RADC-TR-78-147, Vol II (of two)
Final Technical Report
September 1978

PARAMETRIC ANTENNA ANALYSIS SOFTWARE PACKAGE
Computer Program Documentation and User Manuals

Robert J. Hancock
John R. Fricke

Vanderbilt University

Approved for public release; distribution unlimited.

ROME AIR DEVELOPMENT CENTER
Air Force Systems Command
Griffiss Air Force Base, New York 13441

Best
Available
Copy

This report has been reviewed by the RADC Information Office (OI) and is releasable to the National Technical Information Service (NTIS). At NTIS it will be releasable to the general public, including foreign nations.

RADC-TR-78-147, Vol II (of two) has been reviewed and is approved for publication.

APPROVED:

DONALD A. HILDEBRAND
Project Engineer

APPROVED:

JOSEPH L. RYERSON
Technical Director
Surveillance Division

FOR THE COMMANDER:

JOHN P. HUSS
Acting Chief, Plans Office

(19) TR-78-147-VOL-2

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: (18) RADC-TR-78-147, Vol II (of two)	2. GOVT ACCESSION NO.
3. RECIPIENT'S CATALOG NUMBER	
4. TITLE (and Subtitle): (6) PARAMETRIC ANTENNA ANALYSIS SOFTWARE PACKAGE, Computer Program Documentation and User Manuals. Volume II.	5. TYPE OF REPORT & PERIOD COVERED: Final Technical Report. (9)
	6. PERFORMING ORG. REPORT NUMBER: N/A
7. AUTHOR(s): (10) Robert J. Hancock, John R. Fricke	8. CONTRACT OR GRANT NUMBER(s): (15) F30602-75-C-0122
9. PERFORMING ORGANIZATION NAME AND ADDRESS: Vanderbilt University, P.O. Box 1655, Station B, Nashville TN 37235	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: (16) 62702F 95670017 (17) 00
11. CONTROLLING OFFICE NAME AND ADDRESS: Rome Air Development Center (OCDR), Griffiss AFB NY 13441	12. REPORT DATE: (11) September 1978
	13. NUMBER OF PAGES: 188
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office): Same (12) 184 p.	15. SECURITY CLASS. (of this report): UNCLASSIFIED
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE: N/A

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

Same

18. SUPPLEMENTARY NOTES

RADC Project Engineer: Donald A. Hildebrand (OCDR)

This effort was conducted under the RADC Post-Doctoral Program

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Large Aperture Antennas
Numerical Analysis
Fast Fourier Transform
Computer Programs
Parametric Analysis

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

At the present time many programs exist that calculate radiation patterns of particular aperture-type antenna systems or configurations. However, a new set of software is often needed for each antenna that is analyzed. The program described in this report is an effort to overcome this problem, particularly for large aperture array antenna systems. The software package described herein is capable of modeling a wide variety of antenna configurations. The key goals were speed, accuracy, versatility, and ease of use.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

256 740

LB

20 (Cont'd)

The software described provides a tool for accurate quantitative as well as qualitative aperture antenna analysis. Although intended primarily for far-field pattern analysis of large discrete planar arrays, the package can also be used to model reflector antenna systems and optical systems. Any aperture which can be adequately modeled by an array of up to 1000 x 1000 sample points can be treated via the software package.

The package has been designed to enable rapid parameter variations for various analytic purposes. Many commonly used factors, such as Taylor and Bayliss weighting functions; aperture shapes, such as rectangular, circular and elliptical, as well as randomizing and statistical weightings for either amplitude or phase characteristics are built into the program.

The report briefly reviews the theory involved, the parameters available, input and output requirements. Examples to illustrate usage are provided, as is a complete User Manual for the software package.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

A P P E N D I X 4

PAAS USER MANUALS

In this appendix the user manuals for the PAAS software are presented. The user manuals are arranged in alphabetical order and each is self contained except for Table 1. Table 1 is a list of Honeywell GCOS file management supervisor status codes which are used to indicate errors in file handling to the user. Since disc files (PRMFL's) are used by all PAAS software modules this table is printed one time here for reference by all the user manuals. All of the modules discussed herein except PLTDVR can be used via any remote terminal compatible with the RADC H6180 GCOS/TSS. The module PLTDVR is designed to transfer plots to the Dedicated User Interface Subsystem (DUIS).

All TSS file references made herein are to the user master catalog BECACD01 in the RADC H6180 GCOS system. In this documentation all RUN and OLD commands are shown assuming the user has previously accessed the named file or has a copy of the file under his user master catalog. The procedure for accessing files that are stored under BECACD01 is as follows:

From SYSTEM level enter YFORT and then enter the ACCESS subsystem by typing:

ACCE

The H6180 will reply:

Function?

Enter the following:

AF,BECACD01/PLARY,R

TABLE 1

HONEYWELL GCOS FILE MANAGEMENT SUPERVISOR STATUS CODES

Status codes:

4000	NO ERRORS
4001	NAME NOT IN MASTER CATALOG
4002	I/O ERROR ON DEVICE XXX SA = NNN......NNN
4003	PERMISSIONS DENIED
4004	FILE BUSY: TRY LATER
4005	INCORRECT CAT/FILE DESCRIPTION AT AAA......AAA
4006	LLINK SPACE EXHAUSTED, DEVICE XXX
4007	UNDEFINED DEVICE YYY ZZZZZZ
4010	LINK SPACE EXHAUSTED, DEVICE XXX
4011	NON-UNIQUE NAME
4012	SIZE REQUESTED LS THAN ALLOCATED
4013	SPACE REQUEST GR THAN ALLOWED
4014	PASSWORD REQUIRED AT AAA......AAA
	PASSWORD AAA......AAA AT AAA......AAA INCORRECT
4015	I-D-S FILE IN ABORT STATUS
4016	FILE CANNOT BE ALLOCATED FOR TS USE
4017	SEEK ERROR ON DEVICE XXX SA = NNN......NNN
4020	FAILURE IN NAME SCAN (IMP.)
4021	UNDEFINED DEVICE (IMP.)
4022	DEVICE LINK TABLE CHKSUM ERROR
4023	INCONSISTENT FSW BLOCK COUNT
4024	INTERNAL LINK TABLE CKSM ERROR
4025	REQUESTED ENTRY NOT ON-LINE
4026	NON-STRUCTURED FILE ENTRY
4027	FILE IN EFFECTIVE STATUS
4030	ILLEGAL PACK TYPE
4031	ACCESS GRANTED TO I-D-S FILE
4033	CAT/FILE SECURITY LOCKED
4034	ILLEGAL CHARACTER IN CAT/FILE NAME
4035	ILLEGAL CAT/FILE LIST REQUEST
4036	AFT IS FULL
4037	FILE ALREADY IN AFT
4040	MAXIMUM PAT SIZE EXCEEDED
4042	INVALID FILE CODE OR PAT PTR
4043	INVALID CATALOG BLOCK ADDRESS
4044	PERMISSION DENIED - SHARED FILE
4045	INVALID SPACE IDENTIFIER
4051	CHECKSUM ERROR - DEVICE XXX SA = NNN......NNN
4052	DEVICE XXX RELEASED

WHERE:	XXX	=DEVICE NAME (ST1,DS1,...)
	NNN......NNN	=OCTAL REPRESENTATION OF THE SEEK ADDRESS
	AAA......AAA	=12 BCD CHARACTERS OF THE CATALOG ELEMENT IN ERROR
	YYYY	=TYPE /OR/ NAME
	ZZZZZZ	=DEVICE NAME OR CLASS OF DEVICE

The H6180 will reply:

Successful
Function?

Type a carriage return and control will return to YFORT. In the above example the file named PLARY (shown underlined) was accessed. For any other file just substitute the appropriate file name.

PROGRAM FFT2DX

I. PURPOSE

The program FFT2DX transforms an antenna aperture distribution into its equivalent far-field distribution. FFT2DX generates the far-field distribution via a two-dimensional Fast Fourier transform of the aperture illumination function. The far-field is stored on a PRMFL designated by the user.

II. PERIPHERAL DEVICES REQUIRED

LUD	NAME	USE
01	Input PRMFL	Aperture Input
02	Output PRMFL	Far-field Output
03	Temporary disk file	Intermediate FFT Storage
05	Batch data deck	Program Input Data
06	Batch system output	Computer Output

III. OPERATING PROCEDURE

1. Enter the TSS CARDIN system

2. The message

   OLD OR NEW?

   is printed on the operator's console:

3. Type:

   OLD FFT2DX

The program FFT2DX will be loaded into the current file.

4. Go to line 1460 and change the PRMFL name that follows the characters

   'S:PRMFL:01,R/W,R,

   to the appropriate PRMFL name which contains the input aperture which was created using PLARY, RNDERR or FILMOD.

5. Go to line 1470 and change the PRMFL name that follows the characters

   'S:PRMFL:02,R/W,R,'

   to the appropriate PRMFL name in which the far-field energy distribution is to be stored.

6. Go to line 1500 and change the value of the input parameters in the namelist FFT to correspond to the specific input aperture and desired far-field output.

   N2 - The power of 2 which determines the number of points on the side for the 2-d transform

   LRJ - The number of blocks in the x-direction of the input aperture

   LRK - The number of blocks in the y-direction of the input aperture.

7. Go to line 1510 and change the values of the parameter in the namelist FILOUT to correspond to the desired blocks of the far-field output to be stored in the output PRMFL.

LRJIN - The number of blocks to be skipped in the x-direction, starting on the left, before beginning to store the output.

LRJWID - The width in blocks of the desired far-field output.

LRKIN - The number of blocks to be skipped in the y-direction, starting at the top, before beginning to store the output.

LRKWID - The height in blocks of the desired far-field output.

8. At this point the job definition file is complete and the user may SAVE, RESAVE and/or RUN the current file.

9. Type:

   RUN

   to run the prepared file.

10. The message

   SNUMB XXXXT

   is printed, where XXXXT is the job identification number and is used to learn the status of the job at later points in time.

IV. SUBPROGRAMS REQUIRED

FFT2D

V. RESTRICTIONS, REQUIREMENTS, AND MISC. DATA

1. This program must be executed under Honeywell CCOS TSS CARDIN subsystems.

2. N2 must be in the range $4 \le N2 \le 10$.

3. LRJ, LRK must be exactly the same value as was specified in loading the aperture.

4. In picking values for LRJIN, LRJWID, LRKIN, LRKWID remember that the size of the output PRMFL must be large enough to hold all of the specified output.

5. The input PRMFL and the output PRMFL must be in different files.

PROGRAM FILMOD

I. PURPOSE

The program FILMOD allows the user to modify a previously generated aperture that has been stored in a permanent disk file (PRMFL). This program allows the user to: (1) list element values, (2) change element values (one by one), or (3) 'punch holes' with a radius and center both specified by the user. The modified aperture is then stored on a user designed PRMFL which may be the same as the input file.

II. PERIPHERAL DEVICES REQUIRED

LUD	NAME	USE
01	Input PRMFL (Optional output PRMFL)	Aperture Input (Optional output)
02	Output PRMFL	Aperture Output
05	Time sharing terminal keyboard	User Input
06	Time sharing terminal printer	Computer Output

III. OPERATING PROCEDURE

1. Enter TSS YFORT

2. Type

   RUN FILMOD

3. The message

   INPUT FILE NAME

is printed.

The user responds with the name of an existing PRMFL which contains the aperture to be modified. The file name must be followed by a semicolon(;).

4. If the PRMFL name is not acceptable to the computer the message

   UNSUCCESSFUL ATTACH ISTAT = X

   is printed, where X is the first status word returned by the File Management Supervisor (see the Time Sharing System Programmers Reference Manual, BR39, p.(3-39) or Table 1 herein) or will contain:

   1 = file is currently open

   2 = teletypewriter requested in batch mode (illegal)

   3 = additional memory needed, request denied (time sharing user will be terminated)

   4 = CATFIL all blanks

   NOTE: See Honeywell series 600/6000 Fortran manual, BJ67, p.(6-35) - (6-36) for more details on the subroutine ATTACH.

   After the message is printed the program returns to step 3.

5. If the PRMFL name in step 3 is accepted the message

   OUTPUT FILE NAME

is printed.

The user responds with the PRMFL name in which he wishes the output to be stored. The user may specify the input file and output file as the same file if he wishes. The file name must be followed by a semicolon(;). If the file name is not acceptable the program will go through step 4 and return to step 5 so that the user may try another file name. Otherwise the program will proceed to step 6.

6. The message

LRJ, LRK

is printed.

LRJ - The width of the input aperture measured in blocks.

LRK - The height of the input aperture measured in blocks.

NOTE: The numbers entered for LRJ, LRK must be exactly the same as the values specified in the program PLARY which originally loaded the aperture file.

7. The message

MODIFY OR HOLE? (0 or 1)

is printed.

= 0 The program begins the question and answer sequence for modifying individual elements (Proceed to step 8).

- 1 The program begins the question and answer sequence for making holes in the aperture (Proceed to step 18).

8. The message

IBLK

is printed.

The user must respond with the lowest block number in which he wishes to make any modifications. Each subsequent request for IBLK must be answered with a larger number than the previous response.

9. The message

ANY ELEMENTS LISTED? (Y or N)

is printed.

- Y The user wishes to have the values of some of the elements in the specified block listed.

- N No elements are listed (Proceed to step 12).

10. The message

JSTRT, KSTRT, JSTP, KSTP

is printed.

JSTRT - The horizontal coordinate to begin the value listing.

KSTRT - The vertical coordinate to begin the value listing.

Example: (JSTRT, KSTRT) = (1, 1) if the user wishes to begin in the upper left hand corner.

JSTP - The horizontal coordinate to end the value listing.

KSTP - The vertical coordinate to end the value listing.

Example: (JSTP, KSTP) = (16, 16) if the user wishes to end in the lower right hand corner.

NOTE: The user must give all coordinates starting from the upper left hand corner with (1, 1) going to the lower right hand corner with (16, 16). As an example, if the user wished to list the values of the whole block (256 values) he would type

1, 1, 16, 16

in response to the message.

The value listing is printed in two columns with the real part of the element value in the left column and the imaginary part in the right column. The values are double spaced with one element per line starting with the element (JSTRT, KSTRT) and proceeding to (JSTP, KSTRT). The next value will be (JSTRT, KSTRT + 1) and so forth until (JSTP, KSTP) is reached.

11. The message

ANY ELEMENTS CHANGED? (Y or N)

is printed.

= Y The user wishes to change some element values.

(Proceed to step 12)

= N The user does not wish to change any element value.

(Proceed to step 14)

12. The message

HOW MANY ELEMENTS CHANGED?

is printed.

The user responds with the number (up to 100) that he wishes to change in the specified block. If he wishes to change more than 100 the answer to step 14 must be Y. If the user responds with 0 the program will jump to step 15.

13. The message

IELJ, IELK, VREAL, VIMG

is printed.

IELJ - The horizontal coordinate of the element to be changed.

IELK - The vertical coordinate of the element to be changed.

NOTE: The coordinate locations are specified according to the explanation given in step 10.

VREAL - The real part of the new element value.

VIMG - The imaginary part of the new element value.

The above message is repeated the number of times specified in step 12 for the number of elements to be changed.

14. The message

ANY MORE MODS OR LIST? (Y or N)

is printed.

= Y The user wishes to list or modify more elements.

Return to step 9.

= N The user is finished with this block.

The program proceeds to step 15.

15. The message

ANOTHER BLOCK? (Y or N)

is printed.

= Y The user wishes to list or modify the values of some elements in another block.

Return to step 8.

= N The user is finished listing or modifying values of the elements in the aperture.

Proceed to step 16.

16. The message

ANY HOLES? (Y or N)

is printed.

- Y The user wishes to make holes in the aperture.

  Proceed to step 17.

- N The user is finished with the aperture changes.

  Proceed to step 21.

17. The message

    ICNTJ, ICNTK

    is printed.

    ICNTJ - The coordinate for the center of the hole in the horizontal directions.

    ICNTK - The coordinate for the center of the hole in the vertical direction.

    NOTE: Coordinates for the center of the holes are given with respect to the upper left corner element increasing to the right and down. The upper left corner element has the coordinates

    (ICNTJ, ICNTK) = (1, 1)

    Example: If the center of the hole were to be in the lower left corner of a 64 x 64 element aperture the coordinates would be

    (ICNTJ, ICNTK) = (1, 64)

18. The message

    XHOLE

is printed.

XHOLE - The radius of the hole to be 'punched' in the aperture.

19. The hole is punched in the aperture and the following message is printed

ANOTHER HOLE? (Y or N)

The user responds with either of the following:

- Y The user wishes to have another hole punched.

  Return to step 17.

- N The user is finished punching holes in the specified aperture.

20. The message

ANY ELEMENTS CHANGED? (Y or N)

is printed.

- Y The user wishes to list or modify some element values in the specified aperture.

  Return to step 8.

- N The user is finished with the aperture changes.

21. The message

ANOTHER OUTPUT GENERATED? (Y or N)

is printed.

- Y The user wishes to generate another output aperture from the original input aperture.

Proceed to step 22.

= N The user is finished with the original input file.

Proceed to step 23.

22. The message

DETACH OUTPUT FILE ISTAT = X

is printed, where the detach is successful if X = 0; otherwise X = 1.

Return to step 5.

23. The message

ANOTHER FILE MODIFIED? (Y or N)

is printed.

= Y The user wishes to begin the program again with a new input file.

Return to step 3.

= N The user is finished with the program.

In both cases the message

DETACH OUTPUT FILE ISTAT = X

DETACH INPUT FILE ISTAT = X

is printed, where if the detach is successful X = 0; otherwise X = 1.

IV. SUBPROGRAM REQUIRED

None

V. RESTRICTIONS, REQUIREMENTS, AND MISC. DATA

1. If the input file and the output file are the same, then an affirmative answer in step 21 to the question

   ANOTHER OUTPUT GENERATED?

   causes the second generated output to be a modification of the first output. This is a result of the fact that the first modification is written over the original aperture.

2. If the response to the question

   HOW MANY ELEMENTS CHANGED?

   in step 12 is less than zero the program jumps to step 15. If the response is greater than 100 the program will return to the beginning of step 12.

3. This program can be executed only under Honeywell GCOS TSS YFORT subsystem.

PROGRAM PDFESTR

1. PURPOSE

The program PDFESTR generates a histogram of the radiating elements in a statistically loaded aperture. The width of each annulus may be varied and the center of the annuli may be specified.

II. PERIPHERAL DEVICES REQUIRED

LUD	NAME	USE
01	User PRMFL	Input Aperture
05	Time sharing terminal keyboard	User Input
06	Time sharing terminal printer	Computer Output

III. OPERATING PROCEDURES

1. Enter TSS YFORT

2. Type:

   RUN PDFESTR

3. The message

   LRJ, LRK

   is printed.

   LRJ - Number of blocks in the x-direction (horizontally)

   LRK - Number of blocks in the y-direction (vertically)

4. The message

OFSTJ, OFSTK

is printed.

OFSTJ - Value added to the calculated center of the aperture to offset the origin horizontally.

OFSTK - Value added to the calculated center of the aperture to offset the origin vertically.

5. The message

INPUT FILE

is printed.

The user should respond with an existing PRMFL which contains a statistically loaded aperture. Follow the entry with a semicolon(;).

Example:

=/SUBCAT$PSWRD/FILENAM$PSWRD;

If the file name is improper then the message

ATTACH FAILED ISTAT = X

is printed. Where X is the first status word returned by the File Management Supervisor (see the Time Sharing System Programmers Reference

Manual, BR39, p.(3-39) or Table I, herein) or will contain:

1 - file is currently open

2 - teletypewriter requested in batch mode (illegal)

3 - additional memory needed, request denied (time sharing user will be terminated)

4 - CATFIL all blanks

NOTE: See Honeywell series 600/6000 Fortran manual, BJ67, p. (6-35) - p. (6-36) for more details on the subroutine ATTACH.

If the file name is unacceptable the program will return to the beginning of step 5. Otherwise, proceed to step 6.

6. The message

RINC, RLIM

is printed.

RINC - The incremental radius or annuli width used in generating the histogram values.

RLIM - The maximum radius of interest.

7. The message

ICON, NDPACK

is printed.

ICON - Mode flag

= -1 The program halts
= 0 Histogram data is normalized to give a unity cummulative distribution
= 1 Histogram data is converted to probability density estimate data. The new data is divided by the product of the cell width and the total number of elements.
= 2 Raw histogram data.

NDPACK - The number of incremental radius histogram cells, RINC, combined to make each output histogram cell.

8. The program repeats to step 7.

IV. SUBPROGRAMS REQUIRED

None

V. RESTRICTIONS, REQUIREMENTS, AND MISC. DATA

1. This estimator can only be used on apertures that have been statistically loaded.

2. The program TBLS can be used to compare the histogram values of an aperture with similar values calculated for the particular weighting function used to load the aperture. This comparison gives some degree of 'goodness' for the particular load.

PROGRAM PLARY

I PURPOSE

The program PLARY loads a PRMFL with a user specified antenna aperture illumination function.

II. PERIPHERAL DEVICES REQUIRED

LUD	NAME	USE
01	User PRMFL	Aperture Output
05	Time sharing terminal keyboard	User Input
06	Time sharing terminal printer	Computer Output

III. OPERATING PROCEDURES

1. Enter TSS YFORT

2. Type:

   RUN PLARY

3. The message

   OUTPUT FILE NAME

   is printed.

   The user should respond with an existing PRMFL name which is large enough to store the aperture. Follow the entry with a semicolon(;).

   Example:

   */SUBCAT$PSWRD/FILENAM$PSWRD;

If the file name is improper then the message

ATTACH FAILED ISTAT = X

is printed, where X is the first status word returned by the File Management Supervisor (see the <u>Time Sharing System Programmers Reference Manual</u>, BR39, p.(3-39) or Table 1 herein) or will contain:

1 = file is currently open

2 = teletypewriter requested in batch mode (illegal)

3 = additional memory needed, request denied (time sharing user will be terminated)

4 = CATFIL all blanks

NOTE: See Honeywell series 600/6000 Fortran manual, BJ67, p.(6-35) - p.(6-36) for more details on the subroutine ATTACH.

If the file name is unacceptable the program will return to the beginning of step 3. Otherwise proceed to step 4.

4. The message

STATISTICAL TAPER?

is printed. The user responds either Y (yes) or N (no).

YES The user wishes to load the array using a space tapered or 'thinned' loading technique. Proceed to step 5.

NO The user wishes to load the array using element to element amplitude tapering. Proceed to step 6.

5. The message

XKK, MAD1, JRND

is printed.

XKK - This value is the probability that an element will occur at the normalized peak of the design weighting function. (Usually XKK = 1.0).

MAD1 - The starting address for selecting random numbers from the random number array ($1 \leq MAD1 \leq 120$).

JRND - Random number generator initialization constant ($0 \leq JRND \leq 2^{36} - 1$).

6. The message

IAPTFL

is printed.

