

$$F_{ary} = \Sigma\, I_{ym}\, e^{jk\,\alpha_m} \tag{2-49}$$

where

$$\alpha_m = x_m' \sin \theta \cos \phi \tag{2-50}$$

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

$$[V] = [Z][I] \tag{2-51}$$

where

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

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

$$[Z] = \begin{bmatrix} Z_{11} & Z_{12} & \cdots \\ Z_{21} & \cdots & \\ \vdots & & \end{bmatrix}$$ = 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 θ and ϕ components of the electric field are desired to be of a certain relative level as a function of θ and ϕ. The desired $E_\theta(\theta,\phi)$ and $E_\phi(\theta,\phi)$ are converted into desired $F_x(\theta,\phi)$ and $F_y(\theta,\phi)$ using (2-33). The synthesis problem is to find the aperture fields $E_{ax}(x',y')$ and $E_{ay}(x',y')$ which produce sufficiently accurate approximations to the desired $F_x(\theta,\phi)$ and $F_y(\theta,\phi)$, respectively. This amounts to solving the integral equations (2-34) and (2-35). Since these two equations are identical in form we will drop subscripts which refer to polarization, while remembering that two polarizations (alone or together) are possible. The integral equation which we wish to solve is then

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

where

$$u = \sin\theta \cos\phi \tag{3-2}$$

$$v = \sin\theta \sin\phi. \tag{3-3}$$

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

$$\begin{aligned} s &= x'/\lambda \\ t &= y'/\lambda \end{aligned} \tag{3-4}$$

and source function

$$f(s,t) = \begin{cases} \lambda^2 E_a(x',y') & \text{over the aperture} \\ \\ 0 & \text{elsewhere} \end{cases} \tag{3-5}$$

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

$$F(u,v) = \int\int f(s,t)\, e^{j2\pi(su + tv)}\, dsdt \tag{3-6}$$

This integral extends over the whole st-plane and is recognized as a two-dimensional Fourier transform. The analysis problem is straightforward. Given an aperture distribution f we can calculate F from (3-6) by integration. The synthesis problem, however, is very difficult. Suppose we are given a desired pattern $F_d(u,v)$, which can be found from a desired electric field using (2-33). We wish to find an f (which is aperture-limited) giving an F which approximates F_d in some specified manner.

3.4 Mathematical Development of the Method

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

A series of corrections is added to the original pattern giving

$$F^{(K)}\ (u,v) = F^{(o)}\ (u,v) + \sum_{i=1}^{K} \Delta\ F^{(i)}\ (u,v) \qquad (3\text{-}7)$$

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

$$\Delta\ F^{(i)}\ (u,v) = \sum_{n} a_n^{(i)}\ G(u-u_n^{(i)},\ v-v_n^{(i)}) \qquad (3\text{-}8)$$

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

$$a_n^{(i)} = F_d\ (u_n^{(i)},v_n^{(i)}) - F^{(i-1)}\ (u_n^{(i)},v_n^{(i)}) \qquad (3\text{-}9)$$

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:

$$G(u-u_n^{(i)}, v-v_n^{(i)}) = \iint_{\text{aperture}} g_n^{(i)}(s,t)\, e^{j(2\pi su + 2\pi tv)}\, ds\, dt \qquad (3\text{-}10)$$

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

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

where

$$\Delta f^{(i)}(s,t) = \sum_{n} a_n^{(i)} g_n^{(i)}(s,t) \tag{3-12}$$

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

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

$$f^{(K)}(s,t) = \sum_{\ell} \sum_{m} I_{\ell m}^{(K)} \delta(s-s_\ell, t-t_m) \tag{3-13}$$

where δ is the dirac delta function and $I_{\ell m}^{(K)}$ are the currents for the ℓm 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

$$g_n^{(i)}(s,t) = \sum_{\ell} \sum_{m} g_{n\ell m}^{(i)} \delta(s-s_\ell, t-t_m) \tag{3-14}$$

for arrays. Then (3-10) becomes

$$G(u-u_n^{(i)}, v-v_n^{(i)}) = \sum \sum g_{n\ell m}^{(i)} e^{j2\pi(s_\ell u + t_m v)} \tag{3-15}$$

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

$$\Delta f^{(i)}(s,t) = \sum_{n} a_n^{(i)} \sum_{\ell} \sum_{m} g_{n\ell m}^{(i)} \delta(s-s_\ell, t-t_m) \tag{3-16}$$

and let

$$\Delta f^{(i)}(s,t) = \sum_{\ell} \sum_{m} \Delta I_{\ell m}^{(i)} \delta(s-s_\ell, t-t_m) \tag{3-17}$$

So

$$\Delta I_{\ell m}^{(i)} = \sum_n a_n^{(i)} g_{n\ell m}^{(i)} \tag{3-18}$$

and

$$I_{\ell m}^{(K)} = I_{\ell m}^{(o)} + \sum_{i=1}^{K} \Delta I_{\ell m}^{(i)} \tag{3-19}$$

3.5 Common Antenna Types

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

3.5.1 Line Sources

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

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

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

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

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

$$g_n^{(i)}(t) = \begin{cases} L_{y\lambda}^{-1}\,(1 - 2|t|/L_{y\lambda})\,\exp(-j2\pi v_n^{(i)} t) & |t| \le L_{y\lambda}/2 \\ 0 & \text{elsewhere} \end{cases} \tag{3-22}$$

The corresponding pattern found from (3-10) is

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

3.5.2 Linear Array

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

$$g_{nm}^{(i)} = \frac{1}{P}\exp(-j2\pi v_n^{(i)}\,t_m) \tag{3-24}$$

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

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

3.5.3 Rectangular Aperture

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

$$g_n^{(i)}(s,t) = \begin{cases} L_{x\lambda}^{-1}\,L_{y\lambda}^{-1}\,\exp(-j2\pi(u_n^{(i)}s + v_n^{(i)}t)) & \begin{array}{l}|s| \le L_{x\lambda}/2\\ |t| \le L_{y\lambda}/2\end{array} \\ 0 & \text{elsewhere} \end{cases} \tag{3-26}$$

The pattern is

$$G(u-u_n^{(i)},\ v-v_n^{(i)}) = \frac{\sin\,[L_{x\lambda}(u-u_n^{(i)})\pi]}{L_{x\lambda}(u-u_n^{(i)})\pi}\ \frac{\sin\,[L_{y\lambda}(v-v_n^{(i)})\pi]}{L_{y\lambda}(v-v_n^{(i)})\pi} \tag{3-27}$$

3.5.4 Rectangular Array

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

$$g_{n\ell m}^{(i)} = \frac{1}{P_x P_y} \exp\left[-j2\pi(u_n^{(i)} s_\ell + v_n^{(i)} t_m)\right] \tag{3-28}$$

The pattern is

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

3.5.5 Circular Aperture

Consider a uniform amplitude, linear phase, circular source a radius a_λ wavelengths. The source function is

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

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

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

The integral (3-10) over the source (3-31) is

$$G\,(u-u_n^{(i)},\ v-v_n^{(i)})$$
$$= \int_0^{2\pi}\int_0^{a_\lambda} \frac{1}{\pi a_\lambda^2} \exp\{j\ 2\pi\rho_\lambda'\ [(u-u_n^{(i)}) \cos\phi'$$
$$+ (v-v_n^{(i)}) \sin\phi']\}\ \rho_\lambda'\ d\rho_\lambda'\ d\phi' \tag{3-32}$$
$$= \frac{1}{\pi a_\lambda^2} \int_0^{2\pi}\int_0^{a_\lambda} \exp\,[j\ 2\pi\rho_\lambda'\ C \cos\,(\alpha - \phi')]\ \rho_\lambda'\ d\rho_\lambda'\ d\phi' \tag{3-33}$$

where

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

Now (3-33) is easily integrated as

$$G\,(u-u_n^{(i)},\ v-v_n^{(i)}) = \frac{2\pi}{\pi a_\lambda^2} \int_0^{a_\lambda} J_0(2\pi\rho_\lambda' C)\ \rho_\lambda'\ d\rho_\lambda' \tag{3-37}$$
$$= 2\,\frac{J_1(2\pi a_\lambda C)}{2\pi a_\lambda C} \tag{3-38}$$

If $u_n^{(i)} = v_n^{(i)} = 0$ we have

$$G_n\,(u,v) = 2\,\frac{J_1(2\pi a_\lambda \sin\theta)}{2\pi a_\lambda \sin\theta} \tag{3-39}$$

which is the pattern of a uniform amplitude, zero phase, circular source. [4] Also note that when $u = u_n^{(i)}$ and $v = v_n^{(i)}$, $C = 0$ and (3-38) becomes unity. Thus $(u_n^{(i)}, v_n^{(i)})$ is the pattern maximum.

3.5.6 Arbitrary Planar Array

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

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

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 θ and ϕ. 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

$$D = \frac{4\pi}{\Omega_A} \quad (3\text{-}41)$$

The beam solid angle Ω_A is given by

$$\Omega_A = \int_0^{2\pi}\int_0^{\pi} |F(\theta,\phi)|^2 \, d\Omega \quad (3\text{-}42)$$

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

$$d\Omega = \sin\theta \, d\theta \, d\phi \quad (3\text{-}43)$$

It is frequently convenient to transform from the θ,ϕ space to the u,v plane using

$$\begin{aligned} u &= \sin\theta\cos\phi \\ v &= \sin\theta\sin\phi \end{aligned} \tag{3-44}$$

We are collapsing the spherical surface described by θ,ϕ onto a planar surface through its equator giving a circular disk. There is an ambiquity 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

$$\Omega_A = \int_0^{2\pi}\int_0^{\pi/2} |F(\theta,\phi)|^2 \, d\Omega \tag{3-45}$$

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 Ω_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

$$du\,dv = \cos\theta \, d\Omega \tag{3-46}$$

so

$$d\Omega = \frac{du\,dv}{\cos\theta} \tag{3-47}$$

But from (3-44)

$$\cos\theta = \sqrt{1 - u^2 - v^2} \tag{3-48}$$

so

$$d\Omega = \frac{du\ dv}{\sqrt{1 - u^2 - v^2}} \tag{3-49}$$

Thus (3-45) becomes

$$\Omega_A = \int\int_{u^2+v^2\leq 1} |F(u,v)|^2 \frac{du\ dv}{\sqrt{1 - u^2 - v^2}} \tag{3-50}$$

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

$$\Omega_A = \int\int_{u^2+v^2\leq 1} |F(u,v)|^2 \sin\theta\ J\ du\ dv \tag{3-51}$$

where J is the Jacobian given by

$$J = \frac{\partial(\theta,\phi)}{\partial(u,v)} = \frac{1}{\frac{\partial(u,v)}{\partial(\theta,\phi)}} \tag{3-52}$$

and

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

$$= \begin{vmatrix} \cos\theta\cos\phi & -\sin\theta\sin\phi \\ \cos\theta\sin\phi & \sin\theta\cos\phi \end{vmatrix}$$

$$= \cos\theta\sin\theta \tag{3-53}$$

So

$$\sin\theta\ J\ du\ dv = \sin\theta\ \frac{du\ dv}{\cos\theta\sin\theta}$$

$$= \frac{du\ dv}{\sqrt{1 - u^2 - v^2}} \tag{3-54}$$

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

$$F(u,v) = 1.0 \qquad u^2+v^2 \leq 1$$

Using (3-50)

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

Let $r^2 = u^2+v^2$ then

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

$$= \int_0^{2\pi} d\alpha \int_0^1 X^{-1/2} \left(- \frac{dX}{2}\right) = 2\pi$$

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 realtive to the uniform amplitude line source.

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

The patterns of a uniformly excited rectangular array have been omitted because of their similarity to the continuous aperture patterns for element spacings of a half wavelength or less.

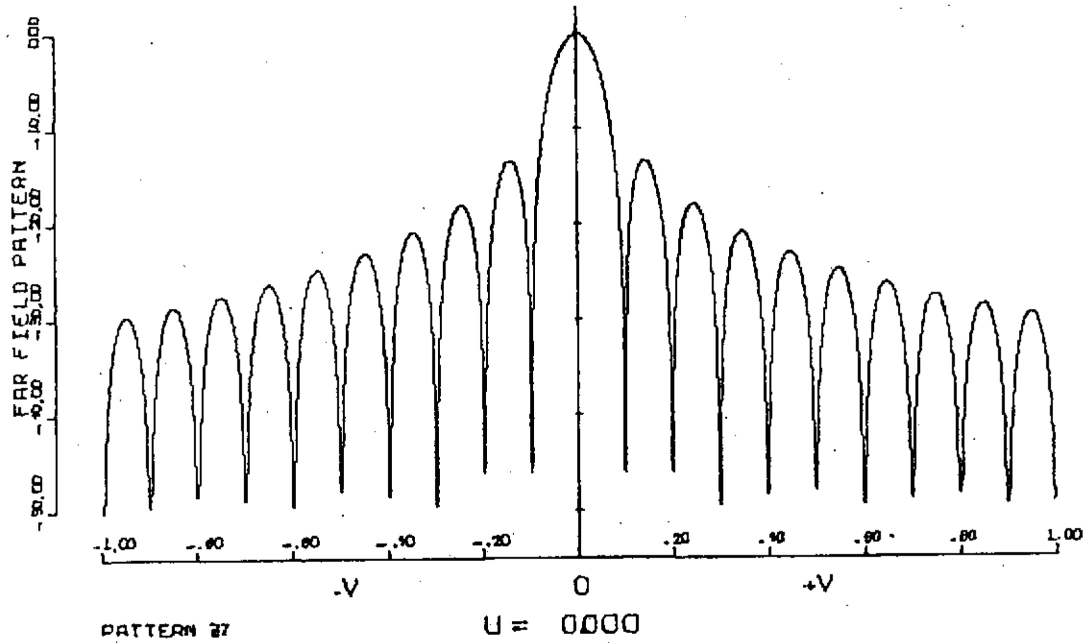


Figure 4.1 Pattern of a uniform amplitude, uniform phase, ten wavelength line source.

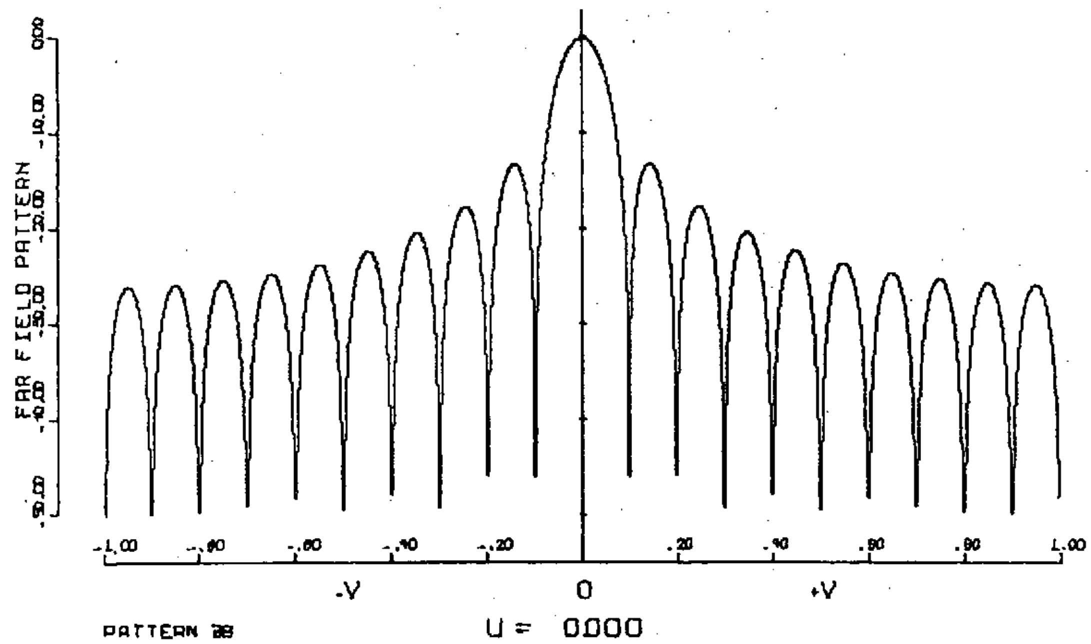


Figure 4.2 Pattern of a uniform amplitude, uniform phase, half-wavelength spaced, 21 element, linear array.

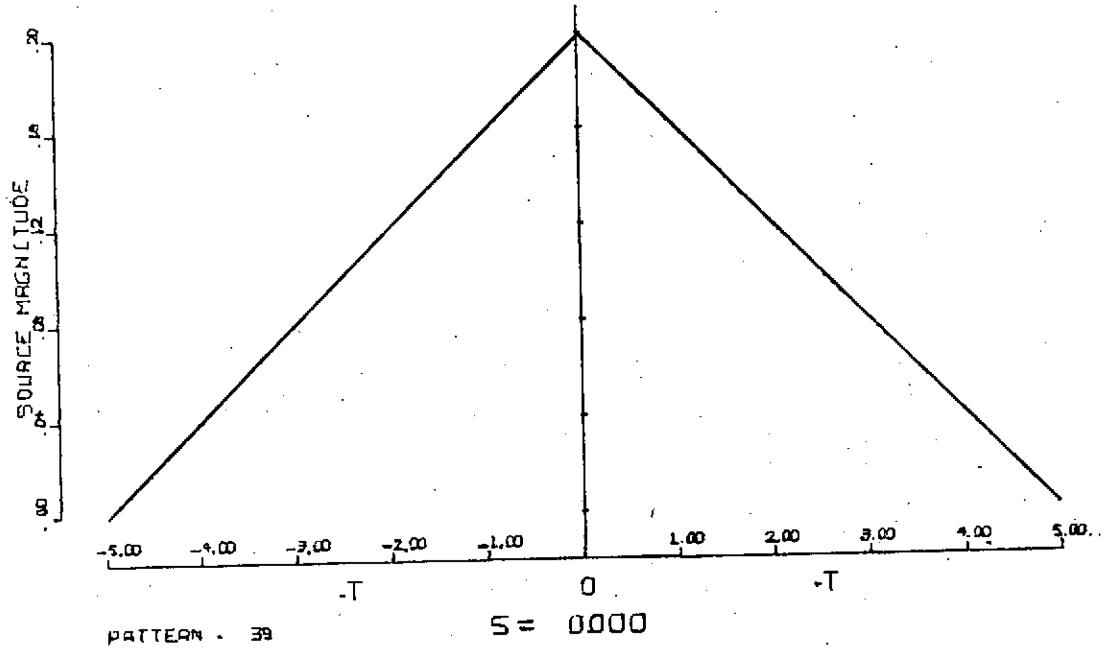


Figure 4.3 Amplitude distribution of a triangular amplitude, ten wavelength line source.

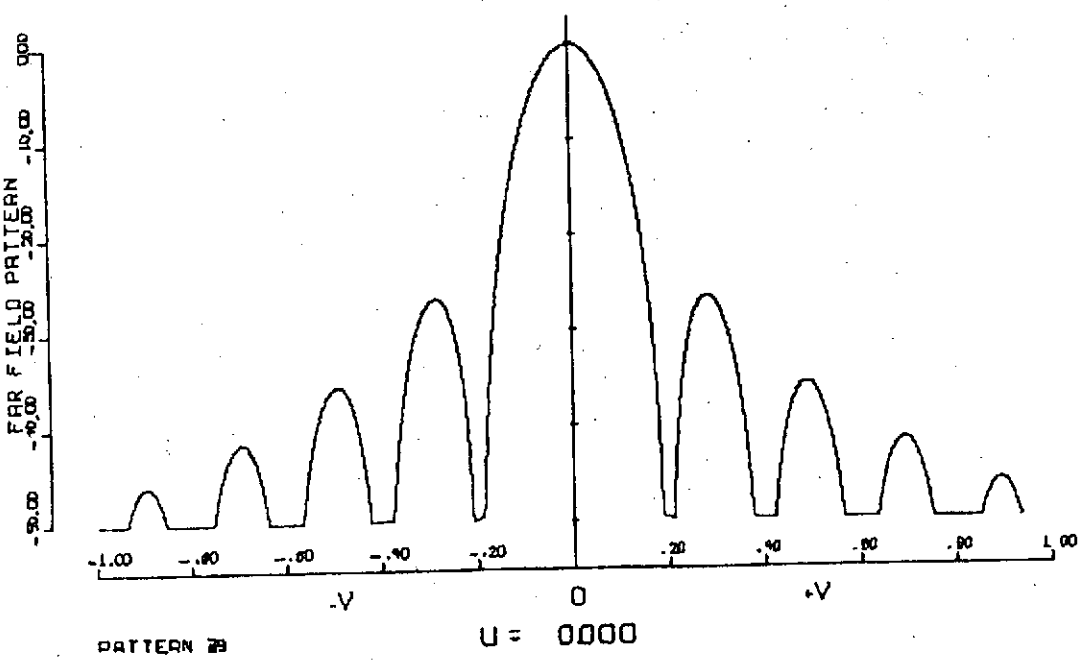


Figure 4.4 Pattern of a triangular amplitude, uniform phase, ten wavelength line source.

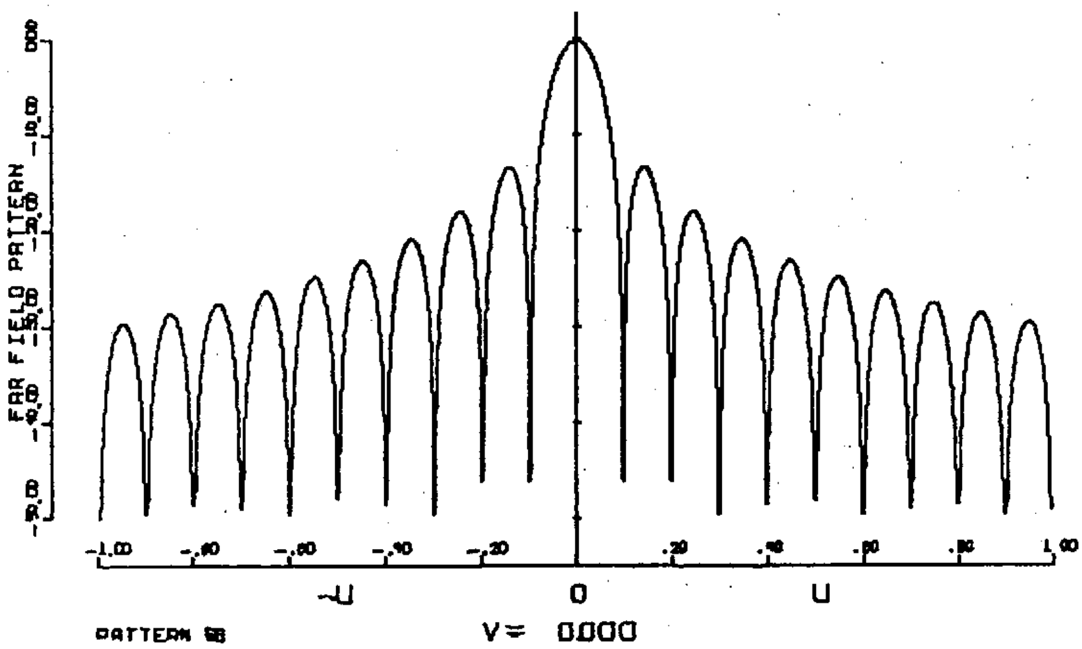


Figure 4.5 Profile along u-axis of the pattern from a uniform amplitude, uniform phase, 10 by 20 wavelength rectangular aperture.

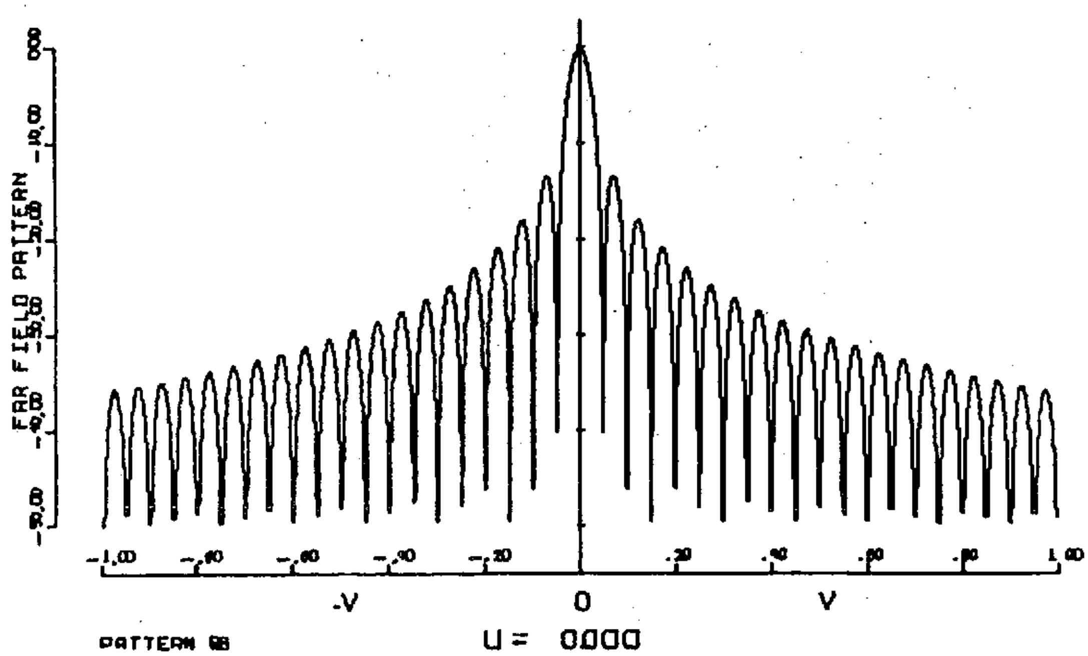


Figure 4.6 Profile along v-axis of the pattern from a uniform amplitude, uniform phase, 10 by 20 wavelength rectangular aperture.

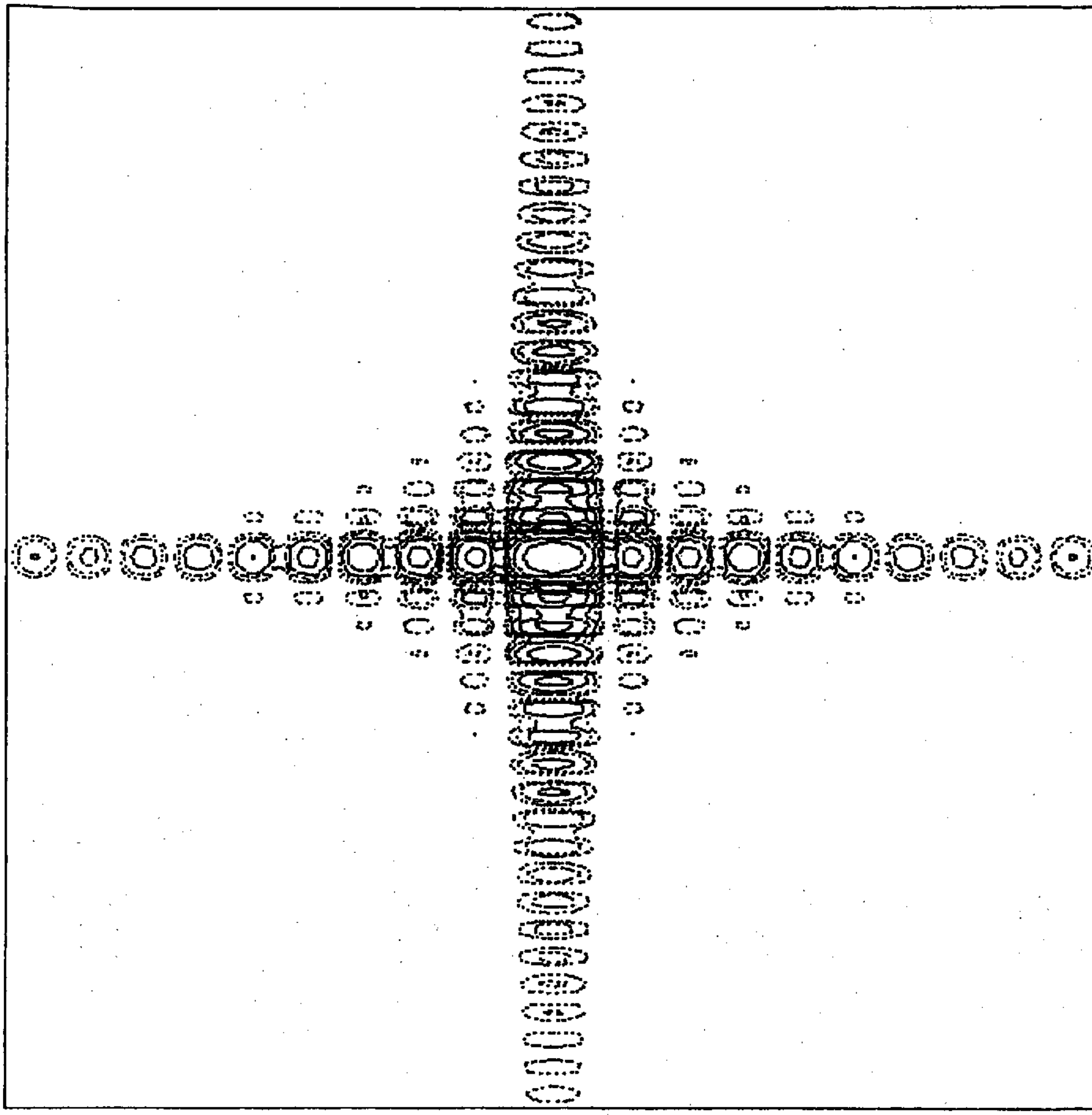


Figure 4.7 Contour map of the pattern from a uniform amplitude, uniform phase, 10 by 20 wavelength rectangular aperture.

PATTERN = 56

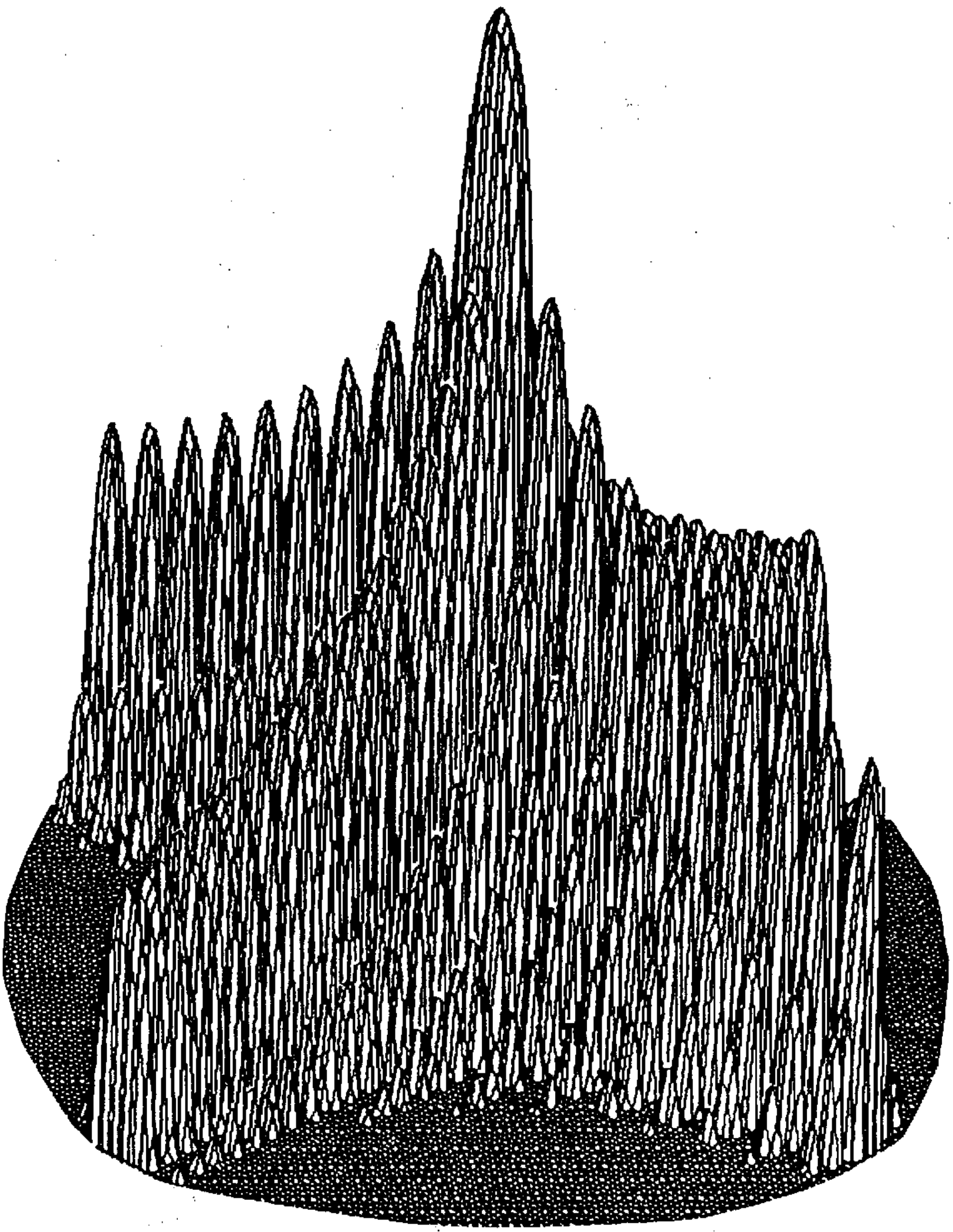


Figure 4.8 Radiation pattern of a uniform amplitude, uniform phase, 10 by 20 wavelength rectangular aperture.

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

v	$F_d(v)$	$F_u(v)$	$F_L(v)$
$\lvert v\rvert \le 0.2$	0. dB	0.5 dB	- 0.5 dB
$0.4 \le \lvert v\rvert \le 1.0$	$-\infty$	- 40.	unspecified

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

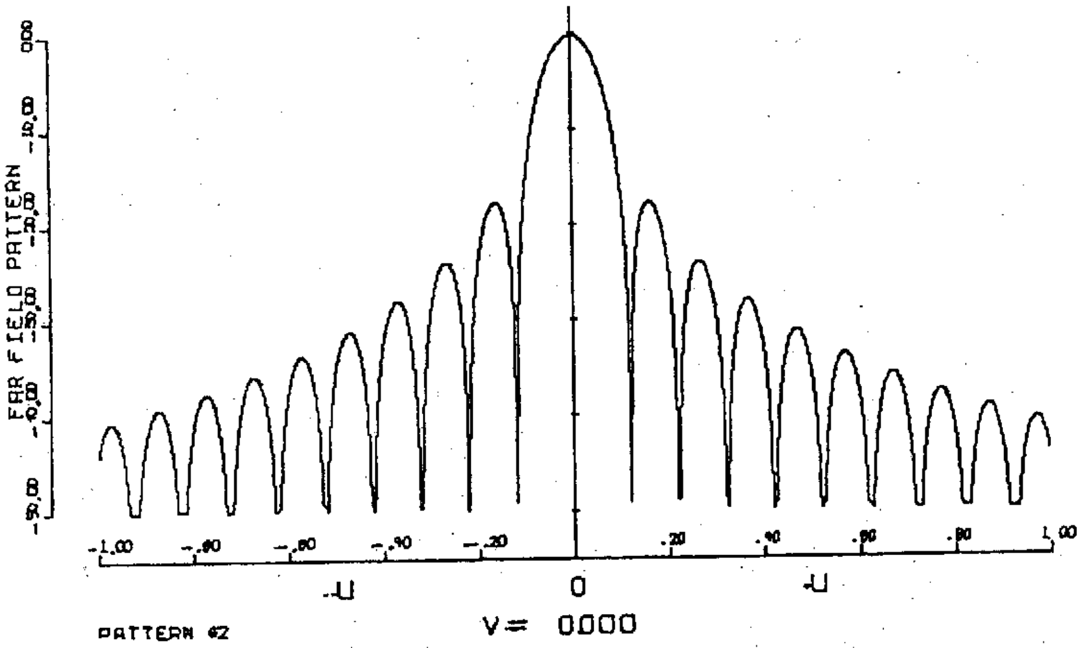


Figure 4.9 Profile along u-axis of the pattern from a uniform amplitude, uniform phase, 10 wavelength diameter circular aperture.

