AD-A060 319 ROME AIR DEVELOPMENT CENTER GRIFFISS AFB N Y F/G 9/5
AN INTRODUCTION TO THE GENERAL ELECTROMAGNETIC MODEL FOR THE AN--ETC(U)
SEP 78 K R SIARKIEWICZ

UNCLASSIFIED RADC-TR-78-181 NL

1 OF 3
ADA
060319

AD A060319

DDC FILE COPY

LEVEL II

RADC-TR-78-181
In-House Report
September 1978

AN INTRODUCTION TO THE GENERAL ELECTROMAGNETIC MODEL FOR THE ANALYSIS OF COMPLEX/SYSTEMS (GEMACS)

Kenneth R. Siarkiewicz

DDC
RECEIVED
OCT 20 1978
F

Approved for public release; distribution unlimited

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

78 10 18 038

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-78-181 has been reviewed and is approved for publication.

APPROVED: Samuel D. Zaccari

SAMUEL D. ZACCARI, Chief
Compatibility Branch
Reliability & Compatibility Division

APPROVED: Joseph J. Naresky

JOSEPH J. NARESKY
Chief, Reliability & Compatibility Division

FOR THE COMMANDER: John P. Huss

JOHN P. HUSS
Acting Chief, Plans Office

If your address has changed or if you wish to be removed from the RADC mailing list, or if the addressee is no longer employed by your organization, please notify RADC (RBCT) Griffiss AFB NY 13441. This will assist us in maintaining a current mailing list.

Do not return this copy. Retain or destroy.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: RADC-TR-78-181	2. GOVT ACCESSION NO.
3. RECIPIENT'S CATALOG NUMBER	
4. TITLE (and Subtitle): AN INTRODUCTION TO THE GENERAL ELECTROMAGNETIC MODEL FOR THE ANALYSIS OF COMPLEX/SYSTEMS (GEMACS)	5. TYPE OF REPORT & PERIOD COVERED: In-House Report, 1 April 77 - 1 June 78
	6. PERFORMING ORG. REPORT NUMBER: N/A
7. AUTHOR(s): Kenneth R. Siarkiewicz	8. CONTRACT OR GRANT NUMBER(s): N/A
9. PERFORMING ORGANIZATION NAME AND ADDRESS: Rome Air Development Center (RBCT), Griffiss AFB NY 13441	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: 62702F, 23380320, 03
11. CONTROLLING OFFICE NAME AND ADDRESS: Same	12. REPORT DATE: September 1978
	13. NUMBER OF PAGES: 286
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office): Same	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

12 286 p.

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

Same 9 Rept. for 1 Apr 77-1 Jun 78,

18. SUPPLEMENTARY NOTES

N/A

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

Electromagnetic Compatibility
Method of Moments
Antenna Analysis
Matrix Equation Solution
GEMACS

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

This report contains a detailed introduction to the General Electromagnetic Model for the Analysis of Complex Systems (GEMACS). It contains a thorough discussion of all of the commands in the GEMACS executive and geometry languages and explains their use by reference to a sample problem that is explained in great detail throughout the document. Suggested uses of GEMACS are provided along with the executive command streams to exercise these capabilities. The document is meant to be the initial exposure to the user of the GEMACS analysis tool. In this it supplements the User's Manual in that the latter is written →

over

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

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

309 050
78 10 18 038

more in the form of a handbook to be used as a reference on the format of the various commands.

TABLE OF CONTENTS

ACCESS
NTIS
DDC
UNANNOUNCED
JUSTIFICATION
BY
DISTRIBUTION/AVAILABILITY CODES
SPECIAL
A

FIGURES

I. INTRODUCTION

The purpose of this document is two-fold. First, it serves as a detailed introduction to the General Electromagnetic Model for the Analysis of Complex Systems (GEMACS). Secondly, its format and presentation of figures are such that it can serve as a teaching guide and source of Vugraph slides. With this in mind a detailed sample problem is presented and continually referred to throughout the report. The geometry of the problem is shown in Figure 1.

It is assumed that this geometry represents the roof of a communications hut. Inside the hut are located a transmitter and a receiver operating at two different frequencies. The concern is centered around the interference in the receiver caused by radiated energy from the transmitter. Two antennas are collocated on the top of the hut and are shown as dots in Figure 1.

It is not the intention of this report to present, derive or justify the basic theory on which GEMACS is based, nor is the structure or make-up of the code itself discussed. The intention is to show how the user interfaces with GEMACS. What are the kinds and quantity of input data; what are the commands and the language of GEMACS? What are the data available from a GEMACS analysis; what form do they take; and, what do the data tell the user?

Section II gives a very brief theoretical background of the method of moments (MOM) pointing out the limitations of many existing codes implementing this formulation. Section III shows how GEMACS reduces the impact of these limitations. This will be followed by a discussion of the language of GEMACS, the use of the language being illustrated by examples taken from