The user responds with a number which determines the shape of the aperture.

= 1 circular

= 2 elliptical

= 3 rectangular

7. If IAPTFL = 1, the message

XEDGE, XHOLE

is printed.

XEDGE = Outside radius of the aperture.

XHOLE = Radius of a hole centered in the aperture (For no hole, XHOLE = 0.0).

Proceed to step 10.

8. If IAPTFL = 2, the message

NMAJOR, NMINOR, XHOLE

is printed.

NMAJOR = Length of the semi-major elliptical axis.

NMINOR = Length of the semi-minor elliptical axis.

XHOLE = Radius of a hole centered at the intersection of the major and minor axes (For no hole, XHOLE = 0.0).

Proceed to step 10.

9. If IAPTFL = 3, the message

NWIDTH, NHIGH

is printed.

NWIDTH = Width of the rectangular aperture.

NHIGH = Height of the rectangular aperture.

10. The message

IWTFLG

is printed.

The user responds with a number which represents the desired weighting function.

= 0 no weighting function

= 1 cosine to a power on a pedestal

= 2 Blackman weighting function

= 3 Kaiser weighting function

= 4 triangular weighting function

= 5 Taylor weighting function

= 6 Bessel weighting function

= 7 cubic weighting function

= 8 Bayliss weighting function

11. If IWTFLG = 0, proceed to step 21.

12. If IAPTFL = 1, the message

WTRAD

is printed.

The user responds with the desired radius of the weighting function.

Proceed to step 14.

13. If IAPTFL = 2, or IAPTFL = 3, the message

ZJRAD, ZKRAD

is printed.

ZJRAD - Half the span of the weighting function in the x-direction.

ZKRAD - Half the span of the weighting function in the y-direction.

NOTE: These refer to the elliptical and rectangular weighting functions which are products of the orthogonal weighting functions.

14. If IWTFLG = 1, the message

WTPED, NWTPOW

is printed.

WTPED - The height of the pedestal.

NWTPOW - The power of the cosine.

Proceed to step 21.

15. If IWTFLG = 2, proceed to step 19.

16. If IWTFLG = 3, the message

WKASIR

is printed.

WKASIR The Kaiser variable for the trade-off between mainlobe width and sidelobe amplitude.

Proceed to step 21.

17. If IWTFLG = 4, proceed to step 21.

18. If IWTFLG = 5, the message

DB, NBAR

is printed.

DB - Sidelobe in dB with reference to the main lobe.

NBAR - Number of zeros used in approximating the Dolph-Chebyschev weighting distribution.

Proceed to step 21.

19. If IWTFLG = 6, the message

BESCAL, BESEDG

is printed.

BESCAL - The maximum amplitude at the center of the aperture.

BESEDG - The scale factor used in calculating the argument for evaluating the Bessel function for the actual radial location on the aperture.

Proceed to step 21.

20. If IWTFLG = 7, the message

CUBK, WTRAD

is printed.

CUBK - The amplitude scaling constant.

WTRAD - The half span of the weighting function.

21. The message

NBITS

is printed.

NBITS - The number of bits used to control the digital phase shifters.

22. The message

ANY BEAM STEERING?

is printed.

The user responds with a Y(yes) or an N(no).

23. If the response in step 22 is Y the message

DELPHJ, DELPHK

is printed.

DELPHJ - The beam steering in degrees in the x-direction.

DELPHK - The beam steering in degrees in the y-direction.

If the response to step 22 is N proceed to step 24.

24. The message

QUADRATIC ERROR?

is printed.

The user responds Y(yes) or N(no).

25. If the response to step 24 is Y, the message

PHERX, PHERY

is printed.

PHERX - The maximum phase error in degrees at the edge in the x-direction.

PHERY - The maximum phase error in degrees at the edge in the y-direction.

If the response to step 24 is N then proceed to step 26.

26. The message

BESSELL ERROR?

is printed.

The user responds Y(yes) or N(no).

27. If the response to 26 is Y, the message

BESERR, BSCAL

is printed.

BESERR - The maximum phase error in degrees at the center of the aperture.

BSCAL - The scaling factor used in calculating the argument for evaluating the Bessell function from the actual radial location on the aperture.

28. The message

LRJ, LRK

is printed.

LRJ - The number of blocks in the x-direction.

LRK - The number of blocks in the y-direction.

29. A message is now printed which shows the user what values he assigned to the program parameters and will be used in calculating the aperture illumination pattern.

30. The message

DETACH ISTAT = X

is printed after the aperture is loaded. X = 0 if the detach was successful; otherwise it is 1.

31. Program aperture terminates and the user is returned to build mode under TSS YFORT. This is indicated by an asterisk (*).

IV. SUBPROGRAMS REQUIRED

EXPND
BESS
CAM
WEIGHT

V. RESTRICTIONS, REQUIREMENTS, AND MISC. DATA

1. If the weighting function span is less than the aperture span then those elements outside the weighting function span are set to zero.

2. The Kaiser variable, WKASIR, should be within the range

   2 ≤ WKASIR ≤ 10

3. The number of zeros, NBAR, in the Taylor approximation must be in the range

   3 ≤ NBAR ≤ 20

4. LRJ and LRK must be even or the message LRJ,LRK will be retyped and the user must respond with two even numbers.

5. It should be noted that if the user has specified a rather complex aperture illumination pattern (Example: a large circular aperture (XEDGE = 40.0) with a Taylor distribution (NBAR = 20)) the time required to load may be quite long (perhaps 5 minutes).

6. This program can be executed only under Honeywell GCOS TSS YFORT subsystem.

PROGRAM PLTDVR

I. PURPOSE

The program PLTDVR formats either an aperture illumination or a far-field energy distribution that is stored in a permanent disk file (PRMFL) for making pseudo-3d plots using the DUIS. The data is formatted and transmitted to the DUIS for recording and subsequent production of 3d plots.

II. PERIPHERAL DEVICES REQUIRED

LUD	NAME	USE
01	Input PRMFL	Aperture or far-field input
05	Time sharing terminal keyboard	User Input
06	Time sharing terminal printer	Computer Output

III. OPERATING PROCEDURE

Load RJE program into DUIS and start program execution. See User's Manual on RJE-300 or RJE-1200.

1. Enter TSS YFORT

2. Type

   RUN PLTDVR

3. The message

   INPUT FILE NAME?

   is printed. The user should respond with the name of an existing file which contains the data to be formatted and recorded. The file name must be followed with a semicolon(;). If the user types STOP the program will terminate execution and the user is returned to build mode under TSS YFORT. This is indicated by an asterisk(*).

4. If the file name is not acceptable the message

UNSUCCESSFUL ATTACH ISTAT = X

is printed, where X is the first status word returned by the File Management Supervisor (see the Time Sharing System Programmers Reference Manual, BR39, p.(3-39), or Table 1 herein) or will contain:

1 = file is currently open

2 = teletypewriter requested in batch mode (illegal)

3 = additional memory needed, request denied (time sharing user will be terminated)

4 = CATFIL all blanks

5. If the file name in step 3 is accepted the message

JWID, JWIDSP, NBMAX

is printed.

JWID - The width in blocks of the part of the array to be formatted and transferred

JWIDSP - The width in blocks of the total input array

NBMAX - The logical record number of the last block to be transferred.

If JWIDSP is greater than 10 the value of JWIDSP is set to 10 and the following warning message is printed:

JWIDSP > 10...SET TO 10

If JWIDSP is greater than JWID, the value of JWIDSP is set to JWID and the following warning message is printed:

JWIDSP > JWID...SET JWIDSP=JWID

6. The message

Enter FIRST LREC, ISCN

is printed.

FIRST LREC - The logical record of the block in the upper left corner of the part of the array to be formatted and transferred. If the response is -1 the program transfers to step 8.

ISCN - = 0 data is processed for magnitude plotting

= 1 real component is processed for bipolar plotting.

7. The data transmission to the DUIS now begins. When all of the requested data has been transmitted the Tektronix display will beep once and erase. The user should then press carriage return and the following message is printed:

TH = X

where X is the maximum value of all the transmitted data. This value is used in scaling the plots produced using the DUIS subprogram H3DPL. The program now returns to step 6.

8. The message

DETACH ISTAT = X

is printed, where X = 0 if the detach is successful and X = 1 otherwise. The program returns to step 3.

IV. SUBPROGRAMS REQUIRED

PLOTD
CHOP

V. RESTRICTIONS, REQUIREMENTS, AND MISC. DATA

1. This program can be executed only under Honeywell GCOS TSS YFORT subsystem.

2. The transmission time for a 64 x 64 point plot is about 10-15 minutes at 300 baud. This should be kept in mind if larger plots are to be attempted.

3. If the response to FIRST LREC is greater than the response to NBMAX the program will repeat step 6.

4. The magnitude of the main lobe determines the value of the least significant bit (LSB). If the ratio of mainlobe to sidelobe level exceeds the accuracy of the 12 bit word length, errors in magnitude representation result. The dynamic range of the 12 bit word is 72 dB (6 dB/bit). Magnitude errors begin to appear if the sidelobes are more than 60 dB down.

PROGRAM RNDERR

I. PURPOSE

The RNDERR program modifies an antenna aperture distribution (resident in PRMFL) by adding a phase error distribution specified by the user. The resulting aperture is stored in another PRMFL where it can be accessed for further processing.

II. PERIPHERAL DEVICES REQUIRED

LUD	NAME	USE
01	User PRMFL	Input Aperture
02	User PRMFL	Output Aperture
05	TSS terminal keyboard	User Input
06	TSS terminal printer	Computer Output

III. OPERATING PROCEDURE

1. Enter TSS YFORT

2. Type

   RUN RNDERR

3. The message

   NTYPE, MAD1, JRND, LRJ, LRK

   is printed.

   NTYPE - Determines the type of error distribution to be added to the ape-ture.

= 1 uniform distribution

= 3 Gaussian distribution

MAD1 - The starting address for selecting random numbers from the random number array. ($1 < MAD1 \leq 128$)

JRND - Random number generator initialization constant. ($0 < JRND \leq 2^{36} - 1$)

LRJ - The number of blocks in the x-direction of the input aperture.

LRK - The number of blocks in the y-direction of the input aperture.

4. If NTYPE = 1, the message

UMEAN, UUEXT

is printed.

UMEAN - Mean value of the uniform distribution in degrees.

UUEXT - Width of the uniform distribution in degrees.

Proceed to step 6.

5. If NTYPE = 3, the message

XMEAN, SIGMA

is printed.

XMEAN - Mean value of the Gaussian distribution in degrees.

SIGMA - Standard deviation of the Gaussian distribution in degrees.

6. The message

INPUT FILE NAME

is printed. The user should respond with an existing PRMFL name which has the input aperture stored in it followed by a semicolon(;). If the file name is not acceptable the following message will be printed:

UNSUCCESSFUL ATTACH ISTAT = X

where X is the first status word returned by the File Management Supervisor (see the Time Sharing System Programmers Reference Manual, BR39, p.(3-39) or Table 1 herein) or will contain:

1 = file is currently open

2 = teletypewriter requested in batch mode (illegal)

3 = additional memory needed, request denied (time sharing user will be terminated)

4 = CATFIL all blanks

NOTE: See Honeywell Series 600/6000 Fortran manual, BJ67, p.(6-35) - (6-36), for more details on the subroutine ATTACH.

If the input file name is unacceptable the program returns to the beginning of step 6. Otherwise, proceed to step 7.

7. If the PRMFL name input in step 6 is acceptable the message

   OUTPUT FILE NAME

   is printed. The user should respond with an existing PRMFL name in which the output aperture is to be stored, followed by a semicolon(;). If the PRMFL name is not acceptable the same procedure as described in step 6 applies here and the program will return to the beginning of step 7. Otherwise, proceed to step 8.

8. The antenna aperture distribution is modified and stored in the specified output PRMFL.

9. When the processing is finished the message

   DETACH ISTAT = X

   is printed twice. The first time is for the input PRMFL and the second is for the output PRMFL. If X = 0 then the detach is successful, otherwise X = 1.

10. Program execution terminates and the user is returned to build mode under TSS YFORT. This is indicated by an asterisk(*).

IV. SUBPROGRAMS REQUIRED

RRAND

V. RESTRICTIONS, REQUIREMENTS, AND MISC. DATA

1. If MAD1 is outside of the range $1 < MAD1 \leq 128$ the computer will print the message of step 3 again and the user must respond correctly.

2. This program can be executed only under Honeywell GCOS TSS YFORT subsystems.

3. The input PRMFL and the output PRMFL can be the same file.

PROGRAM RTI4

I. PURPOSE

The program RTI4 is a subroutine which converts data amplitude to letters of the alphabet. The resulting characters are organized into a matrix which is printed on a remote terminal. Range is usually shown vertically and cross range is displayed horizontally with intensity displayed by the character placed in the cell, e.g. 0 represents 0 dB with respect to the reference, A represents -10 dB with respect to the reference, Z represents -36 dB with respect to the reference. This subroutine processes the same type of data as PLTDVR but does so in a much more compact form.

II. PERIPHERAL DEVICES REQUIRED

LUD	NAME	USE
01	Input PRMFL (permanent disk file)	Aperture far-field input
05	Time sharing terminal keyboard	User Input
06	Time sharing terminal printer	Computer Output

III. OPERATING PROCEDURE

1. Enter TSS YFORT

2. Type

   RUN RTI4

3. The message

   INPUT DESIRED FILE NAME

   is printed.

   The user should respond with the name of an existing PRMFL which contains the data to be displayed. The file name must be followed by a semicolon(;). If the user types STOP the program jumps to step 9.

4. If the PRMFL name is not acceptable the following message is printed:

   UNSUCCESSFUL ATTACH ISTAT = X

   where X is the first status word returned by the File Management Supervisor (see the <u>Time Sharing System Programmers Reference Manual</u>, BR39, p.(3-39) or Table 1, herein) or will contain:

   1 = file is currently open

   2 = teletypewriter requested in batch mode (illegal)

   3 = additional memory needed, request denied (time sharing user will be terminated)

   4 = CATFIL all blanks

   NOTE: See Honeywell Series 600/6000 Fortran manual, BJ67, p.(6-35) - (6-36) for more details on the subroutine ATTACH. After the message is printed the program returns to step 3 and begins again.

5. If the PRMFL name in step 3 is accepted the message

   FLOOR, YINC, JWID, NBMAX

   is printed.

   FLOOR - The reference in dB below which everything is represented in dashes (-) on the RTI plot. All the data with a greater value is represented by a letter, number, or punctuation symbol and the value relative to the FLOOR is calculated using YINC.

   YINC - The increment in dB between each successive letter, number, or symbol is determined by the value assigned to YINC. See Table 2.

   JWID - The width in blocks across the whole side of the data array as determined by the parameter LRJWID in the program FFT2DX or LRJ in PLARY, RNDERR, or FILMOD.

   NBMAX - The number of the last block to be displayed. The final display will be 4 blocks wide and as long as the user chooses depending on the value of NBMAX.

   NOTE: Two examples of the relative dB values for a specified FLOOR and YINC are shown in Table 2.

6. The message

   Enter FIRST LREC, DISPLAY WIDTH

is printed. The user should respond with the number of the block that he wishes to be placed in the upper left hand corner of the RTI and the width of the RTI in samples. The maximum number of samples is 128. If the user types -1 then the program goes to step 8.

7. The computer now begins to transmit the RTI display to the DUIS or a time-sharing terminal printer and will continue until it reaches the specified stopping block number, NBMAX.

8. The message

   DETACH ISTAT = X

   is printed where X = 0 if the detach is successful and X = 1 otherwise. Go to step 3.

9. An asterisk (*) is printed and the program is finished.

IV. SUBPROGRAMS REQUIRED

None

V. RESTRICTIONS, REQUIREMENTS, AND MISC. DATA

1. This program can be executed only under Honeywell GCOS TSS YFORT subsystems.

2. If the response to

   INPUT LREC DESIRED

   is greater than NBMAX then the program will repeat step 6.

3. Any dB level below the value of FLOOR will be represented by a dash (-). Any dB level above the value calculated for '$' will be represented by a '$'.

TABLE 2

EXAMPLES OF RELATIVE VALUE OF LETTERS, NUMBERS, AND SYMBOLS

FLOOR = -20.0
YINC = 1.0

FLOOR = 10.0
YINC = 0.5

dB level (FLOOR = -20.0, YINC = 1.0)	Character	dB level (FLOOR = 10.0, YINC = 0.5)
20.0	$	30.0
19.0	.	29.5
18.0	*	29.0
17.0	+	28.5
16.0	0	28.0
15.0	1	27.5
14.0	2	27.0
13.0	3	26.5
12.0	4	26.0
11.0	5	25.5
10.0	6	25.0
9.0	7	24.5
8.0	8	24.0
7.0	9	23.5
6.0	A	23.0
5.0	B	22.5
4.0	C	22.0
3.0	D	21.5
2.0	E	21.0
1.0	F	20.5
0.0	G	20.0
-1.0	H	19.5
-2.0	I	19.0
-3.0	J	18.5
-4.0	K	18.0
-5.0	L	17.5
-6.0	M	17.0
-7.0	N	16.5
-8.0	O	16.0
-9.0	P	15.5
-10.0	Q	15.0
-11.0	R	14.5
-12.0	S	14.0
-13.0	T	13.5
-14.0	U	13.0
-15.0	V	12.5

dB level

dB level

-16.0	W	12.0
-17.0	X	11.5
-18.0	Y	11.0
-19.0	Z	10.5
-20.0	-	10.0

PROGRAM TBLS

I. PURPOSE

The program TBLS computes and tabulates sample values of selected weighting functions. The program also generates data which is used in checking the probability density function of space tapered arrays.

The program may be used in one of the following three modes:

1. TBLS generates the value of the weights for specific distribution width and weighting function type.

2. For the Taylor and Bayliss functions, the following three modes are available (3):

   (a) For a specific distribution width TBLS generates all the sample weights for sidelobe levels from 20 to 90 dB in steps of 5 dB with $\bar{n}$ ranging from 3 to 20.

   (b) For a specific distribution width and dB level, TBLS generates the sample weights with $\bar{n}$ ranging from 3 to 20.

   (c) For a specific distribution width, dB level, and $\bar{n}$, TBLS generates the sample weighting function.

This program produces tables similar to Hansen's (1) but with more flexibility, greater accuracy, and greater range in dB and $\bar{n}$. Tables may be generated for the Bayliss as well as the Taylor distribution.

3. For all of the above modes TBLS generates data which is either in the form of standard weights or in a form which may be used to compare with

the data generated by PDFESTR to check the density function of space tapered arrays.

II. PERIPHERAL DEVICES REQUIRED

LUD	NAME	USE
05	TSS terminal keyboard	User Input
06	TSS terminal printer	Computer Output

III. OPERATING PROCEDURE

1. Enter TSS YFORT

2. Type

   RUN TBLS

3. The message

   PDFESTR DATA OR TABLES? (0 or 1)

   is printed.

   = 0 The user wishes to generate data that will be compatible with the data from PDFESTR.

   = 1 The user wishes to generate tables of weighting function values.

4. The message

   NTYPE

   is printed.

   The user should respond with a number which determines the weighting function type.

= 1 cosine on a pedestal to a power

(Proceed to 5)

= 2 Blackman

(Proceed to 6)

= 3 Kaiser

(Proceed to 7)

= 4 Bartlett or triangular

(Proceed to 6)

= 5 Taylor

(Proceed to 10)

= 6 Bessel

(Proceed to 8)

= 7 Cubic

(Proceed to 9)

= 8 Bayliss

(Proceed to 10)

5. The message

WTPED, NWTPOW, IRAD, WTRAD

is printed.

WTPED - The height of the pedestal

NWTPOW - The power of the cosine function

IRAD - The radius (or half span of a linear array), in units of elements, of the array.

WTRAD - The radius (or half span of a linear array) of the weighting function.

NOTE: For all subsequent entries, IRAD and WTRAD have the same meaning as above.

Proceed to 15.

6. The message

IRAD, WTRAD

is printed.

Proceed to 15.

7. The message

WKASIR, IRAD, WTRAD

is printed.

WKASIR - The Kaiser variable for the trade-off between main lobe width and sidelobe amplitude.

Proceed to 15.

8. The message

BESEDC, BESCAL, IRAD, WTRAD

is printed.

BESEDG - The scale factor used in calculating the argument used in evaluating the Bessel function from the actual radial location on the aperture.

BESCAL - The maximum amplitude at the center of the aperture.

Proceed to 15.

9. The message

CUBK, IRAD, WTRAD

is printed.

CUBK - The amplitude scaling constant

Proceed to 15.

10. The message

IRAD, WTRAD

is printed.

= 0 The user wishes to generate the complete set of tables for IRAD with NBAR and DB varied.

$3 \leq NBAR \leq 20$ (19 for the Bayliss)

and $20 \leq DB \leq 80$

in steps of 5 dB.

Proceed to 15.

= 1 The user wishes to choose one dB level of interest. Proceed to 12.

12. The message

    IDB

    is printed. The user should respond with an integer value for the specified dB level.

13. The message

    ALL OR SINGLE NBAR? (0 or 1)

    is printed.

    = 0 The user wishes to generate for the specified IRAD and dB level all possible NBAR distributions in the range

    3 ≤ NBAR ≤ 20 (19 for the Bayliss)

    Proceed to 15.

    = 1 The user wishes to choose one NBAR of interest.

    Proceed to 14.

14. The message

    NBAR

    is printed. The user should respond with an integer number for the specific value of NBAR desired. If NBAR is too small, as explained by Taylor (2) and Hansen (1), the message

    INVALID VALUE FOR NBAR

    is printed and the program repeats step 14.

15. The program now generates the appropriate distribution and prints the tables on the TSS terminal printing device. The program returns to step 4 when the requested distribution has been printed.

IV. SUBPROGRAMS REQUIRED

EXPND
BESS
GAM
WEIGHT

V. RESTRICTIONS, REQUIREMENTS, AND MISC. DATA

1. The range of NBAR must be

$$3 \le NBAR \le 20$$

for the Taylor and

$$3 \le NBAR \le 19$$

for the Bayliss.

Reference

1. Hansen, R.C., 'Tables of Taylor Distributions for Circular Aperture Antennas,' IEEE Trans. Anten. Prop., AP-8, #1, (1/60), pp. 23-26.

2. Taylor, T. T., 'Design of Circular Apertures for Narrow Beamwidth and Low Sidelobes,' The Bell System Technical Journal, Vol. 47, No. 5, (May-June 1968), pp. 623-651.

APPENDIX 5

COMPUTER PROGRAM DOCUMENTATION

In this appendix the computer program documentation (CPD) for the PAAS modules is presented. The CPD's are arranged in alphabetical order and each is self-contained. The subprograms BESS and CAM were obtained from RADSIM for use in PAAS. For the reader's convenience and completeness of this document these have been incorporated into this appendix.