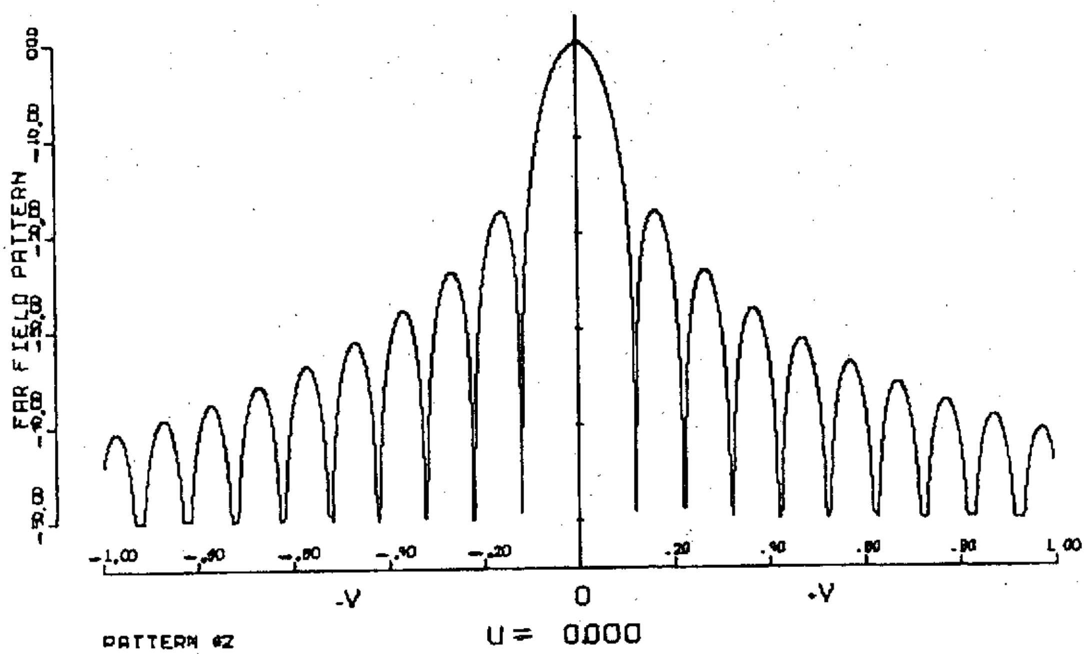


Figure 4.10 Profile along v-axis of the pattern from a uniform amplitude, uniform phase, 10 wavelength diameter circular aperture.

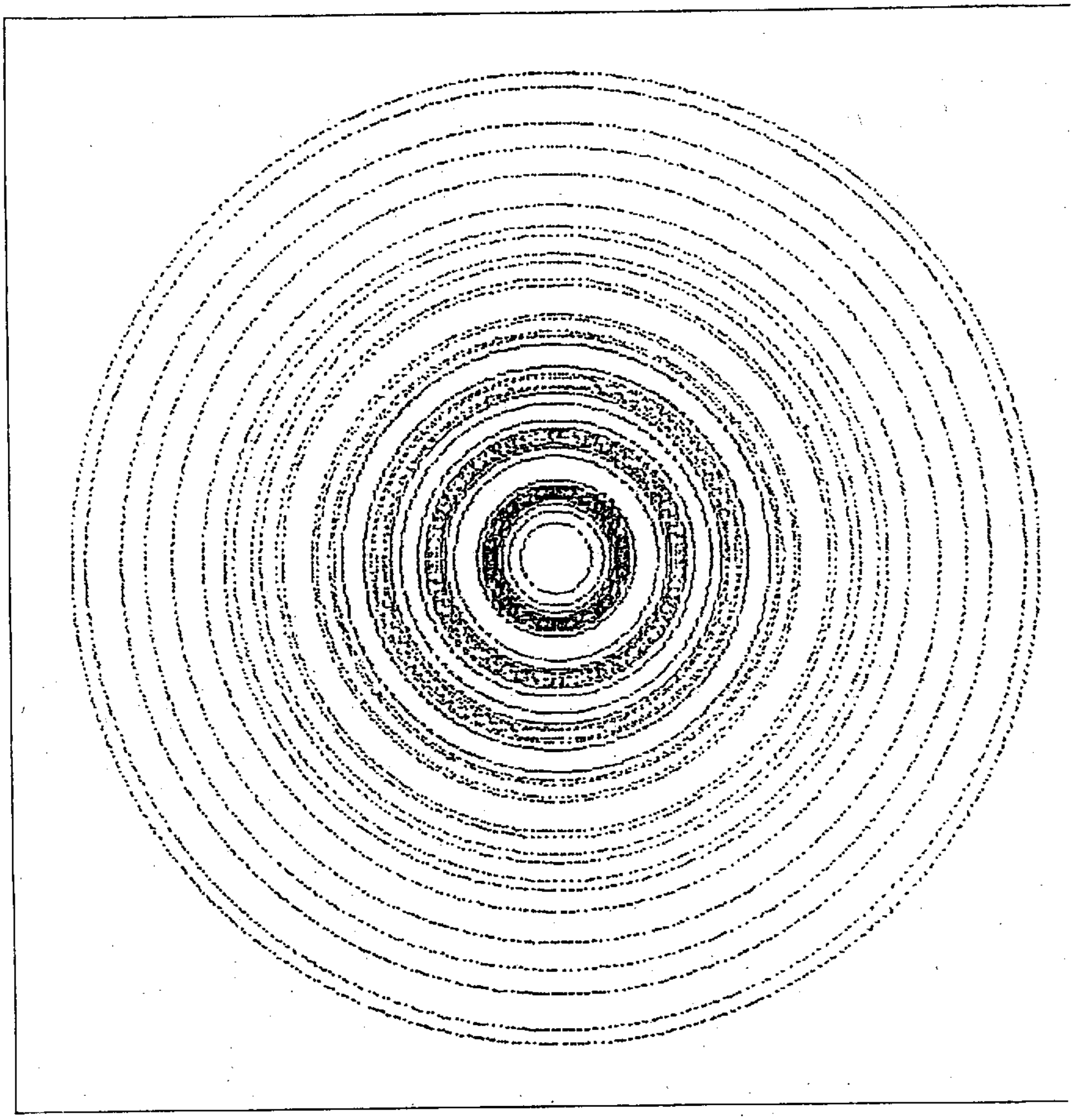


Figure 4.11 Contour map of the pattern from a uniform amplitude, uniform phase 10 wavelength diameter circular aperture.

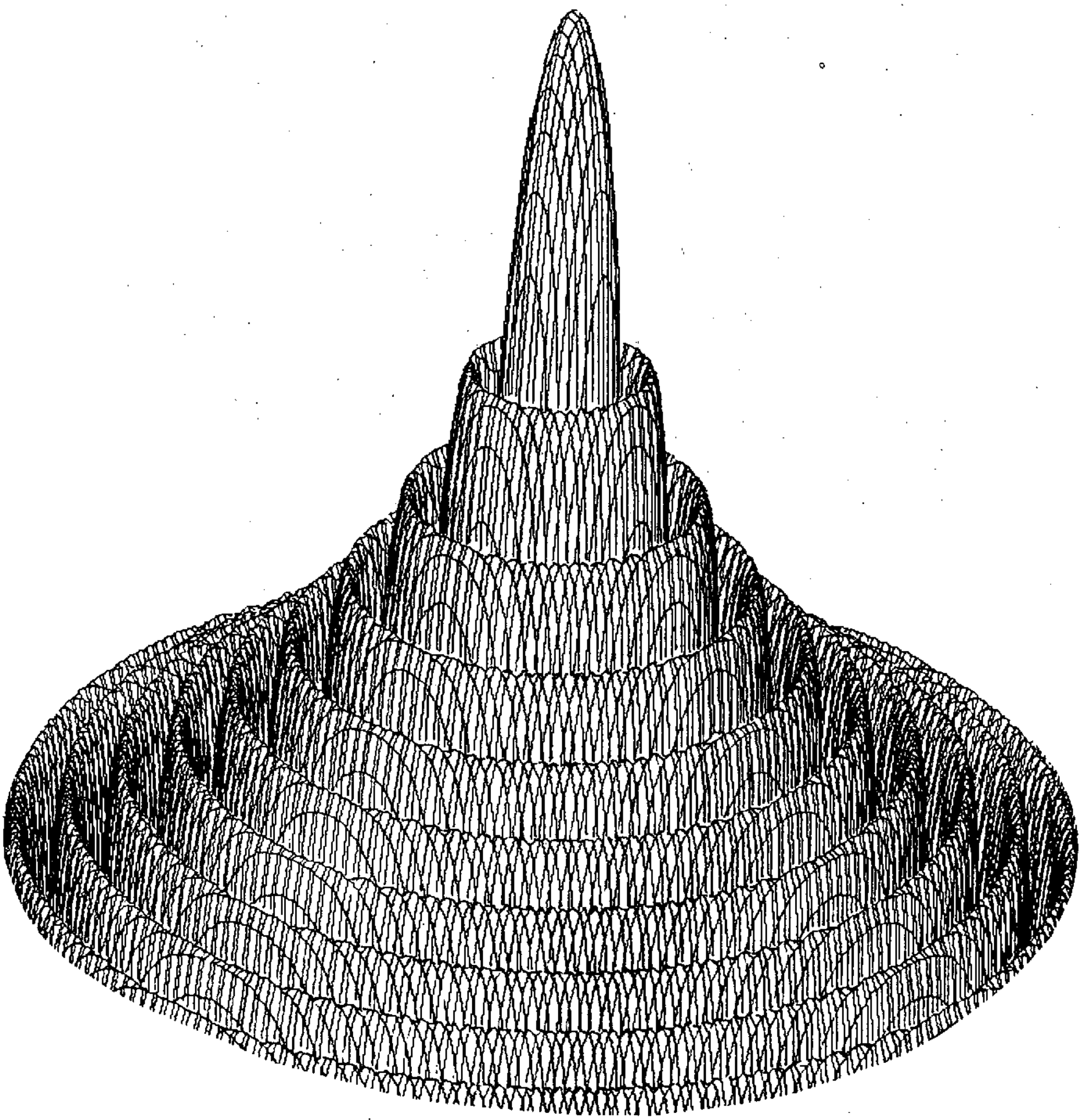


Figure 4.12 Radiation pattern of a uniform amplitude, uniform phase 10 wavelength diameter circular aperture.

PATTERN 42

Table 4.1 Pattern Parameters for Elementary Correction Patterns

Antenna Type - ITYPE	Source Dimensions (λ)	Beam Width		Side Lobe Level (dB)		Directivity (dB)	
		Theory	Computer	Theory	Computer	Theory	Computer
Uniform line source - 1	$L_{y\lambda} = 10$	0.0886	0.0886	-13.3	-13.3	13.1	13.2
Uniform linear array - 2	$P_y = 21$ $d_{x\lambda}=0.5$	0.0886	0.0886	-13.3	-13.2	13.1	13.3
Triangular line source - 3	$L_{y\lambda} = 10$	0.128	0.128	-26.6	-26.6	11.8	12.7
Uniform rectangular aperture - 4	$L_{x\lambda} = 10$ $L_{y\lambda} = 20$	0.0886 0.0443	0.0886 0.0443	-13.3 -13.3	-13.3 -13.3	34.0	34.0
Uniform rectangular array - 5	$P_x = 21$ $d_{x\lambda}=0.5$ $P_y = 41$ $d_{y\lambda}=0.5$					34.0	33.9
Uniform circular aperture - 6	$a_\lambda = 5$	0.102	0.102	-17.6	-17.6	29.95	29.86

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

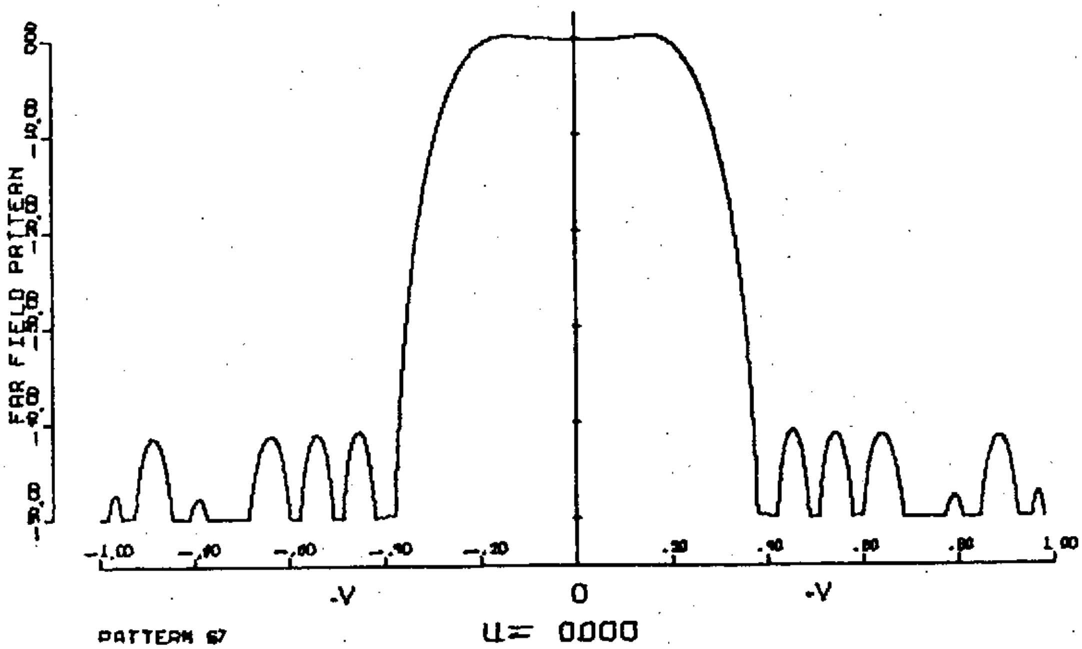


Figure 4.13 Square main beam pattern synthesized using a 10 wavelength line source with uniform amplitude source correction functions.

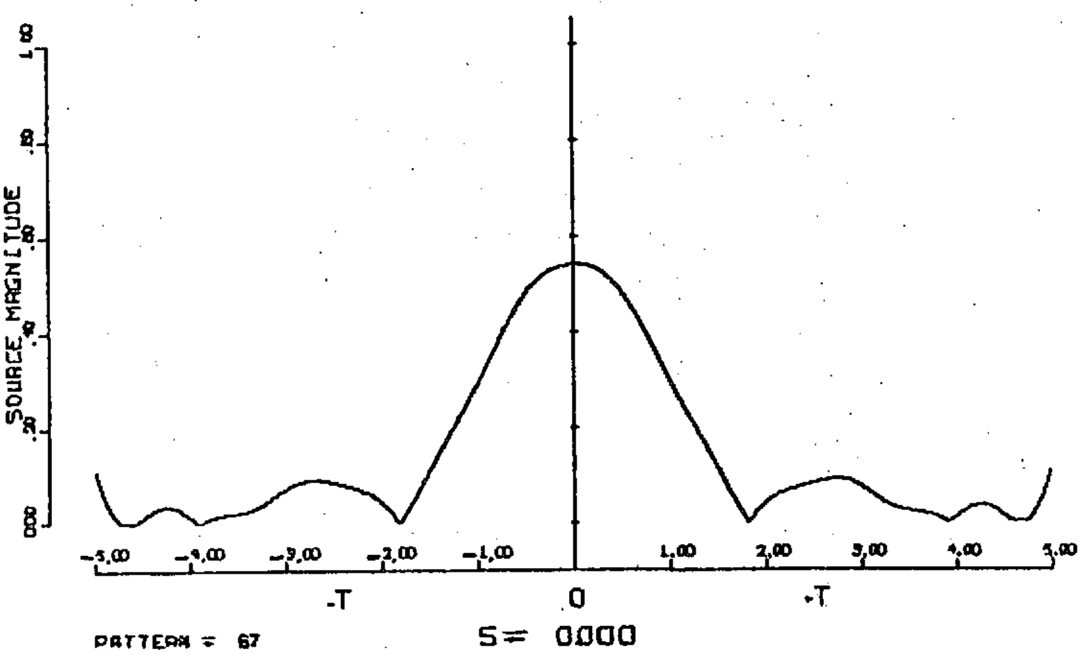


Figure 4.14 Current amplitude distribution required to produce pattern of Fig. 4.13.

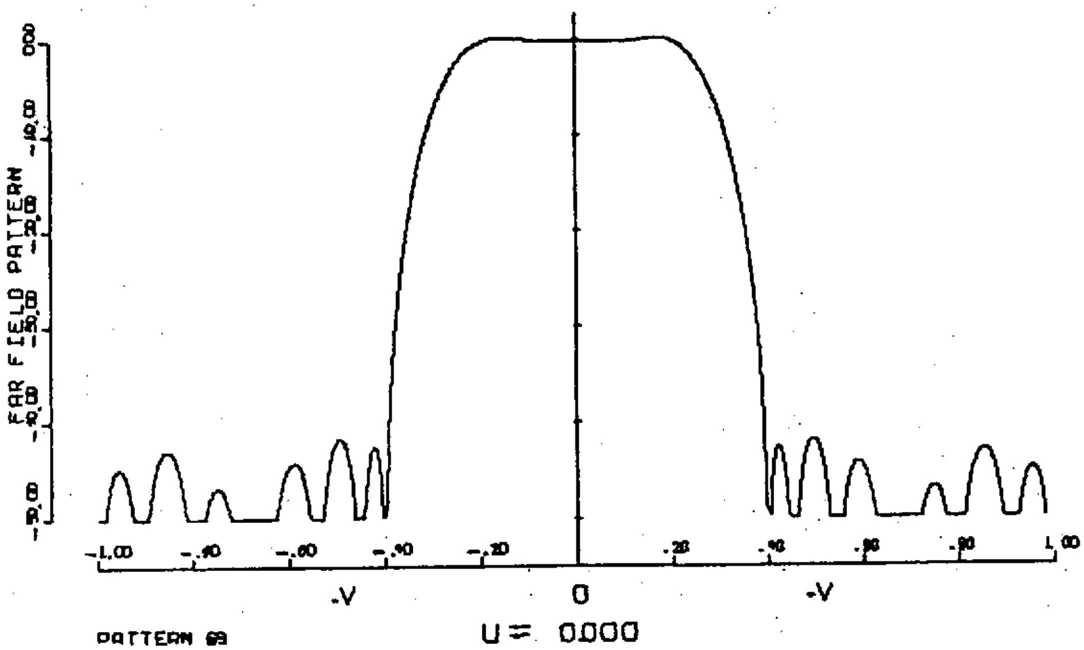


Figure 4.15 Square main beam pattern synthesized using a 10 wavelength line source with triangular amplitude source correction coefficients.

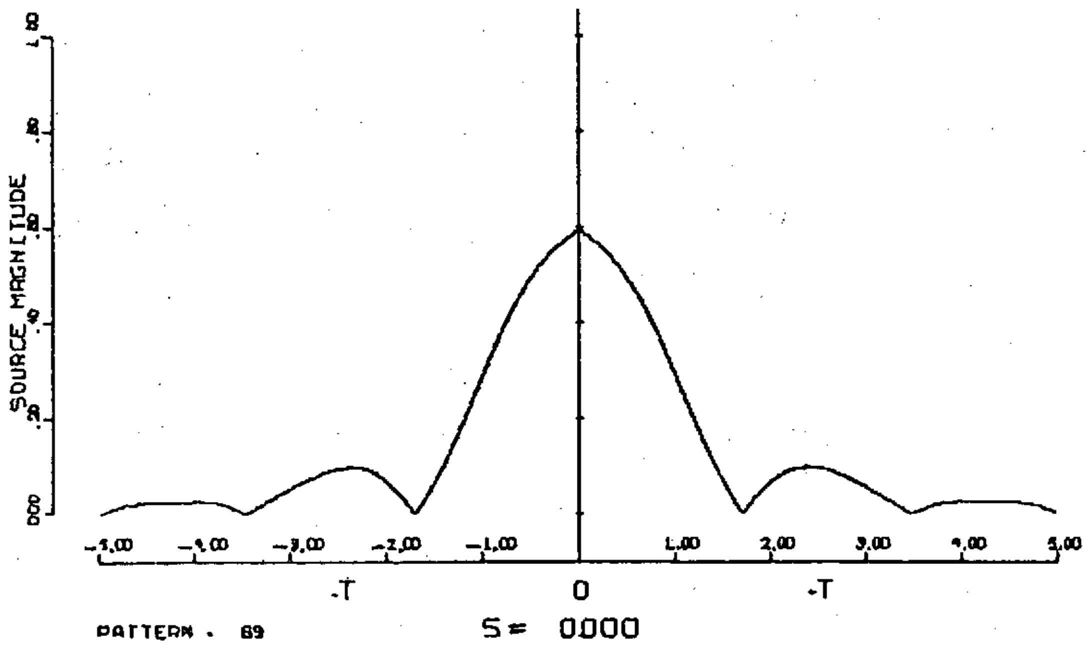


Figure 4.16 Current amplitude distribution required to produce pattern of Fig. 4.15.

v	$F_d(v)$	$F_u(v)$	$F_L(v)$
$\|v\| \leq 0.26$	0. dB	+ .42 dB	- .42 dB
$0.44 \leq \|v\| \leq 1.0$	- 20.	- 18.4	unspecified

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 -25dB in the visible region outside the main beams, i.e. (for example, the beam centered at (0.5, 0.5) was specified for $0.38<u<0.64$ and $0.38<u<0.64$). The other beams were specified in a symmetric fashion. The region outside of these main beam regions had an upper limit of -25dB and no lower limit. The antenna is a 10 by 10 wavelength square aperture. The original pattern was a Woodward-Lawson pattern with a correction coefficient of 1.0 and correction locations at each of the four main beam locations given above. The final pattern was obtained after 21 iterations. Profiles through the centers of the main beams (along u for $v=0.5$ and along v for $u=0.5$) are shown in

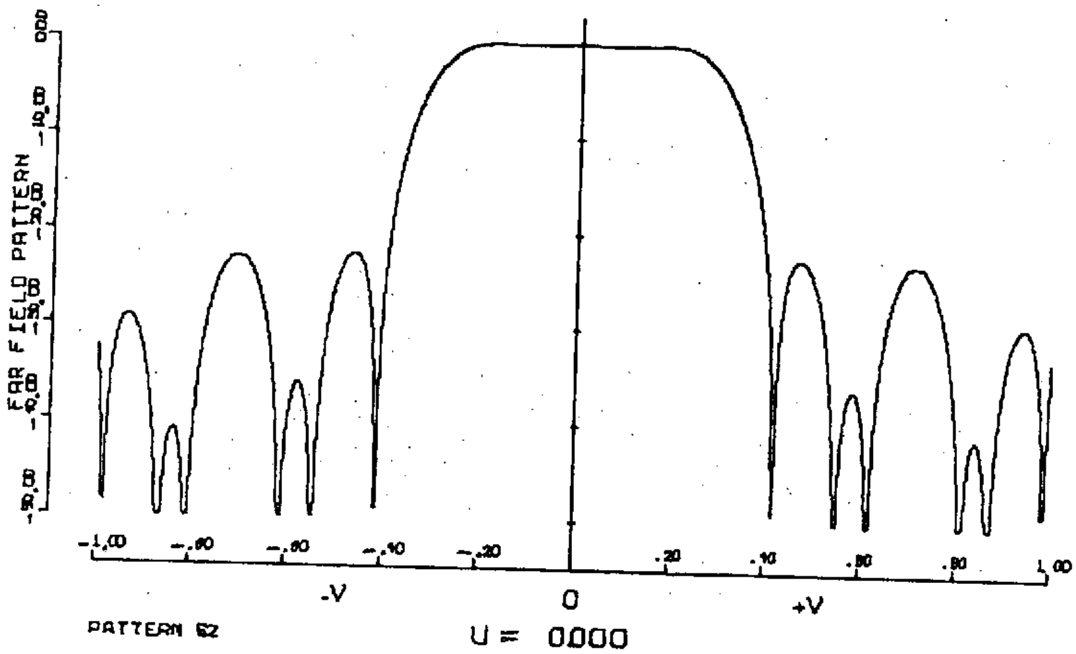


Figure 4.17 Square main beam pattern synthesized using a 13 element, 7 wavelength, nonuniformly spaced linear array.

Figures 4.18 and 4.19. The visible region includes abscissa values between -0.866 and 0.866. Thus, the high side lobes on each end of the profiles are outside the visible region. The contour map of the region $|u|$ and $|v| \leq$ 1.0 is plotted in Figure 4.20. The visible region is a circle inscribed in the square shown. The three dimensional view is given in Figure 4.21.

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

u, v	$F_d(u,v)$	$F_U(u,v)$	$F_L(u,v)$
$\|u\| \leq 0.2, \|u\| \leq 0.05$	0.dB	1.0dB	-0.9dB
$\left.\begin{array}{l} 0.36 \leq \|u\| \leq 0.50 \\ 0.12 \leq \|v\| \leq 0.50 \end{array}\right\}$	-20.0	-18.4	unspecified

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

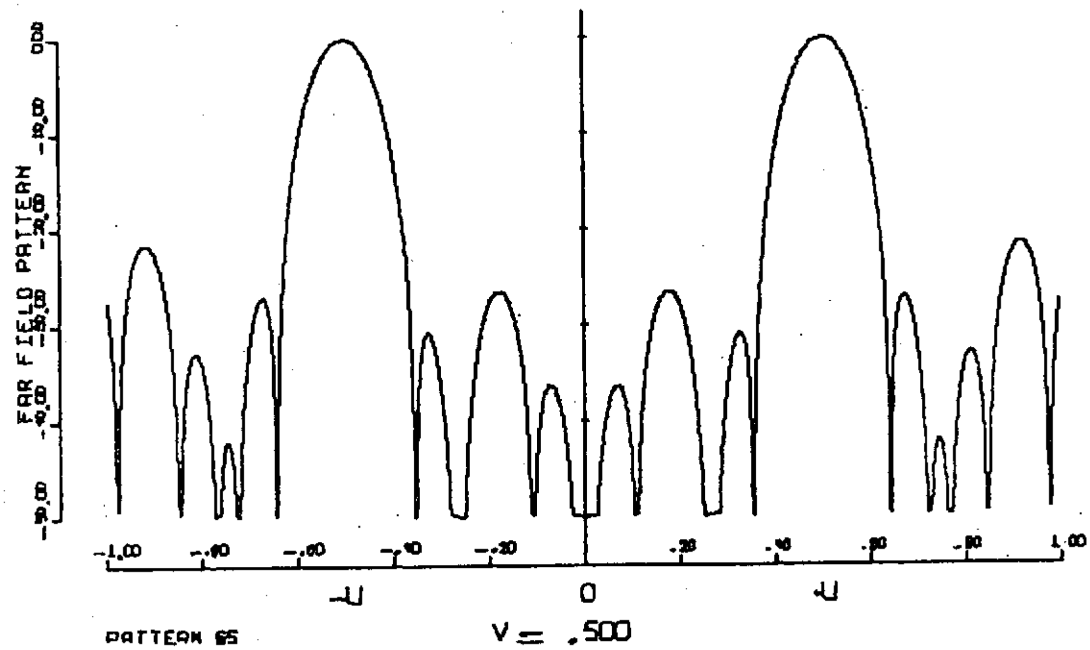


Figure 4.18 A multiple beam radiation pattern profile in the u-direction for v = 0.5 synthesized using a 10 by 10 wavelength aperture antenna. The visible region is for $|u| \leq 0.866$.

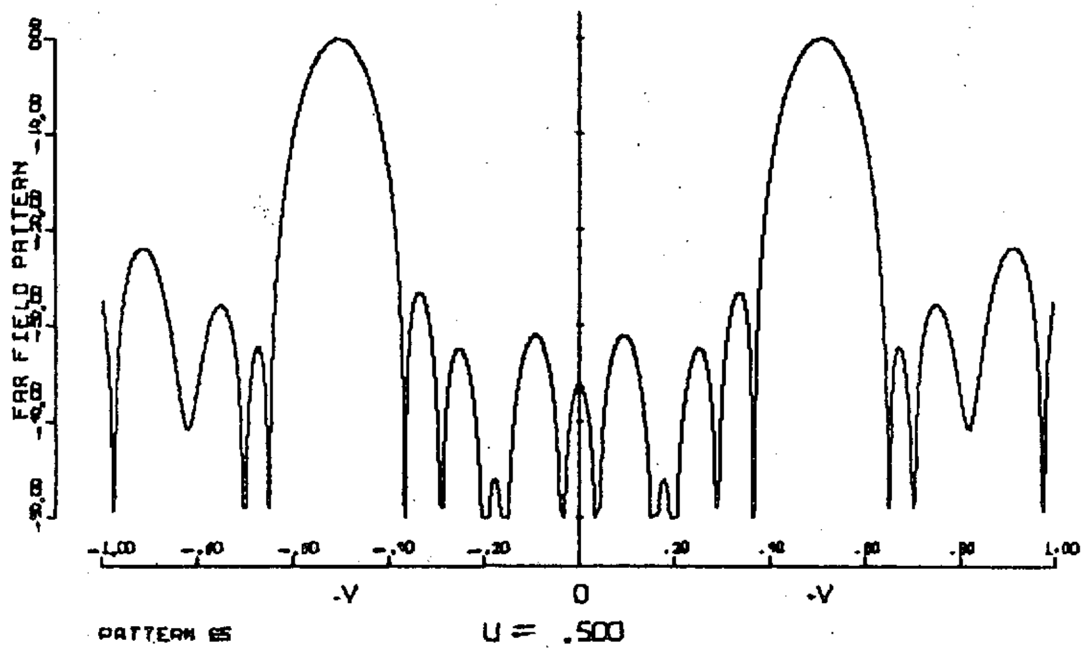


Figure 4.19 A multiple beam radiation pattern profile in the v-direction for u = 0.5 synthesized using a 10 by 10 wavelength aperture. The visible region is for $|v| \leq 0.866$.

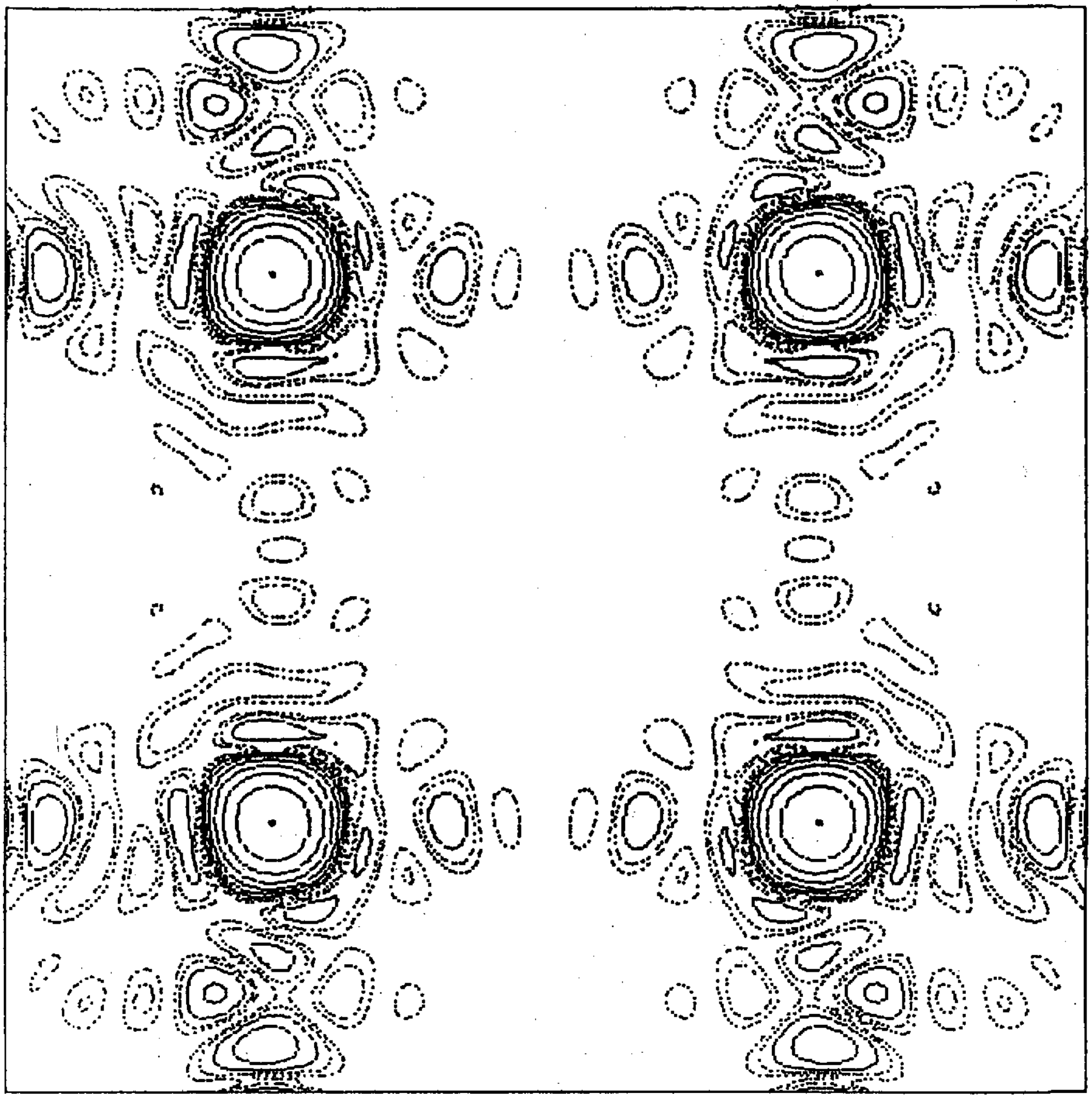


Figure 4.20 Contour map of a multiple beam, low side lobe pattern synthesized from a 10 by 10 wavelength aperture.

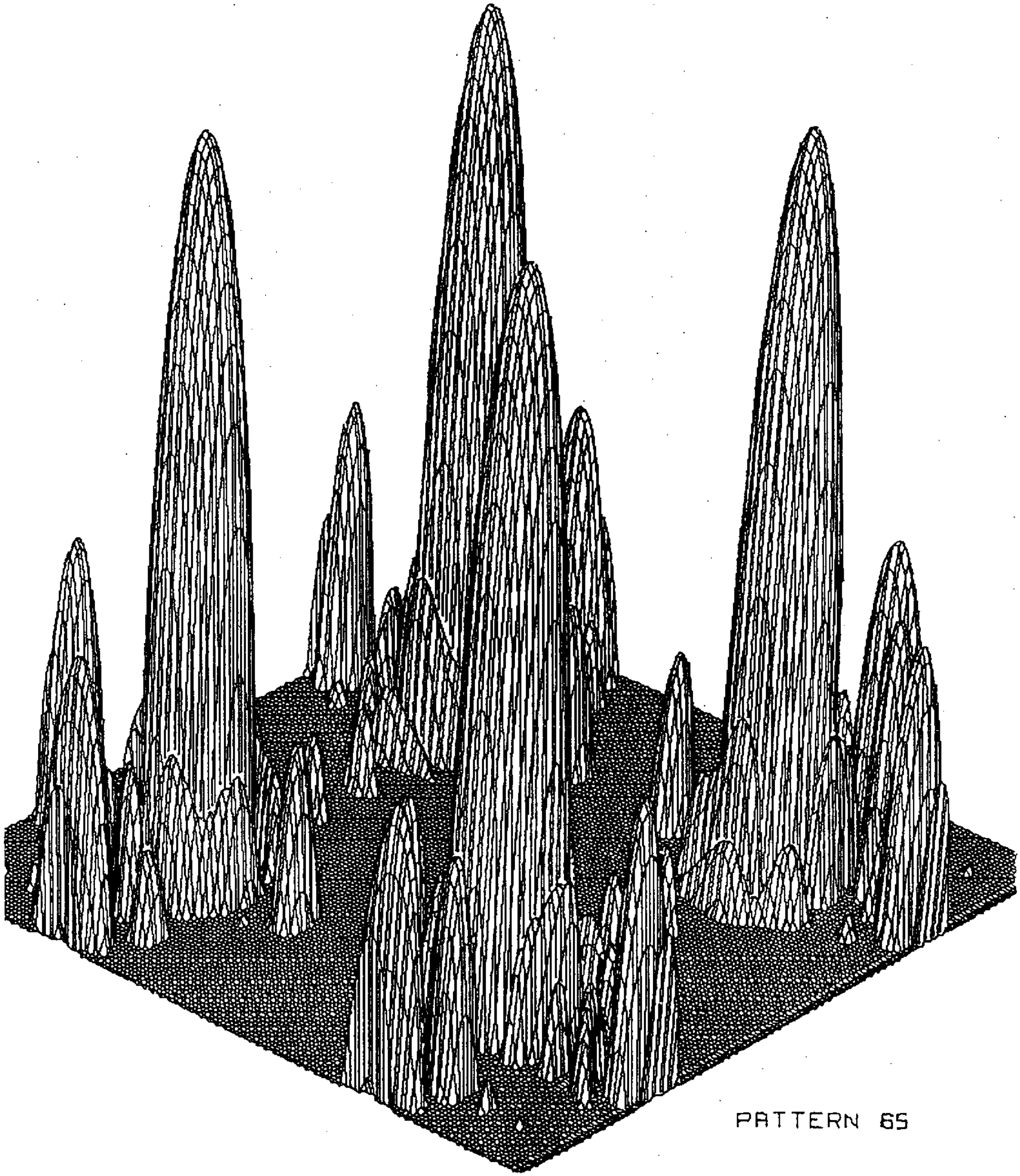


Figure 4.21 Multiple beam, low side lobe pattern synthesized from a 10 by 10 wavelength aperture.