ALL DIMENSIONS IN WAVELENGTHS

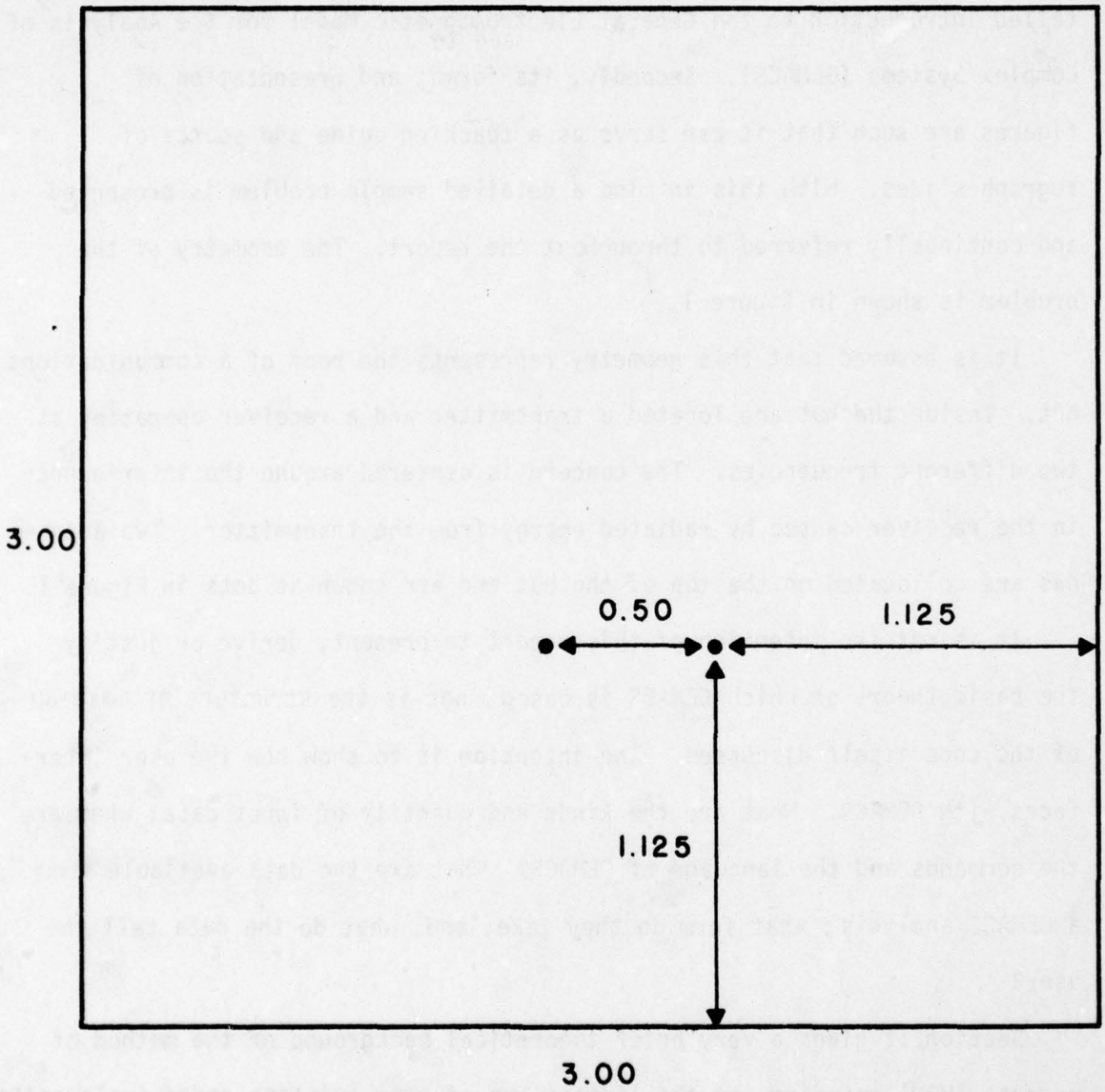


FIGURE 1. SAMPLE PROBLEM

the analysis of the sample problem. Section VI describes the output provided by GEMACS. Section VII describes an alternate solution capability built into GEMACS - a conventional full matrix solution formulation. Sections VIII and IX show how GEMACS is used to analyze scattering problems and how GEMACS may be used to perform parametric studies of a system being designed. The Appendix presents the total computer output from an analysis of the sample problem diagramed in Figure 1.

II. BRIEF THEORETICAL BACKGROUND

GEMACS is based on the Method of Moments (MOM) formulation for the solution of Maxwell's equations. This formulation has been extensively written in readily available open literature. Most of this deals with the more specialized topics. References which give the basic derivation of the MOM algorithms are listed in Section X [1 - 3].

GEMACS is a thin-wire code which means that the geometry is represented by a set of wires whose radii should generally be less than one-tenth of the subsection length. As a rule, subsection lengths should be one-tenth of a wavelength or less at the desired frequency. Somewhat longer subsections may be used on long wires with no abrupt changes while shorter lengths, one-twentieth of a wavelength or less, may be required in modeling the critical regions of an antenna (e.g., the feed region).

However, extremely small subsection lengths and wire radii should be avoided. Approximate minima are 0.001 and 0.00001 wavelength, respectively. These are dependent on the accuracy and word size of the computer.

In addition, the lengths of adjacent subsections should not vary by a ratio greater than two. This is especially important at the junctions of several wires. This is necessary since the code uses an interpolation scheme to represent the current variation over each segment, and rapidly changing lengths for adjacent subsections may result in unrealistic distributions of current.

The radii of adjacent subsections should also be of comparable size. This is in line with the physical geometry as well as allowing for a

smoother interpolation of the boundary conditions, which are applied at the surfaces of the conductors.

Finally, an actual wire in the system should be modeled with its actual radius. This is consistent with the concept of modeling the system in a manner as true-to-life as possible.

If the geometry consists of a surface or a solid sheet of conducting material, then the surface is represented by a wire grid of thin wires. This can be pictured as modeling the surface by window screen. Grid models should use a wire radius of one-fiftieth of the segment length. In addition, grid mesh circumferences should not greatly exceed one-half of a wavelength. Larger circumferences could generate loop resonances which do not exist in the physical system.

In all cases segments with lengths differing by more than a factor of two should not be joined. Angles smaller than about twenty degrees between joined segments should also be avoided. Unjoined segments should be separated by at least the maximum diameter of any of the subsections.

These guidelines are summarized in Figure 2.