Unless otherwise stated, all software documented herein is stored under user master catalog 'BECACD01' in the RADC H6180 GCOS system. The source code for all programs herein are stored in PRMFL's having the same name as the program, e.g. the source code of the program PLARY and all required subroutines is stored in a PRMFL having the name PLARY under user master catalog BECACD01.

The documentation for each PAAS program presented in this appendix is divided into six sections. The order and title of each section is as follows:

1. Purpose

2. Input Parameters

3. Restrictions, Requirements, Miscellaneous data

4. Subprograms Required

5. Theory of Operation

6. FORTRAN Listing

The content of each section is explained in the following paragraphs.

1. Purpose

This section contains a brief description of the purpose of the software module.

2. Input Parameters

This section lists all of the input parameters for the particular software module. Both required and optional input parameters are listed. Each parameter entry is broken into four groups of information and placed into columns for easy reference. The first column contains the parameter name as it appears in the software. The second column tells whether the parameter is required or optional. An R in the second column denotes a required parameter while an O denotes an optional parameter. The variable type, either integer or floating point, is noted in the third column. An I denotes and integer type while an F denotes a floating point variable. The fourth column contains a brief description of the parameter and how it is used in the program.

3. Restrictions, Requirements, Miscellaneous Data

In this section special notes concerning the input parameters, use of the program, potential usage problems, etc. are discussed.

4. Subprograms Required

In this section the subprograms required for the PAAS program are listed. Both subroutine and function subprograms are included.

5. Theory of Operation

In this section the theory behind the programming is discussed using the variable names and notation as they appear in the program. This helps the user in understanding the operation of the program.

6. FORTRAN Source Code Listing

This section contains a listing of the FORTRAN source code.

The documentation for each PAAS subprogram presented in this appendix is divided into seven sections. The order and title of each section is as follows:

1. Purpose
2. Input Parameters
3. Calling Sequence
4. Restrictions, Requirements, Miscellaneous Data
5. Subprograms Required
6. Theory of Operation
7. FORTRAN Source Code Listing

These are the same as those that were previously described for the PAAS program CPD's except that the section entitled 'Calling Sequence' has been added. A description of that section follows:

3. Calling Sequence

This section presents an example of a typical FORTRAN call for the module. For function subprograms the example calling sequence is shown as an assign statement but of course the function reference can be embedded in a FORTRAN arithmetic statement.

FUNCTION BESS

1. MODULE IDENTIFICATION

Name	Classification Code	Reference Number
BESS	Subordinate	Not User Referenced

2. PURPOSE

This function is used to compute the value of a Bessel function.

3. INPUT PARAMETERS

Name	O/R	T	Description
O	R	F	The order of the Bessel function
Z	R	F	The argument of the Bessel function.

4. CALLING SEQUENCE

BS = BESS(O,Z)

Where: O,Z are the Input arguments
BS contains the computed value of the Bessel function

5. RESTRICTIONS, REQUIREMENTS, MISCELLANEOUS DATA

a. This subprogram was obtained from the Computer Program Documentation for AF Contract F30602-67-C-0074.

b. External References:

CAM
DABS

c. Referenced labeled common areas:

None

6. FORTRAN LISTING

```
C
C
       FUNCTION BESS(O,Z)
      DOUBLE PRECISION A1,A2,BS,ADD,GAM,G
C
      IF(Z.NE.0.0) GO TO 40
      IF(O.EQ.0.0) BS=1.0
      IF(O.NE.0.0) BS=0.0
      GO TO 100
   40 SMALL=1.0E-8
      IF(O)100,31,32
   31 BS=1.0
      AKV=0.0
      A1=1.0
   52 AKV=AKV+1.0
      A1=A1*(-1.0)*(Z/2.0)**2/(AKV*AKV)
      BS=BS+A1
      IF(DABS(A1/BS)-SMALL)51,52,52
   51 GO TO 100
   32 A=O
      N=0
   13 IF(A-1.0)10,12,12
   12 A=A-1.0
      N=N+1
      GO TO 13
   10 ARG=A
       G=GAM(ARG)
      A2=1.0
      IF(N)100,76,75
   75 DO 26 NV=1,N
      AJ=NV-1
      IF(A2.GT.1.0E-38) GO TO 26
      BS=0.0
      GO TO 100
   26 A2=A2*(Z/2.0)/(O-AJ)
   76 A2=A2*(Z/2.0)**ARG/G
      BS=A2
      AKV=0.0
      A1=1.0
```

```
   21 AKV=AKV+1.0
      A1=A1*(-1.0)*(Z/2.0)**2/(AKV*(AKV+O))
      ADD=A1*A2
      BS=BS+ADD
      IF(BS.EQ.0.0) GO TO 21
      TEST=DABS(ADD/BS)
      IF(TEST-SMALL)100,21,21
  100  BESS=BS
      RETURN
      END
```

SUBROUTINE CHOP

1. PURPOSE

The purpose of this subroutine is to convert an integer number into two ASCII characters. The ASCII characters are packed, right-adjusted, into an output word.

2. INPUT PARAMETERS

Name	O/R	T	Description
IDAT	R	I	The integer word to be processed
IOUT	R	I	The output word containing two ASCII characters.

3. CALLING SEQUENCE

CALL CHOP(IDAT,IOUT)

4. RESTRICTIONS, REQUIREMENTS, MISCELLANEOUS DATA

a. The input word, IDAT, must be in the following range:

$-2^{11} < \text{IDAT} < 2^{11}-1$

5. SUBPROGRAMS REQUIRED

None

6. THEORY OF OPERATION

The input word, IDAT, contains at most 12 significant bits. The rightmost 12 bits are extracted from IDAT and separated into two 6 bit characters, I1 and I2. The characters I1 and I2 have values which range from 0 to 63 and include the ASCII control character region from Ø to 31. In order to ensure that these characters cannot have values in the control character region, the number 32 is added to each. If this is not done, problems arise with the H6180 TSS processing. These characters are packed, right-adjusted, into the output word, IOUT, and control returns to the calling (sub)program.

7. <u>FORTRAN LISTING</u>

```
SUBROUTINE CHOP(IDAT,IOUT)
I1=FLD(24,6,IDAT)
I2=FLD(30,6,IDAT)
IOUT=0
FLD(18,9,IOUT)=I1+32
FLD(27,9,IOUT)=I2+32
RETURN
END
```

SUBROUTINE CZFFT

1. PURPOSE

This subroutine performs the inverse discrete Fourier transform of a sequence of input data samples. The mechanization is based on the Fast Fourier Transform (FFT) algorithm developed by Langdon and Sande from the approach of J. W. Tukey and J. Cooley. The subroutine described herein has been structured to facilitate the efficient computation of 2-dimensional discrete inverse Fourier transforms.

2. INPUT PARAMETERS

Name	O/R	T	Description
N2	R	I	Power of 2 which determines the total number of points (NTHPOW) transformed by the FFT
IOFST	R	I	Offset of the first sample to be transformed from the front of the array S
IHOP	R	I	Power of 2 which determines the spacing between the samples (NHOP) to be transformed.

3. CALLING SEQUENCES

CALL CZFFT(S,N2,IOFST,IHOP)

Where: S is a complex array containing the data to be processed. The output samples are placed into the array S also.

4. RESTRICTIONS, REQUIREMENTS, MISCELLANEOUS DATA

a. The maximum value of N2 is 11, which gives 2048 samples.

b. In order to minimize the CPU time requirements of this subroutine, a complex exponential look up table is used.

c. Source PRMFL:

BECAVU01/SUPORT$JR/SCZFFT

d. Object PRMFL:

BECAVU01/SUPORT$JR/OCZFFT

5. SUBPROGRAMS REQUIRED

COS
SIN

6. THEORY OF OPERATION

Refer to RADSIM-CPD-ZFFT.

7. FORTRAN LISTING

```
      SUBROUTINE CZFFT(S,N2,IOFST,IHOP)
C
C      S >> /SUPORT/SCZFFT
C      O >> /SUPORT/OCZFFT
C
      DATA PI2/6.28318531/
      DATA IFLAG/0/,NHOPO/-1/
      COMPLEX S(1),C,C1,C2,C3,C4,XEXP(1536)
      REAL I,I1,I2,I3,I4,RX(2)
      INTEGER PASS, SEQLOC, L(15)
      EQUIVALENCE (J,JI),(PASS,J6),(NXTLTH,J7),
     *            (LENGTH,J8),(SEQLOC,J9),(ISCALE,J10),
     *            (IARG,J11),(A1,J12),(RX(1),I4,C),
     *            (RX(2),R4)
      EQUIVALENCE (L15,L(1)),(L14,L(2)),(L13,L(3)),
     *            (L12,L(4)),(L11,L(5)),(L10,L(6)),
     *            (L9,L(7)),(L8,L(8)),(L7,L(9)),
     *            (L6,L(10)),(L5,L(11)),(L4,L(12)),
     *            (L3,L(13)),(L2,L(14)),(L1,L(15))
      NHOP=2**IHOP
      IOFST1=IOFST+1
C
C     IF IFLAG=0 THEN LOAD THE COMPLEX
C     EXPONENTIAL TABLE, XEXP
C
      IF(IFLAG.EQ.1) GO TO 502
      DARG=PI2/2048.0
      ARC=0.0
      DO 500 J=1,1536
      ARC=ARC+DARC
      XEXP(J)=CMPLX(COS(ARC),SIN(ARC))
  500 CONTINUE
      IFLAG=1
  502 IF(NHOP.EQ.NHOPO) GO TO 503
      DO 6 J=1,15
      L(J)=NHOP
  6   IF(J.LE.N2) L(J)=(2**(N2+1-J))*NHOP
      NTHPOW = 2** N2;NHOPO=NHOP
```

```
      N4POW =  N2  /2
 503  NTTL=NTHPOW*NHOP
      IF(N4POW.EQ.0) GO TO 3
C
C     PERFORM RADIX 4 TRANSFORM
C
      DO 2 PASS=1,N4POW
      NXTLTH=2**( N2  -2*PASS)
      LENGTH=4*NXTLTH
      IDEL=2048/LENGTH
      IADDH=NXTLTH*NHOP
      LENGTH=LENGTH*NHOP
      DO 2 J=1,NXTLTH
      IARG1=(J-1)*IDEL
      IARG2=IARG1+IARG1
      IARG3=IARG2+IARG1
      MLOC=IOFST1-LENGTH+(J-1)*NHOP
      DO 2 SEQLOC=LENGTH,NTTL,LENGTH
      J1 = SEQLOC+MLOC
      J2 = J1+IADDH
      J3 = J2+IADDH
      J4 = J3+IADDH
      C1=S(J1)+S(J3)
      C2=S(J1)-S(J3)
      C3=S(J2)+S(J4)
      C=S(J2)-S(J4)
      C4=CMPLX(-R4,I4)
      S(J1)=C1+C3
      IF(J.EQ.1) GO TO 1
      S(J3)=XEXP(IARG1)*(C2+C4)
      S(J2)=XEXP(IARG2)*(C1-C3)
      S(J4)=XEXP(IARG3)*(C2-C4)
      GO TO 2
 1    S(J3)=C2+C4
      S(J2)=C1-C3
      S(J4)=C2-C4
    2 CONTINUE
C
C     PERFORM RADIX 2 TRANSFORM IF REQUIRED
C
    3 IF( N2  .EQ.2*N4POW) GO TO 5
```

```
      NHOP2=NHOP*2
      NSTOP=NTTL+IOFST
      DO 4 J=IOFST1,NSTOP,NHOP2
      C=S(J)+S(J+NHOP)
      S(J+NHOP)=S(J)-S(J+NHOP)
    4 S(J)=C
    5 CONTINUE
C
C     OUTPUT CURRENTLY IS ORGANIZED WITH
C     BIT REVERSED ADDRESSING
C     THIS SECTION PLACES OUTPUT IN THE
C     CORRECT ORDER
C
      IJ=1
      J1=1
      DO 7 J2=J1,L2,L1
      DO 7 J3=J2,L3,L2
      DO 7 J4=J3,L4,L3
      DO 7 J5=J4,L5,L4
      DO 7 J6=J5,L6,L5
      DO 7 J7=J6,L7,L6
      DO 7 J8=J7,L8,L7
      DO 7 J9=J8,L9,L8
      DO 7 J10=J9,L10,L9
      DO 7 J11=J10,L11,L10
      DO 7 J12=J11,L12,L11
      DO 7 J13=J12,L13,L12
      DO 7 J14 =J13,L14,L13
      DO 7 JI=J14,L15,L14
      IF(IJ.GE.JI) GO TO 7
      KJ=IJ+IOFST
      JK=JI+IOFST
      C=S(KJ)
      S(KJ)=S(JK)
      S(JK)=C
    7 IJ=IJ+NHOP
      J1=NTTL/2+IOFST
      J2=J1+1
      DO 14 J3=IOFST1,J1,NHOP
      C=S(J3)
      S(J3)=S(J2)
```

```
      S(J2)=C
      J2=J2+NHOP
14    CONTINUE
      RETURN
      END
```

FUNCTION EXPND

1. PURPOSE

The function EXPND is used to compute the value of a Bessel function of the first kind and the zeroth order.

2. INPUT PARAMETERS

Name	O/R	T	Description
X2	R	F	Independent variable.

3. CALLING SEQUENCE

VAR=EXPND(X2)

Where: X2 is the Input argument
VAR contains the computed value of the Bessel function

4. RESTRICTIONS, REQUIREMENTS, MISCELLANEOUS DATA

a. The Bessel function expansion used herein was obtained from:

Rabiner, L. R., Gold, B., Theory and Application of Digital Signal Processing, Englewood Cliffs, NJ: Prentice-Hall, Inc., 1975, pp. 88-105.

5. SUBPROGRAMS REQUIRED

None

6. THEORY OF OPERATION

EXPND evaluates the series expansion of the Bessel function of the first kind and order zero with an imaginary argument. The series is shown below (4.1).

$$EXPND(X2) = 1+\left(\frac{X2}{2}\right)^2+\frac{(X2/2)^4}{(2!)^2}+\frac{(X2/2)^6}{(3!)^2}+\frac{(X2/2)^8}{(4!)^2}+\ldots \qquad (4.1)$$

7. FORTRAN LISTING

```
      FUNCTION EXPND(X2)
      XB2SQ=X2*X2*0.25
      SUM=1.0+XB2SQ
      ADDON=XB2SQ
      DO 110 J=2,20
      AJ=FLOAT(J)
      IF(ABS(ADDON).LT.ABS(SUM*1.0E-06)) GO TO 200
      ADDON=ADDON*XB2SQ/(AJ*AJ)
  110 SUM=SUM+ADDON
  200 EXPND=SUM
      RETURN
      END
```

SUBROUTINE FFT2D

1. PURPOSE

This subroutine computes the two-dimensional discrete Fourier transform of a planar array of samples.

2. INPUT PARAMETERS

Name	O/R	T	Description
N2	R	I	The power of 2 which defines the length of each side of the 2-D transform.
LRJ	R	I	The number of logical record blocks in the x-direction of the input file (horizontally).
LRK	R	I	The number of logical record blocks in the y-direction of the input file (vertically).
LRJIN	R	I	The number of blocks to be skipped in the x-direction, starting on the left, before storing the output.
LRJWID	R	I	The width in blocks of the desired far-field output.
LRKIN	R	I	The number of blocks to be skipped in the y-direction, starting at the top, before storing the output.
LRKWID	R	I	The height in blocks of the desired far-field output.

3. CALLING SEQUENCE

CALL FFT2D (N2,LRJ,LRK,S,LRSDJ,LRSDK,SL)

Where: N2 = Power of 2 that determines the length of each side of the 2-D transform.

LRJ = The number of logical record blocks in the x-direction of the input file.

LRK = The number of logical record blocks in the y-direction of the input file.

S = A two-dimensional array used to store the blocks of the input file for processing.

LRSDJ = Object-time dimension constant for the x dimension of the array S.

LRSDK = Object-time dimension constant for the y dimension of the array S.

SL = A one-dimensional array equivalenced to S, used for intermediate 2-D processing.

4. RESTRICTIONS, REQUIREMENTS, MISCELLANEOUS DATA

a. Reference:

Hansen, R. C., Microwave Scanning Antennas, Vol. 2, New York: Academic Press, Inc., 1964.

5. SUBPROGRAMS REQUIRED

CZFFT

6. THEORY OF OPERATION

The relationship between a finite linear array of radiators and its corresponding far-field is given by the following equation

$$E(\theta) = \sum_{n=1}^{N} C(n)e^{jnkd_x \sin\theta} \qquad \text{(A3.1)}$$

where $k = 2\pi/\lambda$ and d_x is the interelement spacing. C(n) is the current gain of the n^{th} radiator. This equation assumes isotrophic radiators. Now by letting a new variable p be equal to the following:

$$p = Nkd_x \mathrm{Sin}\theta$$

the expression in Equation A3.1 becomes the following equation

$$\hat{E}(p) = \sum_{n=1}^{N} C(n)e^{j(Pn/N)}$$

which is in the form of the IDFT. This expression can be calculated using standard FFT techniques. The structure of the one dimensional FFT algorithm requires the input data to be in the order that is shown in Figure FFT2D-1. If this reorganization is not implemented the output data will have a 180° phase shift from one point to the next. This problem also arises when the 2D-FFT is performed. A shuffle of blocks of data rather than line segments must be done to prevent the problem from occurring. Figure FFT2D-2 illustrates the organization of the aperture shuffle.

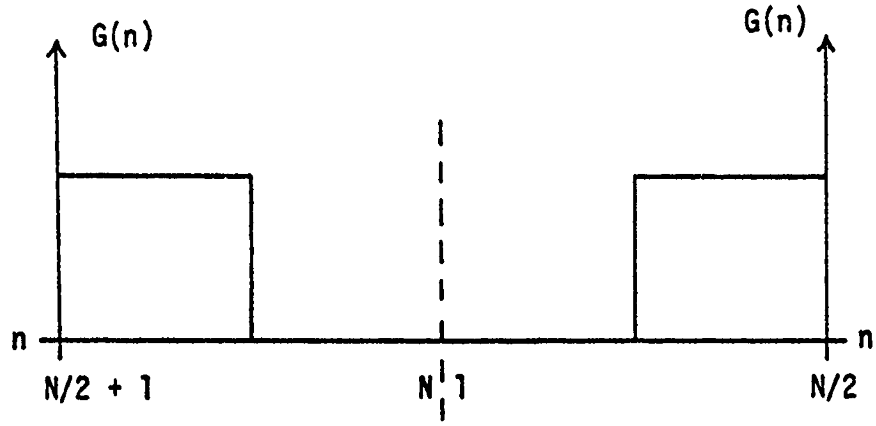
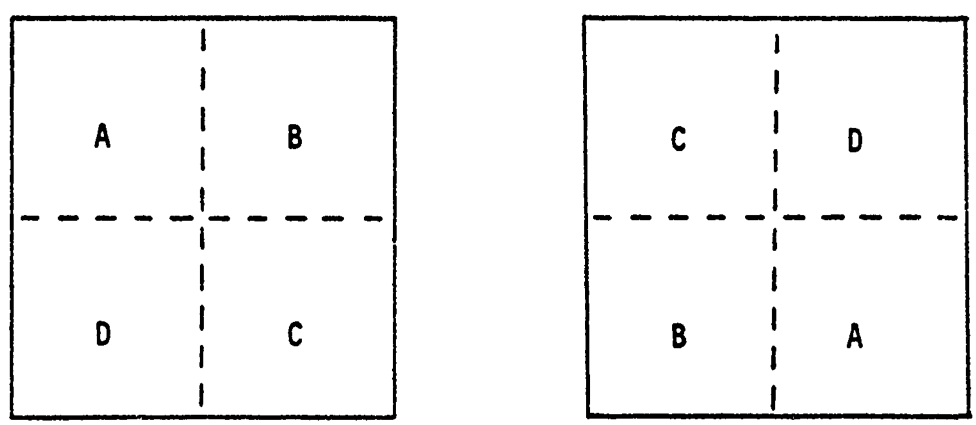


Figure FFT2D-1: Organization of input data required by linear FFT algorithm.



a) Original Antenna Aperture

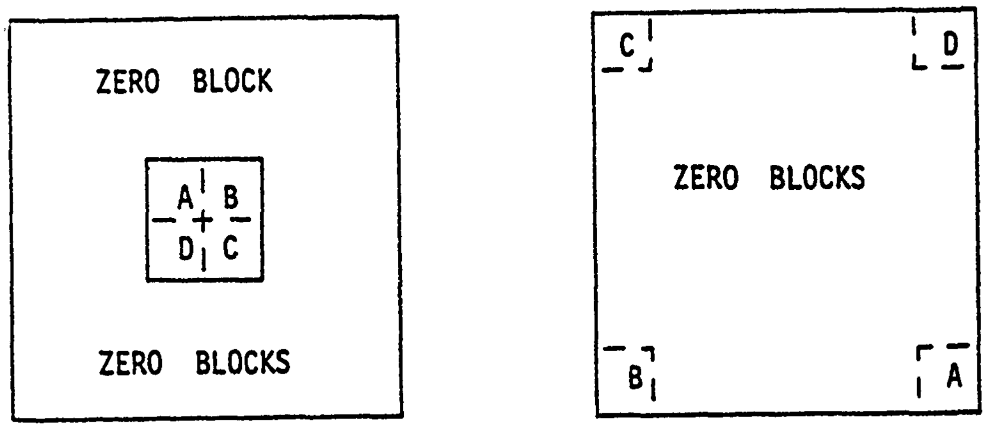
b) Shuffled Antenna Aperture

Figure FFT2D-2: Organization of input data required by 2-D FFT algorithm.

In the case in which the antenna aperture has a smaller number of points than the desired far-field, there must be some 'zero' blocks or blocks loaded with zeros to pad the input aperture into the far-field point configuration. Figure FFT2D-3 illustrates the padding with the zero blocks and the shuffle.

The input aperture field with dimensions LRJ x LRK blocks is split into the four corners of the transform field. The transform field has 2^{N2} points on a side. This is illustrated in Figure FFT2D-4.