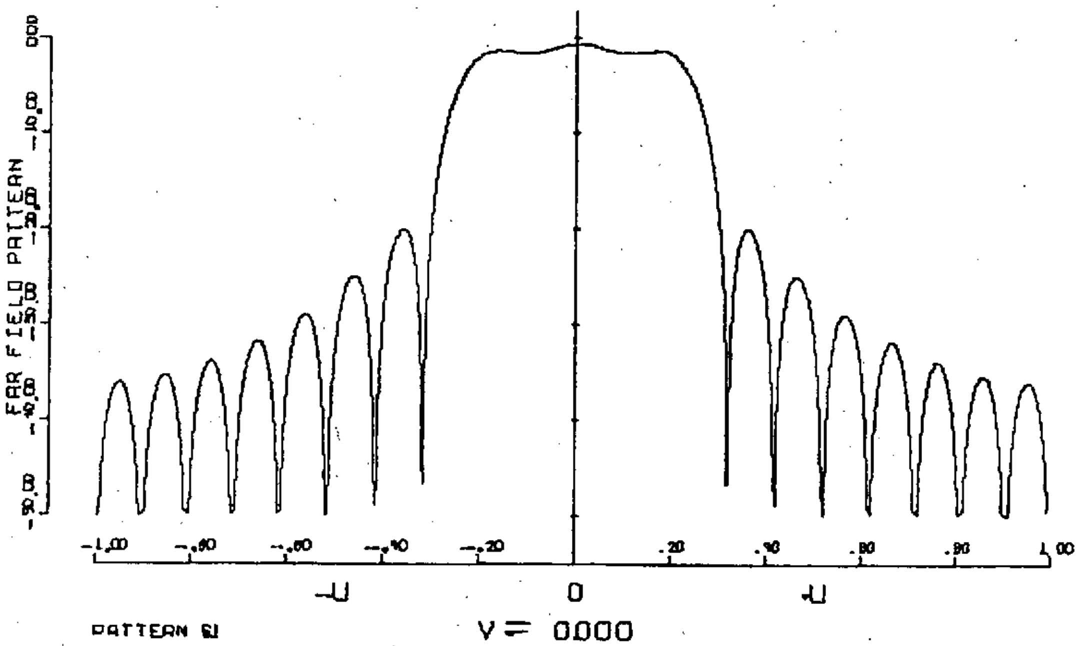


Figure 4.22 Profile along u-axis of a rectangular main beam, low side lobe pattern synthesized from a 20 element, 0.5 wavelength spaced by 40 element, 0.5 wavelength spaced rectangular array.

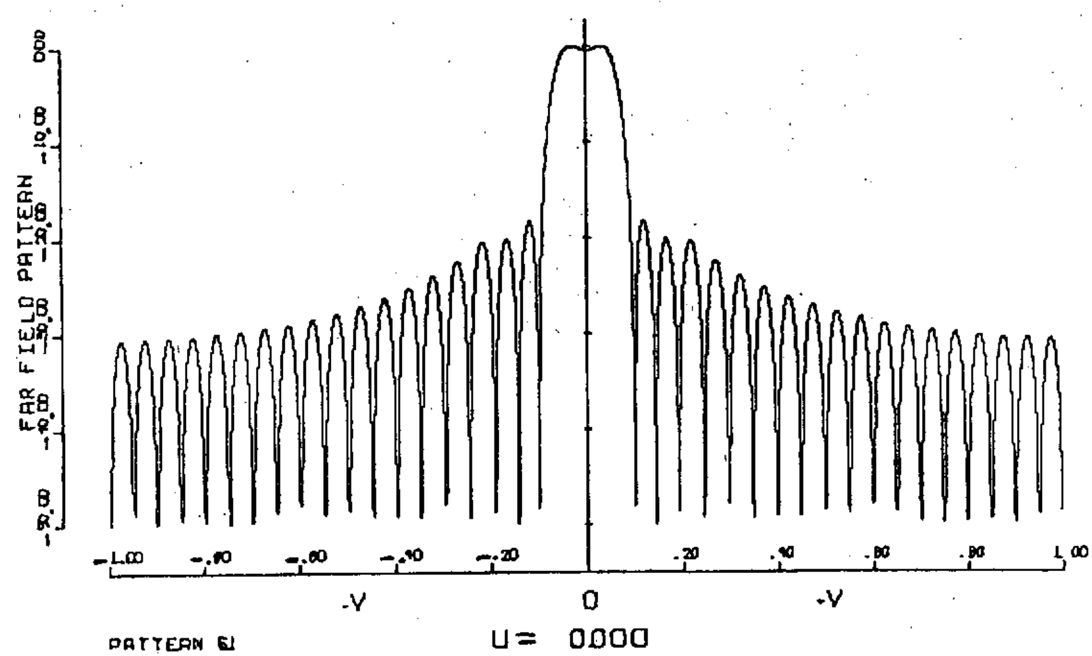


Figure 4.23 Profile along v-axis of a rectangular main beam, low side lobe pattern synthesized from a 20 element, 0.5 wavelength spaced by 40 element, 0.5 wavelength spaced rectangular array.

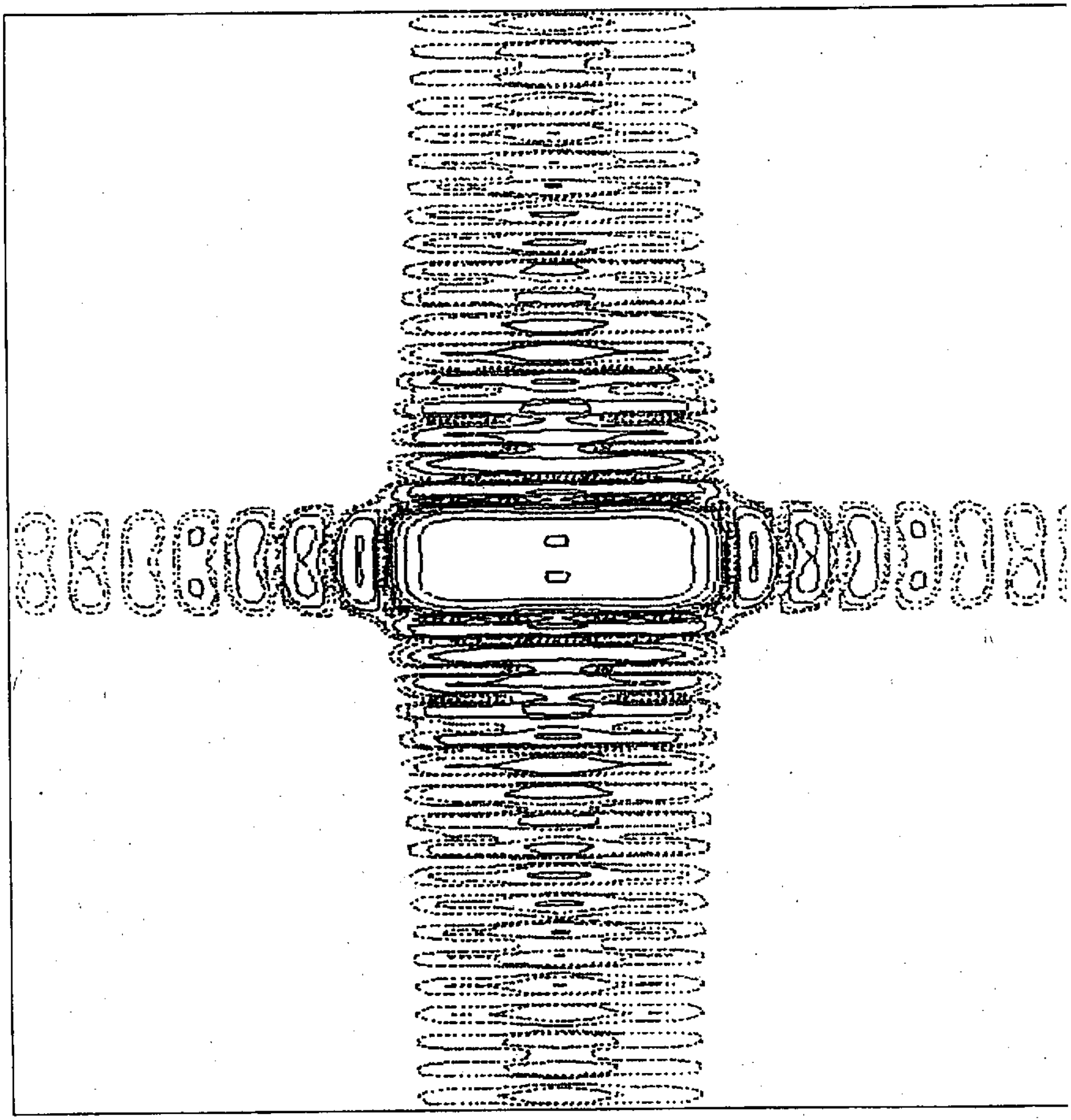


Figure 4.24 Contour map of a rectangular beam, low side lobe pattern synthesized from a 20 element, 0.5 wavelength spaced by 40 element, 0.5 wavelength spaced rectangular array.

PATTERN = 51

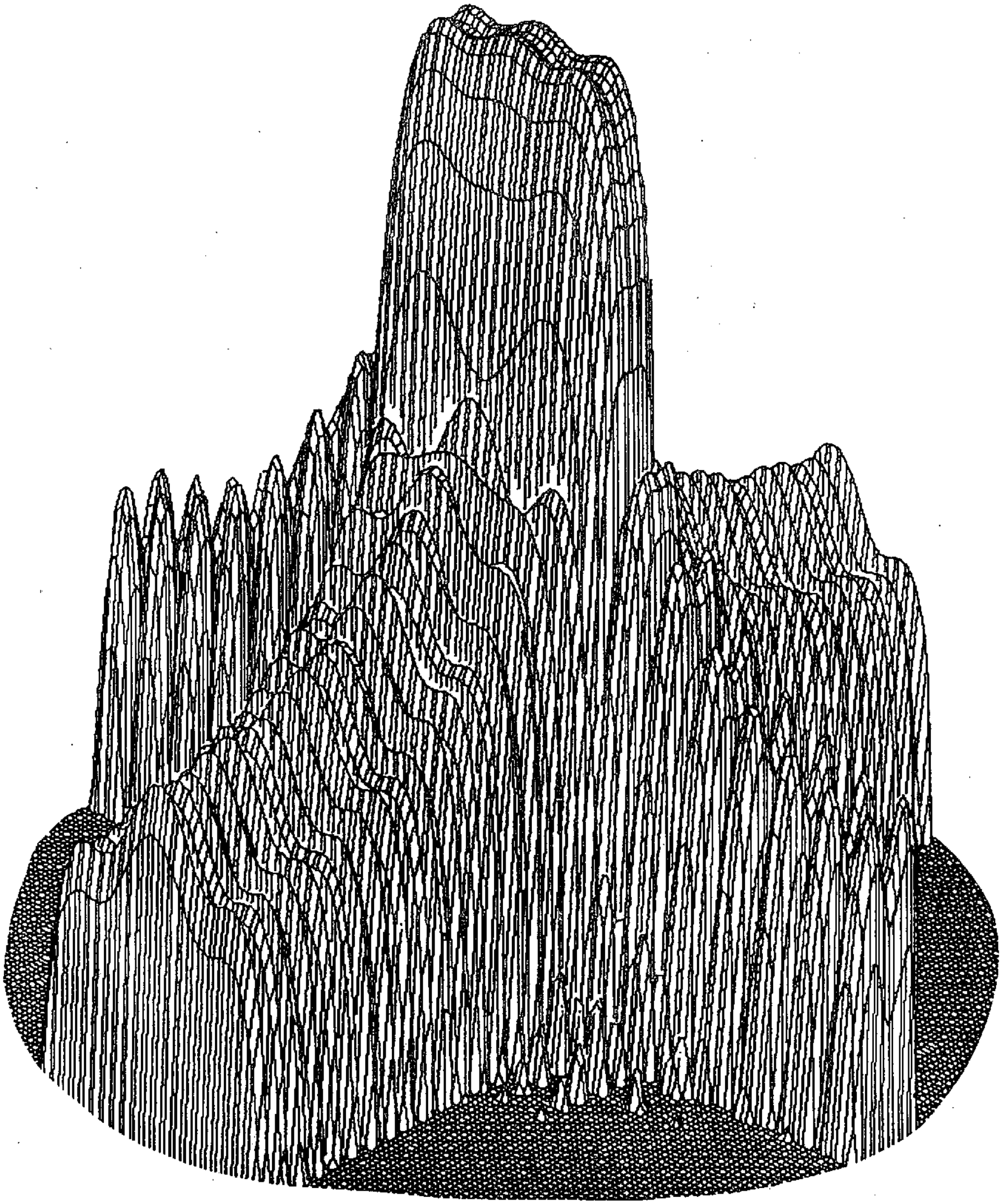


Figure 4.25 Rectangular beam, low side lobe pattern synthesized from a 20 element, 0.5 wavelength spaced by 40 element, 0.5 wavelength spaced rectangular array.

PATTERN 51

Tighter tolerances are easily achieved. This example is a rectangular beam with the following specifications:

(u,v)	$F_d(u,v)$	$F_U(u,v)$	$F_L(u,v)$
$-0.2 \leq u \leq 0.2$ $-0.05 \leq v \leq 0.05$	0.dB	0.5dB	-0.5dB
$0.34 \leq \|u\| \leq 0.50$ $0.12 \leq \|v\| \leq 0.50$	$-\infty$	-25.	unspecified

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 $\pm$0.5dB 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 5.0dB 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.

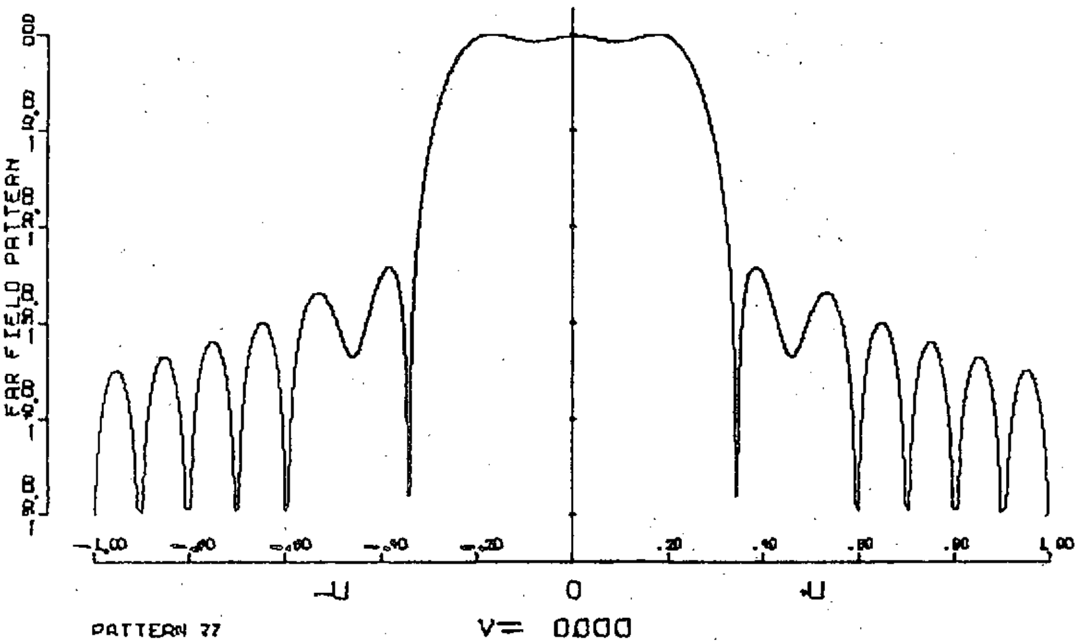


Figure 4.26 Profile along u-axis of a rectangular main beam, low side lobe pattern synthesized from a 10 by 20 wavelength rectangular aperture.

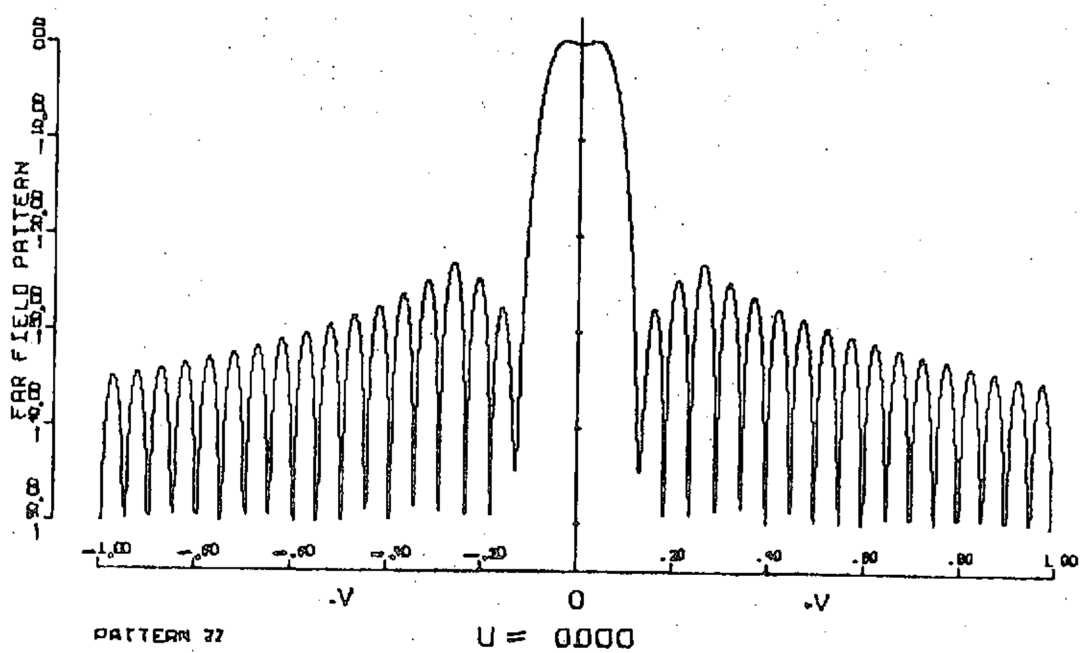


Figure 4.27 Profile along v-axis of a rectangular main beam, low side lobe pattern synthesized from a 10 by 20 wavelength rectangular aperture.

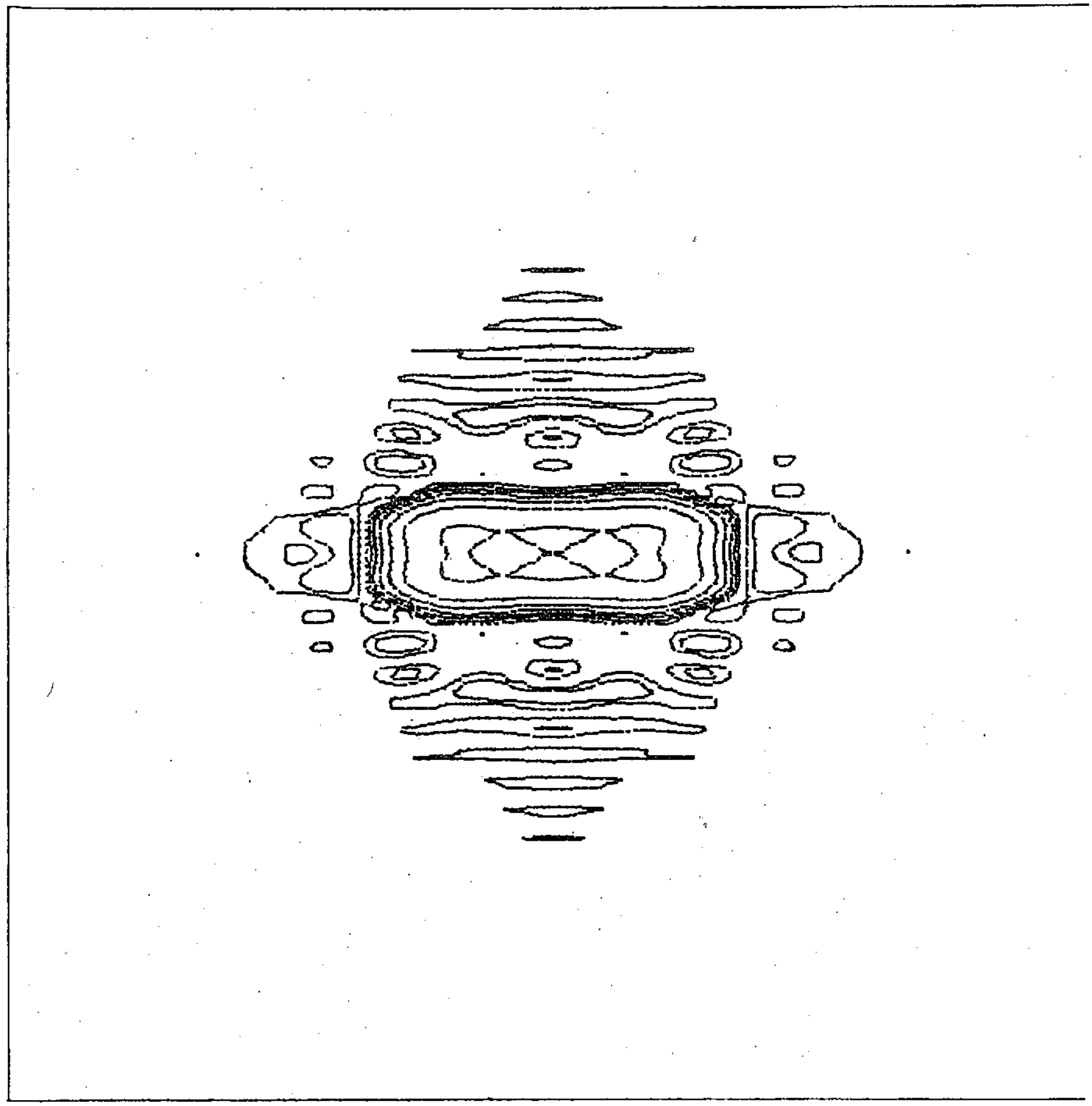
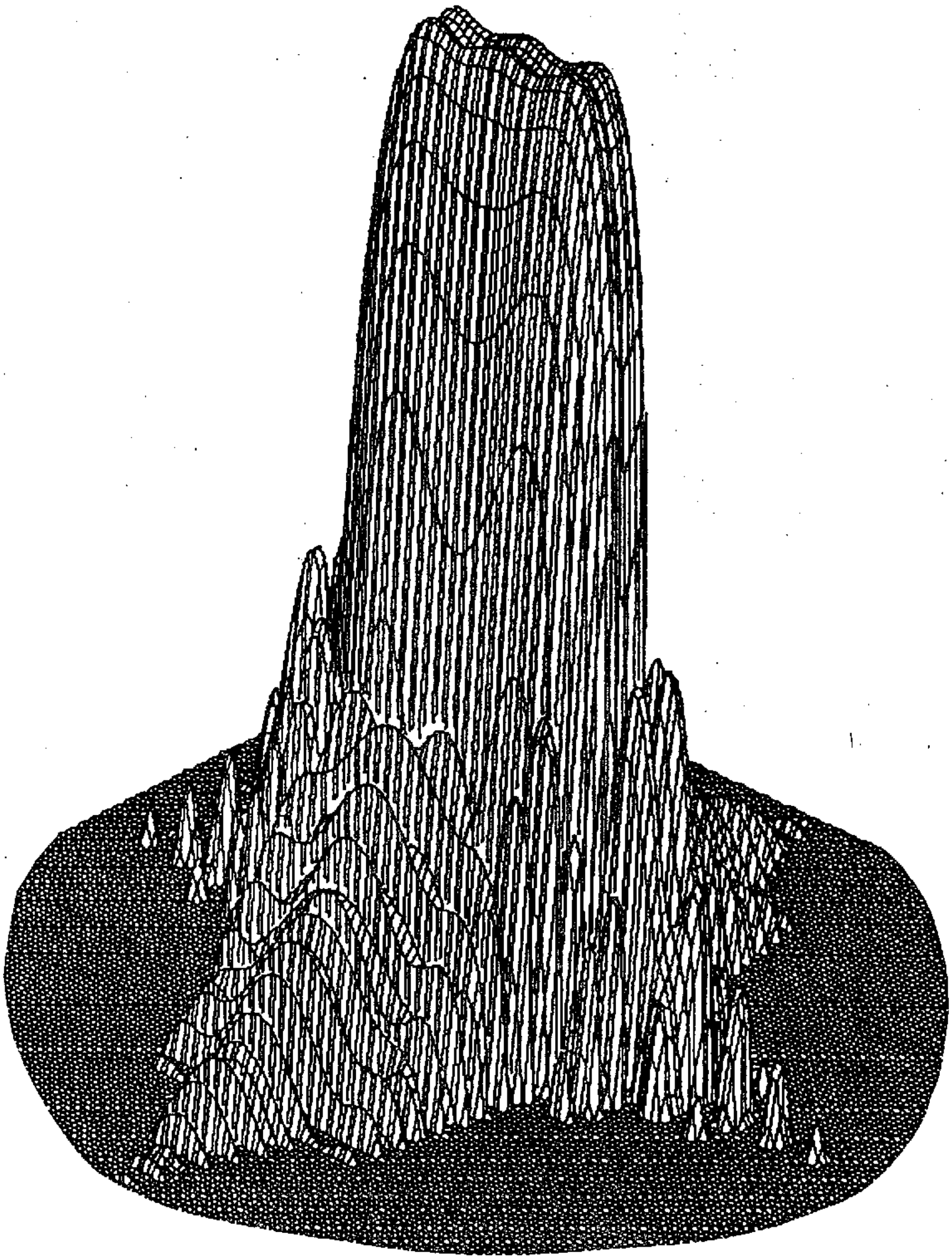


Figure 4.28 Contour map of a rectangular main beam, low side lobe pattern synthesized from a 10 by 20 wavelength rectangular aperture.



PATTERN 77

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 feasability 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 quadralateral 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

1. Virginia Polytechnic Institute and State University Research Proposal ENGR. 72.75, "Synthesis of Multiple Shaped Beam Antenna Patterns," 27 January 1972.

2. R. E. Collin and F. J. Zucker, Antenna Theory, Part I, McGraw-Hill; New York, 1969, Chapter 3.

3. C. H. Walter, Traveling Wave Antennas, McGraw-Hill; New York, 1965, p. 48.

4. E. A. Wolff, Antenna Analysis, John Wiley & Sons, 1967.

5. W. L. Stutzman, "Synthesis of Shaped-Beam Radiation Patterns," Ph.D. Dissertation, The Ohio State University, 1969.

6. W. L. Stutzman, "Synthesis of Shaped-Beam Radiation Patterns Using the Iterative Sampling Method," IEEE Trans. on Antennas and Propagation, Vol. AP-19, pp. 36-41, January, 1971.

7. W. L. Stutzman, "Side Lobe Control of Pencil-Beam Antenna Patterns," 1970 IEEE G-AP International Symposium Digest, pp. 446-454, September, 1970.

8. W. L. Stutzman, "Synthesis of Pencil-Beam Antenna Patterns With Side Lobe Control," Virginia Polytechnic Institute and State University, College of Engineering, Technical Report VPI-E-71-1, January, 1971.

9. W. L. Stutzman, "Side Lobe Control of Antenna Patterns," IEEE Trans. on Antennas and Propagation, Vol. AP-20, pp. 102-104, January, 1972.

10. E. L. Coffey, "Synthesis of Antenna Radiation Patterns Using Rectangular Sources," M.S. Thesis, Vir. Poly. Ins't. and S. U., June 1973.

11. E. D. Sharp, "A Triangular Arrangement of Planar-Array Elements that Reduces the Number Needed," IRE Trans. on Ant. and Prop., Vol. AP-9, pp. 126-129, March 1961.

12. C. H. Walter, Traveling Wave Antennas, McGraw-Hill; New York, 1965, Chapter 3.

13. R. C. Hansen, ed., Microwave Scanning Antennas, Vol. I, Academic Press; New York, 1964, Chapter 1.

14. W. L. Stutzman, "Shaped-Beam Synthesis of Nonumiformly Spaced Linear Arrays," IEEE Trans. on Ant. and Prop., Vol Ap-20, pp. 499-501, July, 1972.

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.

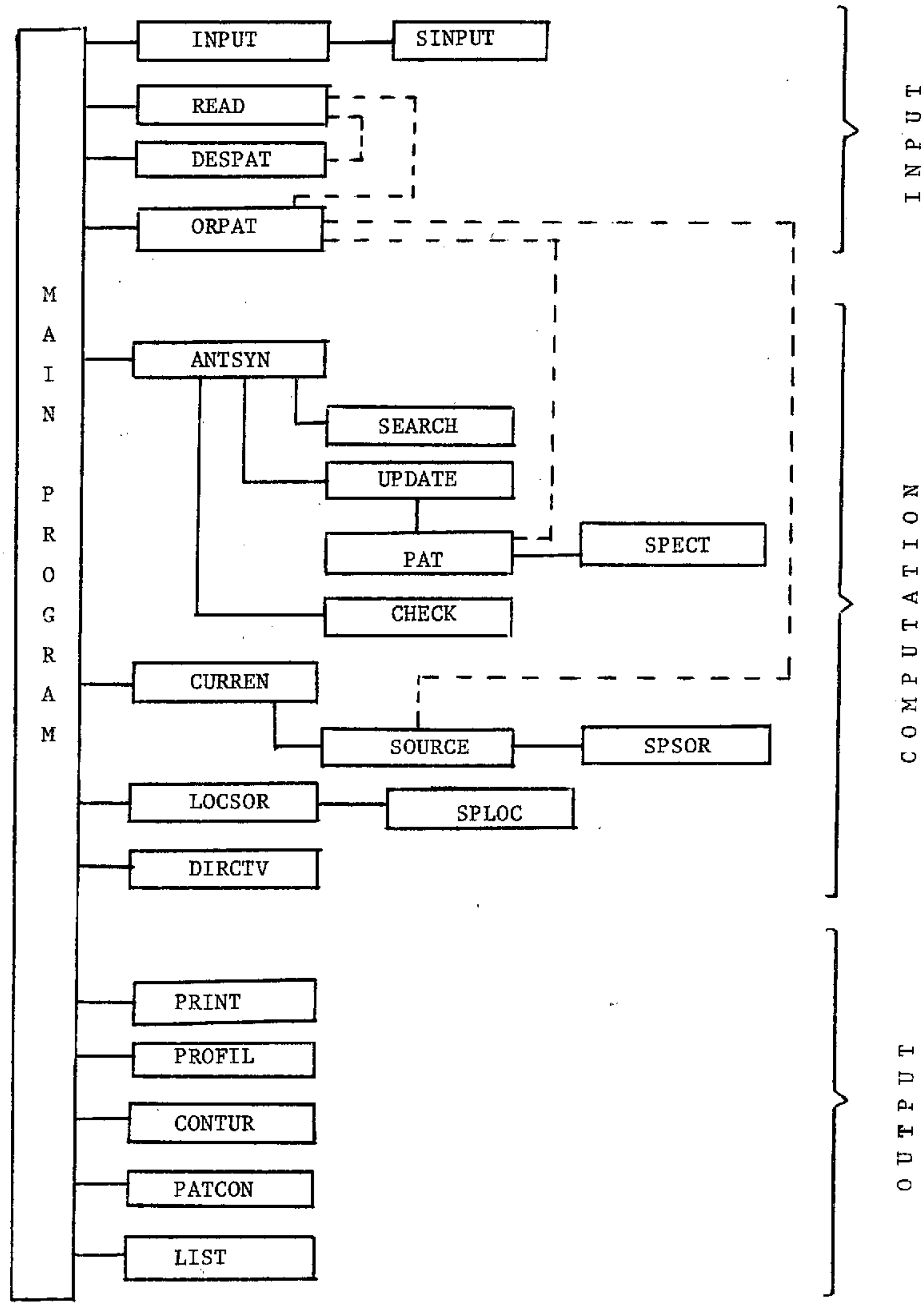


Figure 6.1 Block diagram of ANTSYN program

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.

		Device	Location	Availability
1.	Job assignment	Auxiliary storage	MAIN	Standard
2.	Pattern parameters	Cards; use Namelist PARAM	MAIN	Standard
3.	Output print control	Cards; use Namelist IPRINT	MAIN	Standard
4.	Antenna parameters	Cards; use Namelist PATIN	INPUT	Standard
		If ITYPE >7 also use subroutine SINPUT	SINPUT	Special
5.	Desired pattern	Program statements to load FDES, FU and FL	DESPAT	Special
		or		
		Cards; call subroutine READ to load FDES, FU, and FL	DESPAT	Special
6.	Original pattern and original source	Cards; read NORG, US, VS, CORG to generate original state using Woodward-Lawson method	ORGPAT	Standard
		or		
		Cards or program statements; if Woodward-Lawson is not used, load F, CURR, and CURI, set NORG=0	ORGPAT	Special
7.	Special correction pattern function	Program statements to generate values of PAT, ITYPE >7	SPECPT	Special
8.	Special correction source function	Program statements to generate companion to special pattern, ITYPE >7	SPSOR	Special
9.	Special location	Program statements to generate source coordinates given source array subscripts, use when ITYPE >7	SPLOC	Special
10.	Job storage	Programming to write job data onto storage unit	MAIN	Standard

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, NELMT, ARAD, ITYPE, MCUR, NCUR

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

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

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

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

Step 9. If correction pattern and source functions other than the seven standard ones available are desired by the user, Subroutine SPLOC is to be written. Use ITYPE >7.

Step 10. At the completion of a program data may be stored for future use, such as with the ANTDATA program. After looking at ANTSYN print out the user can decide if further display is desired. We can then use the data stored to plot patterns, currents, etc. IDISK is used to control whether data is to be stored.

6.4 Program Variables

6.4.1 Correspondence Between Symbols Used in the Theory and Program Variables

Symbol	Computer Program Counterpart
$a_n^{(i)}$	CORCOF( )
a_λ	ARAD
$d_{x\lambda}$	DISX
$d_{y\lambda}$	DISY
$f^{(K)}(s,t)$	CURR(M,N) CURI(M,N)
$F^{(i)}(u,v)$	F (M,N)
$F_d(u,v)$	FDES (M,N)
$g_n^{(i)}(s,t)$	SOURCE (J,K,US(L),US(L),ITYPE)
$G(u-u_n^{(i)},v-v_n^{(i)}$	PAT (U-US(L), V-VS(L), ITYPE)
$L_{x\lambda}$	LX
$L_{y\lambda}$	LY
P_x	PX
P_y	PY
s	S
t	T
u	U
$u_n^{(i)}$	US( )
v	V
$v_n^{(i)}$	VS ( )

6.4.2. Definition of Some Integer Variables Used in the Program

DIRECT	Input variable controlling calculation and print out of directivities DIRORG and DIRFNL; 0 No, 1 Yes - Default is 0. Original pattern is to be of Woodward-Lawson type.
FCURCN	Input variable controlling print out of contour map of final current distribution; 0 No, 1 Yes - Default is 0.
FCURPR	Input variable controlling print out of final current distribution profile or list; 0 None, 1 Profile (S and/or T axis) for continuous sources, 1 Table of element currents for arrays, 2 List (primarily for use with ITYPE = 7) - Default is 0.
FCURPT	Input variable controlling print out of a listing of the final current distribution; 0 No, 1 Yes - Default is 0.
FDBCN	Input variable controlling print out of contour map of final pattern in dB; 0 No, 1 Yes - Default is 0.
FDBPR	Input variable controlling print out of final pattern profile; 0 No, 1 Yes - Default is 0.
FDBPT	Input variable controlling print out of final pattern in dB; 0 No, 1 Yes - Default is 0.
FDESPR	Input variable controlling print out of desired pattern profile table; 0 No, 1 Yes (U and/or V axis) - Default is 0.
FDESPT	Input variable controlling print out of a listing of desired pattern; 0 No, 1 Yes - Default is 0.
IC	Subscript of CORCOF( ) array for latest correction.
ICURCN	Input variable controlling print out of contour map of initial current distribution; 0 No, 1 Yes - Default is 0.
ICURPR	Input variable controlling print out of initial current distribution profile or list; 0 None, 1 Profile (U and/or V axis), 2 List (primarily for use with ITYPE = 7) - Default is 0.
ICURPT	Input variable controlling print out of a listing of initial current distribution; 0 No, 1 Yes - Default is 0.
IDISK	Input variable controlling output of data to disk storate; 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 specificiations have not been met, 1 If they have.

ISYMM — Input variable descirbing 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.

ITYPE	Antenna Type	Source Illumination Used to Form Correction Pattern
1	Line Source	Uniform
2	Equally Spaced Linear Array	Uniform
3	Line Source	Triangular
4	Rectangular Aperture	Uniform
5	Rectangular Array	Uniform
6	Circular Aperture	Uniform
7	General Array	Uniform
GT7	SPECPT	SPSOR

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.

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

STARTU	Input variable - Starting point in U direction for pattern comparisons and print outs.
STARTV	Input variable - Starting point in V direction for pattern comparisons and print outs.
T	Source coordinate Y normalized to a wavelength.
TT( )	Antenna array element position in T direction - Input variable for ITYPE - 7.
U	Pattern coordinate.
UORG( )	Input variable - Positions of sample points for original pattern in U direction.
US( )	Positions of corrections (samples) in U direction.
V	Pattern coordinate.
VORG( )	Input variable - Positions of sample points for original pattern in V direction.
VS( )	Positions of corrections (samples) in V direction.

6.5 Subroutine Descriptions

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

SUBROUTINE INPUT

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

ITYPE	Variables to be provided as input for PATIN Namelist
1	LY, INITLT, DELTAT, FINALT, ITYPE
2	LY, PY, DISY, ITYPE
3	LY, INITLT, DELTAT, FINALT, ITYPE
4	LX, LY, INITLS, DELTAS, FINALS, INITLT, DELTAT, FINALT, ITYPE
5	LX, LY, PX, PY, DISX, DISY, ITYPE
6	INITLS, DELTAS, FINALS, INITLT, DELTAT, FINALT, ARAD, ITYPE
7	ITYPE, MCUR, NCUR
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 5^{th} row were to be 0.0, the card corresponding to the 5^{th} row would have 51 in columns 2 and 3 and 0 in column 13.