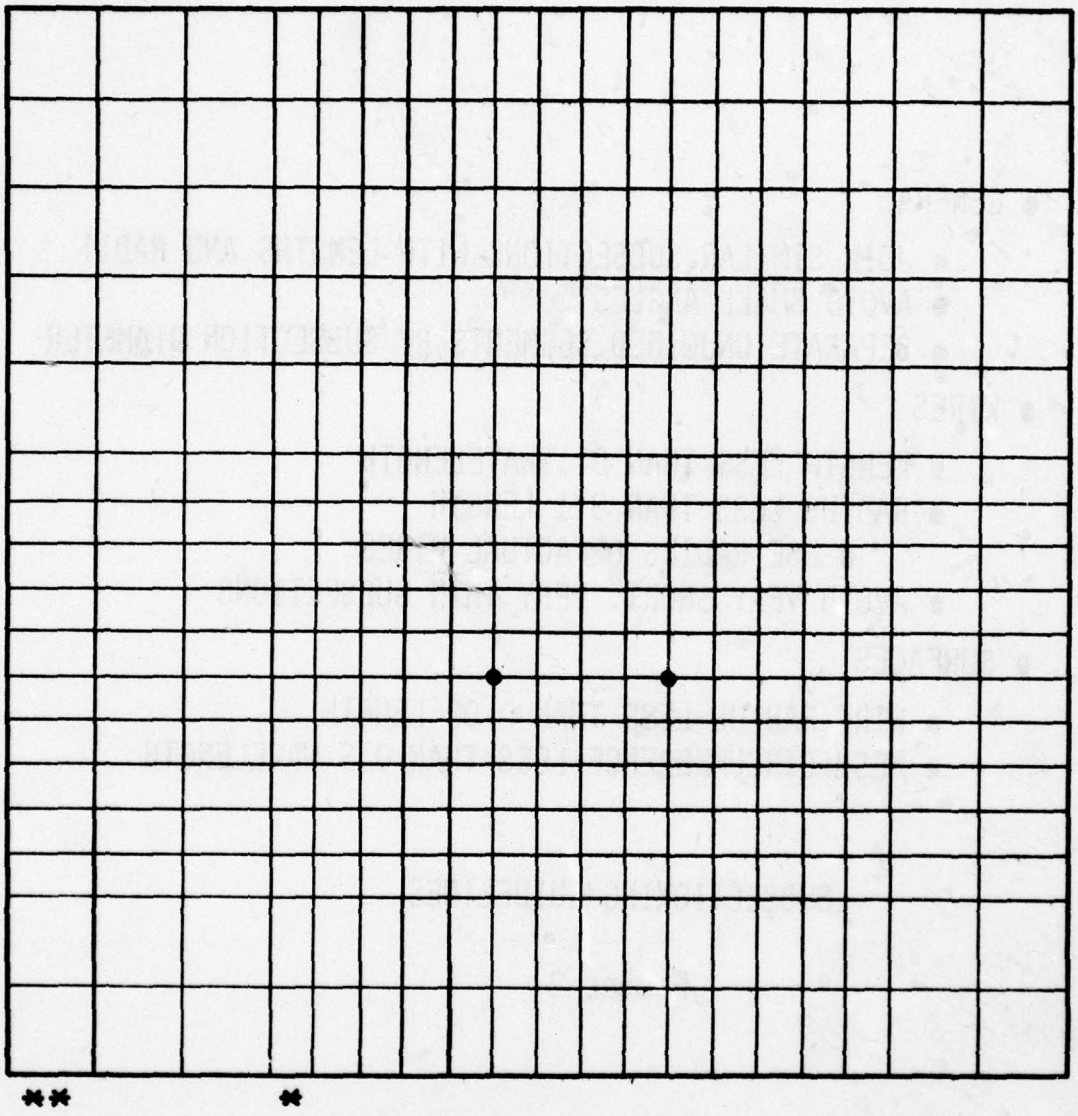
Using the guidelines that have been specified on the problem that has been postulated results in the wire model shown in Figure 3. Each subsection is either a quarter or an eighth of a wavelength in length. In the region around the antennas the length around each loop of the mesh is one-half of a wavelength. The surface is thus made up of 706 subsections. The dots represent the locations of the two antennas. These are subsectioned as shown in Figure 4. It is assumed that one antenna is a quarter-wave monopole at the frequency of interest, and the second is a monopole three-quarters of a wavelength long.

- GENERAL
  - JOIN SIMILAR SUBSECTIONS WITH LENGTHS AND RADII
  - AVOID SMALL ANGLES
  - SEPARATE UNJOINED SEGMENTS BY SUBSECTION DIAMETER
- WIRES
  - LENGTH LESS THAN 0.1 WAVELENGTH
  - RADIUS LESS THAN 0.1 LENGTH
    - USE RADIUS OF ACTUAL WIRES
  - AVOID VERY SHORT, VERY THIN SUBSECTIONS
- SURFACES
  - WIRE RADIUS LESS THAN 0.02 LENGTH
  - MESH CIRCUMFERENCE LESS THAN 0.5 WAVELENGTH