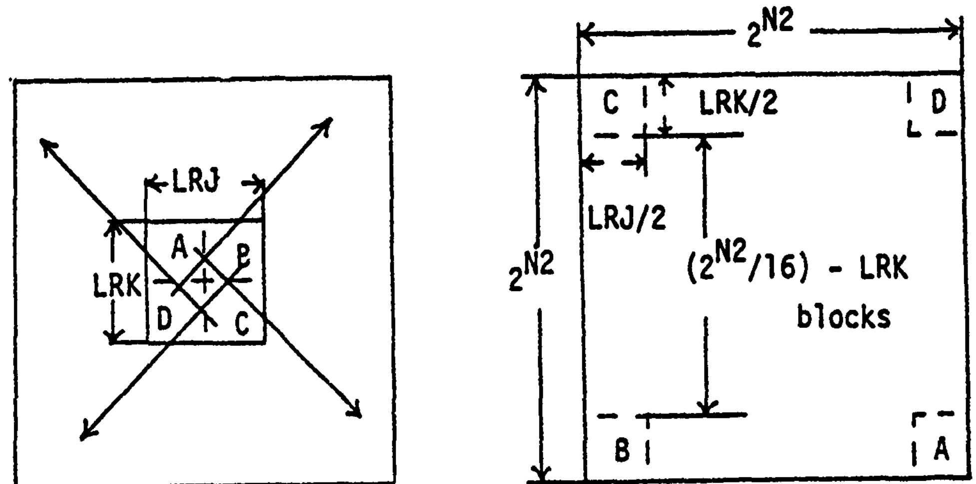
Each row of this matrix is now transformed, one at a time, starting at the top. Since the middle $(2^{N2}/16)$-LRK rows are zero, the transform is equal to zero. Therefore, the program skips these rows and begins at the top of section B. This avoids a waste of computer time. Now that the first transform has been executed, if the whole far-field is required, the second transform, which is the columns of the intermediate result, must execute a complete $2^{N2} \times 2^{N2}$ point transform. In most cases, however, the whole far-field is not required and only a small vertical section needs to be transformed. Figure FFT2D-5 is provided to illustrate this situation. The transform field is $2^{N2}/16$ blocks on a side. If the user only wishes to look at a section of blocks that have dimensions LRJWID x LRKWID, only a vertical stripe LRJWID wide needs to be transformed. Since the transform of the other columns of the matrix have no effect on the transform of the columns in the stripe, it would again be a waste of computer time. Only the double crosshatched area of Figure FFT2D-5 is stored in the output PRMFL. If two complete $2^{N2} \times 2^{N2}$ transforms were executed the total number of complex points processed would be 2^{N2*4}. Using this scheme only (LRK*16)*(LRJWID*16)*2^{N2*2} complex points are processed. For the case of N2 = 8, and LRK = LRJWID = 4, only 6.25% of the total number of complex points are processed.



a) Original Antenna Aperture

b) Shuffled Antenna Aperture

Figure FFT2D-3: Organization of input data with zero blocks required by 2-D FFT algorithm.



a) Orginal Antenna Aperture

b) Shuffled Antenna Aperture

Figure FFT2D-4: Organization of input data showing block measurements for the 2D-FFT algorithm.

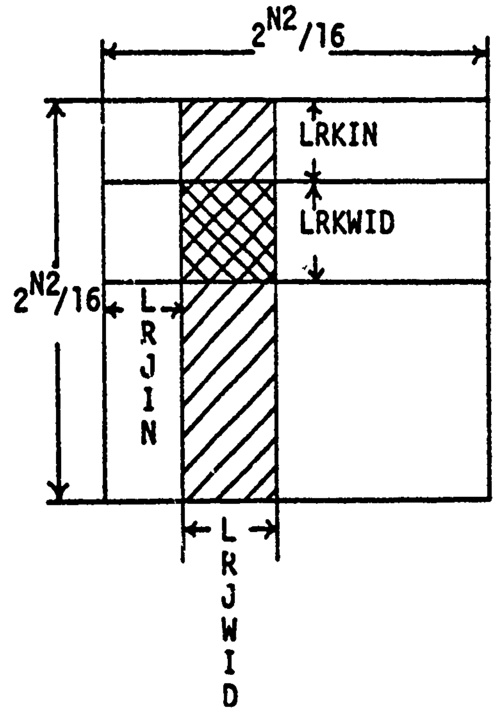


Figure FFT2D-5: Far-field output data from 2-D FFT

7. FORTRAN LISTING

```
C
C    ***************************************
C
      SUBROUTINE FFT2D(N2,LRJ,LRK,S,LRSDJ,LRSDK,S1)
      COMPLEX ALRZ(256),S(LRSDJ,LRSDK),S1(1)
      CALL PTIME(OTIME)
      NAMELIST/FILOUT/LRJIN,LRJWID,LRKIN,LRKWID
      READ(05,FILOUT)
      WRITE(06,FILOUT)
      LRSIDJ=LRSDJ/16
      ICRNR=LRSIDJ*(LRKIN+LRKWID)
      LRSTP=0
      DO 100 I=1,256
  100 ALRZ(I)=(0.0,0.0)
      LRNMR=0
      LR1IN=(1+LRK)*LRJ/2
  220 LRMKR=0
      LR2=LRSTP+1
      LRNMR1=0
      JST=1
      JSTP=16
      GO TO 300
  230 LRST=LRSTP+1
      LRSTP=LRST+(LRSIDJ-LRJ)-1
      DO 240 LR1=LRST,LRSTP
      DO 235 K=1,16
      DO 235 J=JST,JSTP
  235 S(J,K)=(0.0,0.0)
      LRMKR=LRMKR+1
      JST=JSTP+1
  240 JSTP=JST+15
  250 LR1IN=LRR-LRJ/2
  300 LRST=LRSTP+1
      LRSTP=LRST+LRJ/2-1
      LRR=LR1IN
      DO 310 LR1=LRST,LRSTP
      LR1IN=LR1IN+1
```

```
      READ(01'LR1IN)((S(J,K),J=JST,JSTP),K=1,16)
      LRMKR=LRMKR+1
      LRNMR1=LRNMR1+1
      LRNMR=LRNMR+1
      JST=JSTP+1
310   JSTP=JST+15
      IF(LRMKR.EQ.LRSIDJ) GO TO 400
      IF((LRSIDJ-LRJ).NE.0) GO TO 230
      GO TO 250
400   IOFST=0
      IHOP=0
      DO 500 K=1,16
      CALL CZFFT(S,N2,IOFST,IHOP)
500   IOFST=IOFST+LRSDJ
      DO 510 JST=1,LRSDJ,16
      JSTP=JST+15
      WRITE(03'LR2)((S(J,K),J=JST,JSTP),K=1,16)
      LR2=LR2+1
510   CONTINUE
      LRMKR=0
      LR1IN=LR1IN+LRJ
      LRHALF=(LRJ*LRK)/2
      IF(LRNMR1.EQ.LRHALF) GO TO 520
      JST=1
      JSTP=16
      GO TO 300
520   IF(LRNMR.EQ.(LRJ*LRK)) GO TO 800
      IF((LRSIDJ-LRK).EQ.0) GO TO 720
      LRST=LRSTP+1
700   LRSTP=LRST+LRSIDJ-1
      DO 710 LR1=LRST,LRSTP
710   WRITE(03'LR1) ALRZ
      LRZSTP=LRSIDJ*(LRSIDJ-LRK/2)
      IF(LRSTP.GE.LRZSTP) GO TO 720
      LRST=LRSTP+1
      GO TO 700
720   LR1IN=LRJ/2
      GO TO 220
800   LRTTL=0
      LR2RL=0
      LR2=LRJIN
```

```
810  KST=1
     KSTP=256
     LRMKR=0
     LRHOP=0
     LR22=LR2
     LR2=LR2+1
830  READ(03'LR2+LRHOP)(S1(KLOC),KLOC=KST,KSTP)
     LRMKR=LRMKR+1
     LRHOP=LRHOP+LRSIDJ
     KST=KSTP+1
     KSTP=KST+255
     IF(LRMKR.NE.LRSIDJ) GO TO 830
     IOFST=0
     IHOP=4
     DO 840 K=1,16
     CALL CZFFT(S,N2,IOFST,IHOP)
840  IOFST=IOFST+1
     LR2=LR22
     KST=1
     KSTP=256
     LRMKR=0
     LRHOP=0
     LR2=LR2+1
     LRBUMP=0
     LR2RL=LR2RL+1
860  IF(LR2+LRHOP.LT.(LRKIN*LRSIDJ).OR.
    * (LR2+LRHOP).GT.ICRNR) GO TO 865
     WRITE(02'(LR2RL+LRBUMP))(S1(KLOC),
    * KLOC=KST,KSTP)
     LRBUMP=LRBUMP+LRJWID
865  LRTTL=LRTTL+1
     LRMKR=LRMKR+1
     LRHOP=LRHOP+LRSIDJ
     KST=KSTP+1
     KSTP=KST+255
     IF(LRMKR.NE.LRSIDJ) GO TO 860
     IF(LRTTL.NE.(LRSIDJ*LRJWID)) GO TO 810
     CALL PTIME(TIME)
     TIME=(TIME-OTIME)*3600.0
```

```
      WRITE(06,870) TIME
  870 FORMAT('  EXECUTION TIME = ',F10.4)
      RETURN
      END
```

PROGRAM FFT2DX

1. PURPOSE

The program FFT2DX generates a far-field complex voltage pattern from an existing antenna aperture distribution which is stored in a PRMFL. FFT2DX maps the illumination to the far-field using a two-dimensional Fourier transform. The far-field is stored on a PRMFL designated by the user.

2. INPUT PARAMETERS

Name	O/R	T	Description
N2	R	I	The power of 2 which defines the length of each side of the 2-D transform.
LRJ	R	I	The number of logical record blocks in the x-direction of the input file (horizontally).
LRK	R	I	The number of logical record blocks in the y-direction of the input file (vertically).
LRJIN	R	I	The number of blocks to be skipped in the x-direction, starting on the left, before storing the output.
LRJWID	R	I	The width in blocks of the desired far-field output.
LRKIN	R	I	The number of blocks to be skipped in the y-direction, starting at the top, before storing the output.

LRKWID	R	I	The height in blocks of the desired far-field output.

3. RESTRICTIONS, REQUIREMENTS, MISCELLANEOUS DATA

a. N2 must be in the range $4 \leq N2 \leq 10$

b. LRJ,LRK must be even and exactly the same values as were specified in loading the aperture distribution PRMFL.

4. SUBPROGRAMS REQUIRED

FFT2D

5. THEORY OF OPERATION

FFT2DX initializes the array dimensions required for the subroutine FFT2D.

6. FORTRAN LISTING

```
      PARAMETER LENG=8192
      COMMON S(LENG)
      COMPLEX S,S1(1)
      EQUIVALENCE (S1(1),S(1))
      CALL RANSIZ(01,512)
      CALL RANSIZ(02,512)
      CALL RANSIZ(03,512)
      NAMELIST/FFT/N2,LRJ,LRK
      READ(05,FFT)
      WRITE(06,FFT)
      LRSDJ=2**N2
      LRSDK=LENG/LRSDJ
      IF(LRSDK.GT.LRSDJ) LRSDK=LRSDJ
      CALL FFT2D(N2,LRJ,LRK,S,LRSDJ,LRSDK,S1)
      CALL EXIT
      STOP
      END
```

PROGRAM FILMOD

1. PURPOSE

This program modifies existing aperture current distributions that are stored in a PRMFL. The program allows the user to list and/or change individual element values. The program also allows the user to 'punch' holes in the current distribution with specified radius and center. The modified file may be either written over the input PRMFL or may be stored on another user specified PRMFL.

2. INPUT PARAMETERS

Name	O/R	T	Description
LRJ	R	I	Number of logical record blocks in the x-direction (horizontally).
LRK	R	I	Number of logical record blocks in the y-direction (vertically).
IBLK	O	I	A pointer to indicate the logical record block to be modified.
JSTRT	O	I	The horizontal coordinate to begin the element value listing.
KSTRT	O	I	The vertical coordinate to begin the element value listing.
JSTP	O	I	The horizontal coordinate to stop the element value listing.

KSTP	O	I	The vertical coordinate to stop the element value listing.
IELJ	O	I	The horizontal coordinate of the element to be changed.
IELK	O	I	The vertical coordinate of the element to be changed.
VREAL	O	F	The real part of the new element value.
VIMG	O	F	The imaginary part of the new element value.
ICNTJ	O	I	The horizontal coordinate for the center of the hole.
ICNTK	O	I	The vertical coordinate for the center of the hole.
XHOLE	O	F	The radius of the hole to be punched in the aperture illumination.

3. RESTRICTIONS, REQUIREMENTS, MISCELLANEOUS DATA

a. The values of JSTRT,KSTRT,JSTP,KSTP,IELJ,IELK are all assigned with respect to the upper left corner of each block which has the coordinates (1,1).

b. The values of ICNTJ,ICNTK are assigned with respect to the upper left corner of the aperture field having the coordinates (1,1).

4. SUBPROGRAMS REQUIRED

None

5. THEORY OF OPERATION

In the section of the program that lists or changes individual element values, the program reads the specified block into an array A(J,K) that is complex and has dimensions (16,16). The index J increments the fastest. When a list is requested, the values of the elements start with the location (JSTRT,KSTRT), (JSTRT+1,KSTRT),...,(JSTP,KSTRT), (JSTRT,KSTRT+1),..., (JSTP,KSTRT+1), ...,(JSTP,KSTP), i.e., from the upper left hand element down to the lower right hand element. This process is repeated for each block requested.

The element values of each block are changed according to the location and value given by the equation:

A(IELJ,IELK)=CMPLX(VREAL,VIMG)

This is repeated for the total number of element changes requested and for each block requested.

Holes in the aperture illumination are punched with a radius determined by XHOLE. The center of the hole is located at the aperture coordinate (ICNTJ,ICNTK). The values of ICNTJ and ICNTK are assigned with respect to the (1,1) element of logical record block number one having the aperture coordinates (1,1). The upper left element in the aperture field has the coordinates (1,1). The program sequentially steps through the aperture blocks starting with block one and proceeding to block LRTTL (=LRJ*LRK). The distances from the elements in each block to the element located at (ICNTJ,ICNTK) are calculated. A comparison of each distance to the length XHOLE is made. If the distance is less than or equal to XHOLE, the element value is changed to CMPLX(0.0,0.0). Otherwise, the element value is unchanged. In this way holes with radius XHOLE are made in the aperture current distribution.

6. FORTRAN LISTING

```
      COMPLEX A(16,16)
      CHARACTER FILIN*20,FILOUT*20,Y*1,X*1
      DATA IOK/0400000000000/,Y/'Y'/
100   FORMAT(V)
200   WRITE(06,100) 'INPUT FILE NAME'
      READ 100,FILIN
      LUDIN=01
      CALL ATTACH(01,FILIN,3,1,ISTAT, )
      IF(ISTAT.EQ.IOK.OR.ISTAT.EQ.0) GO TO 400
      WRITE(06,300)ISTAT
300   FORMAT('UNSUCCESSFUL ATTACH  ISTAT=',O20)
      GO TO 200
400   CALL RANSIZ(01,512)
      WRITE(06,100) 'OUTPUT FILE NAME'
      READ 100,FILOUT
      LUDOUT=02
      IF(FILIN.NE.FILOUT) GO TO 450
      LUDOUT=01
      GO TO 500
450   CALL ATTACH(02,FILOUT,3,1,ISTAT, )
      IF(ISTAT.EQ.IOK.OR.ISTAT.EQ.0) GO TO 499
      WRITE(06,300) ISTAT
      GO TO 400
499   CALL RANSIZ(02,512)
500   WRITE(06,100) 'LRJ,LRK'
      READ 100,LRJ,LRK
      LRTTL=LRJ*LRK
      LRTTL1=LRTTL+1
      WRITE(06,100) 'MODIFY OR HOLE? (0 OR 1)'
      READ 100,MODFLG
      ITMP=1
      IF(MODFLG.EQ.1) GO TO 1000
600   WRITE(06,100) 'IBLK'
      READ 100,IBLK
      IF(IBLK.EQ.ITMP) GO TO 650
      IF(LUDIN.EQ.LUDOUT) GO TO 650
      DO 620 IBK=ITMP,IBLK-1
      READ(LUDIN'IBK)A
```

```
      WRITE(LUDOUT'IBK)A
620   CONTINUE
650   ITMP=IBLK+1
      READ(LUDIN'IBLK)A
700   WRITE(06,100) 'ANY ELEMENTS LISTED? (Y OR N)'
      READ 100,X
      IF(X.NE.Y) GO TO 750
      WRITE(06,100) 'JSTRT,KSTRT,JSTP,KSTP'
      READ 100,JSTRT,KSTRT,JSTP,KSTP
      WRITE(06,730)((A(J,K),J=JSTRT,JSTP),K=KSTRT,KSTP)
730   FORMAT((2E12.5)/)
      WRITE(06,100) 'ANY ELEMENTS CHANGED? (Y OR N)'
      READ 100,X
      IF(X.NE.Y) GO TO 720
750   WRITE(06,100) 'HOW MANY ELEMENTS CHANGED?'
      READ 100,NELE
      IF(NELE.LE.0) GO TO 770
      IF(NELE.GT.100) GO TO 750
      DO 760 I=1,NELE
      WRITE(06,100) 'IELJ,IELK,VREAL,VIMG'
      READ 100,IJ,IK,VREAL,VIMG
      A(IJ,IK)=CMPLX(VREAL,VIMG)
760   CONTINUE
720   WRITE(06,100) 'ANY MORE MODS OR LIST? (Y OR N)'
      READ 100,X
      IF(X.EQ.Y) GO TO 700
770   WRITE(LUDOUT'IBLK)A
      WRITE(06,100) 'ANOTHER BLOCK? (Y OR N)'
      READ 100,X
      IF(X.EQ.Y) GO TO 600
      IF(ITMP.EQ.LRTTL1) GO TO 900
      IF(LUDIN.EQ.LUDOUT) GO TO 900
      DO 800 IBK=ITMP,LRTTL
      READ(LUDIN'IBK)A
      WRITE(LUDOUT'IBK)A
800   CONTINUE
900   WRITE(06,100) 'ANY HOLES? (Y OR N)'
      READ 100,X
      IF(X.NE.Y) GO TO 1700
      LUDIN=02
      IF(FILIN.EQ.FILOUT) LUDIN=01
```

```
1000 WRITE(06,100) 'ICNTJ,ICNTK'
     READ 100,ICNTJ,ICNTK
     WRITE(06,100) 'XHOLE'
     READ 100,XHOLE
     K1=0
     J1=0
     DO 1500 IBLK=1,LRTTL
     READ(LUDIN'IBLK)A
     DO 1400 KK=1,16
     DO 1300 JJ=1,16
     K=K1+KK
     J=J1+JJ
     XJD=ABS(ICNTJ-J)*ABS(ICNTJ-J)
     XKD=ABS(ICNTK-K)*ABS(ICNTK-K)
     DST=SQRT(XJD+XKD)
     IF(DST.LE.XHOLE) A(JJ,KK)=(0.0,0.0)
1300 CONTINUE
1400 CONTINUE
     WRITE(LUDOUT'IBLK)A
     IF(MOD(IBLK,LRJ).EQ.0) GO TO 1450
     J1=J1+16
     GO TO 1500
1450 K1=(IBLK/LRJ)*16
     J1=0
1500 CONTINUE
     WRITE(06,100) 'ANOTHER HOLE? (Y OR N)'
     READ 100,X
     IF(X.NE.Y) GO TO 1600
     LUDIN=02
     IF(FILIN.EQ.FILOUT) LUDIN=01
     GO TO 1000
1600 WRITE(06,100) 'ANY ELEMENT CHANGES? (Y OR N)'
     READ 100,X
     IF(X.NE.Y) GO TO 1700
     LUDIN=02
     IF(FILIN.EQ.FILOUT) LUDIN=01
     GO TO 600
1700 WRITE(06,100) 'ANOTHER OUTPUT GENERATED? (Y OR N)'
     READ 100,X
     IF(X.NE.Y) GO TO 1800
     WRITE(06,1750) ISTAT
```

```
1750 FORMAT('DETACH OUTPUT FILE  ISTAT=',O20)
     LUDIN=01
     GO TO 400
1800 WRITE(06,100) 'ANOTHER FILE MODIFIED? (Y OR N)'
     READ 100,X
     READ(LUDIN'LRTTL)A
     READ(LUDOUT'LRTTL)A
     WRITE(06,1850) ISTAT
     IF(LUDOUT.EQ.1) GO TO 1899
     WRITE(06,1750) ISTAT
1850 FORMAT('DETACH INPUT FILE  ISTAT=',O20)
1899 IF(X.EQ.Y) GO TO 200
1900 CONTINUE
     STOP
     END
```

DOUBLE PRECISION FUNCTION GAM

1. MODULE IDENTIFICATION

Name	Classification Code	Reference Number
GAM	Subordinate	Not User Referenced

2. PURPOSE

This function is used to compute the value of the Gamma function.

3. INPUT PARAMETERS

Name	O/R	T	Description
X	R	F	Argument of the Gamma function.

4. CALLING SEQUENCE

G = GAM (X)

Where: X is the input argument
G contains the computed value of the Gamma function.

5. RESTRICTIONS, REQUIREMENTS, MISCELLANEOUS DATA

a. The argument, X, must be within the following range: $0 \leq X \leq 1.0$

b. The maximum error in computing the Gamma function is $\pm 3.0 \times 10^{-7}$.

c. Reference: Handbook of Mathematical Functions by M. Abramowitz and I. A. Slegun, Dover, Inc., p. 257.

d. External References:

None

e. Referenced labeled common areas:

None

6. THEORY OF OPERATION

The polynomial expansion for the Gamma function of x+1 is given by the following expression:

$$(x+1) = \sum_{m=0}^{8} b_m x^m + \varepsilon(x)$$

Where: $b_o = 1.0$

$b_1 = 0.577191652$

$b_2 = 0.988205891$

$b_3 = -0.897056937$

$b_4 = 0.918206857$

$b_5 = -0.756704078$

$b_6 = 0.482199394$

$b_7 = -0.193527818$

$b_8 = 0.035868343$

$|\varepsilon(x)| \leq 3.0 \times 10^{-7}$

7. FORTRAN LISTING

```
C
       DOUBLE PRECISION FUNCTION GAM(X)
      DOUBLE PRECISION S,GAM
      S=+0.35868343E-1
      S=S*X-0.193527818
      S=S*X+0.482199394
      S=S*X-0.756704078
      S=S*X+0.918206857
      S=S*X-0.897056937
      S=S*X+0.988205891
      S=S*X-0.577191652
      GAM=S*X+1.0
      RETURN
      END
```

PROGRAM PDFESTR

1. PURPOSE

This program generates a histogram of the radiating elements in a statistically loaded aperture. The width of each radius cell may be varied and the origin of the radius is user specified.

2. INPUT PARAMETERS

Name	O/R	T	Description
LRJ	R	I	Number of logical record blocks in the x-direction (horizontally).
LRK	R	I	Number of logical record blocks in the y-direction (vertically).
OFSTJ	R	F	Value added to the calculated middle of the horizontal aperture field length to give the offset origin.
OFSTK	R	F	Value added to the calculated middle of the vertical aperture field length to give the offset origin.
RINC	R	F	Incremental radius or radius cell width used to accumulate the histogram values.
RLIM	R	F	The maximum radius value of interest.

ICON	R	I	Mode flag = 1 Program halts = 0 Histogram is normalized to a unit cumulative distribution. = 1 Histogram data is converted to probability density estimate data. The raw data is divided by the product of the cell width and the total number of elements. = 2 Raw histogram data
NDPACK	R	I	The number of incremental radius histogram cells, RINC, combined to make each output histogram cell.