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

The purpose of this subroutine is to return arrays FDES, FU, and FL. They are all two dimensional and are loaded with MMAX rows and NMAX columns.

There are several ways that this subroutine can be written to load these arrays. Subroutine READ can be called for each of the arrays, if card input is convenient. If the patterns can be generated from FORTRAN expressions easily, the arrays can be loaded in the subroutine by incrementing thru U and V and assigning values to the arrays. This approach often avoids the need for a large input card deck.

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

<u>SUBROUTINE ORGPAT</u> (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 0^{th} 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 analylitic functions. In ORGPAT, NORG should be set to zero when not using Woodward-Lawson method to generate original state.

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

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

SUBROUTINE SEARCH (I1, 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 I1 and J1 of the pattern matrices. I1 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 I1, J1, and VAL are printed out and flagged with **SEARCH** so that the user can see a "time history" of the corrections applied.

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

This subroutine updates the F array to keep it current. Following each correction the array is recomputed in this subroutine. In ANTSYN the coordinates of the correction point are evaluated after I1 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 = STARTU + (I - 1) * DU

V = STARTV + (J - 1) * 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 gird 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.

Data Type	Variable
Desired pattern	FDESPT = 1
Original pattern	FORGPT = 1
Final pattern	FDBPT = 1
Original current	ICURPT = 1
Final current	FCURPT = 1

SUBROUTINE PROFIL (DATA1, NPT, NUMPAT)

Subroutine PROFIL prints a graph of the data in DATA1 with automatic scaling using NPT(NPT$\leq$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 $\leq$ 51).

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

Source	Variable
Original distribution	ICURCN = 1
Final distribution	FCURCN = 1

Separate contour printouts are given for real and imaginary currents. Not intended for use with ITYPE=7 patterns.

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

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

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

Pattern	ICODE	Variable
Desired	0	FDESCN = 1
Original	1	FORGCN = 1
Final	2	FDBCN = 1

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

```
//F0663AS7 JOB SC7C2,COFFEY
/*MAIN  TIME=3,REGION=220K,LINES=10,CARDS=0
/*PRIORITY  PRIORITY
//   EXEC  FORTGCG,LIB2=SSPLIB
//FORT.SYSIN DD *
/*
//GO.FT22F001 DD DSN=ANTDATA.ASC7C2,UNIT=3330,VOL=SER=USERPK,DISP=SHR
//GO.SYSIN DD *
/*
//
```

6.6.2 Source Listing

```
C
C
C        MAIN PROGRAM TO SYNTHESIZE A PATTERN FOR A GIVEN SOURCE.
C
C
C        VERSION 3   73/164  --  JUNE 13,1973
C
C
C        WRITTEN BY:  E. L. COFFEY
C                     W. L. STUTZMAN
C
C
C        UNDER NASA GRANT: 47-004-103
C
C        LANGUAGE:  FORTRAN IV
C
C        SUBPROGRAMS REQUIRED:
C             DIRCTV
C             INPUT
C             READ
C             ORGPAT
C             ANTSYN
C             SEARCH
C             CHECK
C             UPDATE
C             PAT
C             SOURCE
C             LOCSOR
C             SPLOC
C             SPECPT
C             SPSOR
C             SINPUT
C             CURREN
C             PRINT
C             PROFIL
C             CONTUR
C             PATCON
C             LIST
C             DESPAT
C             DATE
C             STIME
C             BESJ
C             ...STANDARD FORTRAN LIBRARY SUBPROGRAMS...
C
C
C        INPUT/OUTPUT SUPPORT:
C
C             FT05F001  (SYSIN)  --  CARD READER
C
C             FT06F001  (SYSPRINT)  --  LINE PRINTER
C
C             FT22F001  (ANTDATA.A507C2)  --  AUXILIARY STORAGE
C
C
C        STORAGE REQUIREMENTS:  220K
C
```

```
C
C         REFERENCE:  W. L. STUTZMAN, "SYNTHESIS OF PENCIL-BEAM ANTENNA
C              PATTERNS WITH SIDE LOBE CONTROL," VPI&SU TECHNICAL
C              REPORT NO. VPI-71-1,1971.
C
C
C
      DEFINE FILE 22(35,9100,E,NREC)

      INTEGER TITLE(20)
      INTEGER NUMSKP(35),FDESPT,FDESCN,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 DATA1(401,2),DATA2(401,2)
      REAL UORG(100),VORG(100),CORG(100)
      REAL INITLS,INITLT
      COMPLEX SOURCE
      INTEGER FCURPT,FCURCN,FCURPR
C
      COMMON /MPROG/ MCUR,NCUR
      COMMON /START/ NORG,UORG,VORG,CORG
      COMMON /PAT1/ P1,P2,P3,P4,P5,P6,PI,SS(400),TT(400),RR(400)
      COMMON /PAT2/ I1,I2,I3,I4,I5
      COMMON /LOC/ ITYPE
      DATA TITLE / 'ANTE','NNA ','SYNT','HESI','S PR','OGRA',
     $'M   ','VERS','ION ','3  L','EVEL',' 1  ','  73','/164',
     $'    ','VPI ','E.E.',' DEP','T.  ','    '/
C
      NAMELIST /PARAM/ IDISK,ISYMM,ITRMAX,DELTAU,DELTAV,STARTU,STARTV,
     $MMAX,NMAX,MCENT,NCENT
C
         NAMELIST /IPRINT/ FDESPT,FDESPR,FDESCN,FDBPT,FDBCN,FDBPR,
     $FORGPT,FORGCN,FORGPR,ICURPT,ICURCN,ICURPR,FCURPT,FCURCN,FCURPR,
     $DIRECT
C
C
C          BEGIN PROCESSING
C
 9999 CONTINUE
      READ(22'1,8850) NUMPAT,NUMTRK,NUMSKP,IPASS

 8850 FORMAT(75A4,11(200A4))
      NUMPAT=NUMPAT+1
      CALL DATE(I1,J1,K1)
      CALL STIME(IT)
      IHR=IT/10000
      IFR=IT-IHR*10000
      FHR=IFR/10000.
      FM=FHR*60.
      IMIN=FM
      ISEC=(FM-IMIN)*60.
      WRITE(6,1) I1,J1,K1,IHR,IMIN,ISEC,NUMPAT
    1 FORMAT('1  ANTENNA SYNTHESIS PROGRAM   VERSION 3  LEVEL 1',
     $5X,'VPI EE DEPT.',5X,'DATE = ',A2,'-',A2,'-',A2,
     $5X,'TIME= ',I2,'.',I2,'.',I2,5X,'PATTERN ',I4///)
C
```

```
C          DEFAULT PARAMETERS
C
           IDISK=0
           ISYMM=0
           ITRMAX=100
           MMAX=1
           NMAX=1
           MCENT=1
           NCENT=1
           DELTAU=0.
           DELTAV=0.
           STARTU=0.
           STARTV=0.
           MCUR=1
           NCUR=1
           FDESPT=0
           FDESCN=0
           FDBPT=0
           FDBCN=0
           FDBPR=0
           FDESPR=0
           FORGPT=0
           FORGCN=0
           ICURPT=0
           ICURCN=0
           ICURPR=0
           FCURPT=0
           FCURCN=0
           FCURPR=0
           FORGPR=0
           DIRECT=0
C
C
C
           FNORM=1.0
           ISUC=0
           DELTAS=0.
           DELTAT=0.
C
C
C
C          INPUT
C
           READ(5,PARAM)
       WRITE(6,1521) IDISK,STARTU,MMAX,ISYMM,STARTV,NMAX,ITRMAX,DELTAU,
      $MCENT,DELTAV,NCENT
 1521 FORMAT(57X,'PROGRAM PARAMETERS'//30X,'IDISK = ',I1,26X,'STARTU = '
      $,F6.3,20X,'MMAX = ',I3/30X,'ISYMM = ',I1,26X,'STARTV = ',F6.3,
      $20X,'NMAX = ',I3/30X,'ITRMAX = ',I4,22X,'DELTAU = ',F6.3,20X,
      $'MCENT = ',I3/65X,'DELTAV = ',F6.3,20X,'NCENT = ',I3///)
       READ(5,IPRINT)
       WRITE(6,1522) FDESPT,FORGPT,FDBPT,ICURPT,FCURPT,
      $              FDESCN,FORGCN,FDBCN,ICURCN,FCURCN,
      $              FDESPR,FORGPR,FDBPR,ICURPR,FCURPR
 1522 FORMAT(31X,'FDESPT = ',I1,5X,'FORGPT = ',I1,5X,'FDBPT = ',I1,
      $5X,'ICURPT = ',I1,5X,'FCURPT = ',I1/31X,
      $'FDESCN = ',I1,5X,'FORGCN = ',I1,5X,'FDBCN = ',I1,5X,'ICURCN = ',
      $I1,5X,'FCURCN = ', I1,5X,2X/31X,'FDESPR = ',I1,5X,'FORGPR = ',I1,
      $5X,'FDBPR = ',I1,5X,'ICURPR = ',I1,5X,'FCURPR = ',I1///)
```

```
      CALL INPUT
      IF(ITYPE.EQ.7.AND.ICURPR.EQ.1) ICURPR =2
      IF (ITYPE.EQ.7 .AND. FCURPR.EQ.1) FCURPR=2
      CALL LOCSOR(1,1,INITLS,INITLT)
      CALL LOCSOR(MCUR,NCUR,FINALS,FINALT)
      IF(MCUR.NE.1) DELTAS=(FINALS-INITLS)/(MCUR-1)
      IF(NCUR.NE.1) DELTAT=(FINALT-INITLT)/(NCUR-1)
      CALL DESPAT(FDES,FU,FL,MMAX,NMAX,STARTU,STARTV,
     $DELTAU,DELTAV)
C
C
      IF(FDESPT) 300,300,301
  301 WRITE(6,302)
  302 FORMAT(1H1///////////////////////////55X,'DESIRED PATTERN IN DB.')
      CALL PRINT(FDES,MMAX,NMAX,STARTU,STARTV,DELTAU,DELTAV)
C
  300 IF(FDESCN) 303,303,304
  304 CONTINUE
      CALL PATCON(FDES,MMAX,NMAX,0,-0.5,1.3,0.2,STARTU,STARTV,
     $DELTAU,DELTAV,NUMPAT,ISYMM)
  303 IF(FDESPR) 306,306,307
  307 IF(MMAX.LE.1) GO TO 308
      WRITE(6,310) NUMPAT
  310 FORMAT(1H1,10X,'U-AXIS PROFILE OF DESIRED PATTERN ',I4//
     $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)*DELTAU
  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(F9.4,10X))
  308 IF(NMAX.EQ.1) GO TO 306
      WRITE(6,312) NUMPAT
  312 FORMAT(1H1,10X,'V-AXIS PROFILE OF DESIRED PATTERN ',I4//
     $12X,'U',16X,'V',15X,'FDES(U,V)',10X,'FU(U,V)',12X,'FL(U,V)'/)
      U=STARTU+(MCENT-1)*DELTAU
      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)
  306 CONTINUE
C
C        ENTER ORIGINAL PATTERN
C
      CALL ORGPAT(F,MMAX,NMAX,STARTU,STARTV,DELTAU,DELTAV,
     $CURR,CURI,MCUR,NCUR)
C
C        OUTPUT OF ORIGINAL PATTERN
C
      IF(FORGPT) 400,400,401
  401 WRITE(6,402)
  402 FORMAT(1H1///////////////////////////55X,'INITIAL PATTERN')
      CALL PRINT(F,MMAX,NMAX,STARTU,STARTV,DELTAU,DELTAV)
C
  400 IF(FORGCN) 403,403,404
  404 CONTINUE
      CALL PATCON(F,MMAX,NMAX,1,-0.5,1.3,0.2,STARTU,STARTV,DELTAU,
     $DELTAV,NUMPAT,ISYMM)
  403 IF(FORGPR) 406,406,407
  407 CONTINUE
```

```
      DO 408 J=1,401
      U=(J-1)*0.005-1.0
      V=U
      SUMU=0.
      SUMV=0.
      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)
  409 CONTINUE

      DATA1(J,1)=U
      DATA1(J,2)=SUMU
      DATA2(J,1)=V
      DATA2(J,2)=SUMV
  408 CONTINUE
      IF(MMAX.LE.1) GO TO 2801
      WRITE(6,410)
  410 FORMAT(1H1,25X,'U-AXIS PROFILE OF INITIAL PATTERN')
      CALL PROFIL(DATA1,401,NUMPAT)
 2801 IF(NMAX.LE.1) GO TO 406
      WRITE(6,411)
  411 FORMAT(1H1,25X,'V-AXIS PROFILE OF INITIAL PATTERN')
      CALL PROFIL(DATA2,401,NUMPAT)
  406 CONTINUE
C
C        ORIGINAL EXCITATION
C
      IF(ICURPT) 500,500,501
  501 WRITE(6,502)
  502 FORMAT(1H1///////////////////////////////55X,'INITIAL CURR')
      CALL PRINT(CURR,MCUR,NCUR,INITLS,INITLT,DELTAS,DELTAT)
      WRITE(6,582)
  582 FORMAT(1H1///////////////////////////////55X,'INITIAL CURI')
      CALL PRINT(CURI,MCUR,NCUR,INITLS,INITLT,DELTAS,DELTAT)
  500 IF(ICURCN) 503,503,504
  504 WRITE(6,505)
  505 FORMAT(1H1//////10X,'INITIAL CURR')
      CALL CONTUR(MCUR,NCUR,0.005,-0.04,0.04,0.0,CURR,NUMPAT)
      WRITE(6,585)
  585 FORMAT(1H1//////10X,'INITIAL CURI')
      CALL CONTUR(MCUR,NCUR,0.005,-0.04,0.04,0.0,CURI,NUMPAT)
  503 IF(ICURPR-1) 506,507,514
  507 IF(MCUR.LE.1) GO TO 508
      WRITE(6,510)
  510 FORMAT(1H1,10X,'S AXIS PROFILE OF INITIAL CURRENT'//
     $13X,'S',17X,'T',18X,'REAL',12X,'IMAGINARY',10X,'MAGNITUDE',
     $12X,'PHASE'/)
      J=NCUR/2+1
      DO 509 I=1,MCUR
      CALL LOCSOR(I,J,S,T,ITYPE)
      AMAG=SQRT(CURR(I,J)**2+CURI(I,J)**2)
      IF(AMAG.EQ.0.) APH=0.
      IF(AMAG.EQ.0.) GO TO 509
      APH=ATAN2(CURI(I,J),CURR(I,J))*57.2957795
  509 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))
C
```

```
  508 IF(NCUR.LE.1) GO TO 506
      WRITE(6,512)
  512 FORMAT(1H1,10X,'T AXIS PROFILE OF INITIAL CURRENT'//
     $13X,'S',17X,'T',18X,'REAL',12X,'IMAGINARY',10X,'MAGNITUDE',
     $12X,'PHASE'/)
      I=MCUR/2+1
      DO 513 J=1,NCUR
      CALL LOCSOR(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.EC.0.) GO TO 513
      APH=ATAN2(CI,CR)*57.2957795
  513 WRITE(6,511) S,T,CR,CI,AMAG,APH
      GO TO 506
  514 WRITE(6,515)
  515 FORMAT(1H1///10X,'INITIAL ELEMENT CURRENTS'//5X,
     $'J',10X,'S',15X,'T',15X,'CURR',11X,'CURI')
      CALL LIST(CURR,CURI,MCUR,NCUR)
  506 CONTINUE
      IC=0
      WRITE(6,4747)
 4747 FORMAT(1H1)
      CALL ANTSYN (ISUC,MMAX,NMAX,FDES,FU,FL,ITRMAX,ISYMM,CORCOF,IC,US
     $,VS,STARTU,DELTAU,STARTV,DELTAV,MCENT,NCENT,ITER,FNORM,F)
C
C          PRINT OUT RESULTS
C
      WRITE(6,391)
  391 FORMAT(1H1,52X,'--  FINAL COEFFICIENTS  --'//45X,'J',7X,
     $'US(J)',7X,'VS(J)',5X,'CORCOF(J)'//)
      IF(IC.LE.0) WRITE(6,977)
  977 FORMAT(10X,'NO ITERATIONS PERFORMED')
      IF(IC.LF.0) GO TO 978
      CO 498 J=1,IC
  498 WRITE(6,35) J,US(J),VS(J),CORCOF(J)
   35 FORMAT(44X,I3,5X,F7.4,5X,F7.4,5X,F7.4)
      WRITE(6,3874)
 3874 FORMAT(///45X,'J',6X,'UORG(J)',5X,'VORG(J)',6X,'CORG(J)'/)
      DO 3877 J=1,NORG
 3877 WRITE(6,35) J,UORG(J),VORG(J),CORG(J)
      WRITE(6,497) ITER
  497 FORMAT(1H0,9X,'NUMBER OF ITERATIONS = ',I6)
      WRITE(6,496) FNORM
  496 FORMAT(1H0,9X,'FNORM = ',F10.5)
      WRITE(6,976) NUMPAT
  976 FORMAT(1H0,9X,'PATTERN NUMBER = NUMPAT = ',I5)
C
C          OUTPUT FINAL PATTERN IN DB
C
  978 CONTINUE
      DO 29 J=1,MMAX
      DO 29 K=1,NMAX
      IF(F(J,K)) 290,289,290
  289 F(J,K)=-200.
```

```
      GO TO 29
  290 F(J,K)=20.*ALOG10(ABS(F(J,K)))
   29 CONTINUE
C
C
      IF(FDBPT) 600,600,601
  601 WRITE(6,602)
  602 FORMAT(1H1/////////////////////////55X,'FINAL PATTERN IN DB.')
      CALL PRINT(F,MMAX,NMAX,STARTU,STARTV,DELTAU,DELTAV)
C
  600 IF(FDBCN) 603,603,604
  604 CONTINUE
      CALL PATCON(F,MMAX,NMAX,2,-45.,0.0,5.0,STARTU,STARTV,
     $DELTAU,DELTAV,NUMPAT,ISYMM)
  603 IF(FDBPR) 606,606,607
  607 CONTINUE
      DO 608 J=1,401
      U=(J-1)*0.005-1.0
      V=U
      SUMU=0.
      SUMV=0.
      DO 609 K=1,IC
      SUMU=SUMU+CORCOF(K)*PAT(U-US(K),-VS(K),ITYPE)
      SUMV=SUMV+CORCOF(K)*PAT(-US(K),V-VS(K),ITYPE)
  609 CONTINUE
      DATA1(J,1)=U
      DATA2(J,1)=V
      DATA1(J,2)=DATA1(J,2)+SUMU
      DATA2(J,2)=DATA2(J,2)+SUMV
      DATA1(J,2)=DATA1(J,2)*FNORM
      DATA2(J,2)=DATA2(J,2)*FNORM
  608 CONTINUE
      IF(MMAX.LE.1) GO TO 2901
      WRITE(6,610)
  610 FORMAT(1H1,25X,'U-AXIS PROFILE OF FINAL PATTERN')
      CALL PROFIL(DATA1,401,NUMPAT)
 2901 IF(NMAX.LE.1) GO TO 606
      WRITE(6,611)
  611 FORMAT(1H1,25X,'V-AXIS PROFILE OF FINAL PATTERN')
      CALL PROFIL(DATA2,401,NUMPAT)
  606 CONTINUE
C
C
C
      IF(FCURPT+FCURPR+FCURCN .LE. 0) GO TO 706
      CALL CURREN(CURR,CURI,MCUR,NCUR,US,VS,CORCOF,IC)
C
C        FINAL EXCITATION
C
      IF(FCURPT) 700,700,701
  701 WRITE(6,702)
  702 FORMAT(1H1///////////////////////////55X,'FINAL CURR')
      CALL PRINT(CURR,MCUR,NCUR,INITLS,INITLT,DELTAS,DELTAT)
      WRITE(6,782)
  782 FORMAT(1H1///////////////////////////55X,'FINAL CURI')
      CALL PRINT(CURI,MCUR,NCUR,INITLS,INITLT,DELTAS,DELTAT)
  700 IF(FCURCN) 703,703,704
  704 WRITE(6,705)
```

```
      705 FORMAT(1H1//////10X,'FINAL CURR')
          CALL CONTUR(MCUR,NCUR,0.005,-0.04,0.04,0.0,CURR,NUMPAT)
          WRITE(6,785)
      785 FORMAT(1H1//////10X,'FINAL CURI')
          CALL CONTUR(MCUR,NCUR,0.005,-0.04,0.04,0.0,CURI,NUMPAT)
      703 IF(FCURPR-1) 706,707,711
      707 IF(MCUR.LE.1) GO TO 708
          WRITE(6,710)
      710 FORMAT(1H1,10X,'S AXIS PROFILE OF FINAL CURRENT'//
         $13X,'S',17X,'T',18X,'REAL',12X,'IMAGINARY',10X,'MAGNITUDE',
         $12X,'PHASE'/)
          J=NCUR/2+1
          DO 709 I=1,MCUR
          CALL LOCSCR(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 709
          APH=ATAN2(CI,CR)*57.2957795
      709 WRITE(6,511) S,T,CR,CI,AMAG,APH
C
      708 IF(NCUR.LE.1) GO TO 706
          WRITE(6,712)
      712 FORMAT(1H1,10X,'T AXIS PROFILE OF FINAL CURRENT'//
         $13X,'S',17X,'T',18X,'REAL',12X,'IMAGINARY',10X,'MAGNITUDE',
         $12X,'PHASE'/)
          I=MCUR/2+1
          DO 713 J=1,NCUR
          CALL LOCSCR(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 713
          APH=ATAN2(CI,CR)*57.2957795
      713 WRITE(6,511) S,T,CR,CI,AMAG,APH
          GO TO 706
      711 WRITE(6,714)
      714 FORMAT(1H1///10X,'FINAL ELEMENT CURRENTS'//5X,
         $'J',10X,'S',15X,'T',15X,'CURR',15X,'CURI')
          CALL LIST(CURR,CURI,MCUR,NCUR)
      706 CONTINUE
C
C
          ICOUNT=NUMPAT
          WRITE(22'1,8850) ICOUNT,NUMTRK,NUMSKP,IPASS
          IF(DIRECT.EQ.0) GO TO 9998
          CALL DIRCTV(CORG,UORG,VORG,NORG,US,VS,CORCOF,IC,MMAX,NMAX,
         $DIRORG,DIRFNL)
          WRITE(6,6789) DIRORG,DIRFNL
     6789 FORMAT('1   DIRORG = ',F7.2,' DB.'/
         $'0   DIRFNL = ',F7.2,' DB.')
     9998 CONTINUE
```

```
      IF(IDISK.EQ.0) GO TO 9997
      IF(IDISK.EQ.1 .AND. ISUC.NE. 1) GO TO 9997
C        DISK OUTPUT
      DO 7000 J=2,35
      IF(NUMSKP(J) .EQ. 0) GO TO 7001
 7000 CONTINUE
      WRITE(6,7002)
 7002 FORMAT('0     NO DISK SPACE AVAILABLE -- DATA NOT STORED')
      GO TO 9999
 7001 CONTINUE
C
C        SPACE IS AVAILABLE ON RECORD "J"
C
      NUMSKP(J)=1
      WRITE(22'1,8850) NUMPAT,J,NUMSKP,IPASS

      WRITE(22'J,8850) NUMPAT,TITLE,ISYMM,ITER,ISUC,FNORM,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,
     $PI,(SS(M),TT(M),M=1,400),I1,I2,I3,I4,I5,MCUR,NCUR

      WRITE(6,7003) NUMPAT,J
 7003 FORMAT('0      PATTERN NUMBER ',I4,' HAS BEEN STORED ON RECORD',
     $I4,' OF ANTDATA.A507C2')
 9997 GO TO 9999
      END


      SUBROUTINE DIRCTV(CORG,UORG,VORG,NORG,US,VS,CORCOF,IC,MMAX,NMAX,
     1 DIRORG,DIRFNL)
C     THIS SUBROUTINE CALCULATES THE DIRECTIVITY OF THE ORIGINAL PATTERN
C       ,DIRORG, AND OF THE FINAL PATTERN, DIRFNL
      DIMENSION CORG(100),UORG(100),VORG(100),US(500),VS(500),CORCOF(500
     $)
      COMMON /LOC/ ITYPE
      FORGSQ=0.
      FSQ=0.
      FMAX1=0.
      FMAX2=0.
      DO 10 J=1,101
      U=-1.0+(J-1)*0.02
      DO 10 K=1,101
      V=-1.0+(K-1)*0.02
      UVSQ=U*U+V*V
      F=0.
      IF(UVSQ.GE.1.0) GO TO 10
C
C
      IF(NORG.LE.0) GO TO 25
      DO 20 L=1,NORG
   20 F=F+CORG(L)*PAT(U-UORG(L),V-VORG(L),ITYPE)
      FORGSQ=FORGSQ+F**2/SQRT(1.0-UVSQ)
      IF(ABS(F) .GT. FMAX1) FMAX1=ABS(F)
   25 CONTINUE
```

```
      FSQ=FORGSQ
      FMAX2=FMAX1
      IF(IC.LE.0) GO TO 10
      DO 30 L=1,IC
   30 F=F+CORCOF(L)*PAT(U-US(L),V-VS(L),ITYPE)
      FSQ=FSQ+F**2/SQRT(1.0-UVSQ)
      IF(ABS(F).GT.FMAX2) FMAX2=ABS(F)
   10 CONTINUE
      FORGSQ=FORGSQ*0.0004/FMAX1
      FSQ=FSQ*0.0004/FMAX2
      DIRORG=4.0*3.14159265/FORGSQ
      DIRFNL=4.0*3.14159265/FSQ
C
C
      DIRORG=10.*ALOG10(DIRORG)
      DIRFNL=10.*ALOG10(DIRFNL)
      RETURN
      END



      SUBROUTINE INPUT
C
      INTEGER PX,PY
      REAL LX,LY,INITLS,INITLT
C
C
      COMMON /PAT1/ P1,P2,P3,P4,P5,P6,PI,SS(400),TT(400),RR(400)
      COMMON /PAT2/ I1,I2,I3,I4,I5
      COMMON /LOC/ ITYPE
      COMMON /MPROG/ MCUR,NCUR
      COMMON /SYN/ LX,LY
C
C
      NAMELIST /PATIN/ LX,LY,PX,PY,DISX,DISY,INITLS,DELTAS,FINALS,
     $INITLT,DELTAT,FINALT,NELMT,ARAD,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 (100,200,300,400,500,600,700), ITYPE
      WRITE(6,20) ITYPE
   20 FORMAT(1H0,5X,'***ERROR***    ITYPE HAS THE VALUE ',I11,':',2X,
     $'EXECUTION TERMINATED')
      STOP
C
C
  100 P1=LY
      P2=INITLT
      P3=DELTAT
      LX=0.0
      NCUR=(FINALT-INITLT)/DELTAT+1.5
      MCUR=1
```

```
          WRITE(6,101) LY,INITLT,FINALT,DELTAT,NCUR
      101 FORMAT(1CX,'ITYPE=1  --  UNIFORM LINE SOURCE'//15X,'LY = ',F7.3//
         $15X,'INITLT,FINALT,DELTAT:',3(1X,F8.4)//15X,'NUMBER OF SAMPLE POIN
         $TS = NCUR = ',I3)
          GO TO 999
C
C
      200 P1=LY
          I1=PY
          LX=C.C
          P2=DISY
          NCUR=PY
          MCUR=1
          WRITE(6,2C1) LY,PY,DISY
      201 FORMAT(10X,'ITYPE=2  --  UNIFORM LINEAR ARRAY'//
         $15X,'LY = ',F7.3//15X,'NUMBER OF ELEMENTS = ',I3//15X,'INTER-ELEME
         $NT SPACING = ',F6.3)
          GC TO 999
C
C
      3CC P1=LY
          LX=0.0
          P2=INITLT
          P3=DELTAT
          NCUR=(FINALT-INITLT)/DELTAT+1.5
          MCUR=1
          WRITE(6,3C1) LY,INITLT,FINALT,DELTAT,NCUR
      301 FORMAT(10X,'ITYPE=3  --  TRIANGULAR LINE SOURCE'//
         $15X,'LY = ',F7.3//15X,'T VARIES FROM ',F8.4,' TO ',F8.4,5X,
         $'DELTAT = ',F6.3//15X,'NUMBER OF SAMPLE PCINTS = NCUR = ',I3)
          GC TO 999
C
C
      4CC P1=LX
          P2=LY
          P3=INITLS
          P4=INITLT
          P5=DELTAS
          P6=DELTAT
          MCUR=(FINALS-INITLS)/DELTAS+1.5
          NCUR=(FINALT-INITLT)/DELTAT+1.5
          WRITE(6,401) LX,LY,INITLS,DELTAS,FINALS,INITLT,DELTAT,FINALT,MCUR,
         $NCUR
      401 FORMAT(10X,'ITYPE=4  --  UNIFORM RECTANGULAR APERTURE'//
         $15X,'DIMENSIONS = LX,LY = ',F7.4,' , ',F7.4//
         $15X,'INITLS,DELTAS,FINALS: ',3(F8.4,1X)//
         $15X,'INITLT,DELTAT,FINALT: ',3(F8.4,1X)//
         $15X,'MCUR,NCUR: ',2(I3,2X))
          GC TO 999
C
C
      5CC P1=LX
          P2=LY
          I1=PX
```

```
      I2=PY
      P3=DISX
      P4=DISY
      MCUR=I1
      NCUR=I2
      WRITE(6,501) LX,LY,PX,PY,DISX,DISY
  501 FORMAT(10X,'ITYPE=5  --  UNIFORM RECTANGULAR ARRAY'//
     $15X,'DIMENSIONS = LX,LY = ',F7.4,' , ',F7.4//
     $15X,'NUMBER OF ELEMENTS = PX,PY = ',I3,' , ',I3//
     $15X,'INTER-ELEMENT SPACING = DISX,DISY = ',F6.3,' , ',F6.3)
      GO TO 999
C
C
  600 P1=ARAD
      P3=INITLS
      P4=INITLT
      P5=DELTAS
      P6=DELTAT
      LX=ARAD*2.
      LY=LX
      MCUR=(FINALS-INITLS)/DELTAS+1.5
      NCUR=(FINALT-INITLT)/DELTAT+1.5

      WRITE(6,601) ARAD,INITLS,DELTAS,FINALS,INITLT,DELTAT,FINALT,MCUR,
     $NCUR
  601 FORMAT(10X,'ITYPE=6  --  UNIFORM CIRCULAR APERTURE'//

     $15X,'ARAD = ',F7.3//15X,'INITLS,DELTAS,FINALS: ',3(F8.4,1X)//
     $15X,'INITLT,DELTAT,FINALT: ',3(F8.4,1X)//
     $15X,'MCUR,NCUR: ',2(I3,2X))
      GO TO 999
C
C
  700 I1=MCUR
      I2=NCUR
      LX=1.0
      LY=1.0
      NELMT=I1*I2
      WRITE(6,701)
  701 FORMAT(10X,'ITYPE=7  --  GENERAL ARRAY'//
     $15X,'ELEMENT',7X,'SS(J)',14X,'TT(J)')
      DO 702 J=1,NELMT
      READ(5,703) SS(J),TT(J)
  703 FORMAT(3F10.0)
      WRITE(6,704) J,SS(J),TT(J)
  704 FORMAT(17X,I3,5X,3(E14.7,5X))
  702 CONTINUE
      GO TO 999
  990 CALL SINPUT(PX,PY,DISX,DISY,INITLS,DELTAS,FINALS,INITLT,
     $DELTAT,FINALT,NELMT,ARAD,ITYPE)
  999 RETURN
      END
```

```
      SUBROUTINE READ (F,MMAX,NMAX)
      DIMENSION F(51,51),I(6),VAL(6)
      DO 100 J=1,MMAX
      K2=0
  200 CONTINUE
      READ(5,1) (I(L),VAL(L),L=1,6)
    1 FORMAT(6(I3,F10.0))
      DO 20 L=1,6
      I1=I(L)
      IF(I1.EQ.0) GO TO 100
      K1=K2+1
      K2=K1+I1-1
      DO 10 K=K1,K2
   10 F(J,K)=VAL(L)
   20 CONTINUE
      IF(K2.LT.NMAX)  GO TO 200
  100 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE  DESPAT(FDES,FU,FL,MMAX,NMAX,STARTU,STARTV,DELTAU,
     $DELTAV)
      DIMENSION FDES(51,51),FU(51,51),FL(51,51)
C
C        THIS LOADS THE DESIRED PATTERN AND UPPER AND LOWER LIMITS
C
      CALL READ(FDES,MMAX,NMAX)
      CALL READ(FU,MMAX,NMAX)
      CALL READ(FL,MMAX,NMAX)
      RETURN
      END
```

```
      SUBROUTINE ORGPAT(F,MMAX,NMAX,STARTU,STARTV,DELTAU,DELTAV,CURR,
     $CURI,MCUR,NCUR)
      REAL F(51,51),CURR(51,51),CURI(51,51)
      REAL UORG(100),VORG(100),CORG(100)
      COMPLEX SOURCE
      COMPLEX TEMP
      COMMON /START/ NORG,UORG,VORG,CORG
      COMMON /LOC/ ITYPE
C
C        THIS ORGPAT WILL BE "WOODWARD-LAWSON" INPUT.
C
      DO 10 M=1,MMAX
      DO 10 N=1,NMAX
   10 F(M,N)=0.
```

```
      DO 15 M=1,MCUR
      DO 15 N=1,NCUR
      CURR(M,N)=0.
   15 CURI(M,N)=0.
      WRITE(6,17)
   17 FORMAT(1H1,50X,'-- INITIAL COEFFICIENTS --'//45X,'J',6X,
     $'UORG(J)',5X,'VORG(J)',6X,'CORG(J)'/)
      READ(5,1) NORG
    1 FORMAT(I5)
      DO 20 IC=1,NORG
      READ(5,2) US,VS,CORCOF
    2 FORMAT(3F10.0)
      UORG(IC)=US
      VORG(IC)=VS
      CORG(IC)=CORCOF
      DO 30 M=1,MMAX
      U=STARTU+(M-1)*DELTAU
      DU=U-US
      DO 30 N=1,NMAX
      V=STARTV+(N-1)*DELTAV
      DV=V-VS
   30 F(M,N)=F(M,N)+CORCOF*PAT(DU,DV,ITYPE)
      DO 40 M=1,MCUR
      DO 40 N=1,NCUR
      TEMP=SOURCE(M,N,US,VS,ITYPE)
      CURR(M,N)=CURR(M,N)+CORCOF*REAL(TEMP)
   40 CURI(M,N)=CURI(M,N)+CORCOF*AIMAG(TEMP)
      WRITE(6,50) IC,US,VS,CORCOF
   50 FORMAT(44X,I3,5X,F7.4,5X,F7.4,5X,F7.4)
   20 CONTINUE
      RETURN
      END


      SUBROUTINE ANTSYN(ISUC,MMAX,NMAX,FDES,FU,FL,ITRMAX,ISYMM,CORCOF,
     $IC,US,VS,STARTU,DELTAU,STARTV,DELTAV,MCENT,NCENT,ITER,FNORM,F)
      REAL FDES(51,51),FU(51,51),FL(51,51),F(51,51)
      REAL US(500),VS(500),CORCOF(500) UORG(100),VORG(100),CORG(100)
      REAL LX,LY,LXY
      COMMON /START/ NORG,UORG,VORG,CORG
      COMMON /SYN/ LX,LY
      LXY=1./AMAX1(LX,LY)
      ITER=0
   27 ITER=ITER+1
C        NORMALIZE...
      FBIG=F(MCENT,NCENT)
      DO 150 M=1,MMAX
      DO 150 N=1,NMAX
  150 F(M,N)=F(M,N)/FBIG
      FNORM=FNORM/FBIG
      DO 151 I=1,NORG
  151 CORG(I)=CORG(I)/FBIG
      IF(IC.LE.0) GO TO 153
      DO 152 I=1,IC
  152 CORCOF(I)=CORCOF(I)/FBIG
  153 CONTINUE
```

```
C
C                 -- ITERATION PROCEDURE --
C
C
C            SEE IF SPECS ARE MET.
C
      DO 24 J=1,NMAX
      U=STARTU+(J-1)*DELTAU
      DO 24 K=1,NMAX
      V=STARTV+(K-1)*DELTAV
      UVSQ=U*U+V*V
      IF(UVSQ.GT.1.0) GO TO 24
      IF(FDES(J,K).EQ.99.)  GO TO 24
      IF(FL(J,K).LE.0.00001 .AND. ABS(F(J,K)).LE.1.E-4) GO TO 24
      X1=ABS(F(J,K))
      IF(X1.GT.FU(J,K)) GO TO 25
      IF(FL(J,K).EQ. 99.0) GO TO 24
      IF(X1.LT.FL(J,K)) GO TO 25
   24 CONTINUE
      ISUC=1
      IC=IC+1
C
C            SPECS ARE MET -- PROCEDE TO PRINTOUT.
C
      GO TO 750
   25 CONTINUE
C
C            SPECS ARE NOT MET AT POINT (J,K)
C
C
      IC=IC+1
      IF(ITER/100*100 .EQ. ITER)  WRITE(6,7117) ITER
 7117 FORMAT(10X,I6,' ITERATIONS COMPLETED')
      IF(ITER-ITRMAX) 22,22,23
   23 WRITE(6,34) ITRMAX
   34 FORMAT(1H0,9X,'NUMBER  OF ITERATIONS EXCEEDED', I5/
     $10X,'PRINTOUT OF INTERMEDIATE RESULTS FOLLOWS.')
      GO TO 750
   22 CONTINUE
C
C            FIND RELATIVE MAXIMUM ERROR
C
      CALL SEARCH(J,K,VAL,FDES,FU,FL,F,MMAX,NMAX,STARTU,STARTV,DELTAU,
     $DELTAV)
      IF(VAL.NE. 0.0) GO TO 248
C            VAL EQUALS ZERO
      WRITE(6,100)
  100 FORMAT('0    ERROR IN SUBROUTINE SEARCH -- VAL=0.')
      GO TO 750
  248 U1=(J-1)*DELTAU+STARTU
      V1=(K-1)*DELTAV+STARTV
      IF(ABS(U1).LE.0.1*DELTAU) U1=0.
      IF(ABS(V1).LE.0.1*DELTAV)  V1=0.
      IF(LX.EQ.0.) GO TO 1000
      IF(U1.NE.0. .AND. ABS(U1).LE. 0.5/LX)  VAL=VAL/2.
 1000 IF(LY.EQ.0.) GO TO 1001
      IF(V1.NE.0. .AND. ABS(V1) .LE. 0.5/LY)  VAL=VAL/2.
```

```
      IF(ISYMM.NE.4) GO TO 1001
      ITEMP=0
      UV=ABS(ABS(U1)-ABS(V1))
      IF(UV.EQ.0.) GO TO 1001
      IF(UV*1.414.LE.LXY) VAL=VAL/2.
 1001 CONTINUE
C
C        BASIC CORRECTION  --  INDEPENDENT OF ISYMM
C
      US(IC)=U1
      VS(IC)=V1
      CORCOF(IC)=VAL
      CALL UPDATE(IC,US,VS,CORCOF,F,MMAX,NMAX,FNORM,STARTU,STARTV
     $,DELTAU,DELTAV)
      CALL CHECK(IC,VAL,US,VS,CORCOF,DELTAU,DELTAV)
      IF(ISYMM) 26,27,26
   26 CONTINUE
      IF(ISYMM-2) 261,260,260
  260 CONTINUE
C
C        V-AXIS AND QUADRILATERAL SYMMETRY  --  ISYMM = 2,3,4
C
      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
C
C        U-AXIS AND QUADRILATERAL SYMMETRY  -- ISYMM = 1,3,4
C
  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
C
C        QUADRILATERAL SYMMETRY ONLY  --  ISYMM = 3,4
C
      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
C
C        FOR BIQUADRILATERAL SYMMETRY ONLY  --  ISYMM = 4
C
      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
      END


      SUBROUTINE SEARCH(I1,J1,VAL,FDES,FU,FL,F,MMAX,NMAX,STARTU,
     $STARTV,DELTAU,DELTAV)
      REAL FU(51,51),FL(51,51),F(51,51),FDES(51,51)
      VAL=0.
      EMAX=0.
      I2=I1
      J2=J1
      DO 10 J=I2,MMAX
      U=STARTU+(J-1)*DELTAU
      DO 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
C
      IF(FITER.GT.FU(J,K)) GO TO 2000
      IF(FL(J,K).EQ.99.0) GO TO 20
      IF(FL(J,K).LE.0.00001 .AND. FITER.LE.1.E-4) GO TO 20
      IF(FITER.GT.FL(J,K)) GO TO 20
C
 2000 X=FDES(J,K)
      ERROR = FITER-X
      IF(ABS(ERROR)-ABS(EMAX)) 20,20,21
   21 EMAX=ERROR
      VAL=SIGN(ERROR,F(J,K)*(X-FITER))
      I1=J
      J1=K
   20 CONTINUE
   10 CONTINUE
      WRITE(6,100) I1,J1,VAL
  100 FORMAT(5X,'**SEARCH**',I8,I8,5X,F7.4)
      RETURN
      END


      SUBROUTINE CHECK(IC,VAL,US,VS,CORCOF,DELTAU,DELTAV)
      REAL US(500),VS(500),CORCOF(500)
      IF(IC.EQ.1) RETURN
      DU=0.1*DELTAU
      DV=0.1*DELTAV
      IC1=IC-1
      U=US(IC)
      V=VS(IC)
```

```
      DO 10 J=1,IC1
      IF(ABS(U-US(J)).LE.DU.AND.ABS(V-VS(J)).LE.DV)  GO TO 20
   10 CONTINUE
      RETURN
   20 CORCOF(J)=CORCOF(J)+VAL
      IC=IC-1
      RETURN
      END


      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
C        ITYPE = 1  --  UNIFORM LINE SOURCE LOCATED AT S=0.
C                2  --  UNIFORM LINEAR ARRAY LOCATED AT S=0.
C                3  --  TRIANGULAR LINE SOURCE LOCATED AT S=0.
C                4  --  UNIFORM RECTANGULAR APERTURE.
C                5  --  UNIFORM RECTANGULAR ARRAY.
C                6  --  UNIFORM CIRCULAR APERTURE.
C                7  --  GENERAL ARRAY.
C
C        ITYPE > 7  --  SPECIAL SOURCE (FUNCTION SPECPT(U,V,ITYPE) WILL
C                       BE CALLED.
C
C
C        VERSION 1   LEVEL 1
C
C        DATE OF LAST REVISION:  73/193    JULY 12,1973
C
C
C        THIS WORK SUPPORTED BY NASA GRANT NGR 47-004-103
C
C        FOR FURTHER INFORMATION CONTACT:
C             W.L. STUTZMAN   DEPT. OF ELEC. ENGR.  951-6624.
C             E.L. COFFEY     DEPT. OF ELEC. ENGR. 951-5494
C
      COMPLEX TEMP,CEXP,IMAG
      COMMON /PAT1/ P1,P2,P3,P4,P5,P6,PI,SS(400),TT(400),RR(400)
      COMMON /PAT2/ I1,I2,I3,I4,I5
C
C
```

```
            IF(ITYPE.GT.7) GO TO 990
            GO TO (100,200,300,400,500,600,700),ITYPE
C
C               ITYPE .LT. 1
C
            WRITE(6,10) ITYPE
         10 FORMAT(1HC,5X,'***ERROR***    ITYPE HAS THE VALUE ',I11,':',2X,
           $'EXECUTION TERMINATED')
            STOP
C
C
C               ITYPE = 1  --  UNIFORM LINE SOURCE.
C
C           FLEN=P1
        100 CONTINUE
            PAT=1.0
            IF(V.NE.0.) PAT = SIN(PI*P1*V)/(PI*P1*V)
            GO TO 999
C
C
C               ITYPE = 2  --  UNIFORM LINEAR ARRAY
C
        200 CONTINUE
C           FLEN=P1
C           NELMT=I1
            PAT=1.0
            IF(V.NE.0.) PAT=SIN(PI*P1*V)/(I1*SIN(PI*P1*V/I1))
            GO TO 999
C
C
C               ITYPE = 3  --  TRIANGULAR LINE SOURCE.
C
        300 FLEN=P1/2.
            PAT=1.0
            IF(V.NE.0.) PAT = (SIN(FLEN*PI*V)/(FLEN*PI*V))**2
            GO TO 999
C
C
C               ITYPE = 4  --  UNIFORM RECTANGULAR APERTURE
C
        400 CONTINUE
C           FLS=P1
C           FLT=P2
            ARG1=PI*P1*U
            ARG2=PI*P2*V
            IF(ARG1) 401,402,401
        401 IF(ARG2) 403,404,403
        403 PAT=SIN(ARG1)/ARG1*SIN(ARG2)/ARG2
            GO TO 999
        404 PAT=SIN(ARG1)/ARG1
            GO TO 999
        402 IF(ARG2) 405,406,405
        405 PAT=SIN(ARG2)/ARG2
            GO TO 999
        406 PAT=1.0
            GO TO 999
C
C
C               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
  502 IF(ARG2) 505,506,505
  505 PAT=SIN(ARG2)/(I2*SIN(ARG2/I2))
      GO TO 999
  506 PAT=1.0
      GO TO 999
C
C
C          ITYPE = 6  --  UNIFORM CIRCULAR APERTURE.
C
  600 C=SQRT(U*U+V*V)
C     A=P1
      IF(C.EQ.0.) GO TO 601
      X=2.*PI*P1*C
      CALL BESJ(X,1,BJ,0.0001,IER)
      PAT=2.*BJ/X
      GO TO 999
  601 PAT=1.0
      GO TO 999
C
C
C          ITYPE = 7  --  GENERAL ARRAY
C
  700 IMAG=(0.0,1.0)
      NELMT=I1*I2
      TEMP=(0.0,0.0)
      DO 701 J=1,NELMT
      TEMP=TEMP+1.0*CEXP(IMAG*2.*PI*(U*SS(J)+V*TT(J)))
  701 CONTINUE
      PAT=REAL(TEMP)/NELMT
      GO TO 999
  990 PAT=SPECPT(U,V,ITYPE)
  999 RETURN
      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                  2 --  UNIFORM LINEAR ARRAY LOCATED AT S=0.
```

```
C                   3 --  TRIANGULAR LINE SOURCE LOCATED AT S=0.
C                   4 --  UNIFORM RECTANGULAR APERTUR.
C                   5 --  UNIFORM RECTANGULAR ARRAY.
C                   6 --  UNIFORM CIRCULAR APERTURE.
C                   7 --  GENERAL ARRAY.
C
C          ITYPE > 7 --  SPECIAL SOURCE (FUNCTION SPSOR(M,N,U,V,ITYPE)
C                        WILL BE CALLED.)
C
C          VERSION 1  LEVEL 1
C
C          DATE OF LAST REVISION:  73/166    JULY 12,1973
C
C          THIS WORK SUPPORTED BY NASA GRANT NGR 47-004-103.
C
C          FOR FURTHER INFORMATION CONTACT:
C               W. L. STUTZMAN  DEPT. OF ELEC. ENGR.  951-6624.
C               E. L. COFFEY    DEPT. OF ELEC. ENGR.  951-5494.
C
      COMPLEX TEMP,CEXP,IMAG,SPSOR
      COMMON /PAT1/ P1,P2,P3,P4,P5,P6,PI,SS(400),TT(400),RR(400)
      COMMON /PAT2/ I1,I2,I3,I4,I5
C
      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
C
C         ITYPE .LT. 1
C
      WRITE(6,10) ITYPE
   10 FORMAT(1H0,5X,'***ERROR***    ITYPE HAS THE VALUE ',I11,':',2X,
     $'EXECUTION TERMINATED')
      STOP
C
C
C         ITYPE = 1  --  UNIFORM LINE SOURCE
C
  100 CONTINUE
C     FLEN=P1
      SOURCE=CEXP(-IMAG*PI*2.*T*V)/P1
      GO TO 999
C
C
C         ITYPE = 2  -- UNIFORM LINEAR ARRAY
C
  200 CONTINUE
C FLEN=P1
      SOURCE=CEXP(-IMAG*2.*PI*V*T)/P1
      GO TO 999
C
C
C         ITYPE= 3 -- TRIANGULAR LINE SOURCE
C
  300 CONTINUE
C     FLEN=P1
      CON=ABS(2.*T/P1)
```

```
      SOURCE=2./P1*CEXP(-IMAG*2.*PI*T*V)*(1.-CON)
      IF(CON.GT.1) SOURCE=(0.0,0.0)
      GO TO 999
C
C
C        ITYPE = 4  --  UNIFORM RECTANGULAR APERTURE
C
  400 CONTINUE
C     FLS=P1
C     FLT=P2
      SOURCE=CEXP(-IMAG*2.*PI*(S*U+V*T))/(P1*P2)
      GO TO 999
C
C
C        ITYPE = 5 --  UNIFORM RECTANGULAR ARRAY
C
  500 CONTINUE
C     FLS=P1
C     FLT=P2
      SOURCE=CEXP(-IMAG*2.*PI*(S*U+V*T))/(P1*P2)
      GO TO 999
C
C
C        ITYPE = 6  --  UNIFORM CIRCULAR APERTURE
C
  600 RHO=SQRT(S*S+T*T)
C     A=P1
      SOURCE=(0.0,0.0)
      IF(RHO.LE.P1) SOURCE=CEXP(-IMAG*2.*PI*(S*U+T*V))/(2.*PI*P1**2)
      GO TO 999
C
C
C        ITYPE = 7  --  GENERAL ARRAY.
C
  700 CONTINUE
      SOURCE=CEXP(-IMAG*2.*PI*(U*S+V*T))/(I1*I2)
      GO TO 999
  990 SOURCE=SPSOR(M,N,U,V,ITYPE)
  999 RETURN
      END


      SUBROUTINE LOCSOR(M,N,S,T)
      INTEGER PX,PY
      REAL INITLS,INITLT
      COMMON /PAT1/ P1,P2,P3,P4,P5,P6,PI,SS(400),TT(400),RR(400)
      COMMON /PAT2/ I1,I2,I3,I4,I5
      COMMON /LOC/ ITYPE
C
C
      IF(ITYPE.GT.7) GO TO 990
      GO TO (100,200,300,400,500,600,700), ITYPE
      WRITE(6,10) ITYPE
   10 FORMAT(1H0,5X,'***ERROR***    ITYPE HAS THE VALUE ',I11,':',2X,
     $'EXECUTION TERMINATED')
      STOP
C
```

```
C
  100 CONTINUE
C     INITLT=P2
C     DELTAT=P3
      S=0.
      T=P2+(N-1)*P3
      GO TO 999
C
C
  200 CONTINUE
C     PY=I1
C     DISY=P2
      S=0.
      T=(N-I1/2-1)*P2
      IF(I1/2*2.EQ.I1) T=T+0.5*P2
      GO TO 999
C
C
  300 GO TO 100
C
C
  400 CONTINUE
C     INITLS=P3
C     INITLT=P4
C     DELTAS=P5
C     DELTAT=P6
      S=P3+(M-1)*P5
      T=P4+(N-1)*P6
      GO TO 999
C
C
  500 CONTINUE
C     PX=I1
C     PY=I2
C     DISX=P3
C     DISY=P4
      S=(M-I1/2-1)*P3
      T=(N-I2/2-1)*P4
      IF(I1/2*2.EQ.I1) S=S+0.5*P3
      IF(I2/2*2.EQ.I2) T=T+0.5*P4
      GO TO 999
C
C
  600 GO TO 400
C
C
  700 CONTINUE
      NELMT=(M-1)*I2+N
      S=SS(NELMT)
      T=TT(NELMT)
      GO TO 999
C
C
  990 CALL SPLOC(M,N,S,T)
  999 RETURN
      END
```

```
      SUBROUTINE CURREN(CURR,CURI,MCUR,NCUR,US,VS,CORCOF,IC)
C
C
C        THIS SUBROUTINE CALCULATES THE FINAL CURRENT DISTRIBUTION
C        NECESSARY TO PRODUCE THE FINAL PATTERN F(U,V).
C
C
C        DATE:  73/166   JUNE 15,1973.
C
C
      COMPLEX SOURCE,TEMP
      REAL CURR(51,51),CURI(51,51),US(500),VS(500),CORCOF(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 CURI(M,N)=0.
      DO 200 M=1,MCUR
      DO 200 N=1,NCUR
      DO 200 I=1,NORG
      TEMP=SOURCE(M,N,UORG(I),VORG(I),ITYPE)
      CURR(M,N)=CURR(M,N)+CORG(I)*REAL(TEMP)
      CURI(M,N)=CURI(M,N)+CORG(I)*AIMAG(TEMP)
  200 CONTINUE
      IF(IC.LE.0) RETURN
C
C
      DO 10 M=1,MCUR
      DO 10 N=1,NCUR
      DO 10 I=1,IC
      TEMP=SOURCE(M,N,US(I),VS(I),ITYPE)
      CURR(M,N)=CURR(M,N)+CORCOF(I)*REAL(TEMP)
      CURI(M,N)=CURI(M,N)+CORCOF(I)*AIMAG(TEMP)
   10 CONTINUE
C
      RETURN
      END
```



```
      SUBROUTINE PRINT(A,M,N,STARTU,STARTV,DU,DV)
C
C        SUBROUTINE PRINT IS THE GENERAL OUTPUT SUBROUTINE --
C        IT WILL PRINT OUT CO-ORDINATES (U,V) AND VALUES A(I,J)
C        10 ROWS AND 10 COLUMNS TO A PAGE.        72/266  VERSION 3
C
      DIMENSION A(51,51),U(51),V(51)
      WRITE(6,6969)
 6969 FORMAT(1H1)
      DO 10 J=1,51
      U(J)=STARTU+(J-1)*DU
   10 V(J)=STARTV+(J-1)*DV
      N2=N/10.+0.99
      M2=M/10.+0.99
C
```

```
      DO 100 J1=1,M2
      DO 200 K=1,N2
      M3=1+(J1-1)*10
      M4=M3+9
      IF(M4.GT.M) M4=M
      N3=1+(K-1)*10
      N4=N3+9
      IF(N4.GT.N) N4=N
C
C        PRINT OUT A HEADING
C
      WRITE(6,20) (V(I),I=N3,N4)
   20 FORMAT(1H1,16X,F6.3,9(4X,F6.3))
      WRITE(6,30)

C
C        PRINT A PAGE
C
      K2=(M4-M3+1)*6
      DO 4000 J=1,K2
      J2=J/6
      IF(J2*6-J) 27,28,27
   28 J3=J2+M3-1
      WRITE(6,29) U(J3),(A(J3,I),I=N3,N4)
      GO TO 4000
   27 WRITE(6,31)
 4000 CONTINUE
      IF(N4.EQ.N .AND. M4.EQ.M) GO TO 300
  200 CONTINUE
  100 CONTINUE
  300 RETURN
   29 FORMAT(3X,F6.3,' +',5X,10(F9.4,1X))
   30 FORMAT(10X,1H+,10(10H---------+))
   31 FORMAT(10X,'|')
      END


      SUBROUTINE PROFIL(DATA1,NPT,NUMPAT)
      INTEGER SF
      INTEGER OUTPUT(101)
      INTEGER BLANK,PLUS,SLASH,STAR
      REAL DATA(401,2),BOUND(101)
      REAL DATA1(401,2)
      DATA BLANK,PLUS,SLASH,STAR /' ','+','|','*' /
      DO 47 J=1,401
      DATA(J,1)=DATA1(J,1)
      DATA(J,2)=DATA1(J,2)
   47 CONTINUE
C
C        FIND THE RANGE OF DEPENDENT DATA AND SCALE IF NECESSARY
C
      IF(NPT.GT.600) GO TO 999
      BIG=-1.E10
      SMALL = 1.E10
      DO 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 = 0
      IF(DIFF.LT.1.) GO TO 10
      IF(DIFF.LT.100.)GO TO 21
      DO 2 J=1,10
      IF(DIFF*10.**(-J).GT.100.) GO TO 2
      SF=J
      GO TO 20
    2 CONTINUE
  400 WRITE(6,100)
  100 FORMAT('0 YOUR DATA IS TOO LARGE FOR THIS PROGRAM.')
      RETURN
   10 DO 3 J=1,10
      K=11-J
      IF(DIFF*10**K.GT.100.) GO TO 3
      SF=-K
      GO TO 20
    3 CONTINUE
      GO TO 400
   20 DO 4 J=1,NPT
    4 DATA(J,2) = DATA(J,2)*10.**(-SF)
C
C         CALCULATE BOUNDS
C
   21 SCALE=DIFF/100.
      DO 5 J=1,101
      K=J-1
    5 BOUND(J)=(BIG-K*SCALE)*10.**(-SF)
C
C         PRINT TITLE
C
      WRITE(6,640) NUMPAT
  640 FORMAT(26X,'PATTERN NUMBER ',I5//)
      IF (SF.EQ.0) GO TO 200
      WRITE(6,4004) SF
 4004 FORMAT(53X,'SCALE FACTOR IS 10**',I2/)
  200 WRITE(6,650) (BOUND(J),J=1,101,20)
  650 FORMAT(7X,5(F7.3,13X),F7.3,2X,'REAL',5X,'DB.')
      DO 6 J1=1,NPT
      J=NPT+1-J1
      DO 50 K=1,101
   50 OUTPUT(K)=BLANK
      IF((J-1)/10*10-(J-1)) 62,61,62
   61 DO 40 K=1,101,10
   40 OUTPUT(K)=PLUS
      GO TO 87
   62 OUTPUT(1)=SLASH
      OUTPUT(101) = SLASH
```

```
   87 DO 7 K=1,100
      IF(DATA(J,2).GT.BOUND(K)) GO TO 7
      IF(DATA(J,2).LE.BOUND(K+1)) GO TO 7
      OUTPUT(K)=STAR
      GO TO 69
    7 CONTINUE
      OUTPUT(101) = STAR
   69 IF(DATA(J,2).EQ.0.0) DATA(J,2) = 1.0E-6
      DATADB = 20.*ALOG10(ABS(DATA(J,2)))
      IF((J-1)/10*10-(J-1)) 140,141,140
  141 WRITE(6,4000) DATA(J,1),(OUTPUT(K),K=1,101),DATA(J,2),DATADB
 4000 FORMAT(1X,F8.3,1X,101A1,2X,F8.3,2X,F6.2)
      GO TO 6
  140 WRITE(6,4001) (OUTPUT(K),K=1,101),DATA(J,2),DATADB
 4001 FORMAT(10X,101A1,2X,F8.3,2X,F6.2)
    6 CONTINUE
      WRITE(6,650) (BOUND(J),J=1,101,20)
  999 RETURN
      END

      SUBROUTINE CONTUR(K,L,DCON,CLOW,CMAX,CINT,A,NUMPAT)
C***********************PRINT2*******************************************
C     THIS SUBPROGRAM GIVES A CONTOUR MAP OF THE MATRIX A
C     K AND L ARE THE MAXIMUM VALUES OF I AND J
C     IF K=L=51 OR 101 AXES WILL BE SET UP AS FOR A PATTERN PLOT
C     DELCON=DELTA(INCREMENT) BETWEEN CONTOURS FOR CONTUR 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)
C      DIMENSION COL AT LEAST L
      DIMENSION COL(101)
      DATA ALPHA/1H0,1H1,1H2,1H3,1H4,1H5,1H6,1H7,1H8,1H9/
      DATA BLANK,DOT/1H ,1H./
C
      IF(K.LE.1.OR.L.LE.1) RETURN
      CONINT=CINT
      CONMAX=CMAX
      CONLOW=CLOW
      DELCON=DCON
      WRITE(6,87) NUMPAT
 87   FORMAT(1H0,'FOR THE PATTERN NUMBERED',I5)
      IF(CONINT) 99,99,100
   99 BIG=-1.E27
      SMALL=1.E27
      DO 98 I=1,K
      DO 98 J=1,L
      IF(A(I,J) .GT. BIG) BIG=A(I,J)
      IF(A(I,J) .LT. SMALL) SMALL = A(I,J)
   98 CONTINUE
```

```
      CONINT=(BIG-SMALL)/10.
      DELCON=0.5*CONINT
      CONLOW=SMALL+DELCON
      CONMAX=BIG-DELCON
  100 WRITE(6,71) DELCON,CONLOW,CONMAX,CONINT
   71 FORMAT(1H0,     'DELCON=',F10.5,3X,'CONLOW=',F10.5,3X,'CONMAX=',
     1  F10.5,3X,'CONINT=',F10.5)
C     PRINT LEVEL DESIGNATIONS
      MCHAR=ABS((CONMAX-CONLOW)/CONINT+1.1)
      CON=CONMAX+CONINT
      DO 40 M=1,MCHAR
      ICON=M-1
      CON=CON-CONINT
      WRITE(6,72) ICON,CON
   72 FORMAT(1H0,'CONTOUR LEVEL ',I2,'=',F10.5)
   40 CONTINUE
C
C     WRITE HEADING
      DO 32 J=1,101
      COL(J)=BLANK
   32 CONTINUE
      IF(L.GT.51) GO TO 33
      DO 30 J=1,101,2
      COL(J)=DOT
   30 CONTINUE
      GO TO 34
   33 CONTINUE
      DO 35 J=1,101
      COL(J)=DOT
   35 CONTINUE
   34 CONTINUE
      WRITE(6,200)
  200 FORMAT(1H1)
      N1=L*2
      IF(N1.GT.101) N1=101
      WRITE(6,101) (COL(J1),J1=1,N1)
  101 FORMAT(/1H0,14X,101A1)
      DO 1 I=1, K
      DO 31 J=1,101
      COL(J)=BLANK
   31 CONTINUE
      J2=-1
      DO 2 J=1, L
      J2=J2+2
      ICON=-1
      CON=CONMAX+CONINT
      DO 50 M=1,MCHAR
      ICON=ICON+1
      CON=CON-CONINT
      IF(A(I,J).GT.(CON+DELCON)) GO TO 50
      IF(A(I,J).LT.(CON-DELCON)) GO TO 50
C     NOW A(I,J) IS LT CON+DELCON AND GT CON-DELCON
      IF(L.LE.51) COL(J2)=ALPHA(ICON+1)
      IF(L.GT.51) COL(J)=ALPHA(ICON+1)
      GO TO 2
```

```
 50   CONTINUE
 2    CONTINUE
      WRITE(6,140) (COL(J1),J1=1,101)
 140  FORMAT(1H ,13X,'.',101A1)
 1    CONTINUE
C
      RETURN
      END


      SUBROUTINE PATCON(RDATA,MMAX,NMAX,ICODE,CONLOW,CONMAX,CONINT,
     $STARTU,STARTV,DELTAU,DELTAV,NUMPAT,ISYMM)
C
      REAL RDATA(51,51),UAXIS(11),LOW(12),HIGH(12)
      INTEGER OUTPUT(101),LEVEL(12),BLANK
C
      DATA BLANK/' '/
      DATA LEVEL/'0','1','2','3','4','5','6','7','8','9','-','+'/
C
      CALL DATE(I,J,K)
      WRITE(6,10) I,J,K,NUMPAT
   10 FORMAT(1H1,'    PATTERN CONTOUR SUBPROGRAM',34X,'DATE = ',A2,'-',A2
     $,'-',A2,30X,'PATTERN NUMBER',I5//////)
      IF(ICODE.EQ.0) WRITE(6,11)
      IF(ICODE.EQ.1) WRITE(6,12)
      IF(ICODE.EQ.2) WRITE(6,13)
   11 FORMAT(42X,'CONTOUR PLOT OF THE DESIRED PATTERN        '/////)
   12 FORMAT(46X,'CONTOUR PLOT OF THE INITIAL PATTERN'/////)
   13 FORMAT(45X,'CONTOUR PLOT OF THE FINAL PATTERN IN DB.'/////)
C
      FINALU=STARTU+(MMAX-1)*DELTAU
      FINALV=STARTV+(NMAX-1)*DELTAV
      U1=STARTU
      U2=FINALU
      V1=STARTV
      V2=FINALV
      MCOUNT=MMAX
      NCOUNT=NMAX
C
      IF(ISYMM-1) 70,30,20
   20 UBIG=AMAX1(ABS(STARTU),ABS(FINALU))
      U1=-UBIG
      U2=UBIG
      MCOUNT=2*MCOUNT-1
      IF(ISYMM.EQ.2) GO TO 70
C
   30 VBIG=AMAX1(ABS(STARTV),ABS(FINALV))
      V1=-VBIG
      V2=VBIG
      NCOUNT=2*NCOUNT-1
C
   70 CONTINUE
C
C        ESTABLISH LOWER AND UPPER LIMITS
C
```

```
      NUMCON=(CONMAX-CONLOW)/CONINT+1.5
      DELCON=CONINT/2.
      DO 71 J=1,NUMCON
      LOW(J)=CONLOW+(J-1)*CONINT-DELCON
      HIGH(J)=LOW(J)+CONINT+0.00001
   71 CONTINUE
      LOW(11)=-1.E30
      HIGH(12)=1.E30
      HIGH(11)=LOW(1)
      LOW(12)=HIGH(NUMCON)
      MSKIP=100/(MCOUNT-1)
      NSKIP=100/(NCOUNT-1)
C
      DU=(U2-U1)/10.
      DO 40 I=1,11
   40 UAXIS(I)=U1+(I-1)*DU
      WRITE(6,42) (UAXIS(I),I=1,11)
   42 FORMAT(13X,11(F7.4,3X)/16X,11('.',9X))
C
      DU=(U2-U1)/100.
      DV=(V2-V1)/100.
      N1=NSKIP-1
      DO 50 N=1,101,NSKIP
      V=V2-(N-1)*DV
      DO 51 K=1,101
   51 OUTPUT(K)=BLANK
      DO 60 M=1,101,MSKIP
      U=U1+(M-1)*DU
      IF(U*U+V*V.GT.1.0) GO TO 60
C
C         FIND F(U,V)
C
      IJ=1
      IK=1
      J=(U-STARTU)/DELTAU+1.5
      K=(V-STARTV)/DELTAV+1.5
      IF(J.GE.1 .AND. J.LE.MMAX) IJ=0
      IF(K.GE.1 .AND. K.LE.NMAX)  IK=0
C
  101 IF(IJ) 200,102,200
  102 IF(IK) 300,1000,300
C
  200 IF(ISYMM-1) 60,60,201
  201 J=1.5-(U+STARTU)/DELTAU
      IF(J.GE.1 .AND. J.LE.MMAX) IJ=0
      IF(IJ) 60,202,60
  202 IF(IK) 300,1000,300
C
  300 IF(ISYMM.EQ.0 .OR. ISYMM.EQ.2) GO TO 60
      K=1.5-(V+STARTV)/DELTAV
      IF(K.GE.1 .AND. K.LE.NMAX) IK=0
      IF(IK) 60,1000,60
C
 1000 F=RDATA(J,K)
      IF(F.LE.LOW(1)) GO TO 1001
      IF(F.GT.HIGH(NUMCON)) GO TO 1002
C
```

```
      DO 61 K=1,NUMCON
      IF(F.GT.LOW(K) .AND. F.LE.HIGH(K)) GO TO 62
   61 CONTINUE
 1002 OUTPUT(M)=LEVEL(12)
      GO TO 60
 1001 OUTPUT(M)=LEVEL(11)
      GO TO 60
   62 OUTPUT(M)=LEVEL(K)
C
   60 CONTINUE
      WRITE(6,64) V,(OUTPUT(K),K=1,101),V
   64 FORMAT(7X,F7.4,1X,'.',101A1,'.',1X,F7.4)
      IF(N1.EQ.0) GO TO 50
      DO 55 K=1,N1
      WRITE(6,56)
   56 FORMAT('             ')
   55 CONTINUE
C
   50 CONTINUE
      WRITE(6,43) (UAXIS(I),I=1,11)
   43 FORMAT(16X,11('.',9X)/13X,11(F7.4,3X))
C
      WRITE(6,44)
   44 FORMAT(///56X,'CONTOUR LEVEL KEY'//)
      DO 45 I=1,4
   45 WRITE(6,46) (LEVEL(J),LOW(J),HIGH(J),J=I,12,4)
   46 FORMAT(5X,3(A1,':  ',E14.7,' TO ',E14.7,4X))
      RETURN
      END


      SUBROUTINE LIST(CURR,CURI,MCUR,NCUR)
C
C        THIS SUBROUTINE LISTS ARRAY ELEMENT COORDINATES AND CURRENTS
C
C        DATA WRITTEN:  73-192   --    JULY 11,1973.
C
C
      DIMENSION CURR(51,51),CURI(51,51)
      DO 10 M=1,MCUR
      DO 10 N=1,NCUR
      J=(M-1)*NCUR+N
      CALL LOCSOR(M,N,S,T)
      WRITE(6,100) J,S,T,CURR(M,N),CURI(M,N)
   10 CONTINUE
  100 FORMAT(3X,I4,5X,4(E14.7,2X))
      RETURN
      END
```