SUBSECTIONING GUIDELINES

FIGURE 2

• INDICATES ANTENNA LOCATIONS



** *

* ONE EIGHTH WAVELENGTH

** ONE QUARTER WAVELENGTH

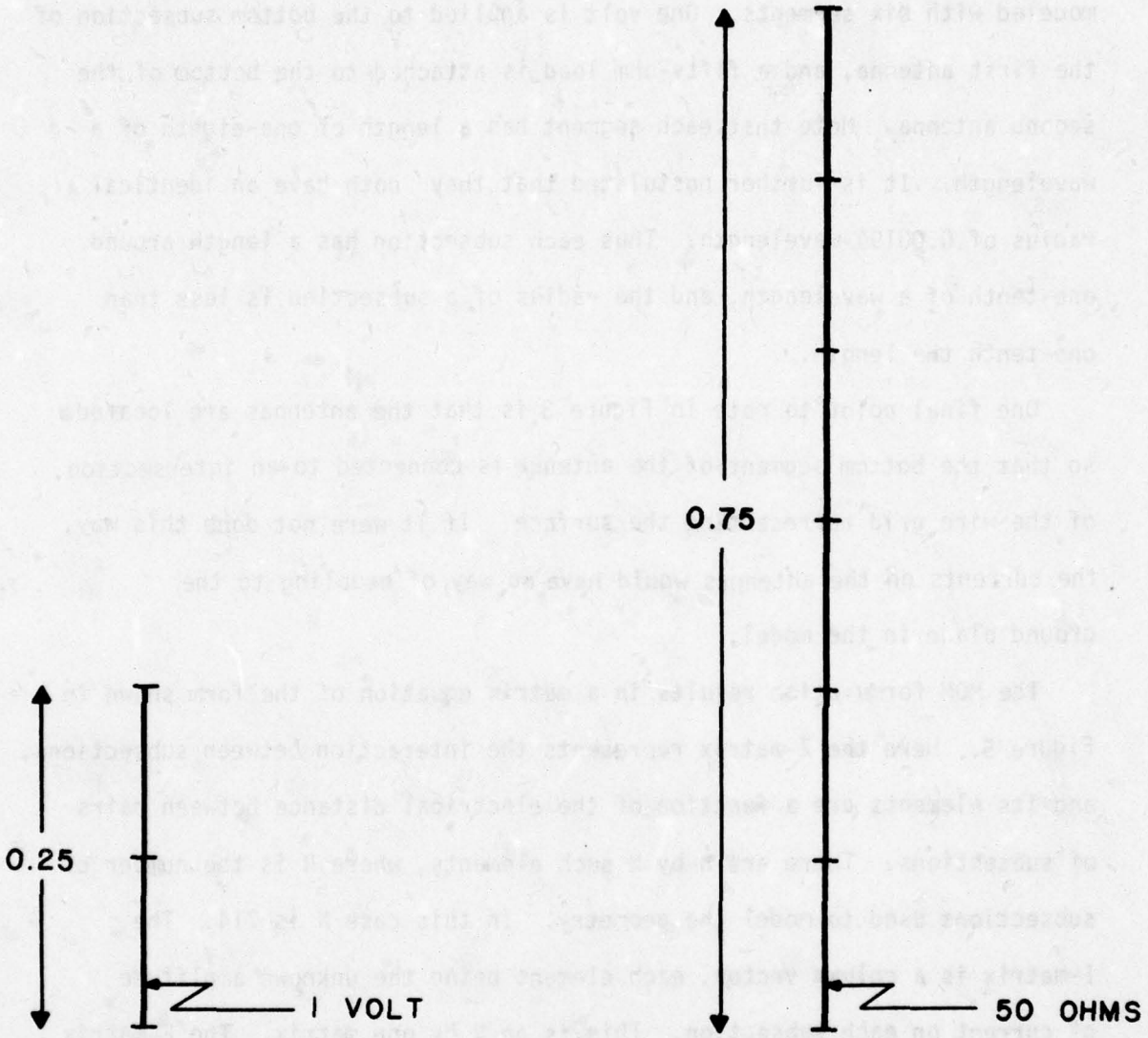
FIGURE 3. PLATE WIRE GRID

The first antenna is modeled with two subsections, and the second is modeled with six segments. One volt is applied to the bottom subsection of the first antenna, and a fifty-ohm load is attached to the bottom of the second antenna. Note that each segment has a length of one-eighth of a wavelength. It is further postulated that they both have an identical radius of 0.00194 wavelength. Thus each subsection has a length around one-tenth of a wavelength, and the radius of a subsection is less than one-tenth the length.

One final point to note in Figure 3 is that the antennas are located so that the bottom segment of the antenna is connected to an intersection of the wire grid representing the surface. If it were not done this way, the currents on the antennas would have no way of coupling to the ground plane in the model.

The MOM formulation results in a matrix equation of the form shown in Figure 5. Here the Z-matrix represents the interaction between subsections, and its elements are a function of the electrical distance between pairs of subsections. There are N by N such elements, where N is the number of subsections used to model the geometry. In this case N is 714. The I-matrix is a column vector, each element being the unknown amplitude of current on each subsection. This is an N by one matrix. The E-matrix is also a column vector, each element of which is the amplitude of the electric field across each subsection. This is also an N by one matrix.

In this equation the elements of the I-matrix are to be calculated knowing the geometry and having specified a particular excitation. This can be done using either a matrix inversion and multiplication process or some form of elimination and iteration scheme. In either case, the



ALL DIMENSIONS IN WAVELENGTHS

FIGURE 4. ANTENNA SUBSECTIONING

$$[Z][I] = [E]$$

Z_{IJ} = ELECTRIC FIELD ON SUBSECTION I DUE TO A CURRENT OF UNIT AMPLITUDE ON SUBSECTION J

I_I = CURRENT ON SUBSECTION I

E_J = FIELD ON SUBSECTION J

MOM MATRIX EQUATION

FIGURE 5

process of filling the interaction matrix and solving for the currents is an expensive one. Computer storage is also a problem, the example requiring approximately one million computer words.

It will be well at this point to list the limitations and assumptions that will be built into a GEMACS analysis. These are shown in Figure 6. As GEMACS now exists the geometry is modeled as a wire screen approximation to the actual surface. The currents in the model exist only on the axis of the wires in the grid, whereas physically they are spread over the entire area of the surface.

It follows then that since the current exists only on the axis of the wire, there is no azimuthal variation of the current around the circumference of the wire, as would exist physically on the antennas. As an example of this point, consider two wires located near each other such that the separation between them is less than a wavelength. Depending on whether the currents in the wires flow in the same direction or in opposite directions, there will be a spreading or a concentration of longitudinal current on the surface of the wires between the wires, with the opposite phenomenon occuring on the opposite side of the respective wires. Thus, there would exist an azimuthal variation of the longitudinal current around the wires.

Also because of the assumed concentration of the current on the axis of the wires there is no radial component of current flow within the wire. This has significance in the generation of near-field phenomena at the ends of the wire.

Antenna sources are generally modeled as a delta voltage source placed across a subsection. This may have no counterpart whatsoever in the physical situation. Moreover, the size of the gap in the model usually does not

ACTUAL GEOMETRY REPRESENTED BY THIN WIRES
CURRENT EXISTS ONLY ON WIRE AXIS
NO AZIMUTHAL VARIATION OF CURRENT
NO RADIAL COMPONENT OF CURRENT
ANTENNA SOURCES ARE GAP MODELS
CURRENTS DETERMINED ONLY AT POINTS

SYSTEM LIMITED IN SIZE
PERFECT CONDUCTORS
EXTERNAL PROBLEMS

MOM ASSUMPTIONS & LIMITATIONS

FIGURE 6

bear any relation to the size of the gap in the physical antenna. The gap in the model is usually the same size as the length of the adjacent subsections, since one of the modeling rules of thumb is to avoid large ratios in the relative lengths of adjacent subsections.

The current on a subsection is computed at the center of the segment, and the variation over the wire is determined by interpolation between adjacent centers, as discussed previously.

Even though all these assumptions are built into a GEMACS analysis, or a MOM analysis in general, good correlation exists between measured data and predicted data, and between other analytical results and the data obtained by using GEMACS [4 - 8].

MOM analyses have been limited to small systems, those that can be represented by 300 subsections or less. Electrically, this roughly corresponds to a size of 30 wavelengths of wires or a surface with an area of one square wavelength. This has not been due to a limitation of the theory or the technique, but has been brought about by the computer resources needed to perform a MOM analysis. The computer core storage goes up as N^2; and the solution time increases as N^3, where N is again the number of subsections.