3. RESTRICTIONS, REQUIREMENTS, MISCELLANEOUS DATA

None

4. SUBPROGRAMS REQUIRED

None

5. THEORY OF OPERATION

The program sequentially reads the logical record blocks into the array A(J,K), which has dimensions (16,16) and is complex. The program starts with block number one and continues through block number LRTTL (=LRJ*LRK). The elements of each block are then processed. If the element value is CMPLX(0.0,0.0) then the program proceeds to the next element. For non-zero element values the radius from the element to the origin is calculated. A counter in the appropriate radius cell of the array XF(IADD) is incremented and the program proceeds to process the next element. After all of the elements in the aperture field have been processed, the raw histogram is contained in the array XF(1) with each cell corresponding to an annulus with radius R = I*RINC and width RINC. The program then combines the radius cells in groups and stores them in the array DATOT(J). NDPACK consecutive radius cells are put in each group, thus reducing the number of histogram cells by a factor of 1/NDPACK. The histogram data is then modified and dumped according to the value of ICON.

For ICON=0

DATOT(J)=DATOT(J)/CUM

where CUM is the total number of elements. This data is normalized to a unit cumulative distribution.

For ICON=1

DATOT(J)=DATOT(J)/(CUM*RINC*NDPACK)

This data is converted to probability density estimate data.

For ICON=2

DATOT(J)=DATOT(J)

The raw histogram data is outputted.

6. FORTRAN LISTING

```
      COMPLEX A(16,16)
      CHARACTER FILIN*20
      DIMENSION XF(1000),DATOT(300)
      DATA IOK/0400000000000/
      WRITE(06,50) 'LRJ,LRK'
 50   FORMAT(V)
      READ 50,LRJ,LRK
      WRITE(06,50) 'OFSTJ,OFSTK'
      READ 50,OFSTJ,OFSTK
 400  WRITE(06,50) 'INPUT FILE'
      READ 50,FILIN
      CALL ATTACH(01,FILIN,1,1,ISTAT, )
      CALL RANSIZ(01,512)
      IF(ISTAT.EQ.IOK.OR.ISTAT.EQ.0) GO TO 300
      WRITE(06,350) ISTAT
 350  FORMAT(' UNSUCCESSFUL ATTACH    ISTAT=',O20)
      GO TO 400
 300  XMIDJ=LRJ*16/2.0+0.5+OFSTJ
      XMIDK=LRK*16/2.0+0.5+OFSTK
      WRITE(06,50) 'RINC,RLIM'
      READ 50,RINC,RLIM
      NIXF=RLIM/RINC
      DO 100 I=1,NIXF
      XF(I)=0.0
 100  CONTINUE
      LRTTL=LRJ*LRK
      J1=0
      K1=0
      DO 200 IBLK=1,LRTTL
      READ(01'IBLK)A
      DO 500 KK=1,16
      K=KK+K1
      DO 600 JJ=1,16
      J=J1+JJ
      IF(A(JJ,KK).EQ.(0.0,0.0)) GO TO 600
      XJD=(XMIDJ-J)*(XMIDJ-J)
      XKD=(XMIDK-K)*(XMIDK-K)
      DV=SQRT(XJD+XKD)
```

```
      IADD=IFIX(DV/RINC)+1
      IF(IADD.GT.NIXF) IADD=NIXF
      XF(IADD)=XF(IADD)+1.0
600   CONTINUE
500   CONTINUE
      IF(MOD(IBLK,LRJ).EQ.0) GO TO 550
      J1=J1+16
      GO TO 200
550   K1=(IBLK/LRJ)*16
      J1=0
200   CONTINUE
700   WRITE(06,50) 'ICON,NDPACK'
      READ 50 ICON,NDPACK
      IF(ICON.EQ.-1) GO TO 1300
      KEND=NIXF/NDPACK
      JJ=-NDPACK
      CUM=0.0
      DO 1000 J=1,KEND
      DEN=0.0
      JJ=JJ+NDPACK
      DO 900 K=1,NDPACK
      DEN=DEN+XF(K+JJ)
900   CONTINUE
      CUM=CUM+DEN
      DATOT(J)=DEN
1000  CONTINUE
      IF(ICON.EQ.2) GO TO 1200
      CUM=1.0/CUM
      IF(ICON.EQ.1) CUM=CUM/(RINC*NDPACK)
      DO 1100 K=1,KEND
      DATOT(K)=DATOT(K)*CUM
1100  CONTINUE
1200  WRITE(06,1210)(DATOT(J),J=1,KEND)
1210  FORMAT(F12.5)
      GO TO 700
1300  CALL DETACH(01,ISTAT, )
      CALL EXIT
      STOP
      END
```

PROGRAM PLARY

1. PURPOSE

This program loads a PRMFL with a user specified antenna aperture current distribution. The aperture parameters include size, shape, weighting, and several deterministic phase options, including beam steering. Thinned or statistically loaded apertures may also be generated.

2. INPUT PARAMETERS

Name	O/R	T	Description
IAPTFL	R	I	Determine the shape of the aperture to be loaded. = 1 Circular = 2 Elliptical = 3 Rectangular
XEDGE	O	F	Radius of the circular aperture.
XHOLE	O	F	Radius of the hole in a circular or elliptical aperture.
NMAJOR	O	I	Length of semi-major elliptical axis.
NMINOR	O	I	Length of semi-minor elliptical axis.
NWIDTH	O	I	Width of rectangular aperture.
NHIGH	O	I	Height of rectangular aperture.

IWTFLG	R	I	Determine the weighting function used in loading the aperture. = 0 Rectangular weighting = 1 Cosine on a pedestal to a power = 2 Blackman = 3 Kaiser = 4 Bartlett or triangular = 5 Taylor = 6 Bessel = 7 Cubic = 8 Bayliss
WTRAD	O	F	Radius of the specified weighting function for circular apertures.
ZJRAD	O	F	Half the span of the weighting function in the x-direction for elliptical and rectangular apertures.
ZKRAD	O	F	Half the span of the weighting function in the y-direction for elliptical and rectangular apertures.
WTPED	O	F	The height of the pedestal for cosine on a pedestal to a power weighting.
NWTPOW	O	I	The power of the cosine function for a cosine on a pedestal to a power weighting.
WKASIR	O	F	The Kaiser variable for the trade-off between main lobe width and side lobe amplitude.

BESCAL	O	F	The maximum weighting amplitude at the center of the aperture for the Bessel weighting.
CUBK	O	F	The weighting amplitude scaling constant for the cubic weight.
BESEDG	O	F	The radius scaling constant for the Bessel weighting.
DB	O	F	The design side lobe amplitude in dB for the Taylor or Bayliss weighting.
NBAR	O	I	The number of zeros used to approximate the Dolph-Chebyschev weighting distribution in the Taylor or Bayliss weighting.
DELPHJ	O	F	Beam steering in degrees in the x-direction.
DELPHK	O	F	Beam steering in degrees in the y-direction.
PHERX	O	F	Maximum quadratic phase error in degrees at the edge of the aperture in the x-direction.
PHERY	O	F	Maximum quadratic phase error in degrees at the edge of the aperture in the y-direction.
NBITS	R	I	Number of bits used to control the digital phase shifters.
BESERR	O	F	The maximum Bessel phase error in degrees at the center of the aperture.

BSCAL	O	F	The radius scaling constant for the Bessel phase error.
LRJ	R	I	The number of logical record blocks in the x-direction.
LRK	R	I	The number of logical record blocks in the y-direction.
XKK	O	F	The probability that an element is located at the peak.
MAD1	O	I	The starting address for selecting random numbers from the random number array. ($1 \leq MAD1 \leq 128$)
JRND	O	I	The random number generator initialization constant. ($0 < JRND < 2^{36} - 1$)

3. RESTRICTIONS, REQUIREMENTS, MISCELLANEOUS DATA

a. The values for LRJ,LRK must be even.

b. Only circular arrays can be statistically loaded.

4. SUBPROGRAMS REQUIRED

EXPND
BESS
GAM
RRAND
WEIGHT
FXOPT

5. THEORY OF OPERATION

The program PLARY loads the generated antenna aperture into a PRMFL. The aperture is divided into 'blocks' which are 16 elements on a side or a total of 256 elements per block. These blocks define the size and dimension of the logical records used to write the aperture into the PRMFL. Thus, the number of blocks needed to load the aperture also corresponds to the total number of logical records required. The aperture field or the total number of elements available is defined by an even number of blocks arranged in columns (LRJ columns) and an even number of blocks in each column (LRK rows). Thus the total number of logical record blocks is LRJ*LRK.

The weights at the edge of a circular aperture are assigned values in a special way to help smooth out the granularity caused by approximating a circular aperture with a grid of rectangularly spaced elements. A square with sides d/2, where d is the interelement spacing, is constructed centered at each element of the array. The distances from the corners of the square to the center of the aperture are calculated for each element. The number of corners contained by the specified aperture determines the weight of the element. Each corner counts a weight of 0.25. Therefore, if all four corners are located within the aperture distribution, then a value of 1.0 is assigned. Similarly, if only three of the corners are within the aperture distribution the value is 0.75, etc. This also holds for the element values around the hole of the aperture distribution. If all four corners are contained by the aperture hole, then the element value is 0.0. The number of corners contained by a radiating part of the aperture determines the value of each element. This technique produces a smoother circular image

The method used to load the elliptical or the rectangular aperture is not quite so sophisticated. The rectangular aperture, since it is being loaded in a rectangular grid, is simply loaded based on whether the element is contained or is outside of the radiating portion of the aperture distribution. Those elements contained inside the specified limits are assigned a value of 1.0. All others are given a value of 0.0. The elliptical aperture is loaded with the same technique as the rectangular aperture. The value of each element is determined by the location of the element with respect to the elliptical edge. If the element is inside the ellipse, the value is 1.0, otherwise it is 0.0. This technique produces a fairly granular edge. However, if the ellipse is large this effect is minimized.

The degree of phase accuracy in beam steering is determined by the number of controlling bits, NBITS. The value of the least significant bit of the beam steering phase shifter, XLSB, can be calculated as follows:

$$XLSB = 360.0/2^{NBITS} \text{ degrees}$$

The beam steering phase shift, PH, for element K,J is computed from the orthogonal steering angles, DELPHJ,DELPHK, as follows

```
PH1 = DELPHK*(K-XMIDK) + DELPHJ*(J-XMIDJ)

PH = FLOAT(IFIX(AMOD(PH1,360.0)/XLSB))*XLSB*DTR
```

Where: K,J = The element location in the y and x coordinates respectively

XMIDK = The middle of the aperture in the y-span

XMIDJ = The middle of the aperture in the x-span

DTR = Degrees to radius conversion constant.

Given the maximum quadratic phase error at the x and y edge of the aperture, PHERX,PHERY, the element to element quadratic phase error is calculated using

```
PHERR = (YMUK*(K-XMIDK)**2+XMUJ*(J-XMIDJ)**2)*DTR
```

Where, for a circle

```
YMUK = PHERY/(XEDGE)**2

XMUJ = PHERX/(XEDGE)**2
```

and the other parameters have the same meaning as above. For the ellipse, the values for XMUJ and YMUK are

```
YMUK = PHERY/(NMINOR)**2

XMUJ = PHERX/(NMAJOR)**2
```

For the rectangular aperture the values are

```
YMUK = PHERY/(NHIGH/2)**2

XMUJ = PHERX/(NWIDTH/2)**2
```

The Bessel phase error is determined by first calculating the radius to each element, then scaling the radius by the constant BSCAL. The scaled radius XRAD, is then used as the argument for evaluation of the Bessel function.

```
PHBSER = BESERR*BESS(0.0,XRAD)*DTR
```

Where: BESS(0.0,XRAD) = The Bessel function of the first kind and order zero, evaluated at XRAD

BESERR = A magnitude scaling factor, determines the value of maximum error at the center of the aperture

DTR = Degrees to radians conversion constant.

The total deterministic phase error at each element is the sum of the three independent phase contributions.

PHTTL = PH + PHERR + PHBSER

Where: PHTTL = The total phase error

PH = Beam steering and quantization phase error

PHERR = Quadratic phase error

PHBSER = Bessel phase error.

The value assigned to the element in the array A(J,K) is

A(J,K) = A(J,K)*CMPLX(COS(PHTTL),SIN(PHTTL))

7. FORTRAN LISTING

```
C
C      28 APR 78
C      1030
C
       COMPLEX A(16,16)
       COMMON IWTFLG,WTPED,NWTPOW,WKASIR,F(20),B(20),ANG,
&              NBAR,BESCAL,CUBK1,PII2,BESS1,IAZ,XKK,WMAX,
&              BESEDG
       COMMON/BLKRND/MAD1,JRND,XMEAN,SIG2SQ,UL,UEXT
       DIMENSION U(20),Z(20),BZERO1(20)
       DATA U/1.2196699,2.2331306,3.2383154,4.2410628,
&      5.2427643,6.2439216,7.2447598,8.2453948,
&      9.2458927,10.2462933,11.246624,12.246900,
&      13.247131,14.247334,15.247508,1..247663,
&      17.247796,18.247920,19.248027,20.248125/
       DATA BZERO1/0.586067,1.6970509,2.7171939,3.728137,
&      4.7312271,5.7345205,6.7368281,7.7385356,
&      8.7398505,9.7408945,10.7417435,11.7424475,
&      12.7430408,13.7435477,14.7439856,15.7443679,
&      16.7447044,17.745003,18.7452697,19.7455093/
       DATA IOK/0400000000000/
       CHARACTER  OUTFIL*20,X*1,Y*1,N*1
       DATA Y,N/'Y','N'/
       CALL FXOPT(68,1,1,0)
       PI=3.1415926
       PII2=2.0/(PI*PI)
       BESS1=1.0/BESS(0.0,0.0)
  590  WRITE(06,140) 'OUTPUT FILE NAME'
       READ 140,OUTFIL
       CALL ATTACH(01,OUTFIL,3,1,ISTAT, )
       IF(ISTAT.EQ.IOK.OR.ISTAT.EQ.0) GO TO 141
       WRITE(06,145) ISTAT
  145  FORMAT('ATTACH FAILED    ISTAT = ',O20)
       GO TO 590
  141  CALL RANSIZ(01,512)
       NAMELIST/APETUR/IAPTFL,XEDGE,XHOLE,NMAJOR,
&      NMINOR,NWIDTH,NHIGH
       WRITE(06,140) 'STATISTICAL TAPER?'
```

```
      READ 140,X
      IAZ=1
      IF(X.EQ.N) GO TO 600
      IAZ=0
      WRITE(06,140) 'XKK,KAD1,JRND'
      READ 140,XKK,MAD1,JRND
 600  WRITE(06,140) 'IAPTFL'
      READ 140,IAPTFL
      GO TO(610,620,630),IAPTFL
 610  WRITE(06,140) 'XEDGE,XHOLE'
      READ 140,XEDGE,XHOLE
      GO TO 640
 620  WRITE(06,140) 'NMAJOR,NMINOR,XHOLE'
      READ 140,NMAJOR,NMINOR,XHOLE
      GO TO 640
 630  WRITE(06,140) 'NWIDTH,NHIGH'
      READ 140,NWIDTH,NHIGH
 640  CONTINUE
      GO TO 670
 660  WRITE(06,140) 'INVALID IWTFLG'
      NAMELIST/WAIT/IWTFLG,WTRAD,ZJRAD,ZKRAD,WTPED,
     &                   NWTPOW,WKASIR,BESCAL,CUBK,BESEDG
 670   WRITE(06,140) 'IWTFLG'
       READ 140,IWTFLG
      IF(IWTFLG.EQ.8.AND.IAPTFL.NE.1) GO TO 660
      IF(IWTFLG.EQ.7.AND.IAPTFL.EQ.1) GO TO 660
       IF(IWTFLG.EQ.0) GO TO 700
      GO TO (810,830,830),IAPTFL
 810  WRITE(06,140) 'WTRAD'
      READ 140,WTRAD
      GO TO 800
 830  WRITE(06,140) 'ZJRAD,ZKRAD'
      READ 140,ZJRAD,ZKRAD
      WTRAD=AMAX1(ZJRAD,ZKRAD)
 800   GO TO (710,700,730,700,720,740,750,720),IWTFLG
  710  WRITE(06,140) 'WTPED,NWTPOW'
       READ 140,WTPED,NWTPOW
       GO TO 700
 720  WRITE(06,140) 'DB,NBAR'
      READ 140,DB,NBAR
      NAMELIST/TAYL/DB,NBAR,SIG
```

```
      RAT=10.0**(DB/20.0)
      AA=ALOG(RAT+SQRT(RAT*RAT-1))/PI
      AASQ=AA*AA
      IF(IWTFLG.EQ.8) GO TO 760
      SIG=U(NBAR)/SQRT(AASQ+(NBAR-0.5)**2)
      SIGSQ=SIG*SIG
      DO 252 I=1,NBAR-1
      FNUM=1.0
      FDNM=1.0
      T=U(I)*U(I)
      XII=-0.5
      DO 254 II=1,NBAR-1
      XII=XII+1.0
      FNUM=FNUM*(1.0-T/(SIGSQ*(AASQ+(XII*XII))))
      IF(II.EQ.I) GO TO 254
      FDNM=FDNM*(1.0-T/(U(II)*U(II)))
  254 CONTINUE
      ARG=PI*U(I)
      F(I)=-BESS(0.0,ARG)*FNUM/FDNM
      F(I)=F(I)/(BESS(0.0,ARG)**2)
  252 CONTINUE
      GO TO 700
  730  WRITE(06,140) 'WKASIR'
       READ 140,WKASIR
      GO TO 700
  740 WRITE(06,140) 'BESCAL,BESEDG'
      READ 140,BESCAL,BESEDG
      GO TO 700
  750 WRITE(06,140) 'CUBK'
      READ 140,CUBK
      XX=SQRT(WTRAD*WTRAD/3.0)
      CUBK1=CUBK/ABS(XX*(XX-WTRAD)*(XX+WTRAD))
      GO TO 700
  760 Z(1)=0.9858302+0.0333885*DB+0.00014064*DB*DB
     &        -0.0000019*DB*DB*DB+0.00000001*DB*DB*DB*DB
      Z(1)=Z(1)*Z(1)
      Z(2)=2.00337487+0.01141548*DB+0.0004159*DB*DB
     &        -0.00000373*DB*DB*DB+0.00000001*DB*DB*DB*DB
      Z(2)=Z(2)*Z(2)
      Z(3)=3.00636321+0.00683394*DB+0.00029281*DB*DB
     &        -0.00000161*DB*DB*DB
```

```
      Z(3)=Z(3)*Z(3)
      Z(4)=4.00518423+0.00501795*DB+0.00021735*DB*DB
     &     -0.00000088*DB*DB*DB
      Z(4)=Z(4)*Z(4)
      IF(NBAR.LE.4) GO TO 762
      DO 761 I=5,NBAR
      Z(I)=AASQ+I*I
 761  CONTINUE
 762  SIG=BZERO1(NBAR+1)/SQRT(Z(NBAR))
      SIGSQ=SIG*SIG
      DO 765 I=1,NBAR
      FNUM=1.0
      T=BZERO1(I)*BZERO1(I)
      FDNM=1.0-T/(BZERO1(1)*BZERO1(1))
      IF(I.EQ.1) FDNM=1.0
      DO 770 II=1,NBAR-1
      FNUM=FNUM*(1.0-T/(SIGSQ*Z(II)))
      IF(I-1.EQ.II) GO TO 770
      FDNM=FDNM*(1.0-T/(BZERO1(II+1)*BZERO1(II+1)))
 770  CONTINUE
      B(I)=(2.0*T/BESS(1.0,BZERO1(I)*PI))*FNUM/FDNM
 765  CONTINUE
      P0=0.4797212+0.01453692*(DB)-0.00018739*(DB*DB)
     &   +0.00000218*(DB*DB*DB)-0.00000001*(DB*DB*DB*DB)
      P0=P0*SIG
      P0SQ=P0*P0
      PIP0SQ=P0SQ*PI*PI
      FNUM=1.0
      FDNM=1.0-P0SQ/(BZERO1(1)*BZERO1(1))
      DO 772 I=1,NBAR-1
      FNUM=FNUM*(1.0-P0SQ/(SIGSQ*Z(I)))
      FDNM=FDNM*(1.0-P0SQ/(BZERO1(I+1)*BZERO1(I+1)))
 772  CONTINUE
      C=PIP0SQ-1.0
      C=C*BESS(1.0,P0*PI)*FNUM/FDNM
      C=1.0/C
      DO 773 I=1,NBAR
      B(I)=-B(I)*C
 773  CONTINUE
 700  WRITE(06,140) 'NBITS'
      NAMELIST/PHASE/DELPHJ,DELPHK,PHERX,PHERY,
```

```
     &                     NBITS,BESERR,BSCAL
           READ 140,NBITS
           WRITE(06,140) 'ANY BEAM STEERING?'
           READ 140,X
           IF(X.EQ.N) GO TO 900
           WRITE(06,140) 'DELPHJ,DELPHK'
           READ 140,DELPHJ,DELPHK
       900 WRITE(06,140) 'QUADRATIC ERROR?'
           READ 140,X
           IF(X.EQ.N) GO TO 910
           WRITE(06,140) 'PHERX,PHERY'
           READ 140,PHERX,PHERY
       910 WRITE(06,140) 'BESSEL ERROR?'
           READ 140,X
           IF(X.EQ.N) GO TO 134
           WRITE(06,140) 'BESERR,BSCAL'
           READ 140,BESERR,BSCAL
       134 WRITE(06,135)
       135 FORMAT('LRJ,LRK')
           NAMELIST/BLOCK/LRJ,LRK
           READ 140,LRJ,LRK
       140 FORMAT(V)
           IF(MOD(LRJ,2).NE.0.OR.MOD(LRK,2).NE.0) GO TO 134
           WRITE(6,APETUR)
           WRITE(6,WAIT)
           WRITE(6,TAYL)
           WRITE(6,PHASE)
           WRITE(6,BLOCK)
           XEDGE2=XEDGE*XEDGE
           XHOLE2=XHOLE*XHOLE
           NSIDEJ=LRJ*16
           NSIDEK=LRK*16
            NCENTJ=NSIDEJ/2+1
            NCENTK=NSIDEK/2+1
            NCNT1J=NSIDEJ/2
            NCNT1K=NSIDEK/2
           XMIDK=NSIDEK/2+0.5
           XMIDJ=NSIDEJ/2+0.5
           LRTOTL=LRJ*LRK
           DTR=0.017453
           XLSB=360.0/2.0**NBITS
```

```
      DXLSB=1.0/XLSB
      XLSB=XLSB*DTR
      GO TO (10,20,30),IAPTFL
10    YMUK=PHERY/XEDGE2
      XMUJ=PHERX/XEDGE2
      GO TO 40
20    YMUK=PHERY/(FLOAT(NMINOR))**2
      XMUJ=PHERX/(FLOAT(NMAJOR))**2
      GO TO 40
30    YMUK=PHERY/(FLOAT(NHIGH)/2.0)**2
      XMUJ=PHERX/(FLOAT(NWIDTH)/2.0)**2
40    CONTINUE
      K1=0
      J1=0
      IF(IAZ.EQ.1) GO TO 680
      WMAX=0.0
      IAZ=1
      DO 650 IRAD=1,IFIX(WTRAD+1)
      RAD=FLOAT(IRAD-1)
      CALL WEIGHT(RAD,WTRAD,WFUNC)
      WMAX=AMAX1(WMAX,WFUNC)
650   CONTINUE
      IAZ=0
680   CONTINUE
      NAMELIST/STAT/XKK,WMAX
      WRITE(06,STAT)
      DO 510 LR1=1,LRTOTL
      DO 50 KK=1,16
      K=K1+KK
      DO 50 JJ=1,16
      J=J1+JJ
      PH=DELPHK*(K-XMIDK)+DELPHJ*(J-XMIDJ)
      PH=FLOAT(IFIX(AMOD(PH,360.0)*DXLSB))*XLSB
      PHERR=(YMUK*(K-XMIDK)**2+XMUJ*(J-XMIDJ)**2)*DTR
      XJ=J-XMIDJ
      XK=K-XMIDK
      XRAD=SQRT(XJ*XJ+XK*XK)*BSCAL
      PHBSER=BESERR*BESS(0.0,XRAD)*DTR
      PH=PH+PHERR+PHBSER
      A(JJ,KK)=CMPLX(COS(PH),SIN(PH))
50    CONTINUE
```

```
      GO TO (100,300,400),IAPTFL
100   DO 210 KK=1,16
      K=K1+KK
      XKSQ=(K-XMIDK)**2
      DO 200 JJ=1,16
      J=J1+JJ
      ANG=ATAN2((K-XMIDK),(J-XMIDJ))
      XLSQ=XKSQ+(J-XMIDJ)**2
      DIST=SQRT(XLSQ)
      IF(IAZ.EQ.0) GO TO 205
      CNR1=(J-NCENTJ)**2+(K-NCENTK)**2
      CNR2=(J-NCNT1J)**2+(K-NCENTK)**2
      CNR3=(J-NCENTJ)**2+(K-NCNT1K)**2
      CNR4=(J-NCNT1J)**2+(K-NCNT1K)**2
      ICNFL=0
      IF(CNR1.GT.XEDGE2.OR.CNR1.LT.XHOLE2) ICNFL=1
      IF(CNR2.GT.XEDGE2.OR.CNR2.LT.XHOLE2) ICNFL=ICNFL+1
      IF(CNR3.GT.XEDGE2.OR.CNR3.LT.XHOLE2) ICNFL=ICNFL+1
      IF(CNR4.GT.XEDGE2.OR.CNR4.LT.XHOLE2) ICNFL=ICNFL+1
      IF(ICNFL.EQ.0) GO TO 205
      GO TO(201,202,203,204),ICNFL
201   A(JJ,KK)=A(JJ,KK)*0.75
      GO TO 205
202   A(JJ,KK)=A(JJ,KK)*0.5
      GO TO 205
203   A(JJ,KK)=A(JJ,KK)*0.25
      GO TO 205
204   A(JJ,KK)=(0.0,0.0)
205   IF(IWTFLG.EQ.0) GO TO 200
      CALL WEIGHT(DIST,WTRAD,WFUNC)
      IF(IWTFLG.NE.8) GO TO 211
      A(JJ,KK)=A(JJ,KK)*CMPLX(0.0,WFUNC)
      GO TO 200
211   A(JJ,KK)=A(JJ,KK)*WFUNC
200   CONTINUE
210   CONTINUE
      GO TO 500
300   XMAJOR=NMAJOR*NMAJOR
      XMINOR=NMINOR*NMINOR
      DO 310 KK=1,16
      K=K1+KK
```

```
      YKSQ=(K-XMIDK)**2
      DO 310 JJ=1,16
      J=J1+JJ
      XKSQ=(J-XMIDJ)**2
      XLSQ=XKSQ+YKSQ
      ELPSQ=YKSQ/XMINOR+XKSQ/XMAJOR
      IF(ELPSQ.GT.1.0.OR.XLSQ.LT.XHOLE2)
     &   A(JJ,KK)=(0.0,0.0)
  310 CONTINUE
      IF(IWTFLG.EQ.0) GO TO 500
      SGN=1.0
      DO 320 KK=1,16
      K=K1+KK
      XKPT=ABS(K-XMIDK)
      IF(IWTFLG.NE.7) GO TO 315
      WFNK=1.0
      GO TO 317
  315 CALL WEIGHT(XKPT,ZKRAD,WFNK)
  317 CONTINUE
      DO 330 JJ=1,16
      J=J1+JJ
      XJPT=ABS(J-XMIDJ)
      CALL WEIGHT(XJPT,ZJRAD,WFNJ)
      IF(IWTFLG.EQ.7.AND.J.LT.XMIDJ) SGN=-1.0
      A(JJ,KK)=A(JJ,KK)*WFNK*WFNJ*SGN
  330 CONTINUE
  320 CONTINUE
      GO TO 500
  400 XWIDTH=NWIDTH/2
      XHIGH=NHIGH/2
      DO 410 KK=1,16
      K=K1+KK
      YK=ABS(K-XMIDK)
      DO 410 JJ=1,16
      J=J1+JJ
      XK=ABS(J-XMIDJ)
      IF(YK.GT.XHIGH.OR.XK.GT.XWIDTH) A(JJ,KK)=(0.0,0.0)
  410 CONTINUE
      IF(IWTFLG.EQ.0) GO TO 500
      SGN=1.0
      DO 420 KK=1,16
```

```
      K=K1+KK
      XKPT=ABS(K-XMIDK)
      IF(IWTFLG.NE.7) GO TO 415
      WFNK=1.0
      GO TO 417
415   CALL WEIGHT(XKPT,ZKRAD,WFNK)
417   CONTINUE
      DO 430 JJ=1,16
      J=J1+JJ
      XJPT=ABS(J-XMIDJ)
      CALL WEIGHT(XJPT,ZJRAD,WFNJ)
      IF(IWTFLG.EQ.7.AND.J.LT.XMIDJ) SGN=-1.0
      A(JJ,KK)=A(JJ,KK)*WFNK*WFNJ*SGN
430   CONTINUE
420   CONTINUE
500   WRITE(01'LR1) A
      IF(MOD(LR1,LRJ).EQ.0) GO TO 505
      J1=J1+16
      GO TO 510
505   K1=(LR1/LRJ)*16
      J1=0
510   CONTINUE
      CALL DETACH(01,ISTAT, )
      WRITE(06,146) ISTAT
146   FORMAT('DETACH   ISTAT=',O20)
      CALL EXIT
      STOP
      END
```