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, LOCSOR, 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.

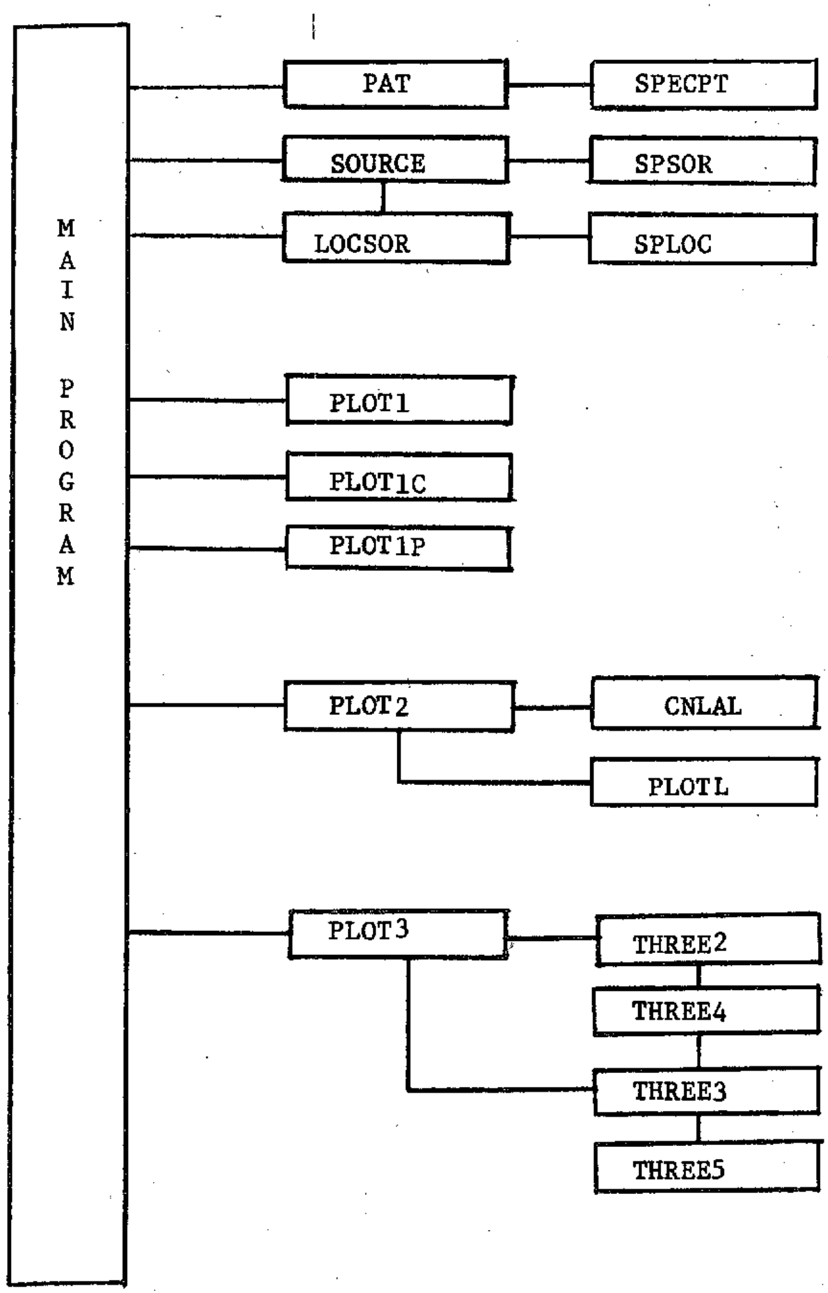


Figure 7.1 Block diagram of ANTDATA program.

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 14I5 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 14I5 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, NORG, IC, (UORG(J), VORG(J), CORG(J), J=1, ITEMP), (US(J), VS(J), CORCOF(J), J=1, ITEMP1), ITYPE, P1, P2, P3, P4, P5, P6, PI, (SS(J), TT(J), J=1, 400), I1, I2, I3, I4, I5, MCUR, NCUR

Refer to the statement listing of subroutine INPUT of the ANTSYN program for a meaning of P1, P2,... and I1, I2,..... These vary with ITYPE.

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

Step 5. Options for pattern magnitude plots.

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

Step 6. U profile location.

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

Step 7. V profile location

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

Step 8. Parameters for PLOT2 and PLOT3 of pattern.

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

Step 9. Pattern contour parameters.

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

Step 10. Options for current magnitude plots.

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

Step 11. S profile location.

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

Step 12. T profile location.

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

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

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

Step 14. Current magnitude contour parameters.

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 OP1U, OPT1V, OPT2, and OPT3 from a card under a 4I1 format. Use zeros for no plot and ones for plot. OPT1U and OPT1V now refer to 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.

Step 17. Current phase contour parameters.

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 and/or the current phase in the S direction. If it is 1 a plot is made, otherwise no plot is made.

OP1V = 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, SPSOR, 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:

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

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

COMMON Blocks Required: COMMON /PLT1/ PTS

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

SUBROUTINE PLOT1C

Purpose:

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

Usage:

CALL PLOT1C (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 ($-180 \leq PTS \leq 180$).

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: CNTROL, 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 suface.

Description of Parameters and Important Variables:

N - Number of data points along first axis.

M - Number of data points along the second axis.

NUMPAT - Pattern number (for labeling)

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

SDISTS - Distance from surface to eye when perspective is calculated -- SKISTS > 6 usually won't show any distortion due to PARALLAX.

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

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

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

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

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

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

Remarks:

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

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

COMMON Blocks Required:

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

Subroutine and Function Subprograms Required: THREE2, THREE3, THREE3, THREE4,

THREE5, PLOT, FACTOR, SYMBOL, NUMBER.

Reference: Howard Jesperson, Iowa State University.

SUBROUTINE THREE2

Purpose:

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

Usage:

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

Description of Parameters:

(X,Y,Z) - Vectors of length 2. Position of rotated vertices.