All the commonly used codes assume that the material of which the system is composed is perfectly conducting. Brute-force techniques can be used to get around this limitation, but they put a heavy burden on the user. Alternate solutions are being pursued at RADC and will be included in future versions of GEMACS.

Finally, the MOM wire grid model can be used only to solve the external problem. Problems that cannot be treated with confidence

include coupling through apertures in the skin of the structure and coupling between antennas located on opposite sides of the structure. Since the structure is modeled by a wire grid, electromagnetic energy will "leak through" the mesh in the model, thus resulting in a form of aperture coupling. For the antenna coupling problem, energy will go directly through the body in addition to going around on the surface. Thus, the coupling will be greater than if the surface were modeled as a solid. A solid surface model will be shortly implemented into the GEMACS code to eliminate these present limitations.

With this brief background it is now time to examine, again briefly, how GEMACS attempts to relieve the problem of large computer resources being required to perform an analysis on a large problem.

III. GEMACS

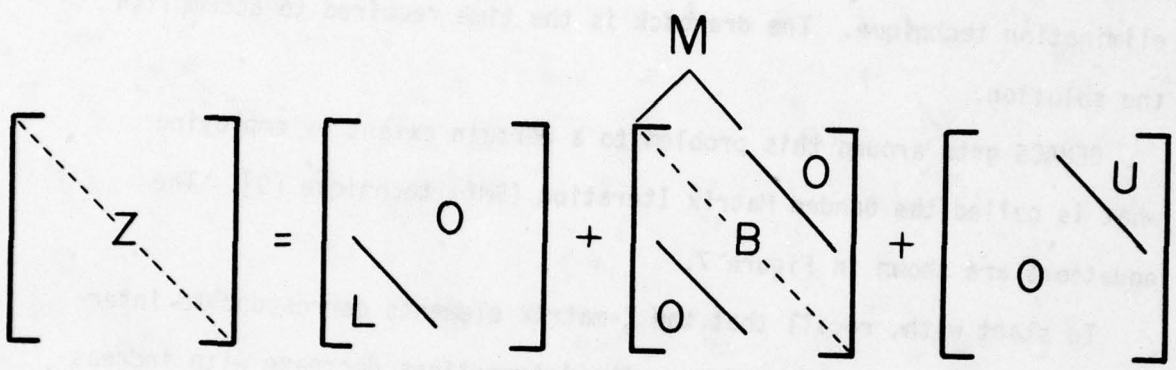
As previously mentioned, the MOM formulation results in a matrix equation which is solved to obtain the current distribution on the structure. The solution is usually obtained by matrix inversion or some elimination technique. The drawback is the time required to accomplish the solution.

GEMACS gets around this problem to a certain extent by employing what is called the Banded Matrix Iteration (BMI) technique [9]. The equations are shown in Figure 7.

To start with, recall that the Z-matrix elements correspond to interactions between wire subsections. The interactions decrease with increasing distance between the subsections. As will be shown later, the segments can be numbered such that the difference between subsection numbers for all close-neighboring segment pairs is small compared to N. The largest matrix elements can then be kept close to the principal diagonal of the interaction matrix, shown dotted in Figure 7.

The first equation on Figure 7 is the basic matrix equation to be solved, relating the field on the subsections to the currents on the subsections by the interaction matrix. The interaction matrix is then separated into three matrices as shown in the second equation in the figure. Here B is called a banded matrix with lower and upper bandwidths M. That is, M is the number of minor diagonals on either side of the main diagonal. L and U contain the remaining elements of the interaction matrix and are called the lower and upper matrices, respectively. They are both triangular matrices.

$$E = ZI$$



$$= L + B + U$$

$$BI = E - (L + U)\,I$$

BMI EQUATION DERIVATION

FIGURE 7.

These two equations can be combined, after some minor manipulation, to result in the third equation shown in Figure 7. An iterative scheme is shown in the first equation in Figure 8, where I_j denotes an approximation to the solution at the jth iteration; and I_{j+1} is an approximation to the solution at the following iteration. Some starting value I_0 (usually zero) is chosen, and I_1 is computed from this equation. Then I_1 is entered on the right-hand side, and I_2 is computed. If the sequence converges, then an approximate solution to the original matrix equation is obtained.

This equation must be solved at each iteration. The cost is minimized by decomposing B into a product of lower and upper triangular matrices. Then the first equation in Figure 8 is solved by forward elimination, as shown in the second equation in the figure. Finally, this is followed by a back substitution, as shown in the third equation in Figure 8.

Note that the decomposition of the banded matrix is similar to the full matrix decomposition, except that the cost is much less since this is a matrix in which many of the elements are zero. The cost for the iterative solution process is shown in the last equation in Figure 8, where K is the number of iterations, N is the rank of the interaction matrix and M is the bandwidth chosen. Note that the best general method for the solution of the matrix equation requires $N^3/3$ operations. In GEMACS the user must make a trade-off between the number of iterations and the bandwidth. The larger the bandwidth, the smaller the number of iterations and vice versa. No rules of thumb exist for the general case to optimize the cost of solving the matrix equation. However, it is apparent that one should choose a large bandwidth. This is so since a larger bandwidth provides lower initial errors and faster convergence rates. If the user picks a

$$B\, I_{j+1} = E-(L+U)\, I_j$$

$$B_L Y_j = E-(L+U)\, I_j$$

$$B_u I_{j+1} = Y_j$$

$$\text{cost} = NM^2-2M^3/3+KN^2$$

ITERATION SOLUTION EQUATIONS

FIGURE 8.

bandwidth too small, slow convergence and low efficiency with respect to the Gauss-Jordan elimination technique will result. Restarting with a larger bandwidth might provide a higher overall efficiency, even with the cost of the aborted run included. This topic is discussed in the BDM interim reports [4-8], especially RADC-TR-75-272 [6].

So much for what the problem is, and how and why GEMACS solves the problem. What is GEMACS, at least in its overall concepts? Basically, GEMACS is a mainframe data handler, whose main purpose is to store data and present data in a place where they are needed and in a format required by some operational routine. In this respect it can be used in many problems that result in large amounts of data, such data being used many times over.

A functional breakdown of the GEMACS structure is shown in Figure 9. The GEMACS executive routines control the interface of the code with the host computer and perform three basic functions: input/output to peripheral files, taking checkpoints and restarting from these checkpoints; and the compilation of statistical information, which can be used to pinpoint areas for further code refinement.