SUBROUTINE PLOTD

1. PURPOSE

The purpose of this subroutine is to transmit a data array to the DUIS for subsequent plotting. The data samples to be plotted must be equally spaced, i.e., the independent variable increment between samples must be a constant.

2. INPUT PARAMETERS

Name	O/R	T	Description
DV	R	F	The array containing the dependent variable values to be plotted
NOUT	R	I	The number of samples to be processed in the array, DV
OR	R	F	The origin of the independent variable, i.e. the value of the independent variable that corresponds to the sample, DV(1)
DEL	R	F	The independent variable increment between samples
TH	O	F	Variable used to accumulate the maximum dependent variable modulus.

3. CALLING SEQUENCE

```
CALL PLOTD(DV,NOUT,OR,DEL,TH)
```

4. RESTRICTIONS, REQUIREMENTS, MISCELLANEOUS DATA

a. This subroutine is structured to process only data which has a fixed independent variable increment between samples.

b. The output data from this subroutine has the standard DUIS plot data format.

c. The output data from this subroutine has an accuracy of 12 bits.

d. The variable TH can be used to accumulate the maximum modulus value over a number of plots. This is mainly used in processing three-dimensional plots.

5. SUBPROGRAMS REQUIRED

ALOG10
CHOP

6. THEORY OF OPERATION

The first operation performed by this subroutine is to scan the input array, DV, to determine the largest, XMAX, and smallest, XMIN, dependent variable values. The parameter TH is then updated.

The dependent variable range XMAX=XMAX-XMIN is computed from the scanner output parameters. The mean of the dependent variable array, BIAS, is then computed. Next, two integers, J and K, are found such that the following condition is satisfied:

$$(J-1)*10^K < XMAX < J*10^K$$

The LSB to be used in digitizing the data is then computed as follows:

$$XLSB=FLOAT(J*10^K)/4000.0$$

Now all of the parameters necessary to characterize the plot data have been determined. Therefore, they are transmitted to the DUIS in the plot header record. This record contains the following parameters:

NOUT,OR,DEL,BIAS,XLSB

Each plot point is converted to an integer number in the following manner:

```
PTL  = (DV(J)-BIAS)/XLSB
IDAT = PTL+SIGN(0.5,PTL)
```

This integer number is converted to two ASCII characters and packed into a plot data record by the subroutine CHOP. Each time a record is filled (33 plot points) it is transmitted to the DUIS.

When all of the plot points have been transmitted to the DUIS a plot terminator record is transmitted to signify that the plot is complete. Control is returned to the calling (sub)program.

7. FORTRAN LISTING

```
C
C     17 APR 78
C     1420
C
      SUBROUTINE PLOTD(DV,NOUT,OR,DEL,TH)
      DIMENSION DV(1),ILINE(35)
      XMAX=DV(1)
      XMIN=DV(1)
C
C     SCAN DATA ARRAY TO FIND LARGEST (XMAX)
C     AND SMALLEST (XMIN) VALUES
C
      DO 100 J=1,NOUT
      XMAX=AMAX1(XMAX,DV(J))
      XMIN=AMIN1(XMIN,DV(J))
 100  CONTINUE
C
C     UPDATE TH
C
      TH=AMAX1(ABS(XMAX),ABS(XMIN),TH)
C
C     SECTION TO DETERMINE BIAS AND XLSB
C
      XMAX=XMAX-XMIN
      IF(XMAX.LT.1.0E-10) XMAX=1.0E-10
      BIAS=0.5*XMAX+XMIN
      PTL=ALOG10(XMAX)
      ITEST=IFIX(PTL)
      IF(PTL) 150,140,140
 140  X=XMAX/(10.0**ITEST)
      GO TO 160
 150  X=XMAX*(10.0**(IABS(ITEST)+1))
 160  N=IFIX(X+0.98)
      IF(PTL) 123,124,124
 123  XMAX=FLOAT(N)/(10.0**(IABS(ITEST)+1))
      GO TO 125
 124  XMAX=FLOAT(N)*(10.0**ITEST)
 125  XLSB=XMAX/4000.0
```

```
C
C     TRANSMIT THE PLOT HEADER RECORD TO DUIS
C
      WRITE(06,1000) NOUT,OR,DEL,BIAS,XLSB
 1000 FORMAT(4HzzX ,I6,4(',',1PE12.5))
C
C     SECTION TO CONVERT DATA TO 12 BIT FORM
C     AND TRANSMIT TO DUIS
C
      IFLG=0
      J=1
      K=1
 181  IF(J-NOUT) 200,200,201
 200  PTL=(DV(J)-BIAS)/XLSB
      IDAT=PTL+SIGN(0.5,PTL)
      CALL CHOP(IDAT,ILINE(K))
      IF(K-33) 220,222,222
 201  IFLG=1
 222  WRITE(06 1002)(ILINE(L),L=1,33)
 1002 FORMAT('>z',33R2,'>z')
      K=1
      IF(IFLG) 180,180,500
 220  K=K+1
 180  J=J+1
      GO TO 181
C
C     PLOT TRANSMISSION IS COMPLETE.
C     TRANSMIT THE TERMINATOR RECORD
C
 500   WRITE(06,1003)
 1003 FORMAT(' zzZ')
      RETURN
      END
```

PROGRAM PLTDVR

1. PURPOSE

The purpose of this subroutine is to process data for transmission to the DUIS for subsequent use in preparing 3-dimensional plots. The types of data normally processed by this subroutine are antenna far field patterns and antenna aperture illumination distributions.

2. INPUT PARAMETERS

Name	O/R	T	Description
AFILE	R	C	The name of the TSS PRMFL which the user wishes to process
JWID	R	I	The data array width in blocks. This parameter corresponds to LRJWID in the program FFT2DX or LRJ in the programs PLARY, RNDERR, or FILMOD.
JWIDSP	R	I	The width in blocks of the vertical strip to be processed
NBMAX	R	I	The number of the last block to be processed
LREC	R	I	The number of the first record to be processed
ISCN	R	I	Data processing mode flag. = 0 modulus data is processed = 1 real component only is processed.

3. RESTRICTIONS, REQUIREMENTS, MISCELLANEOUS DATA

a. The maximum value of the parameter JSID is 10. This corresponds to a row length of 160 samples. This limitation is arbitr ~y and was chosen to minimize the memory required by this program.

b. It should be recognized that the run time of this program is directly proportioned to the number of plot points transmitted to the DUIS. The number of plot points is given by the following expression:

NPTS=JSID*16*NROWS

The time required to transmit each plot point is 16.67 ms at 1200 Baud and 66.67 ms at 300 Baud. For example, a 3-D display 4 blocks wide and 64 rows tall would require 68.27 seconds at 1200 Baud.

4*16*64*16.67ms = 68.27 seconds

At 300 Baud the time required is 4.55 minutes.

c. This program is designed to work only with the DUIS.

4. SUBPROGRAMS REQUIRED

ATTACH
RANSIZ
PLOTD
DETACH
EXIT

5. THEORY OF OPERATION

After startup a message is printed to the user requesting the name of the TSS PRMFL that is to be processed. The user reply is tested to see if it is the word 'STOP'. If so, program execution is terminated. Otherwise, an attempt is made to access the file via the system subroutine, ATTACH. If the file cannot be accessed the system error code is printed and the user is requested to try again.

Providing an input file has been successfully attached, the following parameters are requested from the user:

JWID,JWIDSP,NBMAX

The parameter JWIDSP is tested to ensure that it does not exceed the smaller of JWID or 10. The following parameters are then requested from the user:

LRECF,ISCN

The parameter LRECF is tested to see if it is zero or negative. If so, control is transferred to Statement # 400 and the file is deaccessed via a call to the system subroutine DETACH. The program then returns to the procedure for accessing another file (Statement #110).

The JWIDSP blocks, starting with record number LREC, are loaded into the array A(J,K). Next, the 16 rows of data are processed one row at a time for output to the DUIS via the subroutine PLOTD. If ISCN=1 then the real components of the data samples are placed in the array DV. If ISCN=0 then a bipolar modulus function is computed from the output data. The reason for computing a bipolar modulus is that in the DUIS the data to be plotted is normally interpolated to provide smoother curves. The interpolation functions cannot accurately process data that possesses discontinuities which is the case for a true modulus function. The procedure used to convert the modulus is as follows. It has been noted that the modulus can become zero only if the real(x) and imaginary(y) components are both zero simultaneously. Also, it is known that both x and y are continuous functions. Therefore, if both x and y reverse signs between two sample points then the modulus must have gone through zero. Therefore, the scheme used was to reverse the sign of the modulus function (IP) each time x and y reversed signs at the same time. After each bipolar modulus value has been computed it is placed in the array DV. When the array DV is full (JSTOP values) then the array is rpocessed and transmitted to the DUIS via the subroutine PLOTD.

After each sweep is transmitted the program waits for the DUIS to return a status number, ISTAT. If the DUIS replies with ISTAT=1 then the sweep must be retransmitted since a transmission error was detected. If the DUIS replies with ISTAT=0 then the sweep was received with no errors detected and the processing of the next sweep is begun.

When all 16 rows of data have been transmitted to the DUIS then another 16 rows are read in and processed in the same manner as described above. If the end block NBMAX is reached then the program waits for the user to transmit a carriage return and then the program prints the maximum value of the data transmitted. Next, the TSS PRMFL is detached, control is transferred to Statement # 110, and the user is asked for another file name.

6. FORTRAN LISTING

```
C
C     17 APR 78       1345
C
      DIMENSION DV(256)
      COMPLEX A(160,16)
      CHARACTER AFILE*20,STOP*4,DONE*4
      DATA IOK/O400000000000/,STOP/'STOP'/
      DATA DONE/O007040033014/
 100  FORMAT(V)
C
C     REQUEST TSS FILE NAME FROM USER
C
 110  WRITE(06,100) 'INPUT FILE NAME ?'
      READ 100,AFILE
      IF(AFILE.EQ.STOP) GO TO 600
      CALL ATTACH(01,AFILE,1,1,ISTAT, )
      IF(ISTAT.EQ.IOK.OR.ISTAT.EQ.0) GO TO 70
      WRITE(06,510) ISTAT
      GO TO 110
 70   CALL RANSIZ(01,512)
C
C     REQUEST PARAMETERS FROM USER
C
      WRITE(06,100) 'Enter JWID,JWIDSP,NBMAX'
      READ 100,JWID,JWIDSP,NBMAX
      IF(JWIDSP.LE.10.AND.JWIDSP.LE.JWID) GOTO 75
      JWIDSP=10
      WRITE(06,100) 'JWIDSP > 10... SET TO 10'
      IF(JWIDSP.LE.JWID) GOTO 75
      JWIDSP=JWID
      WRITE(06,100) 'JWIDSP > JWID... SET JWIDSP=JWID'
 75   WRITE(06,100) 'Enter FIRST LREC,ISGN'
      READ 100,LRECF,ISGN
      TH=0.0
C
C     IF LRECF < 1 THEN TERMINATE PROCESSING THIS FILE
C
      IF(LRECF.LE.0) GO TO 400
```

```
      JSTOP=JWIDSP*16
      NST=1
      NSTP=16
      GO TO 200
C
C     SECTION TO READ IN A STRIP OF DATA
C
 210  LRECF=LRECF+JWID
      IF(LRECF.GT.NBMAX) GO TO 85
 200  LR1IN=LRECF
 190  READ(01'LR1IN)((A(J,K),J=NST,NSTP),K=1,16)
      NST=NSTP+1
      NSTP=NSTP+16
      LR1IN=LR1IN+1
      IF(NSTP.LE.JSTOP) GO TO 190
C
C     SECTION TO PROCESS A STRIP FOR OUTPUT
C
      NST=1
      NSTP=16
      DO 310 K=1,16
      IP=1
      DO 300 LL=1,JSTOP
      IF(ISGN.EQ.1) GO TO 770
      E=CABS(A(LL,K))
      X=REAL(A(LL,K))
      Y=AIMAG(A(LL,K))
      IF(LL.NE.1) GO TO 112
      XO=X
      YO=Y
 112  P1=XO*X
      P2=YO*Y
      IF(E.LT.1.0E-30) GO TO 111
      IF(P1.LT.0.0.AND.P2.LT.0.0) IP=-IP
      XO=X
      YO=Y
 111  DV(LL)=SIGN(E,IP)
      GO TO 300
 770  DV(LL)=REAL(A(LL,K))
 300  CONTINUE
 700  CALL PLOTD(DV,JSTOP,0.0,1.0,TH)
```

```
C
C     WAIT FOR DUIS TO REPLY WITH STATUS CODE
C
      READ 100,ISTAT
      IF(ISTAT.EQ.1) GO TO 700
 310  CONTINUE
      GO TO 210
C
C     ALL PLOT RECORDS IN THE FILE HAVE BEEN
C     SUCCESSFULLY TRANSMITTED TO DUIS
C
 85   WRITE(06,100) DONE
      READ 100,AFILE
      WRITE(06,100) 'TH=',TH
      GO TO 75
 400  CALL DETACH(01,ISTAT, )
      WRITE(06,410) ISTAT
 410  FORMAT('DETACH    ISTAT=',O20)
      GO TO 110
 510  FORMAT('UNSUCCESSFUL ATTACH    ISTAT=',O20)
 600  CALL EXIT
      STOP
      END
```

PROGRAM RNDERR

1. PURPOSE

This program adds random phase errors to the elements of an existing aperture distribution stored in a PRMFL. The resulting aperture may be stored on the input PRMFL or may be stored on another user specified PRMFL.

2. INPUT PARAMETERS

Name	O/R	T	Description
NTYPE	R	I	Determines the type of distribution. = 1 Uniform = 3 Gaussian
MAD1	R	I	The starting address for selecting random numbers from the random number array. ($1 \le MAD1 \le 128$)
JRND	R	I	Random number generator initialization constant. ($0 \le JRND \le 2^{36} - 1$)
LRJ	R	I	Number of logical record blocks in the x-direction (horizontally).
LRK	R	I	Number of logical record blocks in the y-direction (vertically).
UMEAN	O	F	Mean value of the uniform distribution in degrees.

UUEXT	O	F	Width of the uniform distribution in degrees.
XMEAN	O	F	Mean value of the Gaussian distribution in degrees.
SIGMA	O	F	Standard deviation of the Gaussian distribution in degrees.

3. RESTRICTIONS, REQUIREMENTS, MISCELLANEOUS DATA

a. The value of MAD1 must be in the range

$$1 \leq MAD1 \leq 128$$

b. The value of JRND must be in the range

$$0 < JRND < 2^{36} - 1$$

4. SUBPROGRAMS REQUIRED

RRAND
FXOPT

5. THEORY OF OPERATION

The program steps sequentially through the LREND (=LRJ*LRK) input logical record blocks starting with block number 1. For each element in a block, a random phase component is generated according to the equation

PHERR = RRAND(NTYPE)*0.017453.

RRAND is the random number generator function and the constant 0.017453 is a degree-to-radian conversion factor. With the phase error expressed in radians a complex representation C is calculated as

C = CMPLX(COS(PHERR),SIN(PHERR)).

This number is then multiplied by the value of the antenna aperture element to be modified. The multiplication is defined by the equation

A(I) = A(I)*C.

Note that this is a complex multiplication by a unit vector and that no change in energy is introduced as a result of the phase errors. This process is repeated for all of the elements in each block. As the elements are processed each block is written off to the appropriate PRMFL, either the input PRMFL or another that is chosen by the user.

```
      IF(ISTAT.EQ.IOK.OR.ISTAT.EQ.0) GO TO 600
      WRITE(06,550) ISTAT
      GO TO 580
600   LREND=LRJ*LRK
      DO 700 LRIN=1,LREND
      READ(01'LRIN) A
      DO 650 I=1,256
      PHERR=RRAND(NTYPE)*0.017453
      C=CMPLX(COS(PHERR),SIN(PHERR))
650   A(I)=A(I)*C
      WRITE(LROUT'LRIN) A
700   CONTINUE
800   CALL DETACH(01,ISTAT, )
      WRITE(06,750) ISTAT
      CALL DETACH(02,ISTAT, )
      WRITE(06,750) ISTAT
750   FORMAT('DETACH   ISTAT=',O20)
      CALL EXIT
      STOP
      END
```

FUNCTION RRAND

1. PURPOSE

This function generates random numbers for use in various subprograms of PAAS. Samples from the uniform, Gaussian, and Rayleigh distributions can be generated. This function is based on the function RRAND used in RADSIM.