XP - Height above paper.

(H,V) - Vectors of length 10. Location of projected vertices on paper.

KODE - Dummy variable

COMMON Blocks Required: None

Subroutine and Function Subprograms Required: THREE4

SUBROUTINE THREE4

Purpose:

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

Usage:

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

Description of Parameters:

(X,Y,Z) - Coordinates of point to be located.

XP Height above paper of point.

(YP,ZP) Coordinates of projection on paper.

COMMON Blocks Required:

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

Subroutine and Function Subprograms Required: None.

SUBROUTINE THREE3

Purpose:

To plot a perspective of a three-dimensional figure.

Usage:

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

Description of Parameters:

X - Vector of length 2

Y - Vector of length 2

N - Number of points in first direction

M - Number of points in second direction

H,V - Vectors of length 10...Coordinates of projected vertices of cube.

K - Grid Line Code (See Subroutine PLOT3)

KODE - Hidden Line Switch (See Subroutine PLOT3)

COMMON Blocks Required:

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

Subroutine and Function Subprograms Required: THREE4, THREE5, PLOT

SUBROUTINE THREE5

Purpose:

To see if a point on the projected three-dimensional figure is visible.

Usage:

CALL THREE5(XI, YJ, M, N, 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

Subroutine and Function Subprograms Required: None.

7.5.2 Virginia Tech Subroutines

VPI UTILITY SUBPROGRAMS

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

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

VPI PLOTTER SUBROUTINES

Subroutine	Purpose
AXIS	To draw a labeled axis of a desired length with annotated tic marks every inch.

FACTOR	To scale the plot in both the X and Y directions.
NUMBER	To draw a floating point number.
PLOT	To move the pen from one point to another, to draw a line between points, to establish a new origin, and to signal the end of a plot.
SYMBOL	To plot a string of alphanumeric characters.

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

INTEGER FUNCTION ICVT

Purpose:

To convert an integer to character format internal coding.

Usage:

```
ICHAR=ICVT(NUM)
```

Remark:

This function was originally written in assembler. Object deck is read in under the SYSLIN dataset.