The input language, task execution and run termination processors simply read the user's data deck, call appropriate subprocessors based on the user's commands and terminate the analysis, respectively. Within these two upper levels in Figure 9 are all of the file handling capabilities built into GEMACS. With a proper interface for the new subprocessors under the task execution processor, these new subprocessors can change the field analysis technique or even apply the mainframe to a different type of problem completely, such as the dynamic load analysis of some structure.

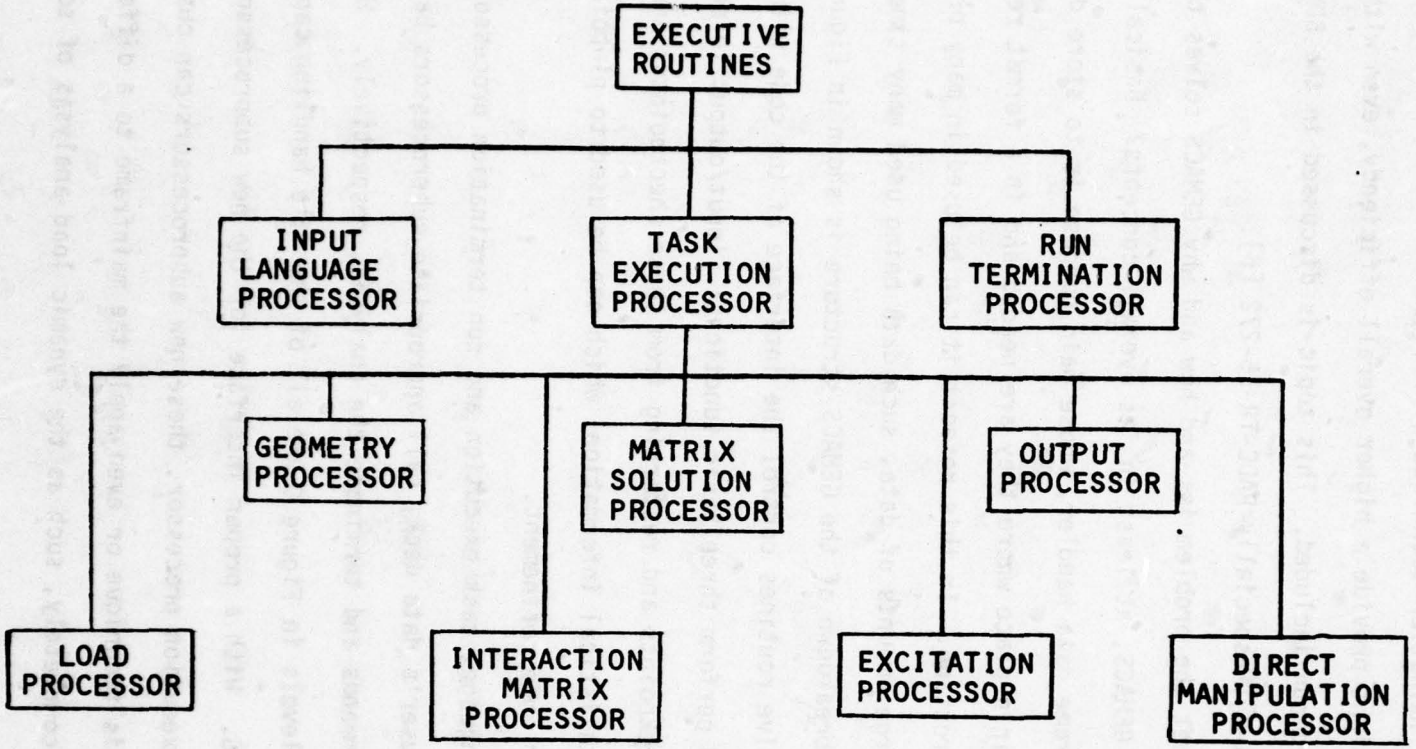


FIGURE 9. GEMACS STRUCTURE

The geometry processor generates the geometry to be analyzed by interpreting the user geometry input commands. The interaction matrix processor generates the elements of the interaction matrix for the frequency and geometry specified by the user. The excitation processor generates the elements of the vector on the right-hand side of the MOM matrix equation. The load processor modifies the interaction matrix to take into account the presence of loads on the wires or the presence of imperfectly conducting materials. The matrix solution processor solves the MOM matrix equation for the currents on the structure. The output processor calculates such quantities as the near- and far-field patterns and terminal impedance for antennas or the backscattering from the structure. The direct manipulation processor sets such variables as the maximum CPU time allowed for the analysis, the number of files in the system available to the code, the frequency of the analysis and the electrical characterisitics of the ground (if present). It also performs arithmetic operations, such as modifying the frequency by some factor, a feature which is useful when a loop is being built into the command stream. This is illustrated by examples in Section IX.

What is the impact of all this to the user? This modular construction has several important advantages that he should be aware of at the start. First, and most obvious, is that one can plug in any technique which has a proper interface or driver to transfer data between the implementing subroutine and the mainframe. This is much like the use of magnetic cards for programs on personal calculators. It is therefore possible to have a complete set of techniques stored in separate files. But the

big difference is that communication with all these techniques is in one common language. There is no need to remember several different sets of input formats, or limit oneself to one specialized code.

A wealth of debug features are built into the code to aid in the execution of an analysis or to aid in the integration of new analysis tools. An explanation of the debug messages accompanies the code documentation supplied with GEMACS.

GEMACS has a checkpoint/restart capability within the code. This means that long analyses can be stopped and started in case the computer should go down or in case there is a need to change the command stream, as, for example, in the case in which the bandwidth of the banded matrix needs to be changed to decrease solution time or the number of iterations. This feature of restarting the same analysis several times can be used to schedule the analysis such that it is accomplished at hours in which the computer load is a minimum.

Finally, and highly significant, is the fact that GEMACS is tied into the Air Force Intrasystem Analysis Program (IAP). This means that it will have the full support of the Air Force, to provide aid in the loading, use and maintenance of the code, additions to the capability of the code, updates to eliminate any bugs that may be in the code; and a common language among all users of the code.

This then has been a very brief introduction to the problem and a technique to solve this problem. A quick run-down on the structure of GEMACS and the advantages of its use to the user have also been given. It is now time to get acquainted with the language of GEMACS and to see how the example problem would be solved using GEMACS.