2. INPUT PARAMETERS (Common Area BLKRND)

Name	O/R	T	Description
MAD1	R	I	Random Number Table pointer
JRND	R	I	Random integer from previous execution of RRAND
XMEAN	R	F	Mean value of the Gaussian distribution
SIG2SQ	R	F	An intermediate parameter used in computing Gaussian and Rayleigh distributions
UL	R	F	An intermediate parameter used in determining the uniform distribution mean value
UEXT	R	F	An intermediate parameter used in determining the uniform distribution width

3. CALLING SEQUENCE:

VAR = RRAND (NTYPE)

NTYPE R I Control integer which specifies the type of distribution to be generated.

NTYPE = 1 Uniform distribution (floating point output)
= 2 Rayleigh
= 3 Gaussian

VAR contains the random sample generated by the function from the NTYPE probability distribution.

4. RESTRICTIONS, REQUIREMENTS, MISCELLANEOUS DATA

a. Before any call can be made to the function RRAND the labeled common area must be loaded as follows:

MAD1 Any integer on the interval: $0 \leq MAD1 \leq 127$

JRND Any integer on the interval: $0 \leq JRND \leq 2^{35}$

XMEAN Mean value of Gaussian distribution

SIG2SQ = -2.0*SIGMA*SIGMA

Where: SIGMA is the standard deviation of the Gaussian or Rayleigh distribution

UL = UMEAN - 0.5 * UEXT

Where: UMEAN is the mean value of the uniform distribution

UEXT is the width of the uniform distribution

UEXT = UEXT/2^{35}

b. For convenience and to minimize program steps, the array IRAND was equivalenced to the arrays NRND1 and NRND2, but displaced by one location. This structure allows an address of zero to be used, i.e., an address of zero will access IRAND (0), which overlays NRND(1). If this were not done, a test would have to be performed on MAD1 to ensure that an address of zero did not occur.

c. The distribution transformations used herein are documented in the following reference:

Robert E. Machol (ed.), System Engineering Handbook, McGraw-Hill, N.Y., 1965, pp. 40-28, 40-29.

d. Referenced labeled common areas:

BLKRND

e. Source PRMFL:

/PLARY

5. SUBPROGRAMS REQUIRED

FLD
FLOAT
SQRT

6. THEORY OF OPERATION

For each call to the function RRAND a number KRND is selected from the random number table, IRAND. The address of the number selected from the table is MAD1, a random number. The number KRND is added to the product of JRND and IMULT and stored in IRND. The variable JRND is the random number which was generated by the previous execution of the function and IMULT is a multiplier chosen because it results in good bit scrambling. The multiplication of JRND by IMULT causes the bits of JRND to be scrambled. This scrambled word is then added to KRND, just retrieved from the table to form the new random number, IRND. The sign bit of IRND is set to zero to ensure a positive number. The random number IRND just generated is an integer having a uniform distribution from 0 to $2^{35}-1$. IRND is placed in the random number table location previously occupied by the KRND. In this manner the random number table is updated by generating new random numbers and inserting them into the table. From the random number IRND, 7 bits are selected to determine the new address MAD1 to be used in the next call to the function. The 7-bit address field allows the addresses to range from 0 to 127. Once the random number is generated JRND is set equal to IRND for use in the subsequent executions of the function. In order to convert this number to a floating point number, r, having a uniform distribution from 0 to 1.0, the following conversion is used:

$$r = \text{FLOAT(IRND)}/2^{35}$$

From this uniform distribution other probability distributions can be generated by using transformations which map a uniform distribution into the desired distribution. The following is a list of the transformations used in this function:

a. Uniform distribution

$$p(x) = 1/(b-a) \qquad ,(a \le x \le b)$$

$$x_n = (b - a)r_n + a$$

b. Rayleigh distribution

$$p(x) = (x/\sigma^2)\exp(-x^2/2\sigma^2) \qquad ,(x \ge 0)$$

$$x_n = -2\sigma^2 \ln r_n$$

c. Gaussian distribution

$p(x) = (1/2\pi\sigma) \exp[(x-\mu)^2/2\sigma^2]$, $(-\infty < x < \infty)$

$x_n = 2\sigma^2 \ln r_n * \text{Cos } 2\pi r_{n+1}$

$y_n = 2\sigma^2 \ln r_n * \text{Sin } 2\pi r_{n+1}$

7. FORTRAN LISTING:

```
C
C      ****************************************
C
       FUNCTION RRAND(NTYPE)
       COMMON/BLKRND/MAD1,JRND,XMEAN,SIG2SQ,UL,UEXT
       DIMENSION IRAND(128),NRND1(65),NRND2(64)
       DATA NRND1/15134181997,27509664464,30323512272,
&       14051007893,16402190290,26306990212,11260717646,
&       16801629773,11849273156,19404991345,06977712830,
&       02883434137,33025570091,11012391622,13411365861,
&       31267410086,13462139250,26463885902,24219774296,
&       11557820695,30512809719,12630506319,17722780814,
&       04722597022,16900280091,16243824041,16388044606,
&       26212698408,13570004754,11188309528,29134237821,
&       13164942096,29908968258,03564986686,24513426529,
&       25262307992,16416251777,32749370939,21116178576,
&       19395173043,20743061171,21319359579,19074491967,
&       19244390324,08846123356,27142309994,15825176938,
&       16410917813,23416520791,28825638452,10800745449,
&       01702686304,17006458873,16841482774,26473264721,
&       17160292937,29260744156,8883554486,3669953728,
&       16068801392,5883873859,14824731880,18081451748,
&       8160418880,30068227383/
       DATA NRND2/12068158044,06847664659,15416782660,
&       25052201840,13988647055,01734737408,07289355507,
&       28120785669,32320902560,19471392797,07683759917,
&       24386072834,29317493972,07114843643,16232718423,
&       29170604246,26866574818,20335880812,14861357546,
&       25072568248,31374670078,13676667951,30463132192,
&       20172084006,16184261842,14974210467,10283018420,
&       18310257399,18938188207,01286074697,19662214195,
&       01577045480,16742867695,11686848767,18174114680,
&       30892487160,30892487160,28360949700,33368415709,
&       17235921632,25322444850,30007056175,13488881553,
&       30224148581,07655423387,32626402591,13101024674,
&       30533512969,07218771539,00229536870,29198604401,
&       33122308420,29107616508,16534467415,3669736170,
&       3491463822,5804776974,30256545186,10832795361,
```

```
     &        18174114680,10556707007,10140208896,9779017119,
     &        19382343178/
              DATA IMULT/1220703125/,IMAX/4294967296/
              DATA N2P16/65536/,CI/2.9103830E-11/
              EQUIVALENCE (IRAND(1),NRND1(2)),(IRAND(65),NRND2(1))
      10      KRND=IRAND(MAD1)
              IRND=KRND+JRND*IMULT
              IRND=FLD(1,35,IRND)
              IRAND(MAD1)=IRND
              MAD1=FLD(15,7,IRND)
              GO TO (200,300,400),NTYPE
      200     RRAND=FLOAT(IRND)*UEXT+UL
              JRND=IRND
              RETURN
      300     RRAND=SQRT(SIG2SQ*ALOG(FLOAT(IRND)*CI))
              JRND=IRND
              RETURN
      400     I1=FLD(1,17,KRND)-N2P16
              I2=FLD(18,17,KRND)-N2P16
              IS=I1*I1+I2*I2
              IF(IS.LT.IMAX) GO TO 20
              JRND=IRND
              GO TO 10
      20      S=1.0/FLOAT(IS)
              VCOS=S*FLOAT(I1*I1-I2*I2)
              VSINE=S*2.0*FLOAT(I1*I2)
              DUM=SQRT(SIG2SQ*ALOG(FLOAT(JRND)*CI))
              RRAND=DUM*VCOS+XMEAN
              DUM=DUM*VSINE+XMEAN
              JRND=IRND
              RETURN
              END
```

PROGRAM RTI4

1. PURPOSE

The purpose of this subroutine is to produce a compact representation of three-dimensional data, e.g. antenna far field pattern, antenna aperture illumination distributions, and radar ambiguity diagrams. The procedure used herein is to represent the modulus of each sample with an alphanumeric character.

2. INPUT PARAMETERS

Name	O/R	T	Description
AFILE	R	C	The name of the TSS PRMFL which the user wishes to process
FLOOR	R	F	The reference in dB below which everything is represented by dashes(-) on the RTI plot
YINC	R	F	The increment in dB between each successive alphanumeric symbol. See Table RTI4-1.
JWID	R	I	The data array width in blocks. This parameter corresponds to LRJWID in the program FFT2DX or LRJ in the programs PLARY, RNDERR, or FILMOD.
NBMAX	R	I	The number of the last block to be displayed
LREC	R	I	The number of the first record to be processed

IWD	R	I	The width of the output character matrix.

3. RESTRICTIONS, REQUIREMENTS, MISCELLANEOUS DATA

   a. The maximum value of the parameter IWD is 128. In addition, the value of IWD should be less than or equal to JWID*16.

   b. Two examples of the relative dB values for specified values of FLOOR and YINC are shown in Table RTI4-1.

4. SUBPROGRAMS REQUIRED

   FPARAM
   ATTACH
   RANSIZ
   DETACH

TABLE RTI4-1

EXAMPLES OF RELATIVE VALUE OF LETTERS, NUMBERS, AND SYMBOLS

FLOOR = -20.0 YINC = 1.0				FLOOR = 10.0 YINC = 0.5
dB level				dB level
	20.0	$	30.0	
	19.0	.	29.5	
	18.0	*	29.0	
	17.0	+	28.5	
	16.0	0	28.0	
	15.0	1	27.5	
	14.0	2	27.0	
	13.0	3	26.5	
	12.0	4	26.0	
	11.0	5	25.5	
	10.0	6	25.0	
	9.0	7	24.5	
	8.0	8	24.0	
	7.0	9	23.5	
	6.0	A	23.0	
	5.0	B	22.5	
	4.0	C	22.0	
	3.0	D	21.5	
	2.0	E	21.0	
	1.0	F	20.5	
	0.0	G	20.0	
	-1.0	H	19.5	
	-2.0	I	19.0	
	-3.0	J	18.5	
	-4.0	K	18.0	
	-5.0	L	17.5	
	-6.0	M	17.0	
	-7.0	N	16.5	
	-8.0	O	16.0	
	-9.0	P	15.5	
	-10.0	Q	15.0	
	-11.0	R	14.5	
	-12.0	S	14.0	
	-13.0	T	13.5	
	-14.0	U	13.0	
	-15.0	V	12.5	

-16.0	W	12.0
-17.0	X	11.5
-18.0	Y	11.0
-19.0	Z	10.5
-20.0	-	10.0

5. THEORY OF OPERATION

After startup a message is printed to the user requesting the name of the TSS PRMFL that is to be processed. The user reply is tested to see if it is the word 'STOP'. If so, program execution is terminated. Otherwise, an attempt is made to access the file via the system subroutine, ATTACH. If the file cannot be accessed the system error code is printed and the user is requested to try again.

Assuming the user has successfully accessed a file, the program then requests the following parameters:

FLOOR, YINC, JWID, NBMAX

Then the following parameters are requested:

LREC,IWD

The parameter LREC is tested to see if it is zero or negative. If so, the file is deaccessed via a call to the system subroutine DETACH. The program then returns to the procedure for accessing another file (Statement # 99). The parameter IWD is tested to ensure that its value is in the range from 1 to 128. The number of blocks (NREC) required for the specified display width (IWD) is computed and compared to JWID. If NREC > JWID then NREC is set equal to JWID and IWD is set equal to NREC*16. In other words, IWD is made as large as possible for the set of data to be processed.

The NREC blocks starting with LREC are read in and processed in the following manner. The modulus of each complex valued sample is computed and converted to dB. Then the dB value (XM) is mapped to an integer number (IADD) on the interval from 1 to 41 by the following procedure:

```
IADD=IFIX((XM-FLOOR)/YINC+0.5)+1
IF(IADD.GT.40) IADD=41
IF(IADD.LT.1) IADD=1
```

The integer numbers 41 and 1 correspond to $ and -, respectively, and represent values either too large or too small to be displayed for the set of parameters, FLOOR and YINC, specified by the user. The integer numbers are used to 'pull' the corresponding character ASCII code from the character table, CTABL. These characters are then stored in the output character matrix, XRTI.

Once the NREC blocks have been processed, the character matrix is transmitted to the user. This character matrix contains 16 rows of data. The above procedure is repeated until the record number NBMAX is encountered. Control then returns to the statement (# 75) requesting the first logical record to be processed.

6. FORTRAN LISTING

```
C
C       26 APR 78      0845
C
      COMPLEX TEMP(16,16)
      DIMENSION XRTI(32,16),CTABL(41)
      DATA CTABL/O137,O132,O131,O130,O127,O126,O125,
&   O124,O123,O122,O121,O120,O117,O116,O115,O114,O113,
&   O112,O111,O110,O107,O106,O105,O104,O103,O102,O101,
&   O71,O70,O67,O66,O65,O64,O63,O62,O61,O60,O53,
&   O52,O56,O44/
      CHARACTER AFILE*20,STOP*4
      DATA IOK/O400000000000/,STOP/'STOP'/
C
C     REQUEST TSS FILE NAME FROM USER
C
      CALL FPARAM(1,130)
 99   WRITE(06,105)
 105  FORMAT('INPUT DESIRED FILE NAME')
      READ(05,50) AFILE
 50   FORMAT(A20)
      IF(AFILE.EQ.STOP) GOTO 310
      CALL ATTACH(01,AFILE,1,1,ISTAT, )
      IF(ISTAT.EQ.IOK.OR.ISTAT.EQ.0) GO TO 101
      WRITE(06,410) ISTAT
      GO TO 99
C
C     REQUEST PARAMETERS FROM USER
C
 101  WRITE(06,100)
 100  FORMAT('FLOOR,YINC,JWID,NBMAX')
      READ 115,FLOOR,YINC,JWID,NBMAX
      CALL RANSIZ(01,512)
 75   WRITE(06,110)
 110  FORMAT('Enter FIRST LREC, DISPLAY WIDTH')
      READ 115,LREC,IWD
 115  FORMAT(V)
C
```

```
C     IF LREC < 1 THEN TERMINATE PROCESSING THIS FILE
C
      IF(LREC.LE.0) GO TO 700
      IF(IWD.LE.128.AND.IWD.GT.0) GOTO 150
      IWD=128
      WRITE(06,115) 'DISPLAY WIDTH TOO BIG. SET TO 128'
 150  NREC=(IWD+15)/16
      IF(NREC.LE.JWID) GOTO 160
      NREC=JWID
      IWD=NREC*16
      WRITE(06,115) 'DISPLAY WIDTH > AVAIL DATA...'
     *  ' CHANGED TO: IWD=JWID*16'
C
C     BEGIN PROCESSING FOR A STRIP
C
 160  IWD4=IWD/4
      LREND=LREC+NREC-1
      IF(LREND.GT.NBMAX) GOTO 75
      NST=-1
      DO 800 IREAD=LREC,LREND
      READ(01'IREAD) TEMP
C
C     PROCEDURE TO PROCESS ONE BLOCK OF DATA
C
      DO 200 J=1,16
      IBIT=MOD(J-1,4)*9
      IWORD=1+(NST+J)/4
      DO 300 IR=1,16
      XM=CABS(TEMP(J,IR))
      IF(XM.LT.1.0E-10) GO TO 120
      XM=20.0*ALOG10(XM)
      GO TO 121
 120  XM=-100.0
 121  IADD=IFIX((XM-FLOOR)/YINC+0.5)+1
      IF(IADD.GT.40) IADD=41
      IF(IADD.LT.1) IADD=1
      FLD(IBIT,9,XRTI(IWORD,IR))=CTABL(IADD)
 300  CONTINUE
 200  CONTINUE
      NST=NST+16
 800  CONTINUE
```

```
C
C     SECTION TO TRANSMIT OUTPUT TO USER
C
      DO 450 IR=1,16
      WRITE(6,1000)(XRTI(J,IR).J=1,IWD4)
 1000 FORMAT(1H ,32A4)
 450  CONTINUE
      LREC=LREC+JWID
      GOTO 160
C
C     DISCONNECT FROM USER TSS FILE
C
 700  CALL DETACH(01,ISTAT, )
 500  WRITE(06,510) ISTAT
      GO TO 99
 410  FORMAT('UNSUCCESSFUL ATTACH','   ISTAT = '.O20)
 510  FORMAT(' DETACH  ISTAT = ',O20)
 310  CALL EXIT
      STOP
      END
```

PROGRAM TBLS

1. PURPOSE

The program TBLS computes and tabulates the sampled values of selected weighting functions. The program also generates data which are used in checking probability density functions of space tapered arrays estimated by program PDFESTR.

2. INPUT PARAMETERS

Name	O/R	T	Description
NTYPE	R	I	Determines the weighting function tabulated. = 1 Cosine on a pedestal to a power = 2 Blackman = 3 Kaiser = 4 Bartlett or triangular = 5 Taylor = 6 Bessel = 7 Cubic = 8 Bayliss
WTPED	O	F	The height of the pedestal for cosine on a pedestal to a power weighting.
NWTPOW	O	I	The power of the cosine function for cosine on a pedestal to a power weighting.
IRAD	R	I	The radius (or half span of a linear array), in units of elements of the array.

WTRAD	R	F	The radius (or half span of a linear array) of the weighting function.
WKASIR	O	F	The Kaiser variable for the trade-off between main lobe width and side lobe amplitude.
BESEDG	O	F	The radius scaling constant for the Bessel weighting.
BESCAL	O	F	The maximum weight amplitude for the Bessel weight. This corresponds to a radius of zero
CUBK	O	F	The weighting amplitude scaling constant for the cubic weight.
IDB	O	I	The design side lobe amplitude in dB for the Taylor or Bayliss weighting.
NBAR	O	I	The number of zeros used to approximate the Dolph-Chebyschev weighting distribution in the Taylor or Bayliss weighting.

3. RESTRICTIONS, REQUIREMENTS, MISCELLANEOUS DATA

a. The range of NBAR must be

$$3 \le NBAR \le 20$$

for the Taylor weighting.

b. The range of NBAR must be

$$3 \le NBAR \le 19$$

for the Bayliss weighting.

4. SUBPROGRAMS REQUIRED

EXPND
BESS
GAM
WEIGHT

5. THEORY OF OPERATION

This program is a driver for the subroutine WEIGHT. A loop is set up that starts at zero and goes to IRAD in steps of one. In each pass of the loop a call to WEIGHT is made and a value of the selected weighting function is returned for the particular radius cell. The values are stored and printed on the time-sharing terminal in a tabular fashion.