7.6 Statement Listing of ANTDATA

7.6.1 Job Control Language

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

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

7.6.2 Source Listing

Listed below are the FORTRAN IV statements of the ANTDATA program.

```
C          PROGRAM DESCRIPTION:
C          ANTDATA IS THE OUTPUT PROGRAM USED IN CONJUNCTION WITH ANTSYN2.
C          BY SPECIFYING APPROPRIATE PARAMETERS ANTDATA WILL GIVE A ONE,
C          TWO, OR THREE DIMENSION PLOT OF THE PATTERN (IN DB.), THE SOURC
C          MAGNITUDE, AND THE SOURCE PHASE.
C
C
C          INPUT/OUTPUT:
C          THE MAJORITY OF INPUT IS TAKEN FROM ANTSYN2 VIA DIRECT-ACCESS
C          UNIT 22 (ANTDATA.A507C2).  PARAMETERS AND JOB OPTIONS ARE SUPPL
C          THROUGH UNIT 5 (SYSIN).  ALL OUTPUT IS CHANNELLED TO UNIT 6
C          (SYSPRINT) AND THE PLOTTER (PLOT1).
C
C
C          VERSION 1 73 - 094 -- APRIL 5, 1973
C
C
C          WRITTEN BY: W. L. STUTZMAN
C                      S. R. KAUFFMAN
C                      E. L. COFFEY
C
C
C          UNDER NASA GRANT: 47-004-103
C
C
C          ADDITIONAL SUBPROGRAMS REQUIRED:
C
C          ICVT   -- ASSEMBLER LANGUAGE SUBPROGRAM TO CONVERT AN I2 INTEGE
C                    TO A2 CHARACTER FORMAT.
C
C
C
C
      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 AU(151),AV(151)
      INTEGER TITLE(20)
      REAL INITLS,INITLT
      REAL LOWCON
      INTEGER OPT1U,OPT1V,OPT2,OPT3,PX,PY
      COMPLEX CTEMP,CI
      COMMON /PLT1/ PTS
      COMMON /ARRAY/ A
      COMMON /PAT1/ P1,P2,P3,P4,P5,P6,PI,SS(400),TT(400),RR(400)
      COMMON /PAT2/ I1,I2,I3,I4,I5
      COMMON /LOC/ ITYPE
      IPAGE=0
      PI=3.14159265
      CI=CMPLX(0.0,1.0)
      CALL STIME(ITIME)
      TIME=0.
 9999 CONTINUE
```

```
      CALL TIMEON
      IPAGE=IPAGE+1
      CALL DATE(I1,J1,K1)
      CALL STIME(IT)
      IHR=IT/10000
      IFR=IT-IHR*10000
      FHR=IFR/10000.
      FM=FHR*60.
      IMIN=FM
      ISEC=(FM-IMIN)*60
      IHR=ICVT(IHR)
      IMIN=ICVT(IMIN)
      ISEC=ICVT(ISEC)
      IPG=ICVT(IPAGE)
      WRITE(6,1) I1,J1,K1,IHR,IMIN,ISEC,IPG
    1 FORMAT(1H1,2X,'ANTDATA I  VERSION 1  LEVEL 2',
     $8X,'VPI EE DEPT.',5X,'DATE = ',A2,'-',A2,'-',A2,
     $5X,'TIME = ',A2,'.',A2,'.',A2,10X,'PAGE 00',A2 ///)
C
C        READ PATTERN NUMBER AND TRACK -- VERIFY THAT PATTER IS ACTUALLY
C        STORED
C
      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
   20 FORMAT(A4)
      IF(NUM.EQ.NUMPAT) GO TO 51
      DO 30 I=2,25
      READ(22'I,20) NUM
      IF(NUM.EQ.NUMPAT) GO TO 50
   30 CONTINUE
C
C        NUMPAT IS NOT ON DISK
C
      WRITE(6,40) NUMPAT
   40 FORMAT(1H0,'PATTERN NUMBER',I5,' WAS NOT LOCATED -- PROGRAM HALT')
      GO TO 999
C
C        NUMPAT FOUND ON UNEXPECTED TRACK
C
   50 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
C
C        BEGIN PROCESSING
C
      READ(5,10) MMAX,NMAX
      READ(22'NUMTRK,70) ITEMP,ITEMP1
   70 FORMAT(104X,2A4)
```

```
      READ(22'NUMTRK,101) NUMPAT,TITLE,ISYMM,ITER,ISUC,FNORM,IDISK,
     $NORG,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
  101 FORMAT(75A4,11(200A4))
C
C         READ OPTIONS FOR PATTERN MAGNITUDE
C
      READ(5,29) OPT1U,OPT1V,OPT2,OPT3
   29 FORMAT(4I1)
      IF(OPT1U-1) 80,81,80
   81 CONTINUE
      READ(5,31) CONST
      IF(MMAX.LE.1) GO TO 80
      DO 90 J=1,4001
      U=(J-1)*0.0005-1.0
      SUM=0.
      DO 911 K=1,NORG
  911 SUM=SUM+CORG(K)*PAT(U-UORG(K),CONST-VORG(K),ITYPE)
      IF(IC.LE.0) GO TO 90
      DO 91 K=1,IC
      SUM=SUM+CORCOF(K)*PAT(U-US(K),CONST-VS(K),ITYPE)
   91 CONTINUE
   90 PTS(J)=20.*ALOG10(ABS(SUM)       )
      WRITE(6,92) CONST
   92 FORMAT('0U-AXIS PROFILE PLOT REQUESTED -- V = ',F6.3)
      CALL PLOT1(-1.0,1.0,4001,2,CONST,NUMPAT)
   80 IF(OPT1V-1) 82,83,82
   83 CONTINUE
      READ(5,31) CONST
      IF(NMAX.LE.1) GO TO 82
      DO 900 J=1,4001
      V=(J-1)*0.0005-1.0
      SUM=0.
      DO 909 K=1,NORG
  909 SUM=SUM+CORG(K)*PAT(CONST-UORG(K),V-VORG(K),ITYPE)
      IF(IC.LE.0) GO TO 900
      DO 910 K=1,IC
  910 SUM=SUM+CORCOF(K)*PAT(CONST-US(K),V-VS(K),ITYPE)
  900 PTS(J)=20.*ALOG10(ABS(SUM)       )
      WRITE(6,93) CONST
   93 FORMAT('0V-AXIS PROFILE PLOT REQUESTED -- U = ',F6.3)
      CALL PLOT1(-1.0,1.0,4001,1,CONST,NUMPAT)
   82 IF(OPT2+OPT3) 85,85,84
   84 CONTINUE
C
C         GENERATE PATTERN ARRAY
C
      READ(5,31) LOWCON,DASH
      IF(MMAX.LE.1 .OR. NMAX.LE.1) GO TO 239
      DELTAU=2.0/(MMAX-1)
      DELTAV=2.0/(NMAX-1)
      WRITE(6,701) LOWCON,LOWCON
  701 FORMAT('0PATTERN IS NOW BEING GENERATED.  IF PATTERN < ',F7.2,
     $' PATTERN = ',F7.2)
      IF(ITYPE .GT. 5) GO TO 5000
```

```
C
C        LOAD UP AU AND AV.
C
      DO 2000 I=1,MMAX
      U=(I-1)*DELTAU
 2000 AU(I)=PAT(U,0.,ITYPE)
      DO 2010 J=1,NMAX
      V=(J-1)*DELTAV
 2010 AV(J)=PAT(0.,V,ITYPE)
C
C        BEGIN
C
      U=-1.0-DELTAU
      DO 2020 M=1,MMAX
      U=U+DELTAU
      V=-1.0-DELTAV
      DO 2020 N=1,NMAX
      V=V+DELTAV
      TEMP=0.
      DO 2030 K=1,NORG
      I=ABS(U-UORG(K))/DELTAU+1.5
      J=ABS(V-VORG(K))/DELTAV+1.5
 2030 TEMP=TEMP+CORG(K)*AU(I)*AV(J)
      IF(IC.LE.0) GO TO 2020
      DO 2040 K=1,IC
      I=ABS(U-US(K))/DELTAU+1.5
      J=ABS(V-VS(K))/DELTAV+1.5
 2040 TEMP=TEMP+CORCOF(K)*AU(I)*AV(J)
 2020 A(M,N)=20.*ALOG10(       ABS(TEMP))
      GO TO 239
 5000 CONTINUE
      DO 200 M=1,MMAX
      U=-1.0+(M-1)*DELTAU
      DO 201 N=1,NMAX
      V=-1.0+(N-1)*DELTAV
      TEMP=0.
      DO 242 I=1,NORG
  242 TEMP=TEMP+CORG(I)*PAT(U-UORG(I),V-VORG(I),ITYPE)
      IF(IC.LE.0) GO TO 2021

      DO 202 I=1,IC
  202 TEMP=TEMP+CORCOF(I)*PAT(U-US(I),V-VS(I),ITYPE)
 2021 CONTINUE

      A(M,N)=20.*ALOG10(       ABS(TEMP))
  201 CONTINUE
  200 CONTINUE
  239 CONTINUE
      IF(OPT2) 210,210,211
  211 READ(5,31) CONLOW,CONMAX,CONINT
   31 FORMAT(8F10.0)
      IF(MMAX.LE.1 .OR. NMAX.LE.1) GO TO 230
      DO 257 M=1,MMAX
      DO 257 N=1,NMAX
      IF(A(M,N) .LT. LOWCON) A(M,N)=LOWCON
  257 CONTINUE
      WRITE(6,220) CONLOW,CONMAX,CONINT
```

```
  220 FORMAT('0CONTOUR PLOT OF PATTERN REQUESTED'/
     $'        LOWEST CONTOUR   = ',F7.2/
     $'        HIGHEST CONTOUR  = ',F7.2/
     $'        CONTOUR INTERVAL = ',F7.2)
      CALL PLOT2(MMAX,NMAX,CONLOW,CONMAX,CONINT,NUMPAT,DASH)
  210 IF(OPT3) 230,230,231
  231 WRITE(6,240)
  240 FORMAT(1H0,'THREE - DIMENSIONAL PLOT OF PATTERN REQUESTED')
      CALL PLOT3(MMAX,NMAX,NUMPAT)
  230 CONTINUE
      IF(MMAX.LE.1.OR.NMAX.LE.1)  WRITE(6,23)
   23 FORMAT('0        TWO AND THREE DIM. PLOTS CANCELLED SINCE SOURCE IS
     $ONE DIMENSIONAL')
   65 CONTINUE
C
C          END OF PATTERN
C
C
      IA=0
      IF(ITYPE.EQ.1) GO TO 401
      IF(ITYPE.EQ.3) GO TO 401
      IF(ITYPE.EQ.4) GO TO 401
      IF(ITYPE.EQ.6) GO TO 401
  400 WRITE(6,402)
  402 FORMAT(1H0,'THE SOURCE IS AN ARRAY -- THIS PGM IS FOR CONTINOUS SO
     $URCES ONLY')
      IA=1
  401 CONTINUE
      P3TEMP=P3
      P5TEMP=P5
      P6TEMP=P6
      IF(ITYPE-1) 404,403,404
  404 IF(ITYPE-3) 405,403,405
  403 CONTINUE
C       ITYPE= 1 OR 3
      INITLS=0.
      DELTAS=0.
      FINALS=0.
      INITLT=P2
      FINALT=P2+P1
      P3=P1/4000.
      DELTAT=P3
      GO TO 410
  405 IF(ITYPE-4) 407,406,407
  406 CONTINUE
C         ITYPE=4
      INITLS=P3
      FINALS=P3+P1
      INITLT=P4
      FINALT=P4+P2
      P5=P2/4000.
      P6=P2/4000.
      DELTAT=P6
      DELTAS=P5
      GO TO 410
  407 CONTINUE
      IF(ITYPE-6) 410,409,410
```

```
  409 INITLS=P3
      FINALS=P3+2.*P1
      INITLT=P4
      FINALT=P4+2.*P1
      P5=P1/2000.
      P6=P1/2000.
      DELTAT=P6
      DELTAS=P5
  410 CONTINUE
      READ(5,29) OPT1U,OPT1V,OPT2,OPT3
      IF(OPT1U-1) 302,301,302
  301 CONTINUE
      READ(5,31) CONST
      IF(IA.EQ.1) GO TO 3000
      IF(NMAX.LE.1) GO TO 302
      J=1
      IF(DELTAT.NE.0.) J=1.5+(CONST-INITLT)/DELTAT
      DO 303 I=1,4001
      CTEMP=(0.0,0.0)
      IF(NORG.LE.0) GO TO 304
      DO 305 K=1,NORG
  305 CTEMP=CTEMP+CORG(K)*SOURCE(I,J,UORG(K),VORG(K),ITYPE)
  304 IF(IC.LE.0) GO TO 303
      DO 306 K=1,IC
  306 CTEMP=CTEMP+CORCOF(K)*SOURCE(I,J,US(K),VS(K),ITYPE)
  303 PTS(I)=CABS(CTEMP)
      WRITE(6,307) CONST
  307 FORMAT('0S-AXIS PROFILE PLOT REQUESTED --  T = ',F6.3)
      CALL PLOT1C(INITLS,FINALS,4001,2,CONST,NUMPAT)
  302 CONTINUE
      IF(OPT1V-1) 311,310,311
  310 CONTINUE
      READ(5,31) CONST
      IF(IA.EQ.1) GO TO 3000
      IF(NMAX.LE.1) GO TO 322
      I=1
      IF(DELTAS.NE.0.0) I=1.5+(CONST-INITLS)/DELTAS
      DO 313 J=1,4001
      CTEMP=(0.0,0.0)
      IF(NORG.LE.0) GO TO 314
      DO 315 K=1,NORG
  315 CTEMP=CTEMP+CORG(K)*SOURCE(I,J,UORG(K),VORG(K),ITYPE)
  314 IF(IC.LE.0) GO TO 313
      DO 316 K=1,IC
  316 CTEMP=CTEMP+CORCOF(K)*SOURCE(I,J,US(K),VS(K),ITYPE)
  313 PTS(J)=CABS(CTEMP)
      WRITE(6,317) CONST
  317 FORMAT('0T-AXIS PROFILE PLOT REQUESTED  --   S = ',F6.3)
      CALL PLOT1C(INITLT,FINALT,4001,1,CONST,NUMPAT)
  311 CONTINUE
  322 CONTINUE
 3000 CONTINUE
      P3=P3TEMP
      P5=P5TEMP
      P6=P6TEMP
      MCUR=51
      NCUR=51
```

```
      IF(OPT2+OPT3) 320,320,321
  321 CONTINUE
      READ(5,31) LOWCON,DASH
      IF(IA.EQ.1) GO TO 333
C
C        GENERATE CURRENT MAGNITUDE ARRAY
C
      DO 330 M=1,MCUR
      DO 331 N=1,NCUR
      CALL LOCSOR(M,N,S,T)
      CTEMP=0.
      DO 339 K=1,NORG
  339 CTEMP=CTEMP+CORG(K)*SOURCE(M,N,UORG(K),VORG(K),ITYPE)
      DO 332 K=1,IC
  332 CTEMP=CTEMP+CORCOF(K)*SOURCE(M,N,US(K),VS(K),ITYPE)
      A(M,N)=       CABS(CTEMP)
  331 CONTINUE
  330 CONTINUE
  333 CONTINUE
      IF(OPT2) 350,350,351
  351 READ(5,31) CONLOW,CONMAX,CONINT
      IF(IA.EQ.1) GO TO 360
      IF(MMAX.LE.1.OR.NMAX.LE.1) GO TO 360
      WRITE(6,340) CONLOW,CONMAX,CONINT
  340 FORMAT('CCONTOUR PLOT OF CURRENT MAGNITUDE REQUESTED'/
     $'      LOWEST CONTOUR  = ',F7.4/
     $'      HIGHEST CONTOUR  = ',F7.4/
     $'      CONTOUR INTERVAL = ',F7.4)
      CALL PLOT2 (MCUR,NCUR,CONLOW,CONMAX,CONINT,NUMPAT,DASH)
  350 IF(OPT3) 360,360,361
  361 IF(IA.EQ.1) GO TO 360
      WRITE(6,355)
  355 FORMAT(1H0,'THREE DIMENSION PLOT OF CURRENT MAGNITUDE REQUESTED')
      CALL PLOT3 (MCUR,NCUR,NUMPAT)
  360 CONTINUE
      IF(MMAX.LE.1.OR.NMAX.LE.1)  WRITE(6,23)
  320 CONTINUE
C
C        END OF CURRENT MAGNITUDE
C
C
C        READ OPTIONS FOR CURRENT PHASE
C
      READ(5,29) OPT1U,OPT1V,OPT2,OPT3
      IF(OPT2+OPT3) 520,520,521
  521 CONTINUE
      IF(IA.EQ.1) GO TO 533
      READ(5,31) LOWCON,DASH
C
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)
```

```
  549 CONTINUE
      DO 532 K=1,IC
  532 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
  531 CONTINUE
  530 CONTINUE
  533 CONTINUE
      IF(OPT2) 550,550,551
  551 READ(5,31) CONLOW,CONMAX,CONINT
      IF(IA.EQ.1) GO TO 560
      IF(MMAX.LE.1.OR.NMAX.LE.1) GO TO 560
      WRITE(6,540) CONLOW,CONMAX,CONINT
  540 FORMAT('0CONTOUR PLOT OF CURRENT PHASE REQUESTED '/
     $'      LOWEST CONTOUR   = ',F7.2/
     $'      HIGHEST CONTOUR  = ',F7.2/
     $'      CONTOUR INTERVAL = ',F7.2)
      CALL PLOT2 (MCUR,NCUR,CONLOW,CONMAX,CONINT,NUMPAT,DASH)
  550 IF(OPT3) 560,560,561
  561 IF(IA.EQ.1) GO TO 560
      WRITE(6,555)
  555 FORMAT('0THREE DIMENSION PLOT OF CURRENT PHASE REQUESTED')
      CALL PLOT3(MCUR,NCUR,NUMPAT)
  560 CONTINUE
      IF(MMAX.LE.1.OR.NMAX.LE.1)  WRITE(6,23)
  520 CONTINUE
      CALL TIMECK(ISEC)
      FMIN=ISEC/6000.
      WRITE(6,897) FMIN
  897 FORMAT('0EXECUTION TIME: ',F6.2,' MINUTES.')
      TIME=TIME+FMIN
      GO TO 9999
  999 WRITE(6,600)
  600 FORMAT(1H1,'*** END OF EXECUTION ***' //)
      CALL PLOT(0.0,0.0,-4)
      WRITE(6,898) TIME
  898 FORMAT('0TOTAL EXECUTION TIME: ',F7.2,' MINUTES.')
      CALL STIME(JTIME)
      IT=JTIME-ITIME
      FMIN=IT/10000.*60.
      WRITE(6,899) FMIN
  899 FORMAT('0TOTAL ELAPSED TIME:    ',F7.2,' MINUTES.')
      STOP
      END


      SUBROUTINE PLOT1(PSTRT,PEND,IP,CODE,CONST,NUMPAT)
C
C
C        SUBROUTINE PLOT1
C
C
C        WRITTEN BY: S. R. KAUFMAN
C
C
C        DATE:  73-113   APRIL 23,1973
```

```
C
C
C        INPUT:
C
C        PSTRT   -- BEGINNING OF PLOT
C        PEND    -- END OF PLOT
C        IP      -- NUMBER OF POINTS TO BE PLOTTED
C        CODE    -- LABELLING VARIABLE.  IF CODE=0:  LABEL='THETA = ';
C                   IF CODE=1:  LABEL='U = '; IF CODE = 2: LABEL = 'T = '
C        CONST   -- CONSTANT PARAMETER FOR LABEL
C        NUMPAT  -- NUMBER OF PATTERN FOR LABEL.
C
C
C
      INTEGER NAME(2),CODE
      DIMENSION PTS(4001)
      COMMON /PLT1/ PTS
      CALL FACTOR(0.5)
      CALL PLOT(8.,1.,-3)
      IF(CODE.GT.0) GO TO 3
      CALL SYMBOL(-1.2,-.6,.2,8HTHETA = ,0.,8)
      CALL NUMBER(.3,-.8,.2,CONST,0.,3)
      GO TO 6
    3 IF(CODE.GT.1) GO TO 4
      CALL SYMBOL(-1.,-.8,.2,1HU,0.,1)
      CALL SYMBOL(-.9,-.8,.2,3H = ,0.,3)
      CALL NUMBER(-.2,-.8,.2,CONST,0.,3)
      CALL SYMBOL(-2.6,-.4,.2,2H-V,0.,2)
      CALL SYMBOL(2.4,-.4,.2,2H+V,0.,2)
      GO TO 6
    4 IF(CODE.GT.2) GO TO 5
      CALL SYMBOL(-1.,-.8,.2,1HV,0.,1)
      CALL SYMBOL(-.9,-.8,.2,3H = ,0.,3)
      CALL NUMBER(-.2,-.8,.2,CONST,0.,3)
      CALL SYMBOL(-2.6,-.4,.2,2H-U,0.,2)
      CALL SYMBOL(2.4,-.4,.2,2H+U,0.,2)
    6 CONTINUE
      PDEL=(PSTRT-PEND)/IP
      PTIC=((ABS(PSTRT-PEND))/10.)
      CALL AXIS(-5.,0.,1H ,1,4.,0.,PSTRT,PTIC)
      PSTRE=PSTRT+(6.*PTIC)+.00001
      PTIC2=PTIC+0.00001
      CALL AXIS(1.,0.,1H ,1,4.,0.,PSTRE,PTIC2)
      CALL PLOT(-1.,0.,3)
      CALL PLOT(1.,0.,2)
      CALL PLOT(0.,0.,3)
      CALL PLOT(0.,5.8,2)
      CALL PLOT(0.,0.,3)
      CALL SYMBOL(-.05,-.4,.2,1H0,0.,1)
      X=0.05
      DO 10 J=1,6
      Y=0.5+(J-1)*1.0
      CALL PLOT(-X,Y,3)
   10 CALL PLOT(X,Y,2)
      CALL PLOT(0.0,0.0,3)
      IF(PTS(1).LE.-50.) PTS(1)=-50.
      FS=((PTS(1))/10.)+5.5
```

```
      CALL PLOT(-5.,FS,3)
      DO 1 IW1=1,IP
      THETS=((PSTRT-(IW1*PDEL))*5.)/(ABS(PSTRT))
      FOBS=((PTS(IW1))/10.)+5.5
      IF(FOBS.LT.0.5) GO TO 1
      CALL PLOT(THETS,FOBS,2)
    1 CONTINUE
      CALL SYMBOL(-5.0,-0.8,0.125,10HPATTERN = ,0.,10)
      FNUM=FLOAT(NUMPAT)
      CALL NUMBER(-3.87,-0.8,0.125,FNUM,0.,-1)
      CALL AXIS(-5.5,0.5,17HFAR FIELD PATTERN,17,5.0,90.,-50.,10.)
      CALL PLOT(8.,-1.,-3)
    5 RETURN
      END


      SUBROUTINE PLOT1C(PSTRT,PEND,IP,CODE,CONST,NUMPAT)
C
C
C         SUBROUTINE PLOT1C
C
C
C         WRITTEN BY: S. R. KAUFMAN
C
C
C         DATE:  73-113    APRIL 24,1973
C
C
C         INPUT:
C
C         PSTRT   -- BEGINNING OF PLOT
C         PEND    -- END OF PLOT
C         IP       -- NUMBER OF POINTS TO BE PLOTTED
C         CODE    -- LABELLING VARIABLE.  IF CODE=0: LABEL IS 'THETA = ';
C                    IF CODE=1: LABEL IS 'S = '; IF CODE=2: LABEL IS 'T = '
C         CONST   -- CONSTANT PARAMETER FOR LABEL.
C         NUMPAT  -- PATTERN NUMBER FOR LABEL.
C
C
      INTEGER NAME(2),CODE
      CALL FACTOR(0.5)
      CALL PLOT(8.,1.,-3)
      DIMENSION PTS(4001)

      COMMON /PLT1/ PTS

      I=0
      IF(CODE.GT.0) GO TO 3
      CALL SYMBOL(-1.2,-0.6,0.2,8HTHETA = ,0.0,8)
      CALL NUMBER(0.3,-0.8,0.2,CONST,0.0,3)
      GO TO 6
    3 IF(CODE.GT.1) GO TO 4
      CALL SYMBOL(-1.0,-0.8,0.2,1HS,0.0,1)
      CALL SYMBOL(-0.9,-0.8,0.2,3H = ,0.0,3)
      CALL NUMBER(-0.2,-0.8,0.2,CONST,0.0,3)
      CALL SYMBOL(-2.6,-0.4,0.2,2H-T,0.0,2)
      CALL SYMBOL(2.4,-0.4,0.2,2H+T,0.0,2)
```

```
      GO TO 6
    4 IF(CODE.GT.2) GO TO 5
      CALL SYMBOL(-1.0,-0.8,0.2,1HT,0.0,1)
      CALL SYMBOL(-0.9,-0.8,0.2,3H = ,0.0,3)
      CALL NUMBER(-0.2,-0.8,0.2,CONST,0.0,3)
      CALL SYMBOL(-2.6,-0.4,0.2,2H-5,0.0,2)
      CALL SYMBOL(2.4,-0.4,0.2,2H+S,0.0,2)
    6 CONTINUE
      PDEL=(PSTRT-PEND)/IP
      PTIC=((ABS(PSTRT-PEND))/10.0)
      CALL AXIS(-5.0,0.0,1H ,1,4.0,0.0,PSTRT,PTIC)
      PSTRE=PSTRT+(6.0*PTIC)+0.00001
      PTIC2=PTIC+0.00001
      CALL AXIS(1.0,0.0,1H ,1,4.0,0.0,PSTRE,PTIC2)
      CALL PLOT(-1.0,0.0,3)
      CALL PLOT(1.0,0.0,2)
      CALL PLOT (0.0,0.0,3)
      CALL PLOT(0.0,5.8,2)
      CALL PLOT(0.0,0.0,3)
      CALL SYMBOL(-0.05,-0.4,0.2,1H0,0.0,1)
      X=0.05
      DO 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.,3)
      PSTRS=5.0*PSTRT
      GMAX=0.0
      DO 1IW1=1,IP
      IF(PTS(IW1).GT.GMAX) GMAX=PTS(IW1)
    1 CONTINUE
      IF(GMAX.GT.0.5) ASCLE=1.
      IF(GMAX.LE.0.5)  ASCLE=0.5
      IF(GMAX.LE.0.2)  ASCLE=0.2
      IF(GMAX.LE.0.1) ASCLE=0.1
      IF(GMAX.LE.0.05) ASCLE=0.05
      PTSA=((PTS(1))/ASCLE)*5.+0.5
      CALL PLOT(-5.0,PTSA,3)
      DO 7 IW1=1,IP
      THETA=(PSTRT-(IW1*PDEL))
      THETS=(THETA/(ABS(PSTRT)))*5.
      APTS=((PTS(IW1))/ASCLE)*5.+0.5
      CALL PLOT(THETS,APTS,2)
    7 CONTINUE
      IF(GMAX.GT.0.5)  ATIC=0.2+0.0001
      IF(GMAX.LE.0.5)  ATIC=0.1+0.0001
      IF(GMAX.LE.0.20) ATIC=0.04+0.0001
      IF(GMAX.LE.0.1) ATIC=0.02+0.0001
      IF(GMAX.LE.0.05) ATIC=0.01+0.0001
      CALL AXIS(-5.5,0.5,16HSOURCE MAGNITUDE,16,5.0,90.0,0.0,ATIC)
      CALL SYMBOL(-5.0,-0.8,0.125,10HPATTERN = ,0.,10)
      FNUM=FLOAT(NUMPAT)
      CALL NUMBER(-3.5,-0.8,0.125,FNUM,0. ,-1)
      CALL PLOT(8.,-1.,-3)
    5 RETURN
      END
```

```
      SUBROUTINE PLOT1P(PSTRT,PEND,IP,CODE,CONST,NUMPAT)
C
C
C        SUBROUTINE PLOT1P
C
C        WRITTEN BY:  S. R. KAUFMAN
C
C
C        DATE:  73-113   APRIL 24,1973
C
C
C       INPUT:
C
C        PSTRT  -- BEGINNING OF PLOT
C        PEND    -- END OF PLOT
C        IP      -- NUMBER OF POINTS TO BE PLOTTED  (IP < 4002)
C        CODE    -- LABELLING PARAMETER.  IF CODE = 0: LABEL IS
C                  'THETA = '; IF CODE = 1: LABEL IS 'S = '; IF CODE = 2
C                  LABEL = 'T = '.
C        CONST   -- CONSTANT PARAMETER IN LABEL.
C        NUMPAT  -- PATTERN NUMBER FOR LABEL.
C
C
C
      INTEGER NAME(2),CODE
      DIMENSION PTS(4001)
      COMMON /PLT1/PTS
      CALL FACTOR(0.5)
      CALL PLOT(8.,1.,-3)
      IF(CODE.GT.0) GO TO 3
      CALL SYMBOL(-1.2,-6.,.2,8HTHETA = ,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,.2,1HS,0.,1)
      CALL SYMBOL(-.9,-.8,.2,3H = ,0.,3)
      CALL NUMBER(-.2,-.8,.2,CONST,0.,3)
      CALL SYMBOL(-2.6,-.4,.2,2H-T,.0,2)
      CALL SYMBOL(2.4,-.4,.2,2H+T,.0,2)
      GO TO 6
    4 IF(CODE.GT.2) GO TO 5
      CALL SYMBOL(-1.,-.8,.2,1HT,0.,1)
      CALL SYMBOL(-.9,-.8,.2,3H = ,0.,3)
      CALL NUMBER(-.2,-.8,.2,CONST,0.,3)
      CALL SYMBOL(-2.6,-.4,.2,2H-5,0.,2)
      CALL SYMBOL(2.4,-.4,.2,2H+S,0.,2)
    6 PDEL=(PSTRT-PEND)/IP
      PTIC=((ABS(PSTRT-PEND))/10.0)
      CALL AXIS(-5.0,0.0,1H ,1,4.0,0.0,PSTRT,PTIC)
      PSTRE=PSTRT+(6.0*PTIC)+0.00001
      PTIC2=PTIC+0.00001
      CALL AXIS(1.,0.,1H ,1,4.,0.,PSTRE,PTIC2)
      CALL PLOT(-1.,0.,3)
      CALL PLOT(1.,0.,2)
      CALL PLOT(0.,0.,3)
      CALL PLOT(0.,5.8,2)
      CALL PLOT(0.,0.,3)
```

```
      CALL SYMBOL(-.05,-.4,.2,1H0,0.,1)
      CALL PLOT(0.0,0.0,3)
      X=0.05
      DO 10 J=1,9
      Y=0.5+(J-1)*1.0
      CALL PLOT(-X,Y,3)
   10 CALL PLOT(X,Y,2)
      CALL PLOT(0.0,0.0,3)
      DO 1 IW1=1,IP
      THETA=(PSTRT-(IW1*PDEL))
      THETS=(THETA/(ABS(PSTRT)))*5.
      PANGS=PTS(IW1)/180.*4.+4.5
      IF(IW1.EQ.1)CALL PLOT(THETS,PANGS,3)
      IF(IW1.EQ.1)GO TO 1
      CALL PLOT(THETS,PANGS,2)
    1 CONTINUE
      CALL AXIS(-5.5,0.5,14HAPERTURE PHASE,14,8.,90.,-180.,45.)
      CALL SYMBOL (-5.0,-0.8,0.125,10HPATTERN = ,0.,10)
      FNUM=FLOAT(NUMPAT)
      CALL NUMBER(-3.5,-0.8,0.125,FNUM,0.,-1)
      CALL PLOT(8.,-1.,-3)
    5 RETURN
      END


      SUBROUTINE PLOT2(N,M,CONLOW,CONMAX,CONINT,NUMPAT,DASH)
C
C     A= N BY M MATRIX OF DATA POINTS
C     CONLOW= LOWEST CONTOUR TO BE PLOTTED
C     CONMAX= HIGHEST CONTOUR TO BE PLOTTED
C     CONINT= INTERVAL BETWEEN CONTOURS
C     WORDS= TEXT OF PLOT LABEL
C     NCHAR= NUMBER OF CHARACTERS INPLOT LABEL
C     CONTOURS BELOW -40. ARE PLOTTED AS DASHED LINES
C
C
      DIMENSION A(151,151),RA(151),RB(151),X(151),Y(151)
      COMMON /ARRAY/ A
      CALL PLOT(8.,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
    1 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,M,CNTRLO,CMAX,CNTRAL,0)
      GO TO 7
```

```
    5 CNTRAL=CONINT
      IF(CONMAX.EQ.CONLOW)GO TO 6
      CMAX=CONMAX
      CNTRLO=CONLOW
      GO TO 7
    6 CALL CNLAL(N,M,CNTRLO,CMAX,CNTRAL,1)
    7 CONTINUE
      CONLOW=CNTRLO
      CONMAX=CMAX
      CONINT=CNTRAL
      CALL PLOTL(SS,SYS,0.,SYS,SCALE)
      CALL PLOTL(0.,0.,SS,0.,SCALE)
      CALL PLOTL(SS,0.,SS,SYS,SCALE)
      CALL PLOTL(0.,SYS,0.,0.,SCALE)
      CALL PLOT(1.00,0.25,3)
      CALL PLOT(0.60,0.25,2)
      CALL PLOT(0.60,8.25,2)
      CALL PLOT(1.00,8.25,2)
      CALL PLOT(1.00,0.25,2)
      CALL SYMBOL (0.88,0.45,0.12,10HPATTERN = ,90.,10)
      FNUM=NUMPAT
      CALL NUMBER(0.88,2.075,0.12,FNUM,90.,-1)
 1125  YCONA=1.0/SMAX
      DELTAX=SX/FLOAT(N-1)
      X(1)=0.0
      Y(1)=0.0
      RB(1) = A(1,1)
      DO 27 J=2,N
       RB(J)=A(J,1)
   27  X(J)=X(J-1)+DELTAX
      DELTAY=SY/FLOAT(M-1)
      DO 28 J=2,M
   28  Y(J)=Y(J-1)+DELTAY
      DO 118  K=2,M
       DO 30 J=1,N
       RA(J)=RB(J)
   30  RB(J)=A(J,K)
      DO 118  J=2,N
   35  ASSIGN 112 TO L
      RR=RA(J)
      XX=X(J)
      YY=Y(K-1)
   37 RL=RR
      XL=XX
      YL=YY
   39 IF(RL-RA(J-1))   41,40 ,40
   40 IF(RL-RB(J))42,50 ,50
   41 RL=RA(J-1)
      XL=X (J-1)
      YL= Y(K-1)
      GO TO 40
   42 RL=RB(J)
      XL=X (J)
      YL=Y(K)
      GO TO 50
   50 RS=RR
      XS=XX
```

```
      YS=YY
      IF(RS-RA(J-1)) 52, 52,53
   52 IF(RS-RB(J)) 60,60,54
   53 RS=RA(J-1)
      XS=X (J-1)
      YS =Y(K-1)
      GO TO 52
   54 RS=RB(J)
      XS=X (J)
      YS=Y (K)
      GO TO 60
   60 RM=RR
      XM=XX
      YM=YY
      IF(RM-RS) 62, 62,61
   61 IF(RM-RL)70 ,62 ,62
   62 RM=RA(J-1)
      XM=X (J-1)
      YM=Y (K-1)
      IF(RM-RS) 64,64,63
   63 IF(RM-RL) 70,64,64
   64 RM = RB(J)
      XM=X (J)
      YM=Y (K)
   70 YCS=YS*YCONA
      YCM=YM*YCONA
      YCL=YL*YCONA
   71 YS=YS-SY
      YM=YM-SY
      YL=YL-SY
   72  XCS=XS/SMAX
       XCM=XM/SMAX
       XCL=XL/SMAX
      RC = CNTRLO
   80 IF (RC.GT.CMAX )      GO TO 110
      IF ( RC .NE. RM ) GO TO 91
   81 IF ( RM .NE. RS ) GO TO 91
   82 IF ( RL .EQ. RM ) GO TO 100
   91 IF(RC-RS)100,95,92
   92 IF(RC-RM)96,93,94
   93 XPA=XCM
      YPA=YCM
      GO TO 99
   94 IF(RC-RL)106,103,110
   95 Q=0.0
      GO TO 97
   96 Q = (RC-RS)/(RM-RS)
   97 XPA = XCS-Q*(XCS-XCM)
      YPA = YCS-Q*(YCS-YCM)
   99 Q = (RC-RS)/(RL-RS)
      XPB = XCS-Q*(XCS-XCL)
      YPB = YCS-Q*(YCS-YCL)
      IF(RC-DASH) 10115,10115,10116
10115 XPB1=0.5*(XPA+XPB)
      YPB1=0.5*(YPA+YPB)
      IF(ABS (XPA-XPB1)-.001)5001,5002,5002
 5001 IF(ABS (YPA-YPB1)-.001)100,5002,5002
```

```
 5002 CALL PLOT(SCALE*XPA+2.,SCALE*YPA+0.25,3)
      CALL PLOT(SCALE*XPB1+2.,SCALE*YPB1+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 PLOT(SCALE*XPA+2.,SCALE*YPA+0.25,3)
      CALL PLOT(SCALE*XPB+2.,SCALE*YPB+0.25,2)
  100 RC = RC + CNTRAL
      GO TO 80
  103 XPA = XCL
      YPA = YCL
      GO TO 99
  106 Q=(RC-RM)/(RL-RM)
      XPA=XCM-Q*(XCM-XCL)
      YPA=YCM-Q*(YCM-YCL)
      GO TO 99
  110  GO 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
      CALL PLOT (SCALE+6.,0.,-3)
       RETURN
       END


      SUBROUTINE CNLAL(N,M,CNTRLO,CMAX,CNTRAL,NC)
      DIMENSION X(151,151)
      COMMON /ARRAY/ X
      XMAX=X(1,1)
      XMIN=X(1,1)
      DO 10  J=1,M
      DO 10  I=1,N
      XMAX=AMAX1(XMAX,X(I,J))
   10 XMIN=AMIN1(XMIN,X(I,J))
      IF(NC.EQ.1) GO TO 40
      IF(XMAX.EQ.0.)GO TO 20
      SN=XMIN/XMAX
      IF(SN)20,20,30
   20 XCON=ABS(XMAX)
      IF(ABS(XMIN).GT.ABS(XMAX))XCON=ABS(XMIN)
      CNTRAL=XCON/10.
      CMAX=XMAX
      CNTRLO=CNTRAL*AINT(XMIN/CNTRAL)
      RETURN
   30 XCON=ABS(XMAX-XMIN)
      CNTRAL=XCON/10.
      CNTRLO=XMIN
      CMAX=XMAX
      RETURN
   40 CMAX=CNTRAL*AINT(XMAX/CNTRAL)
      CNTRLO=CNTRAL*AINT(XMIN/CNTRAL)
      RETURN
      END
```

```
      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 PLOT(X(1),Y(1),3)
      CALL PLOT(X(2),Y(2),2)
      RETURN
      END
C
C
C         SUBROUTINE PLOT3
C
C         PURPOSE:  TO DRAW A PERSPECTIVE VIEW OF A CONTOURED SURFACE.
C
C
C         DESCRIPTION OF PARAMETERS AND IMPORTANT VARIABLES:
C
C         N        - NUMBER OF DATA POINTS ALONG FIRST AXIS.
C
C         M        - NUMBER OF DATA POINTS ALONG THE SECOND AXIS.
C
C         NUMPAT   - PATTERN NUMBER (FOR LABELLING).
C
C         K        - CODE THAT TELLS WHETHER TO DRAW THE GRID LINES:
C                         K=1: ALONG THE N-DIMENSION ONLY.
C                         K=2: ALONG THE M-DIMENSION ONLY.
C                         K=3: ALONG BOTH DIMENSIONS.
C
C         DISTS    - DISTANCE FROM SURFACE TO EYE WHEN PERSPECTIVE IS
C                    CALCULATED -- SDISTS > 6 USUALLY WONT'T SHOW ANY
C                    DISTORTION DUE TO PARALLAX.
C
C         YAW      - (IN DEGREES) HOW FAR THE OBJECT IS TURNED AWAY FROM
C                    THE VIEWER.
C
C         PITCH    - (IN DEGREES) HOW THE SURFACE IS LOWERED OR RAISED AT
C                    THE FRONT EDGE.  (POSITIVE PITCH TENDS TO EXPOSE THE
C                    TOP OF THE FIGURE).
C
C         SIZE     - (IN INCHES) THE SIZE OF THE CUBE THAT ENCLOSES THE
C                    FIGURE.
C
C         KODE     - "HIDDEN LINE" SWITCH.  IF KODE=0 DO NOT DRAW HIDDEN
C                    LINES...IF KODE=1, ALL HIDDEN LINES ARE PLOTTED.
C
C         MGN      - WHETHER TO DRAW THE OUTLINE OF THE CUBE TO HELP ORIENT
C                    THE VIEWER.  MGN=0: DO NOT DRAW ANY OUTLINE OF THE
C                    CUBE.  MGN=1:  DRAW THE OUTLINE OF THE CUBE SEPARATE
C                    FROM THE FIGURE.  MGN=2:  DRAW THE OUTLINE OF THE
C                    CUBE SUPERIMPOSED ON THE SURFACE PLOT.  MGN=3:  DRAW
C                    ONLY THE THREE EDGES OF THE CUBE THAT MEET AT THE
C                    ORIGIN, SUPERIMPOSED ON THE SURFACE PLOT.
C
C         SCALE    - HOW TALL TO MAKE THE SURFACE RELATIVE TO THE HEIGHT
C                    OF THE CUBE.  SCALE=0: DO NOT SCALE THE DATA AT ALL
```

```
C                    BUT TRUST THE USER THAT THE DATA IS NOT SO HIGH THAT
C                    IT RUNS OFF THE PAPER.  SCALE=1: SCALE THE DATA SO
C                    THE TOP OF THE DATA JUST TOUCHES THE TOP OF THE CUBE.
C                    SCALE=0.3: SCALE THE DATA SO THE TOP OF THE SURFACE IS
C                    THREE TENTHS AS HIGH AS THE CUBE.
C
C
C
C        REMARKS.
C
C     I. IT IS VERY EXPENSIVE TO DRAW OPAQUE SURFACES, BECAUSE THE
C        PROGRAM HAS TO DETERMINE THE VISIBILITY OF EVERY POINT, THE
C        COMPUTER TIME DOUBLES OR TRIPLES...DEPENDING ON HOW MANY LINE
C        SEGMENTS ARE PARTIALLY VISIBLE.
C
C    II. THE CONTENTS OF ARRAY A ARE DESTROYED IN COMPUTATION.
C
C
C        COMMON BLOCKS REQUIRED:
C
C             COMMON /ARRAY/ A
C             COMMON /THREE6/ ANGA,ANGB,HV,D,SH,SV
C             COMMON /THREE7/ SL,SM,SN,CX,CY,CZ,QX,QY,QZ,SD
C
C
C        SUBROUTINE AND FUNCTION SUBPROGRAMS REQUIRED:
C             THREE2
C             THREE3
C             THREE4
C             THREE5
C             PLOT
C             FACTOR
C             SYMBOL
C             NUMBER
C
C
C
C        REFERENCE:HOWARD JESPERSON, IOWA STATE UNIVERSITY.
C
C        MODIFIED FOR USE AT VPI BY:  ROBERT D. KEPHART.
C                                     S. R. KAUFFMAN
C                                     W. L. STUTZMAN
C                                     E. L. COFFEY
C
C
C


      SUBROUTINE PLOT3(N,M,NUMPAT)
C
C*****A= N BY M MATRIX OF DATA POINTS
C*****WORDS= PLOT LABELING
C*****NCHAR= NUMBER OF CHARACTERS IN THE PLOT LABLE+SPACES
C
      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=3
      SDISTS=6.0
      PITCH=30.
      YAW=45.
      SIZE=10.
      KODE=0
      MGN=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 SYMBOL(0.3,1.0  ,0.12,10HPATTERN = ,90.,10)
      FNUM=FLOAT(NUMPAT)
      CALL NUMBER(0.3,2.130 ,0.12,FNUM,90.,-1)
      CALL PLOT(1.5,-.2,-3)
C***** ********
      ANGA = (YAW+270.) * .0174532
      ANGB = PITCH * .0174532
      HV = SIZE
C  DIRECTION COMPONENTS TO THE EYE.
       SL = -COS( ANGA ) * COS( ANGB )
       SM = -SIN( ANGA ) * COS( ANGB )
       SN = -SIN ( ANGB )
            IF ( ABS( SN ) .NE. 1.0 ) GO TO 10
          WRITE(  6       ,     20  )
   20    FORMAT (  '1' , 20X,  20('*') , / '0', 'YOU ARE ATTEMPTING TO LO
     :K  STRAIGHT DOWN ( OR UP ) AT THE SURFACE ' )
               GO TO 2150
   10  CONTINUE
      SD = 1.0 / SQRT( 1.0 - SN ** 2  )
       X(1) = 1
       X(2) = N
       Y(1) = 1
       Y(2) = M
      T=MAX0(M,N)
C  FIND THE DIAGONAL OF THE "CUBE".
       D = M ** 2  +  N ** 2  +  T ** 2
       D  =  SQRT ( D )
       SCL = SDISTS * 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      PPPPPPPPPPPPPPPPPPPPPPPPPPP
       GO TO 2060
C      WRITE(6,100) CX,CY,CZ
C      WRITE(6,100) QX,QY,QZ
 100  FORMAT(1X,3F15.3)
 2060 Z(2)=A(1,1)
```

```
      Z(1)=A(1,1)
      DO 1000 J=1,N
      DO 1000 K=1,M
      Z(1)=AMIN1(Z(1),A(J,K))
      Z(2)=AMAX1(Z(2),A(J,K))
 1000 CONTINUE
      RANGE= (Z(2)-Z(1))
       DOL=1.0
       IF(SCALE.NE.0)  DOL=T/RANGE*SCALE
C  SCALE THE SURFACE TO MAKE A "CUBE".
       DO 30 I = 1 , N
       DO 30 J = 1 , M
       A ( I , J ) = ( A ( I , J ) - Z ( 1 ) ) * DOL
   30 CONTINUE
       Z(1) = 0.0
       Z(2) = T
 2080 CALL THREE2  ( X, Y, Z, XP , H , V ,KODE)
       DO 2130 I = 1 , 8
       H( I ) = ( (XP(I) - OX ) * SM - (  H(I) - QY ) * SL ) * SD
       V ( I ) = (  V( I ) - QZ ) * SD
 2130 CONTINUE
 2100 H(10)=H(1)
      H(9)=H(1)
      DO 1001 J=1,8
      H(9)=AMIN1(H(9),H(J))
      H(10)=AMAX1(H(10),H(J))
 1001 CONTINUE
 2120 V(9)=V(1)
      V(10)=V(1)
      DO 1002 J=1,8
      V(9)=AMIN1(V(9),V(J))
      V(10)=AMAX1(V(10),V(J))
 1002 CONTINUE
      IF( MGN .EQ. 0) GO TO 2140
      S=HV
      IF(MGN .EQ. 1) S=1.5
      SH = S/ (H(10)-H(9) )
      SV = S/ (V(10)-V(9) )
      SH = SIGN( AMIN1(SH,SV),SH )
      SV = SIGN(SH,SV)
      IF(MGN .EQ. 1)CALL PLOT (0.,2.,-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(.03,.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(3)-H(9))*SH,  (V(3)-V(9))*SV,2)
```

```
          CALL PLOT ( (H(7)-H(9))*SH, (V(7)-V(9))*SV,2)
          CALL PLOT ( (H(5)-H(9))*SH, (V(5)-V(9))*SV,2)
          CALL PLOT ( (H(6)-H(9))*SH, (V(6)-V(9))*SV,2)
          CALL PLOT ( (H(8)-H(9))*SH, (V(8)-V(9))*SV,2)
          CALL PLOT ( (H(4)-H(9))*SH, (V(4)-V(9))*SV,2)
          CALL PLOT ( (H(8)-H(9))*SH, (V(8)-V(9))*SV,2)
          CALL PLOT ( (H(7)-H(9))*SH, (V(7)-V(9))*SV,2)
     2139  IF(MGN .NE. 1) GO TO 2140
          CALL PLOT (AINT((H(10)-H(9))*SH+2.),-2.05,-3)
     2140 CALL THREE3(X,Y,N,M,H,V,K,KODE)
     2150 CONTINUE
          CALL PLOT(16.,-1.5,-3)
             RETURN
          END


          SUBROUTINE THREE2 ( X, Y, Z, XP , H , V ,KODE)
C FIND THE CORNERS OF THE ROTATED CUBE.
C
           DIMENSION X(2),Y(2),Z(2),H(10),V(10),XP(8)
C
  050  L = 0
  070  DO 180 I = 1, 2
C
  090   DO 170 J = 1, 2
C
  110     DO 160 K = 1, 2
C
  130     L = L + 1
  140     CALL THREE4 ( X(I), Y(J), Z(K), XP( L ),
     1      H(L) , V( L ),KODE )
  160 CONTINUE
  170 CONTINUE
  180 CONTINUE
  190 RETURN
       END


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


          SUBROUTINE THREE3 (X,Y,N,M,H,V,K,KODE)
C  DRAW THE FIGURE.
         COMMON /THREE6/ ANGA , ANGB , HV , D, SH,SV
        COMMON /THREE7/SL,SM,SN,CX,CY,CZ,QX,QY,QZ,SD
C
        DIMENSION X(2),Y(2),H(10),V(10),A(151,151)
```

```
      COMMON /ARRAY/ A
      INTEGER UP , DOWN , PEN , P , Q
        INTEGER P1 , PO
C
C
C
             END = 1.0 / 16.0
C  CAN USE 1 / 32 OR  1 / 64 FOR FINER INTERPOLATION
C
C
      UP   = 3
      DOWN = 2
      SH =  HV / ( H ( 10 ) - H ( 9 ) )
      SV = HV / ( V ( 10 ) - V ( 9 ) )
      SH = SIGN(AMIN1(SH,SV),SH)
      SV = SIGN(SH,SV)
       MM = M
       NN = N
C 080 IF(K-1) 100,120,100
C
C 100 IF(K-3) 1110,120,1110
C
C  DRAW LINES ALONG THE Y-AXIS
  120 CONTINUE
      L = 0
      LD = 1
        DD = 0.5 * LD
C
  140  DO 1060 J = 1, M
      Q   = 0
      YJ = J
  160  DO 1030 I = 1, NN
C
           L = L + LD
       XI = L
      CALL THREES (    XI ,YJ , N , M , P ,KODE)
       PEN = UP
           IF ( P ) 510 , 520 , 530
  510 CONTINUE
         IF ( Q ) 540 , 550 , 540
  520 CONTINUE
         IF ( Q ) 610 , 1020 , 610
  530  CONTINUE
         IF ( Q ) 540 , 550 , 540
  540  CONTINUE
        PEN = DOWN
       GO TO 170
  550  CONTINUE
          IF ( I .EQ. 1 ) GO TO 170
         DI = DD
      TO = L - LD
        T = TO + DI
        P1 = Q
  560  IF (  ABS( DI )  .LT.  END  )  GO TO 570
      CALL THREES (T,YJ,N,M,PO,KODE)
        DI = DI * 0.5
       IF ( PO  .EQ.  0  )  GO TO 565
```

```
      TC = T
      P1 = PO
      T =  T - DI
      GO TO 560
  565 T = T + DI
      GO TO 560
  570 CONTINUE
      T = TO
      IF ( P1 * P  )  170 , 170 , 580
  580 CONTINUE
  590 CONTINUE
      ZP = A(L-LD,J)+(T-L+LD)*(A(L,J)-A(L-LD,J))/LD
      CALL THREE4(T,YJ,ZP,XP,HH,VV,KODE)
      HH = ( ( XP-QX)*SM- (HH - QY )*SL ) * SD
      VV =  ( VV - QZ ) * SD
      HH = ( HH - H(9) ) * SH
      VV = ( VV - V(9) ) * SV
      CALL PLOT ( HH , VV , PEN )
  600 PEN = 5 - PEN
      GO TO 170
  610 CONTINUE
      PEN = DOWN
      DI = DD
      TO = L - LD
      T = TO + DI
      P1 = Q
  620 IF ( ABS( DI )  .LT.  END  )  GO TO 630
      CALL THREE5 (T,YJ,N,M,PO,KODE)
      DI = DI * 0.5
      IF ( PO  .EQ.  0  )  GO TO 625
      TO = T
      P1 = PO
      T = T + DI
      GO TO 620
  625 T = T - DI
      GO TO 620
  630 CONTINUE
      T = TO
      IF (  P1 * Q )  600 , 600 , 590
  170 CALL THREE4 ( XI , YJ , A( L, J ), XP , HH ,VV ,KODE)
      VV =  ( VV - QZ ) * SD
      HH = ( ( XP-QX)*SM- (HH - QY )*SL ) * SD
  190 HH = ( HH - H(9) ) * SH
  200 VV = ( VV - V(9) ) * SV
      CALL PLOT (  HH , VV , PEN )
 1020 Q = P
 1030 CONTINUE
C
C
      L = L + LD
      LD = -LD
      DD =-DD
C
 1060 CONTINUE
C
C
C1090 IF(K-3) 2060,1110,2060
```

```
C
C  DRAW LINES ALONG THE X-AXIS.
 1110 CONTINUE
C
      L = 0
      LD = 1
        DD = 0.5 * LD
 1140 DO 2040 I = 1 , N
          XI = I
       Q = 0
 1160    DO 2020 J = 1 , MM
              L = L + LD
            YJ = L
      CALL THREE5 (XI,YJ,N,M,P,KODE)
       PEN = UP
             IF ( P ) 1510 , 1520 , 1530
 1510 CONTINUE
          IF ( Q ) 1540 , 1550 , 1540
 1520   CONTINUE
         IF ( Q )  1610 , 2010 , 1610
1530   CONTINUE
           IF ( Q ) 1540 , 1550 , 1540
 1540   CONTINUE
         PEN = DOWN
        GO TO 1170
 1550       CONTINUE
        IF ( J .EQ.  1  )  GO TO 1170
         DI = DD
      TO=L-LD
       T = TO + DI
       P1 = Q
 1560  IF (  ABS( DI )  .LT.  END  )  GO TO 1570
      CALL THREE5 (XI,T,N,M,PO,KODE)
        DI = DI * 0.5
      IF ( PO  .EQ.  0  )  GO TO 1565
         TO = T
       P1 = PO
       T =  T - DI
         GO TO 1560
 1565 T = T + DI
          GO TO 1560
 1570   CONTINUE
       T = TO
      IF ( P1 * P   )  1170 , 1170 , 1580
 1580 CONTINUE
 1590 CONTINUE
      ZP=A(I,L-LD) + (T-L+LD) * (A(I,L) - A(I,L-LD))/LD
         CALL THREE4 ( XI , T , ZP , XP,HH,VV ,KODE)
      HH = ( ( XP-QX)*SM- (HH - QY )*SL ) * SD
      VV =   ( VV - QZ ) * SD
        HH = ( HH - H(9) ) * SH
        VV = ( VV - V(9) ) * SV
       CALL PLOT ( HH , VV , PEN )
 1600 PEN = 5 - PEN
        GO TO 1170
 1610  CONTINUE
         PEN = DOWN
```

```
            DI = DD
           TO = L - LC
          T = TO + DI
             PI = C
   1620  IF ( ABS( DI )  .LT.  END  )  GO TO 1630
         CALL THREE5 (XI,T,N,M,PO,KODE)
            DI = DI * 0.5
           IF ( PO .EQ. 0 )  GO TO 1625
            TO = T
          P1 = PO
           T = T + DI
           GO TO 1620
   1625    T = T - DI
            GO TO 1620
   1630     CONTINUE
           T = TO
           IF ( P1 * C ) 1600 , 1600 , 1590
   1170    CALL THREE4 ( XI, YJ, A( I, L ), XP , HH ,VV ,KODE)
          HH = ( ( XP-QX)*SM- (HH - QY )*SL ) * SD
          VV =  ( VV - CZ ) * SD
   1180      HH = ( HH - H(9) ) * SH
   1190      VV = ( VV - V(9) ) * SV
            CALL PLOT (  HH , VV ,  PEN  )
   2010  C = P
   2020   CONTINUE
C
          L = L + LD
          LD = - LD
             DD =-DD
   2040  CONTINUE
C
   2060 CONTINUE
C
   2130  RETURN
          END


         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/SL,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
       2  IF ( YJ .EQ.  JC )        GO TO 4
          ZB = Z(IR , JC) + (YJ-JC)*(Z(IR,JC+1) - Z(IR,JC ))
       4  CONTINUE
          XEND = 0.0
          DX = 0.0
```

```
      YMULT = 0.0
      ZMULT = 0.0
        IF (XI   .EQ.  CX  ) GO TO 10
     YMULT = (YJ - CY ) /  (XI - CX )
       ZMULT = ( ZB - CZ ) / ( XI - CX )
     DX = 1.0
       XEND = M + 1
       IF ( XI  .LT.  CX)  GO TO 10
       DX = -1.0
      XEND = 0.0
   10 CONTINUE
      YEND = 0.0
      DY = 0.0
       XMULT = 0.0
       IF ( YJ  .EQ.  CY  )  GO TO 20
      XMULT = ( XI - CX ) / (YJ - CY )
       IF (  ZMULT  .EQ.  0.0  )  ZMULT=(ZB - CZ ) / ( YJ - CY )
      DY = 1.0
      YEND = N + 1
      IF ( YJ  .LT.  CY )  GO TO 20
      DY = -1.0
      YEND = 0.0
   20 CONTINUE
      CUM = 0
      CNT = 0
      P = 0
      XB = XI
      YB = YJ
   30 CONTINUE
      II = AINT( XB )
      JJ = AINT( YB )
      XSTEP = DX
      YSTEP = DY
        IF ( XB  .EQ.  II )  GO TO 40
        IF ( DX  .LT.  0.0 )  XSTEP = 0.0
      GO TO 45
   40  IF ( YB  .EQ.  JJ )  GO TO 45
        IF ( DY  .LT.  0.0 )  YSTEP = 0.0
   45 CONTINUE
        I = II + XSTEP
        J = JJ + YSTEP
        IF ( I  .EQ.  XEND )  GO TO 80
        IF ( J  .EQ.  YEND )  GO TO 80
      XB = CX + XMULT * ( J - CY )
      YB = CY + YMULT * ( I - CX )
        IF ( DX  .LT.  0.0 )  GO TO 55
        IF ( XB  .LT.  I  )  GO TO 60
   50 XB = I
       GO TO 65
   55 IF ( XB  .LT.  I )  GO TO 50
   60 YB = J
   65  CONTINUE
      ZB = CZ + ZMULT * ( XB - CX )
      IR = I
      JC = J
        IF ( YB  .NE.  J )  GO TO 70
       ICX = I - DX
```

```
      ZS = Z( IR, JC ) - DX * ( XB - I ) * (Z(IDX,JC) - Z(IR,JC))
      GO TO 75
 70   JDY=J-DY
      ZS = Z( IR,JC ) - DY * ( YB-J ) * (Z( IR,JDY ) - Z( IR,JC ) )
   75 CONTINUE
      SGN = 1
      IF (  ZB  .LT.  ZS  )  SGN = -1
      CUM = CUM + SGN
      CNT = CNT + 1
      IF (  IABS ( CUM)  .EQ.  CNT  )  GO TO 30
       GO TO 90
  78  P=1
      GO TO 95
   80 CONTINUE
      P = 1
          IF ( CUM ) 84 , 86 , 90
   84 P = -1
             GO TO 90
   86  CONTINUE
              IF ( ZB  .LE.  CZ  )  GO TO 90
       P = -1
   90  CONTINUE
  95    RETURN
      END


      FUNCTION PAT(U,V,ITYPE)
C
C        THIS SUBPROGRAM GIVES THE BASIC CORRECTION PATTERN F(U,V).
C
C        ITYPE = 1  --  UNIFORM LINE SOURCE LOCATED AT S=0.
C                2  --  UNIFORM LINEAR ARRAY LOCATED AT S=0.
C                3  --  TRIANGULAR LINE SOURCE LOCATED AT S=0.
C                4  --  UNIFORM RECTANGULAR APERTURE.
C                5  --  UNIFORM RECTANGULAR ARRAY.
C                6  --  UNIFORM CIRCULAR APERTURE.
C                7  --  GENERALARRAY.
C
C        ITYPE > 7  --  SPECIAL SOURCE (FUNCTION SPECPT(U,V,ITYPE) WILL
C                       BE CALLED.
C
C
C        VERSION 1  LEVEL 1
C
C        DATE OF LAST REVISION:  AUGUST 29, 1973.
C
C
C        THIS WORK SUPPORTED BY NASA GRANT NGR 47-004-103
C
C        FOR FURTHER INFORMATION CONTACT:
C             W.L. STUTZMAN   DEPT. OF ELEC. ENGR.  951-6624.
C             E.L. COFFEY     DEPT. OF ELEC. ENGR. 951-5494
C
      COMPLEX TEMP,CEXP,IMAG
      COMMON /PAT1/ P1,P2,P3,P4,P5,P6,PI,SS(100),TT(100),RR(100)
      COMMON /PAT2/ I1,I2,I3,I4,I5
C
```

```
      C
              IF(ITYPE.GT.7) GO TO 990
              GO TO (100,200,300,400,500,600,700),ITYPE
      C
      C          ITYPE .LT. 1
      C
              WRITE(6,10) ITYPE
           10 FORMAT(1H0,5X,'***ERROR***     ITYPE HAS THE VALUE ',I11,':',2X,
             $'EXECUTION TERMINATED')
              STOP
      C
      C
      C          ITYPE = 1  --  UNIFORM LINE SOURCE.
      C
      C       FLEN=P1
          100 CONTINUE
              PAT=1.0
              IF(V.NE.0.) PAT=SIN(PI*P1*V)/(PI*P1*V)
              GO TO 999
      C
      C
      C          ITYPE = 2  --  UNIFORM LINEAR ARRAY
      C
          200 CONTINUE
      C       FLEN=P1
      C       NELMT=I1
              PAT=1.0
              IF(V.NE.0.) PAT=SIN(PI*P1*V)/(I1*SIN(PI*P1*V/I1))
              GO TO 999
      C
      C
      C          ITYPE = 3  --  TRIANGULAR LINE SOURCE.
      C
          300 FLEN=P1/2.
              PAT=1.0
              IF(V.NE.0.) PAT = (SIN(FLEN*PI*V)/(FLEN*PI*V))**2
              GO TO 999
      C
      C
      C          ITYPE = 4  --  UNIFORM RECTANGULAR APERTURE
      C
          400 CONTINUE
      C       FLS=P1
      C       FLT=P2
              ARG1=PI*P1*U
              ARG2=PI*P2*V
              IF(ARG1) 401,402,401
          401 IF(ARG2) 403,404,403
          403 PAT=SIN(ARG1)/ARG1*SIN(ARG2)/ARG2
              GO TO 999
          404 PAT=SIN(ARG1)/ARG1
              GO TO 999
          402 IF(ARG2) 405,406,405
          405 PAT=SIN(ARG2)/ARG2
              GO TO 999
          406 PAT=1.0
              GO TO 999
```

```
C
C
C          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
     502 IF(ARG2) 505,506,505
     505 PAT=SIN(ARG2)/(I2*SIN(ARG2/I2))
         GO TO 999
     506 PAT=1.0
         GO TO 999
C
C
C          ITYPE = 6  --  UNIFORM CIRCULAR APERTURE.
C
     600 C=SQRT(U*U+V*V)
C        A=P1
         IF(C.EQ.0.) GO TO 601
         X=2.*PI*P1*C
         CALL BESJ(X,1,BJ,0.0001,IER)
         PAT=BJ/X*2.0
         GO TO 999
     601 PAT=1.0
         GO TO 999
C
C
C          ITYPE = 7  --  GENERAL ARRAY
C
     700 IMAG=(0.0,1.0)
         NELMT=I1*I2
         TEMP=(0.0,0.0)
         DO 701 J=1,NELMT
         TEMP=TEMP+1.0*CEXP(IMAG*2.*PI*(U*SS(J)+V*TT(J)))
     701 CONTINUE
         PAT=REAL(TEMP)/NELMT
         GO TO 999
     990 PAT=SPECPT(U,V,ITYPE)
     999 RETURN
         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                 2 --  UNIFORM LINEAR ARRAY LOCATED AT S=0.
C                 3 --  TRIANGULAR LINE SOURCE LOCATED AT S=0.
C                 4 --  UNIFORM RECTANGULAR APERTUR.
C                 5 --  UNIFORM RECTANGULAR ARRAY.
C                 6 --  UNIFORM CIRCULAR APERTURE.
C                 7 --  GENERAL (3-D) ARRAY.
C
C         ITYPE > 7 --  SPECIAL SOURCE (FUNCTION SPSOR(M,N,U,V,ITYPE)
C                       WILL BE CALLED.)
C
C         VERSION 1  LEVEL 0
C
C         DATE OF LAST REVISION:  73/166    JUNE 15,1973
C
C         THIS WORK SUPPORTED BY NASA GRANT NGR 47-004-103.
C
C         FOR FURTHER INFORMATION CONTACT:
C              W. L. STUTZMAN  DEPT. OF ELEC. ENGR.  951-6624.
C              E. L. COFFEY    DEPT. OF ELEC. ENGR.  951-5494.
C
      COMPLEX TEMP,CEXP,IMAG,SPSOR
      COMMON /PAT1/ P1,P2,P3,P4,P5,P6,PI,SS(100),TT(100),RR(100)
      COMMON /PAT2/ I1,I2,I3,I4,I5
C
      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
C
C         ITYPE .LT. 1
C
      WRITE(6,10) ITYPE
   10 FORMAT(1H0,5X,'***ERROR***     ITYPE HAS THE VALUE ',I11,':',2X,
     $'EXECUTION TERMINATED')
      STOP
C
C
C         ITYPE = 1  --  UNIFORM LINE SOURCE
C
  100 CONTINUE
C     FLEN=P1
      SOURCE=CEXP(-IMAG*PI*2.*T*V)/P1
      GO TO 999
C
C
C         ITYPE = 2  -- UNIFORM LINEAR ARRAY
C
  200 CONTINUE
C     FLEN=P1
      SOURCE=CEXP(-IMAG*2.*PI*V*T)/P1
      GO TO 999
C
C
C         ITYPE= 3 -- TRIANGULAR LINE SOURCE
C
```

```
1243      300 CONTINUE
     C        FLEN=P1
1244          CON=ABS(2.*T/P1)
1245          SOURCE=2./P1*CEXP(-IMAG*2.*PI*T*V)*(1.-CON)
1246          IF(CON.GT.1) SOURCE=(0.0,0.0)
1247          GO TO 999
     C
     C
     C           ITYPE = 4  --  UNIFORM RECTANGULAR APERTURE
     C
1248      400 CONTINUE
     C        FLS=P1
     C         FLT=P2
1249          SOURCE=CEXP(-IMAG*2.*PI*(S*U+V*T))/(P1*P2)
1250          GO TO 999
     C
     C
     C           ITYPE = 5 --  UNIFORM RECTANGULAR ARRAY
     C
1251      500 CONTINUE
     C        FLS=P1
     C        FLT=P2
1252          SOURCE=CEXP(-IMAG*2.*PI*(S*U+V*T))/(P1*P2)
1253          GO TO 999
     C
     C
     C           ITYPE = 6  --  UNIFORM CIRCULAR APERTURE
     C
1254      600 RHO=SQRT(S*S+T*T)
     C        A=P1
1255          SOURCE=(0.0,0.0)
1256          IF(RHO.LE.P1) SOURCE=CEXP(-IMAG*2.*PI*(S*U+T*V))/(2.*PI*P1**2)
1257          GO TO 999
     C
     C
     C           ITYPE = 7  --  GENERAL ARRAY.
     C
1258      700 CONTINUE
1259          SOURCE=CEXP(-IMAG*2.*PI*(U*S+V*T))/(I1*I2)
1260          GO TO 999
1261      990 SOURCE=SPSOR(M,N,U,V,ITYPE)
1262      999 RETURN
1263          END



1264          SUBROUTINE LOCSOR(M,N,S,T)
1265          INTEGER PX,PY
1266          REAL INITLS,INITLT
1267          COMMON /PAT1/ P1,P2,P3,P4,P5,P6,PI,SS(100),TT(100),RR(100)
1268          COMMON /PAT2/ I1,I2,I3,I4,I5
1269          COMMON /LOC/ ITYPE
     C
     C
1270          IF(ITYPE.GT.7) GO TO 990
1271          GO TO (100,200,300,400,500,600,700), ITYPE
```

```
      WRITE(6,10) ITYPE
   1C FORMAT(1HC,5X,'***ERROR***    ITYPE HAS THE VALUE ',I11,':',2X,
     $'EXECUTICN TERMINATED')
      STCP
C
C
  10C CCNTINUE
C     INITLT=P1
C      CELTAT=P3
      S=C.
      T=P2+(N-1)*P3
      GO TO 999
C
C
  20C CONTINUE
C     PY=I1
C      DISY=P2
      S=C.
      T=(N-I1/2-1)*P2
      IF(I1/2*2.EQ.I1) T=T+0.5*P2
      GO TO 999
C
C
  30C GC TO 100
C
C
  40C CONTINUE
C     INITLS=P3
C     INITLT=P4
C     CELTAS=P5
C      DELTAT=P6
      S=P3+(M-1)*P5
      T=P4+(N-1)*P6
      GC TO 999
C
C
  5CC CONTINUE
C     PX=I1
C     PY=I2
C     DISX=P3
C     DISY=P4
      S=(M-I1/2-1)*P3
      T=(N-I2/2-1)*P4
      IF(I1/2*2.EQ.I1) S=S+0.5*P3
      IF(I2/2*2.EQ.I2) T=T+0.5*P4
      GO TO 999
C
C
  60C GO TO 400
C
C
  70C CCNTINUE
      NELMT=(M-1)*I2+N
      S=SS(NELMT)
      T=TT(NELMT)
      GO TO 999
C
```

```
C
  990 CALL SPLOC(M,N,S,T)
  999 RETURN
      END

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

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

      SUBROUTINE SPLOC(M,N,S,T)
C        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

(u,v)	$F_d(u,v)$	$F_u(u,v)$	$F_L(u,v)$
$-0.2 \leq u \leq 0.2$ $-0.05 \leq v \leq 0.05$	0. dB	0.5 dB	-0.5 dB

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

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:

```
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
      READ(5,1) ULIM,VLIM
    1 FORMAT(8F10.0)
C
C        READ TRANSITION REGION LIMITS UTRAN AND VTRAN
C
      READ(5,1) UTRAN,VTRAN
C
      DO 10 M=1,MMAX
      U=STARTU+(M-1)*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         11        21        31        41        51        61
          +---------+---------+---------+---------+---------+---------+

 1         &PARAM
 2         IDISK=1,ISYMM=3,DELTAU=0.02,DELTAV=0.01,MMAX=26,NMAX=51,
 3         &END
 4         &IPRINT
 5         FDESCN=1,FDBPR=1,FDBCN=1,FCURPR=1,
 6         &END
 7         &PATIN
 8         ITYPE=4,LX=10.,LY=20.,
 9         INITLS=-5.0,DELTAS=0.2,FINALS=5.0,
10         INITLT=-10.0,DELTAT=0.4,FINALT=10.0,
11         &END
12        0.2       0.05
13        0.34      0.12
14        00015
15        0.0       0.0       1.0
16        0.0       0.05      1.0
17        0.0       -0.05     1.0
18        0.1       0.0       1.0
19        -0.1      0.0       1.0
20        0.1       0.05      1.0
21        0.1       -0.05     1.0
22        -0.1      0.05      1.0
23        -0.1      -0.05     1.0
24        0.2       0.05      1.0
25        0.2       -0.05     1.0
26        -0.2      0.05      1.0
27        -0.2      -0.05     1.0
28        0.2       0.0       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 10λ by 20λ (LX=10.,LY=20.)
iii. The value of current will be calculated at 51 x 51 points from s = -5.0 to 5.0 by 0.2, and t = -10.0 to 10.0 by 0.4. (INITLS=-5.,FINALS=5.,DELTAS=0.2;INITLT=-10.,FINALT=10.,DELTAT=0.4)

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

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

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

8.2 Output from ANTSYN

This section is devoted to the actual output from the computer program ANTSYN with data as specified in Section 8.1. Due to page size limitations, some of the output has been edited. The omissions are indicated by an ellipsis(...).