IV GEMACS GEOMETRY LANGUAGE

The function of the geometry processor shown in the block diagram of Figure 9 is to translate the user's input regarding the geometry into a data set which may then be operated on by the interaction matrix generator to generate the interaction matrix for the structure under analysis. The geometry processor is entered when the GMDATA command is encountered in the command language stream (see Section V.C).

The basic elements for GEMACS are points and line segments. These in turn may belong to larger groups with a given name, such as WING or PLATE. Any reference to this given name will also reference all the points and segments within that group. In addition, line segments may also be identified as a group by having the same tag number. Thus, all the segments belonging to the wing may have a tag number "six" and may be referenced as a group by specifying a tag of six instead of using the label WING. Another use, as shown later, would be to give all segments loaded by the same impedance an identical tag number. This will obviate the need for multiple load commands. The same is true for excited segments.

The subsection may thus be identified by either a segment number or a tag number. The difference is that the first is unique in the model while the second may be shared by any number (or all) of the subsections within the geometry model.

The following basic conventions must be observed when the geometry deck is being built. The first nonblank item (not necessarily the first two columns) of the card must be a two-letter alpha code. Then the data following this type code must be separated into items by either a comma

or a blank. Blanks within items are not allowed, as the data would then be interpreted as two separate pieces of information when in fact only one was meant.

If a dollar sign ($) is encountered on a card, the data following are treated as a comment and are ignored for further processing. Also, keep in mind that should a comment be used on a card, a blank must separate the dollar sign from the last piece of geometric input. A dollar sign as the first nonblank character on a card will cause the entire card to be treated as a comment card, and the data on that card will be ignored for further processing. Comments are useful when a second party is using the deck. This allows easy interpretation of what is being done with each command.

Data for the same geometry command may appear on more than one card. This is accomplished through the use of continuation cards. A continuation card is indicated by the presence of an asterisk (*) in column one of each succeeding card. There is no limit to the number of continuation cards that may be used, providing that no more than 256 pieces of data are associated with any single command. About the only command that this will occur in, as will be seen, is the renumber (RN) command (IV.E).

The general form of the commands is shown in Figure 10. In this figure TYPE is one of the mnemonic codes in Figure 11; and P1, P2, etc. are the ordered parameter items required for the processing of the command. As shown in the figure they are separated by commas or by blanks. Note that the order of the appearance of the data items is fixed, and any variation

TYPE P1, P2, P3, P4, P5

—OR—

TYPE P1 P2 P3 P4 P5

FIGURE 10. GENERAL FORMS FOR GEOMETRY COMMAND

1. GENERAL COMMANDS

CS	RA
END	SC

2. SUBSECTION GENERATION

CP	PT
MP	WR

3. ELEMENT GENERATION

AT	DE
CE	DF

4. MACROGENERATORS

RF
RT
XL

5. RENUMBER

RN

FIGURE 11. GEOMETRY GENERATION GROUPING

will cause an abort of the analysis or an incorrect representation of the geometry.

Finally, some of the commands have items which may be defaulted. These are pointed out in the discussion, and their existence is indicated by the use of brackets [ ] in the figures. In general, for a command of the form shown in Figure 10 in which P3, P4 and P5 may be defaulted, it must be kept in mind that the defaults can be achieved only from right to left. For example, to default P5, both P3 and P4 must be specified. To default P4, only P3 needs to be specified, while P5 must assume its default value. When one or more parameters are to be defaulted, then all items to their right must also be defaulted. As a final example assume that it is desired to default P3. Then both P4 and P5 must also be defaulted. If P4 must be specified, then a value for P3 must be also input. If a value for P5 must be given, then both P3 and P4 must be input.

The geometry commands may be arbitrarily grouped into five categories as shown in Figure 11. The general commands essentially set what may be called the global parameters of the problem. They are defined in the beginning of the geometry deck and referenced thereafter. The only exception is the END card, which is the last card of the deck and indicates to the geometry processor that it now has processed all of the input cards relating to the geometry.

The basic units of geometry are points and subsections and these are defined by the commands of group two. Large units can be built from these

points and lines by defining and combining elements using the commands of group three. These units may be generated quite easily by the use of the reflect, rotate and translate commands of group four.

The renumber command can be used to resequence the numbering of the subsections. Thus, the geometry may be entered in any convenient manner and then very simply be resequenced in order to rearrange the interaction matrix in preparation for use of the BMI solution technique. Examples of this will be shown during the discussion of the renumber command.

A. General Commands

The four commands within this group perform the following functions:

1. Define all coordinate systems (CS)
2. Set up a radius table for all subsections (RA)
3. Scale the geometry data to meters (SC)
4. Define the end of the data deck (END).

The first two commands will always be located at the beginning of the geometry data deck. The SCALE command appears before a deck of cards with consistent units which are not meters. If a deck of cards then follows with a different set of units, this latter deck must be preceded by an appropriate SCALE command. The END command is the last card of the geometry deck at all times.

Figure 12 again shows the wire grid model of the top of the hut. Also shown outlined is an area of finer mesh which will be defined in a separate coordinate system. Figure 13 shows the general format for the Coordinate System (CS) specification command, as well as specific examples suitable for the coupling problem. The mnemonic CS identifies this