6. FORTRAN LISTING

```
      COMMON NTYPE,WTPED,NWTPOW,WKASIR,F(20),B(20),ANG,
     &       NBAR,BESCAL,CUBK1,PII2,BESS1,IAZ,KKK,WMAX,
     &       BESEDG
      INTEGER DB
      DIMENSION U(20),OUT(205),SIGG(5),IX(20),BZERO1(20)
      DIMENSION Z(20)
      DATA U/1.2196699,2.2331306,3.2383154,4.2410628,
     &  5.2427643,6.2439216,7.2447598,8.2453948,
     &  9.2458927,10.2462933,11.246624,12.246900,
     &  13.247131,14.247334,15.247508,16.247663,
     &  17.247796,18.247920,19.248027,20.248125/
      DATA BZERO1/0.586067,1.6970509,2.7171939,3.726137,
     &  4.7312271,5.7345205,6.7368281,7.7385356,
     &  8.7398505,9.7408945,10.7417435,11.7424475,
     &  12.7430408,13.7435477,14.7439856,15.7443679,
     &  16.7447044,17.745003,18.7452697,19.7455093/
      WRITE(06,50) 'PDFESTR DATA OR TABLES? (0 OR 1)'
      READ 50,IAZ1
      IAZ=1
      ANG=0.0
      PI=3.1415926
      PII2=2.0/(PI*PI)
      BESS1=1.0/BESS(0.0,0.0)
 1850 WRITE(06,50) 'NTYPE'
      READ 50,NTYPE
      IF(NTYPE.EQ.-1) GO TO 1830
      GO TO (1100,1200,1300,1200,1500,1360,1370,1500),
     &      NTYPE
 1100 WRITE(06,50) 'WTPED,NWTPOW,IRAD,WTRAD'
      READ 50,WTPED,NWTPOW,IRAD,WTRAD
      GO TO 1600
 1200 WRITE(06,50) 'IRAD,WTRAD'
      READ 50,IRAD,WTRAD
      GO TO 1600
 1300 WRITE(06,50) 'WKASIR,IRAD,WTRAD'
      READ 50,WKASIR,IRAD,WTRAD
      GO TO 1600
 1360 WRITE(06,50) 'BESEDG,BESCAL,IRAD,WTRAD'
```

```
      READ 50,BESEDC,BESCAL,IRAD,WTRAD
      GO TO 1600
1370  WRITE(06,50) 'CUBK,IRAD,WTRAD'
      READ 50,CUBK,IRAD,WTRAD
      XX=SQRT(WTRAD*WTRAD/3.0)
      CUBK1=CUBK/ABS(XX*(XX-WTRAD)*(XX+WTRAD))
      GO TO 1600
1500  WRITE(06,50) 'IRAD,WTRAD'
      READ 50,IRAD,WTRAD
      WRITE(06,50) 'ALL OR SINGLE DB LEVEL? (0 OR 1)'
      READ 50,IA
      IF(IA.EQ.0) GO TO 60
      WRITE(06,50) 'IDB'
      READ 50,IDB
      IDB1=IDB
      IDB2=IDB1-1
      IDB3=1
      WRITE(06,50) 'ALL OR SINGLE NBAR? (0 OR 1)'
      READ 50,IAX
      IF(IAX.EQ.0) GO TO 70
410   WRITE(06,50) 'NBAR'
      READ 50,INBAR
      IBAR1=INBAR
      IBAR2=IBAR1-1
      GO TO 80
60    IDB1=20
      IDB2=80
      IDB3=5
70    IBAR1=3
      IBAR2=20
80    CONTINUE
50    FORMAT(V)
      DO 20 I=1,20
      IX(I)=I
20    CONTINUE
      I1=IRAD+1
      DO 300 III=IDB1,IDB2,IDB3
      DB=FLOAT(III)
      IOUT=1
      SIG1=1
      XA=10.0**(III/20.0)
```

```
      A=ALOG(XA+SQRT(XA*XA-1))/PI
      AASQ=A*A
      IT=0
      IT1=0
      NSGFL=0
      DO 200 NBAR=IBAR1,IBAR2
      IF(NBAR.EQ.20.AND.NTYPE.EQ.8) GO TO 200
      IF(NTYPE.EQ.8) GO TO 1900
      IF(NSGFL.NE.0) GO TO 30
      SIGP1=U(NBAR+1)/SQRT(AASQ+(NBAR+0.5)**2)
 30   SIGG(SIG1)=U(NBAR)/SQRT(AASQ+(NBAR-0.5)**2)
      SG=SIGG(SIG1)
      SGSQ=SG*SG
      IF(SG.LE.SIGP1.AND.IAX.EQ.1) GO TO 400
      IF(SG.LE.SIGP1) GO TO 200
      NSGFL=1
      SIGP1=0.0
      DO 252 I=1,NBAR-1
      FNUM=1.0
      FDNM=1.0
      T=U(I)*U(I)
      XII=-0.5
      DO 254 II=1,NBAR-1
      XII=XII+1.0
      FNUM=FNUM*(1.0-T/(SGSQ*(AASQ+(XII*XII))))
      IF(II.EQ.I) GO TO 254
      FDNM=FDNM*(1.0-T/(U(II)*U(II)))
 254  CONTINUE
      ARG=PI*U(I)
      F(I)=-BESS(0.0,ARG)*FNUM/FDNM
      F(I)=F(I)/(BESS(0.0,ARG)**2)
 252  CONTINUE
      GO TO 1910
 1900 Z(1)=0.9858302+0.0333885*DB+0.00014064*DB*DB
&            -0.0000019*DB*DB*DB+0.00000001*DB*DB*DB*DB
      Z(1)=Z(1)*Z(1)
      Z(2)=2.00337487+0.01141548*DB+0.0004159*DB*DB
&            -0.00000373*DB*DB*DB+0.00000001*DB*DB*DB*DB
      Z(2)=Z(2)*Z(2)
      Z(3)=3.00636321+0.00683394*DB+0.00029281*DB*DB
&            -0.00000161*DB*DB*DB
```

```
      Z(3)=Z(3)*Z(3)
      Z(4)=4.00518423+0.00501795*DB+0.00021735*DB*DB
     &      -0.00000088*DB*DB*DB
      Z(4)=Z(4)*Z(4)
      DO 761 I=5,NBAR+1
      Z(I)=AASQ+I*I
 761  CONTINUE
       IF(NSCFL.NE.0) GO TO 1930
      SIGP1=BZERO1(NBAR+2)/SQRT(Z(NBAR+1))
 1930 SIGG(SIG1)=BZERO1(NBAR+1)/SQRT(Z(NBAR))
      SC=SIGG(SIG1)
      SCSQ=SC*SC
      IF(SC.LE.SIGP1.AND.IAX.EQ.1) GO TO 400
      IF(SC.LE.SIGP1) GO TO 200
      NSCFL=1
      SIGP1=0.0
      DO 1940 I=1,NBAR
      FNUM=1.0
      T=BZERO1(I)*BZERO1(I)
      FDNM=1.0-T/(BZERO1(1)*BZERO1(1))
      IF(I.EQ.1) FDNM=1.0
      DO 1950 II=1,NBAR-1
      FNUM=FNUM*(1.0-T/(SCSQ*Z(II)))
      IF(I-1.EQ.II) GO TO 1950
      FDNM=FDNM*(1.0-T/(BZERO1(II+1)*BZERO1(II+1)))
 1950 CONTINUE
      B(I)=(2.0*T/BESS(1.0,BZERO1(I)*PI))*FNUM/FDNM
 1940 CONTINUE
      P0=0.4797212+0.01456692*(DB)-0.00018739*(DB*DB)
     &   +0.00000218*(DB*DB*DB)-0.00000001*(DB*DB*DB*DB)
      P0=P0*SC
      P0SQ=P0*P0
      PIP0SQ=P0SQ*PI*PI
      FNUM=1.0
      FDNM=1.0-P0SQ/(BZERO1(1)*BZERO1(1))
      DO 772 I=1,NBAR-1
      FNUM=FNUM*(1.0-P0SQ/(SCSQ*Z(I)))
      FDNM=FDNM*(1.0-P0SQ/(BZERO1(I+1)*BZERO1(I+1)))
 772  CONTINUE
      C=PIP0SQ-1.0
      C=C*BESS(1.0,P0*PI)*FNUM/FDNM
```

```
      C=1.0/C
      DO 773 I=1,NBAR
      B(I)=-B(I)*C
 773  CONTINUE
 1910 SIG1=SIG1+1
      IF(IAZ1.EQ.1) GO TO 55
      WMAX=0.0
      DO 51 I=1,IRAD+1
      RAD=FLOAT(I-1)
      CALL WEIGHT(RAD,WTRAD,WFUNC)
      WMAX=AMAX1(WMAX,WFUNC)
 51   CONTINUE
 55   IT=IT+1
      DO 100 K=1,IRAD+1
      RAD=FLOAT(K-1)-0.5*(IAZ1-1)
      CALL WEIGHT(RAD,WTRAD,WFUNC)
      IF(IAZ1.EQ.0) WFUNC=WFUNC*2.0*PI*RAD/WMAX
      OUT(IOUT)=WFUNC
      IOUT=IOUT+1
      IF(IAX.EQ.1.AND.K.EQ.IRAD+1) GO TO 500
      IF(NTYPE.NE.8) GO TO 56
      IF((MOD(IT,5).EQ.0.OR.NBAR.EQ.19).AND.K.EQ.IRAD+1)
&       GO TO 120
      GO TO 100
 56   IF((MOD(IT,5).EQ.0.OR.NBAR.EQ.20).AND.K.EQ.IRAD+1)
&           GO TO 120
      GO TO 100
 500  WRITE(06,995) III,A
      WRITE(06,997) NBAR
 997  FORMAT(' NBAR=',4X,I2,//)
      WRITE(06,996) SIGG(1)
 996  FORMAT(' SIGMA=',/,3X,F12.10,//)
      DO 975 LL=1,IRAD+1
 975  WRITE(06,994) LL-1,OUT(LL)
 994  FORMAT(I3,F12.10)
      GO TO 100
 120  IT1=IT1+1
      IF(IT1.GT.1) GO TO 130
      WRITE(06,995) III,A
 995  FORMAT(//,'DB=',I2,//'A=',F12.10,//)
 130  WRITE(06,998)(IX(I),I=NBAR-IT+1,NBAR)
```

```
998  FORMAT('NBAR=',4X,I2,4(10X,I2))
     WRITE(06,1998)
     WRITE(06,993)(SIGG(JJ),JJ=1,IT)
993  FORMAT('SIGMA=',/,3X,5(F12.10))
     WRITE(06,1998)
1998 FORMAT(/)
     ITT=5
     IF(NTYPE.NE.8) GO TO 57
     IF(NBAR.EQ.19.AND.IT.NE.5) ITT=MOD(IT,5)
     GO TO 58
57   IF(NBAR.EQ.20.AND.IT.NE.5) ITT=MOD(IT,5)
58   DO 75 LL=1,IRAD+1
75   WRITE(06,999) LL-1,(OUT(LL+I-I1),I=I1,I1*ITT,I1)
999  FORMAT(I3,5(F12.10))
     WRITE(06,1999)
1999 FORMAT(//)
     IOUT=1
     IT=0
     SIG1=1
100  CONTINUE
200  CONTINUE
300  CONTINUE
     GO TO 420
400  WRITE(06,50) 'INVALID VALUE FOR NBAR'
     GO TO 410
1600 WMAX=1.0
     IF(NTYPE.EQ.6) WMAX=BESCAL
     IF(NTYPE.EQ.7) WMAX=CUBK
     DO 1700 K=1,IRAD+1
     RAD=FLOAT(K-1)-0.5*(IAZ1-1)
     CALL WEIGHT(RAD,WTRAD,WFUNC)
     IF(IAZ1.EQ.0) WFUNC=WFUNC*2.0*PI*RAD/WMAX
     OUT(K)=WFUNC
1700 CONTINUE
     WRITE(06,1705) NTYPE
1705 FORMAT(//,' NTYPE=',I3,//)
     GO TO (1710,1800,1730,1800,1830,1740,1750),NTYPE
1710 WRITE(06,1715) WTPED,NWTPOW
1715 FORMAT(' WTPED=',F12.5,'     NWTPOW=',I3,//)
     GO TO 1800
1730 WRITE(06,1735) WKASIR
```

```
1735 FORMAT(' WKASIR=',F12.5,//)
     GO TO 1800
1740 WRITE(06,1745) BESCAL,BESEDG
1745 FORMAT(' BESCAL=',F12.5,'    BESEDG=',F12.5,//)
     GO TO 1800
1750 WRITE(06,1755) CUBK
1755 FORMAT(' CUBK=',F12.5,//)
1800 DO 1820 LL=1,IRAD+1
     WRITE(06,1810) LL-1,OUT(LL)
1810 FORMAT(I3,F12.5)
1820 CONTINUE
420  CONTINUE
     GO TO 1850
1830 CALL EXIT
     STOP
     END
```

SUBROUTINE WEIGHT

1. PURPOSE

This program is used to compute the values of various weighting functions. The weighting functions include the cosine on a pedestal to a power, Blackman, Kaiser, Bartlett or triangular, Taylor, cubic, and Bayliss.

2. INPUT PARAMETERS

Name	O/R	T	Description
RAD	R	F	Independent variable for the weighting function evaluation.
WTRAD	R	F	Radius of the specified weighting function. (For a linear array, WTRAD is the half span of the weighting function).
IAZ	R	I	Flag that determines whether the subroutine generates amplitude or statistical weighting data. = 0 Statistical weighting specified by the probability density function defined by the chosen weighting function. = 1 Normal amplitude weighting.

IWTFLG	R	I	Determines the weighting function evaluated. = 1 Cosine on a pedestal to a power = 2 Blackman = 3 Kaiser = 4 Bartlett or triangular = 5 Taylor = 6 Bessel = 7 Cubic = 8 Bayliss

<u>For IWTFLG = 1</u>

WTPED	O	F	Height of the pedestal for cosine on a pedestal to a power weighting.
NWTPOW	O	I	Power of the cosine for cosine on a pedestal to a power weighting.

<u>For IWTFLG = 3</u>

WKASIR			Kaiser variable for the trade-off between main lobe width and side lobe amplitude.

<u>For IWTFLG = 5</u>

F(20)	O	F	A set of constants used in evaluating the Taylor weighting (See Section 4).
DB	O	F	The design side lobe amplitude in dB for the Taylor weighting.
NBAR	O	I	Number of zeros used to approach the ideal pattern function for the Taylor weighting.

PII2	O	F	A constant, PII2 = $2.0/\pi^2$.
BESS1	O	F	A constant, BESS1 = $1.0/J_0(0.0)$.

<u>For IWTFLG = 6</u>

BESCAL	O	F	Maximum weighting amplitude for the Bessel weighting (at RAD = 0.0).
BESEDG	O	F	Radius scaling constant for the Bessel weighting.

<u>For IWTFLG = 7</u>

CUBK	O	F	Amplitude scaling constant for the cubic weight.

<u>For IWTFLG = 8</u>

B(20)	O	F	A set of constants used in evaluating the Bayliss weighting (See Section 4).
ANG	O	F	Azimuth angle independent variable for evaluation of the Bayliss weighting.
DB	O	F	The design side lobe design amplitude in dB for the Bayliss weighting.
NBAR	O	I	Number of zeros used to approximate the ideal pattern friction for the Bayliss weighting.

<u>IAZ = 0</u>

XKK	O	F	A thinning factor used in the statistical loading. Equals the probability of an element occurring at the normalized peak of the chosen weighting function.
WMAX	O	F	Peak of the chosen weighting function used for weight normalization in statistical loading.
MAD1	O	I	Starting address for selecting random numbers from the random number array in the call to Function RRAND (5) ($1 \leq MAD1 \leq 128$).
JRND	O	I	Random number generator initialization constant used in the call to Function RRAND ($0 < JRND \leq 2^{36} - 1$).
UL	O	F	Constant used in Function RRAND to set up uniform random number distribution (UL = 0.0).
UEXT	O	F	Constant used in Function RRAND to set up uniform random number distribution (UEXT = 2.910383046E-11).

3. CALLING SEQUENCE

CALL WEIGHT (RAD,WTRAD,WFUNC)

Where: RAD = Independent variable for the weighting function evaluation.

WTRAD = Radius (half span) of the weighting function.

WFUNC = Returns the value of the weighting function evaluated at RAD.

4. RESTRICTIONS, REQUIREMENTS, MISCELLANEOUS DATA

a. Two common statements are required for the subroutine WEIGHT. These contain the input variables to the subroutine. The statements must be in the form shown

```
COMMON IWTFLG,WTPED,NWTPOW,WKASIR,F(20),B(20),
       ANG,NBAR,BESCAL,CUBK,PII2,BESS1,IAZ,XKK,
       WMAX,BESEDG
```

The labeled common block, BLKRND, is used in the call to the Function RRAND.

All the variable names are the same as those described above in Section 2.

b. The constants, F(20), are used in evaluating the Taylor weighting function. To decrease execution time the constants should be calculated once in the calling program for each weighting design and the values stored in the array F(20). The equations for these constants are the following:

$$
F(m) = \begin{cases} 1 & ,m=0 \\[1em] \dfrac{1}{J_0(\pi\mu_m)} \dfrac{\prod_{n=1}^{NBAR-1}\left\{1 - \dfrac{\mu_m^2}{\sigma^2(A^2+(n+\frac{1}{2})^2)}\right\}}{\prod_{\substack{n=1\\ n\neq m}}^{NBAR-1}\left\{1 - \left(\dfrac{\mu_m}{\mu_n}\right)^2\right\}} & ,m=1,\ldots NBAR \\[1em] 0 & ,m=NBAR+1\ldots \end{cases}
$$

Where: $A = \dfrac{\cosh^{-1}\eta}{\pi}$

$\eta = 10.0^{DB/20}$

$\sigma = \mu_{NBAR}/(A^2+(NBAR-1/2)^2)^{1/2}$

μ_n The zeros of the Bessel function

$J_1(\pi\mu_n) = 0,\ n = 1,2,\ldots$

c. The constants, B(20), are used in evaluating the Bayliss weighting function. To decrease execution time the constants should be calculated once in the calling program for each weighting design and the values stored in the array B(20). The equations for the constants are given below.

$$
B(m) = \frac{-jC2\mu_m^2}{J_1(\mu_m\pi)} \frac{\prod_{n=1}^{NBAR-1}\left\{1 - \left(\dfrac{\mu_m}{\sigma z_n}\right)^2\right\}}{\prod_{\substack{k=0\\ k\neq n}}^{NBAR-1}\left\{1 - \left(\dfrac{\mu_m}{\mu_k}\right)^2\right\}} \quad ,m=0,1,\ldots NBAR-1
$$

$$
= 0 \quad ,m=NBAR,NBAR+1\ldots
$$

Where: μ_m = The zeros of the Bessel function

$J_1'(\mu_m\pi) = 0,\ m = 0,\ 1,\ \ldots$

$$z_n = \begin{cases} 0 & , n = 0 \\ \pm\, \xi_n & , n = 1, 2, 3, 4 \\ \pm\, (A^2+n^2)^{1/2} & , n = 5, 6, \ldots \end{cases}$$

$$\xi_1 = 0.9858302 + 0.0333885 \cdot DB + 0.000140 \cdot DB^2$$
$$-0.0000019 \cdot DB^3 + 0.00000001 \cdot DB^4$$

$$\xi_2 = 2.00337487 + 0.1141548 \cdot DB + 0.0004159 \cdot DB^2$$
$$-\ 0.00000373 \cdot DB^3 + 0.00000001 \cdot DB^4$$

$$\xi_3 = 3.00636321 + 0.00683394 \cdot DB + 0.00029281 \cdot DB^2$$
$$-\ 0.00000161 \cdot DB^3$$

$$\xi_4 = 4.00518423 + 0.00501795 \cdot DB + 0.0021735 \cdot DB^2$$
$$-\ 0.00000088 \cdot DB^3$$

$$A = \frac{\cosh^{-1}\eta}{\pi}$$

$$\eta = 10.0^{DB/20}$$

$$\sigma = \frac{\mu_{nBAR}}{z_{nBAR}}$$

$$1/C = ((p_0\sigma\pi)^2-1)J_1(p_0\sigma\pi)\frac{\prod_{n=1}^{NBAR-1}\left\{1 - \left(\frac{p_0}{z_n}\right)^2\right\}}{\prod_{n=0}^{NBAR-1}\left\{1 - \left(\frac{p_0\sigma}{\mu_n}\right)^2\right\}}$$

$$P_o = 0.4797212 + 0.1456692 \cdot DB - 0.0018739 \cdot DB^2$$

$$+ 0.00000218 \cdot DB^3 - 0.00000001 \cdot DB^4$$

The normalization constant, C, is selected such that the weighting function will produce a peak of unit height in the far-field.

d. The value of CUBK must be normalized to the peak of the cubic weighting. The equation for this is shown below.

```
CUBK=CUBK/ABS(XX*(XX.WTRAD)*(XX+WTRAD))
```

Where: $XX = (WTRAD)^2/(3.0)$

e. References:

Taylor, T. T., 'Design of Circular Apertures for Narrow Beamwidth and Low Sidelobes,' IRE Trans. on Antennas and Propagation, Vol. AP-8, pp. 17-22, (1/60).

Hansen, R. C., 'Tables of Taylor Distributions for Circular Aperture Antennas,' IRE Trans. on Antennas and Propagation, Vol. AP-8, pp. 23-26, (1/60).

Bayliss, E. T., 'Design of Monopulse Antenna Difference Patterns with Low Sidelobes,' Bell Sys. Tech. Journal, Vol. 47, pp. 623-650, (5/68).

Oppenheim, A.V., Schafer, R.W., Digital Signal Processing, Englewood Cliffs, NJ, Prentice-Hall, Inc., 1975, pp. 243-244.

5. SUBPROGRAMS REQUIRED

RRAND
EXPND
BESS
GAM

6. THEORY OF OPERATION

The cosine on a pedestal to a power, Blackman, Kaiser, and Bartlett weighting are described in Oppenheim and Schafer (4.e). Details of the Taylor weighting function may be seen in the articles by Taylor and Hansen (4.e). Details of the Bayliss weighting function may be seen in the article by Bayliss (4.e). The value of the weighting function, WFUNC, for a cosine on a pedestal to a power is described in the equation below.

```
WFUNC=WTPED+(1-WTPED)*(COS(RAD*PI/WTRAD*2)))**NWTPOW
```

For the Blackman window the equation is given below.

```
WFUNC=0.42 - 0.5*COS(ARG)+0.08*COS(ARG+ARG)
```

```
Where:     ARG=((RAD/WTRAD)+1)*PI
```

The equation for the Kaiser window is given below.

```
WFUNC=EXPND(CONK*SQRT(SQN-RAD*RAD))*DENOM
```

```
Where:  DENOM = 1.0/EXPND(WKASIR)
        CONK  = WKASIR/WTRAD
        SQN   = WTRAD*WTRAD
```

The equation for the triangular weighting is given below.

```
WFUNC=1 - RAD/WTRAD
```

The equation for the Bessel weighting is given below.

$$WFUNC=BESCAL*J_o(RAD*BESEDG)$$

The equation for the cubic weighting is given below.

$$WFUNC=CUBK*RAD*(RAD - WTRAD)*(RAD+WTRAD)$$

7. FORTRAN LISTING

```
C
C     ****************************************
C
      SUBROUTINE WEIGHT(RAD,WTRAD,WFUNC)
      COMMON IWTFLG,WTPED,NWTPOW,WKASIR,F(20),B(20),ANG,
&              NBAR,BESCAL,CUBK,PII2,BESS1,IAZ,XKK,WMAX,
&              BESEDG
      COMMON/BLKRND/MAD1,JRND,XMEAN,SIG2SQ,UL,UEXT
      DIMENSION U(20),BZERO1(20)
      DATA U/1.2196699,2.2331306,3.2383154,4.2410628,
&     5.2427643,6.2439216,7.2447598,8.2453948,
&     9.2458927,10.2462933,11.246624,12.246900,
&     13.247131,14.247334,15.247508,16.247663,
&     17.247796,18.247920,19.248027,20.248125/
      DATA BZERO1/0.586067,1.6970509,2.7171939,3.726137,
&     4.7312271,5.7345205,6.7368281,7.7385356,
&     8.7398505,9.7408945,10.7417435,11.7424475,
&     12.7430408,13.7435477,14.7439856,15.7443679,
&     16.7447044,17.745003,18.7452697,19.7455093/
      UL=0.0
      UEXT=2.910383046E-11
      IF(RAD.LE.WTRAD) GO TO 280
      WFUNC=0.0
      GO TO 200
  280 PI=3.1415926
      CON=PI/(WTRAD*2)
      GO TO(210,220,230,240,250,260,270,281),IWTFLG
  210 TEMP=1.0-WTPED
      WFUNC=WTPED+TEMP*COS(RAD*CON)**NWTPOW
      GO TO 200
  220 ARG=(RAD+WTRAD)*2*CON
      WFUNC=0.42-0.5*COS(ARG)+0.08*COS(ARG+ARG)
      GO TO 200
  230 DENOM=1.0/EXPND(WKASIR)
      CONK=WKASIR/WTRAD
      SQN=WTRAD*WTRAD
      WFUNC=EXPND(CONK*SQRT(SQN-RAD*RAD))*DENOM
      GO TO 200
```

```
240  WFUNC=1.0-RAD/WTRAD
     GO TO 200
250  P=PI*RAD/WTRAD
     CSTRT=0.0
     DO 256 I=1,NBAR-1
     CSTRT=CSTRT+(BESS(0.0,U(I)*P)*F(I))
256  CONTINUE
     WFUNC=PII2*(BESS1+CSTRT)
     GO TO 200
260  RAD=RAD*BESEDG
     WFUNC=BESS(0.0,RAD)*BESCAL
     GO TO 200
270  X=CUBK*RAD
     WFUNC=X*(RAD+WTRAD)*(RAD-WTRAD)
     GO TO 200
281  P=PI*RAD/WTRAD
     CSTRT=0.0
     DO 285 I=1,NBAR
     CSTRT=CSTRT+(-B(I-1)*BESS(1.0,BZERO1(I-1)*P))
285  CONTINUE
     WFUNC=COS(ANG)*CSTRT
200  IF(IAZ.EQ.1) GO TO 300
     WFUNC=XKK*WFUNC/WMAX
     WTMP=SIGN(1.0,WFUNC)
     RRND=RRAND(1)
     IF(RRND.GT.ABS(WFUNC)) WTMP=0.0
     WFUNC=WTMP
300  RETURN
     END
```

MISSION
of
Rome Air Development Center

RADC plans and conducts research, exploratory and advanced development programs in command, control, and communications (C^3) activities, and in the C^3 areas of information sciences and intelligence. The principal technical mission areas are communications, electromagnetic guidance and control, surveillance of ground and aerospace objects, intelligence data collection and handling, information system technology, ionospheric propagation, solid state sciences, microwave physics and electronic reliability, maintainability and compatibility.