ANTENNA SYNTHESIS PROGRAM VERSION 3 LEVEL 1 VPI EE DEPT.

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

PROGRAM PARAMETERS

IDISK = 1	STARTU = 0.0	MMAX = 26
ISYMM = 3	STARTV = 0.0	NMAX = 51
ITRMAX = 200	DELTAU = 0.020	MCENT = 1
	DELTAV = 0.010	NCENT = 1

FDESPT = 0	FORGPT = 0	FDBPT = 0	ICURPT = 0	FCURPT = 0
FDESCN = 1	FORGCN = 0	FDBCN = 1	ICURCN = 0	FCURCN = 0
FDESPR = 0	FORGPR = 0	FDBPR = 1	ICURPR = 0	FCURPR = 1

ITYPE=4 -- UNIFORM RECTANGULAR APERTURE

DIMENSIONS = LX,LY = 10.0000 , 20.0000

INITLS,DELTAS,FINALS: -5.0000 0.2000 5.0000

INITLT,DELTAT,FINALT: -10.0000 0.4000 10.0000

MCUR,NCUR: 51 51

CONTOUR PLOT OF THE DESIRED PATTERN.

```
         -0.5000   -0.4000   -0.3000   -0.2000   -0.1000   -0.0000   0.1000
            .         .         .         .         .         .         .
 0.5000 .2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 0.4900 .2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 0.4800 .2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 0.4700 .2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 0.4600 .2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 0.4500 .2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 0.4400 .2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 0.4300 .2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 0.4200 .2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 0.4100 .2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 0.4000 .2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 0.3900 .2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 0.3800 .2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 0.3700 .2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 0.3600 .2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 0.3500 .2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 0.3400 .2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 0.3300 .2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 0.3200 .2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 0.3100 .2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 0.3000 .2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 0.2900 .2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 0.2800 .2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 0.2700 .2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 0.2600 .2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 0.2500 .2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 0.2400 .2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 0.2300 .2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 0.2200 .2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 0.2100 .2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 0.2000 .2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 0.1900 .2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 0.1800 .2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 0.1700 .2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 0.1600 .2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 0.1500 .2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 0.1400 .2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 0.1300 .2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 0.1200 .2 2 2 2 2 2 2 2 + + + + + + + + + + + + + + + + + + + + + + +
 0.1100 .2 2 2 2 2 2 2 2 + + + + + + + + + + + + + + + + + + + + + + +
 0.1000 .2 2 2 2 2 2 2 2 + + + + + + + + + + + + + + + + + + + + + + +
 0.0900 .2 2 2 2 2 2 2 2 + + + + + + + + + + + + + + + + + + + + + + +
 0.0800 .2 2 2 2 2 2 2 2 + + + + + + + + + + + + + + + + + + + + + + +
 0.0700 .2 2 2 2 2 2 2 2 + + + + + + + + + + + + + + + + + + + + + + +
 0.0600 .2 2 2 2 2 2 2 2 + + + + + + + + + + + + + + + + + + + + + + +
 0.0500 .2 2 2 2 2 2 2 2 + + + + + + + 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7
 0.0400 .2 2 2 2 2 2 2 2 + + + + + + + 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7
 0.0300 .2 2 2 2 2 2 2 2 + + + + + + + 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7
 0.0200 .2 2 2 2 2 2 2 2 + + + + + + + 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7
 0.0100 .2 2 2 2 2 2 2 2 + + + + + + + 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7
 0.0000 .2 2 2 2 2 2 2 2 + + + + + + + 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7
-0.0100 .2 2 2 2 2 2 2 2 + + + + + + + 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7
-0.0200 .2 2 2 2 2 2 2 2 + + + + + + + 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7
-0.0300 .2 2 2 2 2 2 2 2 + + + + + + + 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7
-0.0400 .2 2 2 2 2 2 2 2 + + + + + + + 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7
-0.0500 .2 2 2 2 2 2 2 2 + + + + + + + 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7
```

CONTOUR LEVEL KEY

0:	-0.6000000E 00	TO	-0.3999900E 00		4:	0.2000000E 00	TO	0.4000099E 00
1:	-0.4000000E 00	TO	-0.1999900E 00		5:	0.4000000E 00	TO	0.6000099E 00
2:	-0.2000000E 00	TO	0.1000000E-04		6:	0.5999998E 00	TO	0.8000098E 00
3:	0.0	TO	0.2000099E 00		7:	0.7999997E 00	TO	0.1000010E 01

8:	0.9999995E 00	TO	0.1200008E 01
9:	0.1199999E 01	TO	0.1400008E 01
-:	-0.9999999E 30	TO	-0.6000000E 00
+:	0.1400008E 01	TO	0.9999999E 30

-- INITIAL COEFFICIENTS --

J	UORG(J)	VORG(J)	CORG(J)
1	0.0	0.0	1.0000
2	0.0	0.0500	1.0000
3	0.0	-0.0500	1.0000
4	0.1000	0.0	1.0000
5	-0.1000	0.0	1.0000
6	0.1000	0.0500	1.0000
7	0.1000	-0.0500	1.0000
8	-0.1000	0.0500	1.0000
9	-0.1000	-0.0500	1.0000
10	0.2000	0.0500	1.0000
11	0.2000	-0.0500	1.0000
12	-0.2000	0.0500	1.0000
13	-0.2000	-0.0500	1.0000
14	0.2000	0.0	1.0000
15	-0.2000	0.0	1.0000

SEARCH	9	4	-0.1842
SEARCH	9	14	0.1524
SEARCH	1	14	0.2088
SEARCH	11	6	0.1875
SEARCH	5	6	0.1962
SEARCH	9	18	-0.1572
SEARCH	1	18	-0.2146
SEARCH	1	22	0.1065
SEARCH	19	6	0.1304
SEARCH	8	14	0.1241
SEARCH	8	22	0.1394
SEARCH	1	14	0.1209
SEARCH	11	6	0.1322
SEARCH	4	6	0.1436
SEARCH	23	6	-0.1201
SEARCH	7	18	-0.1188
SEARCH	1	4	-0.0716
SEARCH	9	5	-0.1592
SEARCH	1	5	-0.1155
SEARCH	9	5	-0.0856
SEARCH	1	5	-0.0900
SEARCH	1	4	-0.0643
SEARCH	9	5	-0.1467
SEARCH	1	5	-0.1023
SEARCH	9	5	-0.0757
SEARCH	7	16	0.1005
SEARCH	11	6	0.0889
SEARCH	14	14	0.1093
SEARCH	1	5	-0.0670
SEARCH	9	5	-0.0946
SEARCH	11	1	0.1536
SEARCH	19	1	0.1523
SEARCH	1	5	-0.0784
SEARCH	23	1	-0.1235
SEARCH	9	4	-0.1375
SEARCH	11	1	0.1179
SEARCH	6	1	0.1176
SEARCH	6	6	0.0956
SEARCH	9	4	-0.0875
SEARCH	11	6	0.1438
SEARCH	11	1	0.0898
SEARCH	5	1	0.0873
SEARCH	5	6	0.1050
SEARCH	9	4	-0.0652
SEARCH	6	1	0.1144
SEARCH	11	1	0.1130
SEARCH	11	6	0.0960
SEARCH	9	4	-0.0996
SEARCH	6	6	0.1263
SEARCH	11	1	0.1018
SEARCH	6	1	0.0968
SEARCH	9	4	-0.0924
SEARCH	11	6	0.1390
SEARCH	5	6	0.0783
SEARCH	11	1	0.0947
SEARCH	9	5	-0.0938
SEARCH	11	6	0.0748
SEARCH	1	4	-0.0624
SEARCH	10	5	-0.1378
SEARCH	1	5	-0.0909
SEARCH	9	5	-0.0783
SEARCH	1	5	-0.0715

-- FINAL COEFFICIENTS --

J	US(J)	VS(J)	CORCOF(J)
1	0.1600	0.0300	-0.6602
2	-0.1600	0.0300	-0.6602
3	0.1600	-0.0300	-0.6602
4	-0.1600	-0.0300	-0.6602
5	0.1600	0.1300	0.1689
6	-0.1600	0.1300	0.1689
7	0.1600	-0.1300	0.1689
8	-0.1600	-0.1300	0.1689
9	0.0	0.1300	0.3694
10	0.0	-0.1300	0.3694
⋮	⋮	⋮	⋮
80	-0.0800	0.0	0.0840
81	0.1800	0.0400	-0.1449
82	-0.1800	0.0400	-0.1449
83	0.1800	-0.0400	-0.1449
84	-0.1800	-0.0400	-0.1449

J	UORG(J)	VORG(J)	CORG(J)
1	0.0	0.0	1.0352
2	0.0	0.0500	1.0352
3	0.0	-0.0500	1.0352
4	0.1000	0.0	1.0352
5	-0.1000	0.0	1.0352
6	0.1000	0.0500	1.0352
7	0.1000	-0.0500	1.0352
8	-0.1000	0.0500	1.0352
9	-0.1000	-0.0500	1.0352
10	0.2000	0.0500	1.0352
11	0.2000	-0.0500	1.0352
12	-0.2000	0.0500	1.0352
13	-0.2000	-0.0500	1.0352
14	0.2000	0.0	1.0352
15	-0.2000	0.0	1.0352

NUMBER OF ITERATIONS = 62

FNORM = 1.03517

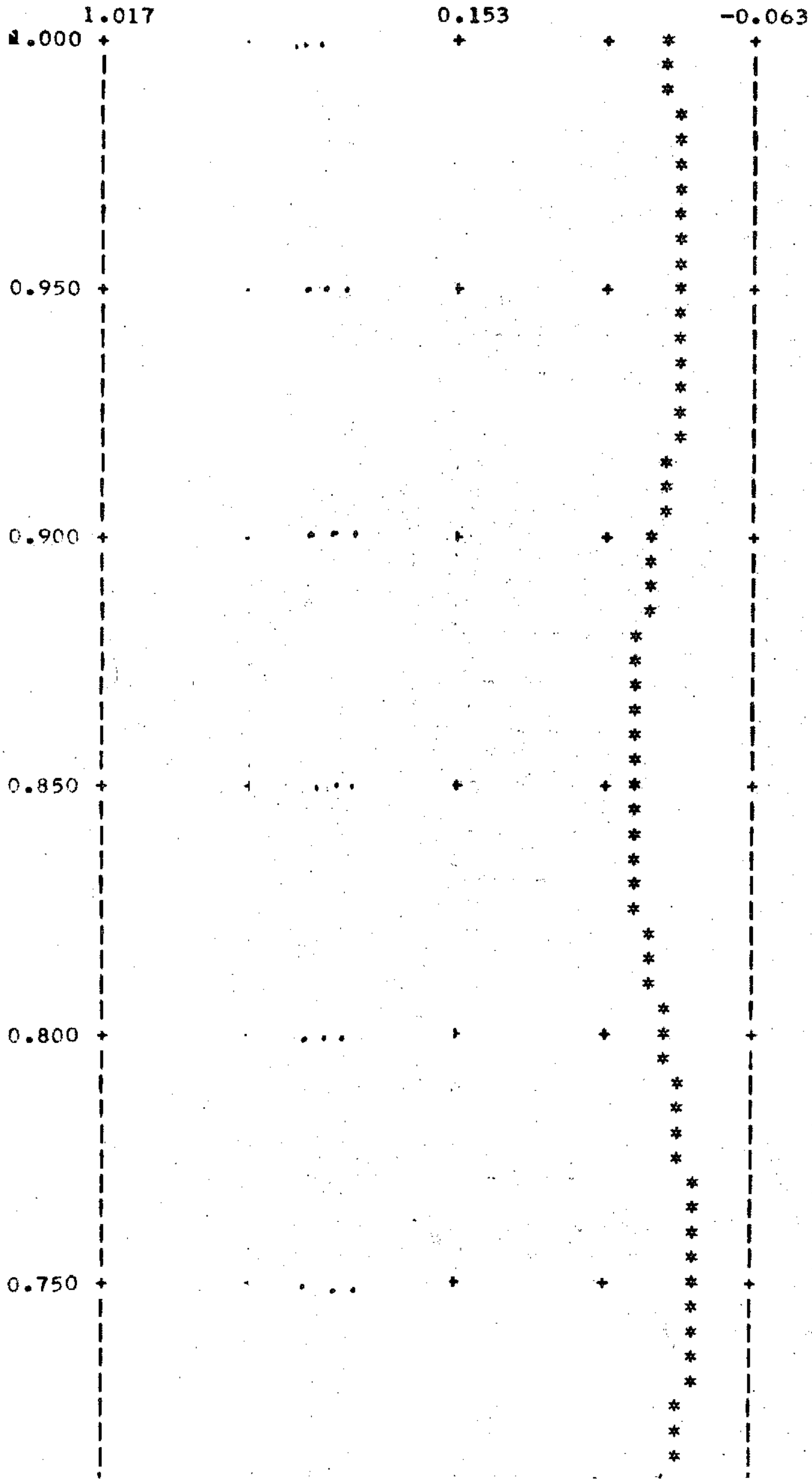
PATTERN NUMBER = NUMPAT = 77

CONTOUR PLOT OF THE FINAL PATTERN IN DB.

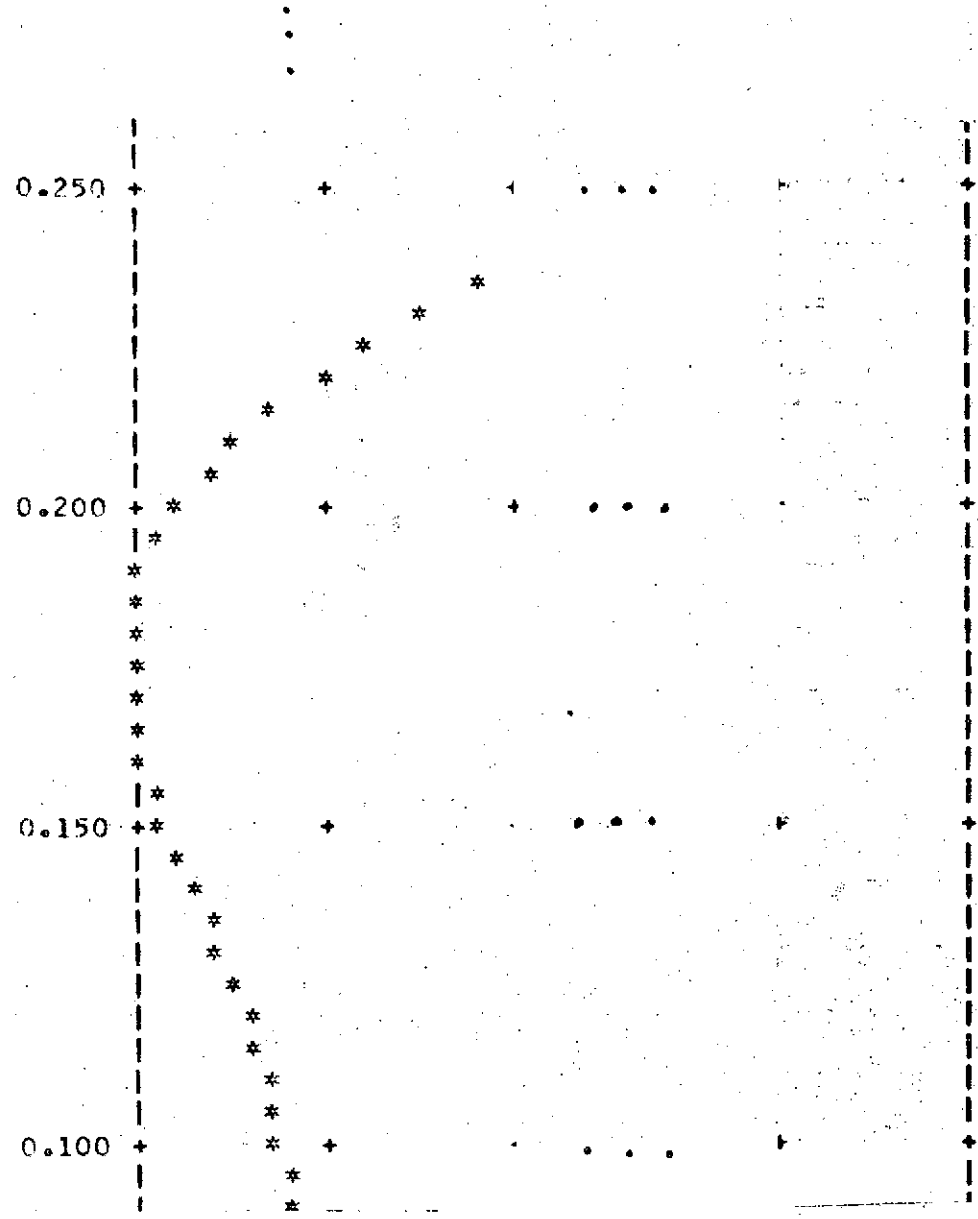
```
          -0.5000   -0.4000   -0.3000   -0.2000   -0.1000   -0.0000    0.1000
             .         .         .         .         .         .         .
 0.5000 .- - - - - - - - - - - - - - - 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 0.4900 .- - - - 0 - - - - 0 1 1 1 1 1 1 1 0 0 1 1 1 1 1 2 2 2 1 1 1 1
 0.4800 .- - 0 0 0 0 - - 0 1 2 2 2 2 2 2 2 2 2 2 3 3 3 3 3 3 3 3 3 3 3
 0.4700 .- - 0 0 0 0 - - 0 1 2 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
 0.4600 .- - - - 0 - - - - 0 1 2 2 2 2 2 2 2 3 3 3 3 3 3 3 3 3 3 3 3 3
 0.4500 .- - - - - - - - - - - - - - 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 0.4400 .- - - 0 0 0 - - 0 1 1 2 2 2 1 1 1 0 0 0 1 1 1 2 2 2 2 2 1 1 1
 0.4300 .- - 0 0 1 0 - - 0 1 2 2 3 3 3 3 2 2 2 3 3 3 3 3 3 3 3 3 3 3 3
 0.4200 .- - 0 1 1 0 - - 0 2 2 3 3 3 3 3 3 3 3 3 3 3 3 3 4 4 4 3 3 3 3
 0.4100 .- - - 0 0 0 - - 0 1 1 2 2 2 2 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
 0.4000 .- - - - - - - - - - - - - - 0 1 1 2 2 2 2 2 2 2 1 1 1 2 2 2 2
 0.3900 .- - - 0 0 0 - - 0 1 1 2 2 2 1 1 0 0 0 0 0 1 1 1 2 2 2 1 1 1 0
 0.3800 .- - 0 1 1 1 0 - 1 2 2 3 3 3 3 3 3 2 3 3 3 3 3 3 3 3 3 3 3 3 3   .
 0.3700 .- - 0 1 1 1 0 - 1 2 2 3 3 3 3 3 3 3 3 3 3 4 4 4 4 4 4 4 4 4 3
 0.3600 .- - - 0 0 0 - - 0 1 2 2 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3   .
 0.3500 .- - - - - - - - - - - - - 0 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 0.3400 .- - 0 0 0 0 - - 0 1 2 2 2 2 1 1 0 - - - - 0 1 1 2 2 2 1 1 0 -   .
 0.3300 .- - 0 1 1 1 0 - 1 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 4 4 4 3 3 3 3
 0.3200 .- - 0 1 1 1 0 - 1 2 3 3 3 3 3 3 3 3 3 4 4 4 4 4 4 4 4 4 4 4 4
 0.3100 .- - 0 0 0 0 - - 0 1 2 2 3 3 3 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4
 0.3000 .- - - - - - - - - - - - - 1 2 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
 0.2900 .- - 0 1 1 1 0 - 0 2 2 2 2 2 1 - - 1 1 1 1 0 - 0 1 1 1 0 - 0 1
 0.2800 .- - 1 1 2 1 0 - 1 2 3 3 3 3 3 3 2 2 2 2 2 3 3 3 4 4 4 3 3 3 2
 0.2700 .- 0 1 1 2 1 0 - 1 2 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4
 0.2600 .- - 0 0 0 0 - - 0 1 2 3 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4
 0.2500 .- - - - - - - - - - - - 0 2 3 4 4 4 4 5 5 4 4 4 4 4 4 4 4 4 5
 0.2400 .- - 0 1 2 2 1 - 1 2 3 3 3 2 - 2 4 4 4 4 4 4 3 3 2 2 2 3 3 4 4
 0.2300 .- - 1 2 2 2 1 - 2 3 4 4 4 4 2 - 3 4 4 4 4 3 1 0 2 2 2 0 1 3 4
 0.2200 .- - 1 2 2 2 2 - 2 3 4 4 4 4 3 2 0 3 3 3 3 1 1 3 4 4 4 3 1 1 3
 0.2100 .- - 0 1 2 2 1 - 1 3 4 4 4 4 3 3 0 1 2 2 1 - 2 4 4 4 4 4 2 - 1
 0.2000 .- - - - - - 0 0 - - 1 2 3 3 3 2 1 - 0 1 - 0 2 3 4 4 4 3 2 0 -
 0.1900 .- 0 2 2 2 1 - 1 2 3 3 2 2 0 - 1 1 0 - - - - 1 2 2 2 2 2 1 - -
 0.1800 .- 1 2 3 3 3 1 1 3 4 4 4 4 3 2 0 0 1 1 1 - - 0 1 1 1 1 1 0 - -
 0.1700 .0 1 2 3 3 3 1 1 3 4 5 5 5 4 3 2 0 2 2 2 1 - 2 3 3 3 3 3 2 - 1
 0.1600 .0 0 2 3 3 3 1 1 3 4 5 5 5 4 3 2 1 2 3 3 2 - 2 3 3 4 3 3 2 - 2
 0.1500 .1 - 0 2 2 2 0 1 3 4 4 4 4 4 3 1 1 3 3 3 3 1 - 2 3 3 3 2 - 1 3
 0.1400 .0 1 1 0 0 - - - 0 1 2 3 3 3 2 1 - 1 2 3 2 2 0 - 1 2 1 - 0 2 2
 0.1300 .0 2 2 3 3 2 1 - 2 2 2 1 - 1 2 3 3 2 2 0 - - - - 0 0 0 - - - -   .
 0.1200 .- 2 3 3 3 3 1 1 3 4 4 3 1 2 4 4 5 5 5 4 4 4 3 3 3 3 3 3 3 4 4
 0.1100 .0 2 3 4 4 3 1 2 4 4 4 2 2 4 5 6 6 6 6 6 6 6 5 5 5 5 5 5 5 6 6   .
 0.1000 .1 0 2 3 3 2 - 3 3 3 2 3 5 6 6 7 7 7 7 7 7 7 7 6 6 6 6 6 7 7 7
 0.0900 .2 0 - 1 0 0 2 3 3 0 4 5 6 7 7 8 8 8 8 8 8 8 7 7 7 7 7 7 7 8 8
 0.0800 .3 2 2 2 3 3 3 3 1 4 5 6 7 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8   .
 0.0700 .3 3 3 4 4 4 4 3 2 5 6 7 8 8 8 9 9 9 9 9 9 9 9 8 8 8 8 8 9 9 9
 0.0600 .3 4 4 4 5 5 5 3 3 6 7 7 8 8 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9
 0.0500 .3 4 4 5 5 5 5 3 4 6 7 8 8 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9
 0.0400 .3 4 4 4 5 5 5 3 4 6 7 8 8 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9
 0.0300 .3 3 4 4 5 5 5 3 3 6 7 8 8 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9
 0.0200 .3 3 3 4 4 4 4 3 3 6 7 8 8 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9
 0.0100 .3 3 3 3 4 4 4 3 3 5 7 8 8 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9
 0.0000 .3 3 2 3 3 4 4 3 2 5 7 8 8 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9
-0.0100 .3 3 3 3 4 4 4 3 3 5 7 8 8 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9
-0.0200 .3 3 3 4 4 4 4 3 3 6 7 8 8 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9
-0.0300 .3 3 4 4 5 5 5 3 3 6 7 8 8 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9
-0.0400 .3 4 4 4 5 5 5 3 4 6 7 8 8 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9
-0.0500 .3 4 4 5 5 5 5 3 4 6 7 8 8 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9
```

A-118

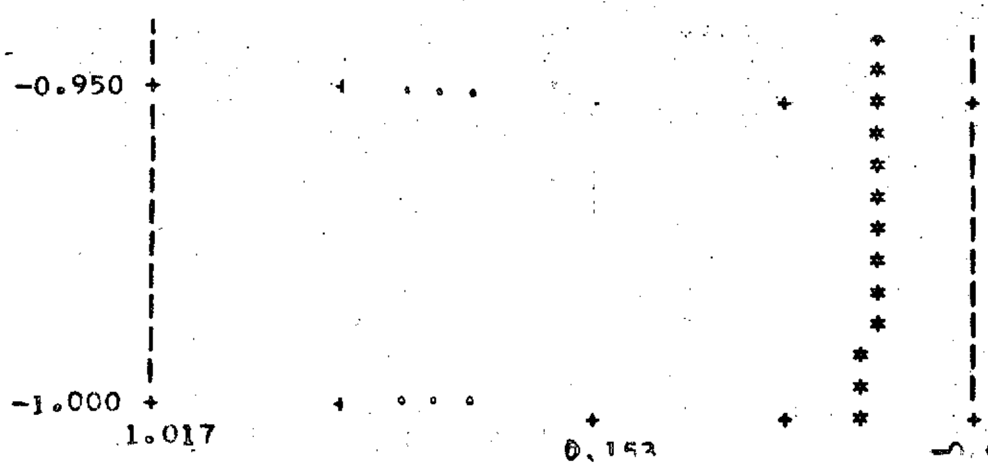
U-AXIS PROFILE OF FINAL PATTERN -- V= 0.0
PATTERN NUMBER 77



	REAL	DB.
1.000	-0.003	-51.37
	-0.005	-45.46
	-0.008	-42.08
	-0.010	-39.79
	-0.012	-38.13
	-0.014	-36.91
	-0.016	-36.01
	-0.017	-35.39
	-0.018	-35.00
	-0.018	-34.83
0.950	-0.018	-34.87
	-0.018	-35.14
	-0.017	-35.64
	-0.015	-36.41
	-0.013	-37.51
	-0.011	-39.03
	-0.009	-41.19
	-0.006	-44.43
	-0.003	-50.15
	-0.000	-81.36
0.900	0.003	-50.51
	0.006	-44.40
	0.009	-40.95
	0.012	-38.61
	0.014	-36.92
	0.016	-35.66
	0.018	-34.74
	0.020	-34.09
	0.021	-33.68
	0.021	-33.48
0.850	0.021	-33.50
	0.021	-33.73
	0.020	-34.20
	0.018	-34.93
	0.016	-35.97
	0.013	-37.43
	0.011	-39.49
	0.007	-42.53
	0.004	-47.75
	0.001	-65.08
0.800	-0.003	-50.30
	-0.007	-43.53
	-0.010	-39.85
	-0.013	-37.40
	-0.017	-35.62
	-0.019	-34.30
	-0.022	-33.32
	-0.023	-32.62
	-0.025	-32.15
	-0.025	-31.90
0.750	-0.026	-31.86
	-0.025	-32.03
	-0.024	-32.42
	-0.022	-33.06
	-0.020	-33.99
	-0.017	-35.30
	-0.014	-37.12
	-0.010	-39.77



[illegible]	
0.666	-3.53
0.707	-3.01
0.747	-2.54
0.784	-2.11
0.819	-1.73
0.852	-1.39
0.882	-1.09
0.909	-0.83
0.933	-0.60
0.954	-0.41
0.972	-0.25
0.986	-0.12
0.998	-0.02
1.007	0.06
1.013	0.11
1.016	0.14
1.017	0.15
1.016	0.14
1.013	0.11
1.009	0.07
1.003	0.02
0.996	-0.04
0.988	-0.10
0.980	-0.17
0.972	-0.24
0.964	-0.32
0.957	-0.38
0.950	-0.45
0.943	-0.51
0.938	-0.55
0.934	-0.59
0.931	-0.62
0.929	-0.64
[illegible]	[illegible]



-0.017	-35.64
-0.018	-35.14
-0.018	-34.87
-0.018	-34.83
-0.018	-35.00
-0.017	-35.39
-0.016	-36.01
-0.014	-36.91
-0.012	-38.13
-0.010	-39.79
-0.008	-42.08
-0.005	-45.46
-0.003	-51.37
REAL	[illegible]

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

1.056 0.163 -0.061

V	REAL	DB.
1.000	-0.002	-55.08
	0.004	-48.45
	0.009	-40.90
	0.013	-37.45
	0.017	-35.63
	0.018	-34.86
	0.018	-34.96
	0.016	-35.97
	0.012	-38.17
	0.008	-42.44
0.950	0.002	-54.16
	-0.004	-48.22
	-0.009	-40.53
	-0.014	-37.04
	-0.017	-35.19
	-0.019	-34.40
	-0.019	-34.48
	-0.017	-35.48
	-0.013	-37.64
	-0.008	-41.85
0.900	-0.002	-53.19
	0.004	-48.00
	0.010	-40.15
	0.015	-36.60
	0.018	-34.73
	0.020	-33.91
	0.020	-33.98
	0.018	-34.95
	0.014	-37.08
	0.009	-41.22
0.850	0.002	-52.15
	-0.004	-47.80
	-0.010	-39.75
	-0.016	-36.15
	-0.019	-34.24
	-0.021	-33.40
	-0.021	-33.44
	-0.019	-34.38
	-0.015	-36.48
	-0.009	-40.54
0.800	-0.003	-51.06
	0.004	-47.62
	0.011	[illegible]
⋮	⋮	⋮
	0.009	-40.91
	0.004	-48.45
1.000	-0.002	-55.07

1.056 0.163 -0.061 REAL DB.

S AXIS PROFILE OF FINAL CURRENT

S	T	REAL	IMAGINARY
-5.0000	-0.0000	0.5700823E-02	0.0
-4.8000	-0.0000	0.7003162E-02	0.0
-4.6000	-0.0000	0.6880980E-02	0.2095476E-08
-4.4000	-0.0000	0.5990162E-02	0.3026798E-08
-4.2000	-0.0000	0.4723225E-02	0.0
-4.0000	-0.0000	0.3161762E-02	0.0
-3.8000	-0.0000	0.1145484E-02	0.0
-3.6000	-0.0000	-0.1570189E-02	0.0
-3.4000	-0.0000	-0.5123518E-02	0.0
-3.2000	-0.0000	-0.9384781E-02	0.0
-3.0000	-0.0000	-0.1387722E-01	0.0
-2.8000	-0.0000	-0.1780687E-01	0.0
-2.6000	-0.0000	-0.2019734E-01	0.0
-2.4000	-0.0000	-0.2009303E-01	0.0
-2.2000	-0.0000	-0.1677621E-01	0.0
-2.0000	-0.0000	-0.9939682E-02	-0.2095476E-08
-1.8000	-0.0000	0.2297403E-03	0.0
-1.6000	-0.0000	0.1307478E-01	0.0
-1.4000	-0.0000	0.2758893E-01	0.0
-1.2000	-0.0000	0.4259761E-01	-0.3492460E-08
-1.0000	-0.0000	0.5695011E-01	-0.3492460E-08
-0.8000	-0.0000	0.6967366E-01	-0.3492460E-08
-0.6000	-0.0000	0.8006346E-01	-0.2793968E-08
-0.4000	-0.0000	0.8768129E-01	-0.4656613E-09
-0.2000	-0.0000	0.9230632E-01	0.0
-0.0000	-0.0000	0.9385431E-01	-0.7105427E-14
0.2000	-0.0000	0.9230632E-01	0.0
0.4000	-0.0000	0.8768129E-01	0.2095476E-08
0.6000	-0.0000	0.8006346E-01	0.2793968E-08
0.8000	-0.0000	0.6967390E-01	0.3725290E-08
1.0000	-0.0000	0.5695021E-01	0.3725290E-08
1.2000	-0.0000	0.4259773E-01	0.3725290E-08
1.4000	-0.0000	0.2758904E-01	0.0
1.6000	-0.0000	0.1307483E-01	0.0
1.8000	-0.0000	0.2297729E-03	0.0
2.0000	-0.0000	-0.9939581E-02	0.0
2.2000	-0.0000	-0.1677619E-01	0.0
2.4000	-0.0000	-0.2009305E-01	0.0
2.6000	-0.0000	-0.2019734E-01	0.0
2.8000	-0.0000	-0.1780686E-01	0.0
3.0000	-0.0000	-0.1387723E-01	-0.1396984E-08
3.2000	-0.0000	-0.9384889E-02	0.0
3.4000	-0.0000	-0.5123563E-02	0.0
3.6000	-0.0000	-0.1570206E-02	0.0
3.8000	-0.0000	0.1145449E-02	0.0
4.0000	-0.0000	0.3161710E-02	0.0
4.2000	-0.0000	0.4723225E-02	0.0
4.4000	-0.0000	0.5990129E-02	-0.2561137E-08
4.6000	-0.0000	0.6880980E-02	-0.2095476E-08
4.8000	-0.0000	0.7003162E-02	0.0
5.0000	-0.0000	0.5700838E-02	0.0

T AXIS PROFILE OF FINAL CURRENT

S	T	REAL	IMAGINARY
-0.0000	-10.0000	-0.2455123E-01	0.4423782E-08
-0.0000	-9.6000	-0.1838829E-01	0.2793968E-08
-0.0000	-9.2000	-0.1313753E-01	0.3725290E-08
-0.0000	-8.8000	-0.9560350E-02	0.5122274E-08
-0.0000	-8.4000	-0.7850584E-02	0.1396984E-08
-0.0000	-8.0000	-0.7672068E-02	0.1396984E-08
-0.0000	-7.6000	-0.8307122E-02	0.1396984E-08
-0.0000	-7.2000	-0.8870248E-02	0.3492460E-08
-0.0000	-6.8000	-0.8532621E-02	0.3026798E-08
-0.0000	-6.4000	-0.6700888E-02	0.3026798E-08
-0.0000	-6.0000	-0.3115309E-02	0.1862645E-08
-0.0000	-5.6000	0.2147641E-02	-0.1164153E-08
-0.0000	-5.2000	0.8754689E-02	0.1396984E-08
-0.0000	-4.8000	0.1624599E-01	-0.1164153E-08
-0.0000	-4.4000	0.2417203E-01	0.9313226E-09
-0.0000	-4.0000	0.3219855E-01	0.1862645E-08
-0.0000	-3.6000	0.4015272E-01	0.2328306E-08
-0.0000	-3.2000	0.4800258E-01	0.2793968E-08
-0.0000	-2.8000	0.5578430E-01	-0.2328306E-09
-0.0000	-2.4000	0.6350774E-01	0.1629815E-08
-0.0000	-2.0000	0.7107019E-01	-0.2328306E-09
-0.0000	-1.6000	0.7821018E-01	-0.1629815E-08
-0.0000	-1.2000	0.8452117E-01	0.5820766E-10
-0.0000	-0.8000	0.8951896E-01	-0.1600711E-09
-0.0000	-0.4000	0.9274071E-01	-0.4656613E-09
-0.0000	-0.0000	0.9385431E-01	-0.7105427E-14
-0.0000	0.4000	0.9274071E-01	0.7130438E-09
-0.0000	0.8000	0.8951896E-01	-0.8731149E-10
-0.0000	1.2000	0.8452106E-01	-0.8731149E-10
-0.0000	1.6000	0.7821023E-01	0.0
-0.0000	2.0000	0.7107037E-01	0.2095476E-08
-0.0000	2.4000	0.6350780E-01	0.6984919E-09
-0.0000	2.8000	0.5578437E-01	-0.1396984E-08
-0.0000	3.2000	0.4800263E-01	-0.1396984E-08
-0.0000	3.6000	0.4015278E-01	-0.3026798E-08
-0.0000	4.0000	0.3219860E-01	-0.1862645E-08
-0.0000	4.4000	0.2417206E-01	-0.1862645E-08
-0.0000	4.8000	0.1624606E-01	0.2328306E-09
-0.0000	5.2000	0.8754738E-02	-0.1396984E-08
-0.0000	5.6000	0.2147641E-02	0.1164153E-08
-0.0000	6.0000	-0.3115296E-02	-0.1862645E-08
-0.0000	6.4000	-0.6700821E-02	-0.3026798E-08
-0.0000	6.8000	-0.8532569E-02	-0.3026798E-08
-0.0000	7.2000	-0.8870188E-02	-0.2561137E-08
-0.0000	7.6000	-0.8307122E-02	0.2328306E-09
-0.0000	8.0000	-0.7672135E-02	-0.3492460E-08
-0.0000	8.4000	-0.7850584E-02	0.2328306E-09
-0.0000	8.8000	-0.9560246E-02	-0.4423782E-08
-0.0000	9.2000	-0.1313743E-01	-0.3725290E-08
-0.0000	9.6000	-0.1838810E-01	-0.4423782E-08
-0.0000	10.0000	-0.2455102E-01	-0.1862645E-08

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

        1        11        21
        +---------+---------+--------
  1     0007700020
  2     0015100151
  3     1111
  4     0.0
  5     0.0
  6     -35.0    -35.0
  7     -30.0    0.0       5.0
  8     0000
  9     0000
```

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

8.4 Output from ANTDATA

The following is the printout from computer program ANTDATA.

```
PLOT OUTPUT FOR PATTERN   77:

U-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.

Figure	Description
4.26	U-axis profile
4.27	V-axis profile
4.28	Contour plot
4.29	Three-dimensional plot