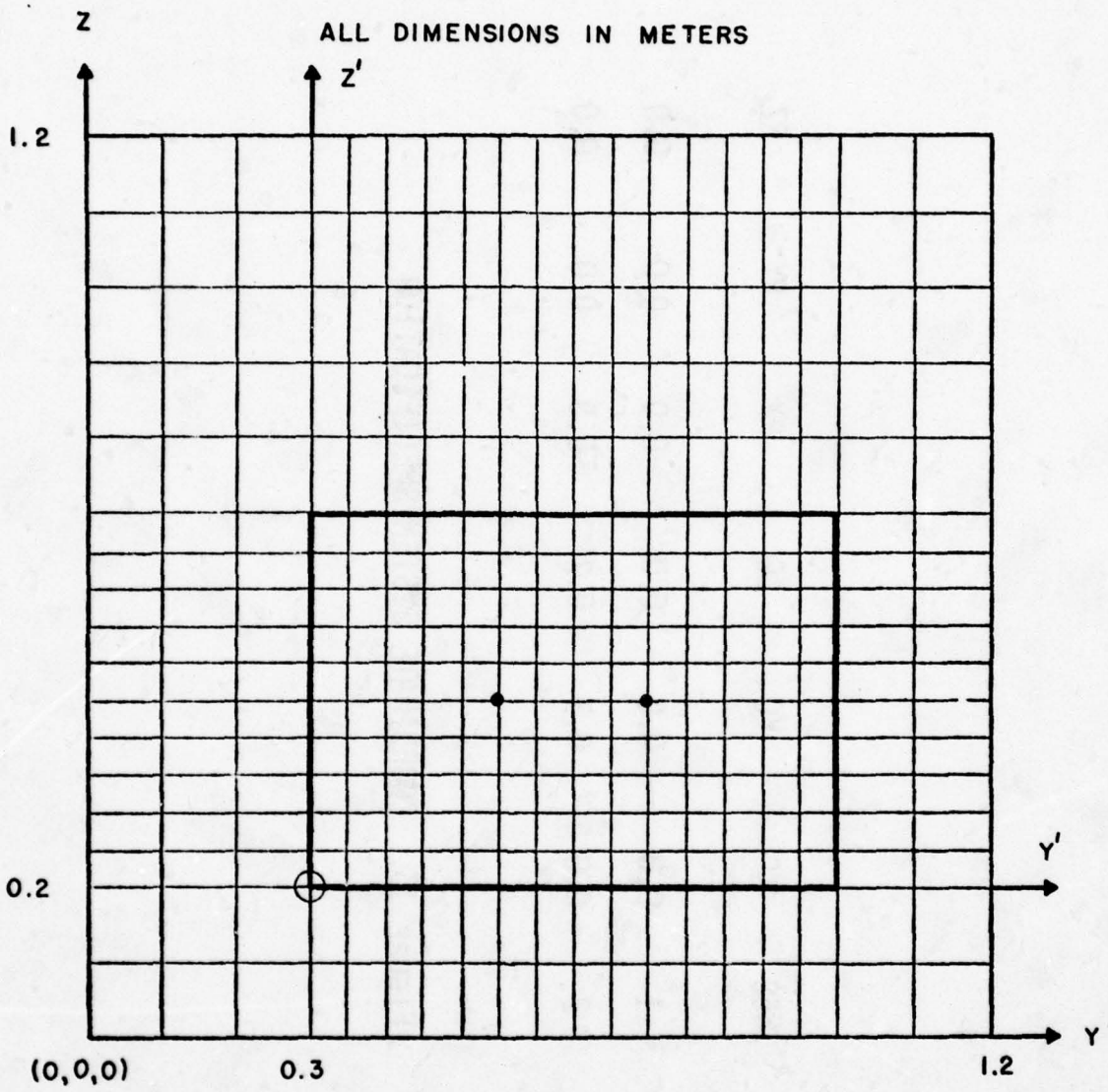


FIGURE 12. COORDINATE SYSTEM DEFINITIONS

CS	NCS	XC	YC	ZC	RX	RY	RZ
CS	1	0.0	0.0	0.0	0.0	0.0	0.0
CS	2	0.0	0.3	0.2	0.0	0.0	0.0

FIGURE 13. COORDINATE SYSTEM SPECIFICATION

as a coordinate system specification command. The parameter NCS is a unique integer identifier of the coordinate system being defined. In this example the value "two" has been chosen for the fine mesh area. The parameters XC, YC and ZC are the (x, y, z) coordinates of the global coordinate system at which the origin of this local coordinate system is to be located. As seen from figure 12 XC=0.0, YC=0.3 and ZC=0.2, where the units are in meters. The final set of parameters RX, RY and RZ are the rotation angles in degrees about the global coordinate system X, Y, and Z axes. An example of this can be seen on page 61 of the user's manual [10] for GEMACS. Note that for this command none of these parameters may be defaulted.

It should be noted that when more than one coordinate system is present, even the global system should be given an integer identifier. This is due to the fact that, as will be seen, the coordinate system parameter on all other commands can be defaulted, but the default will not be to the global system. It will be to the last previous coordinate system referenced. Therefore, the first example in Figure 13 identifies the global coordinate system as system number one. All the parameters with the exception of NCS are zero.

The only time the global coordinate system is not labeled is when all of the geometry will be defined in a single coordinate system.

The Radius (RA) specification command sets up an internal table which contains the radius in meters of each of the subsections used in the analysis. This means that the user needs to specify each different radius only once and then references these radii by their position in the radius

command card. Up to ten different radii may be specified.

Figure 14 shows the general form of the command. The letters RA identify this as the radius specification command. The symbols R1 through Rn represent the different radii that will be used in defining the geometry. For the example only one radius is being used and that is equal to 0.000776 meters. Examples of references to the radius table will be shown during the discussion of other commands. Note that there are no default parameters for this command.

The Scale (SC) specification command allows the user to input the geometry data in any convenient set of units and then modify these data by a scale factor to convert the units of length to meters. If all data are input in meters, then this command is not needed at all. Conversely, if this command is not present, GEMACS will assume that the input data are in meters.

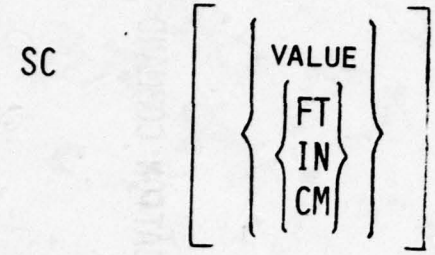
The general format for this command and a specific example are shown in Figure 15. Note that the example is fictitious and not related to the problem being used as an example. The mnemonic SC identifies the command type. The parameter may be defaulted (with a default of one). This would indicate that the following geometry data are in meters.

When this command is used, a double option is available. First, a conversion factor, that is, a scale value, may be specified which would convert the input units to meters. Second, for the convenience of the user conversion factors for units of feet (FT), inches (IN) and centimeters (CM) are provided within GEMACS. Then it is necessary to input only the proper two letter code applicable to the units being used.

RA R1 R2 R3 ··· R_N

RA 0.000776

FIGURE 14. RADIUS SPECIFICATION COMMAND



SC CM

[FINE MESH]

(FICTITIOUS)

SC FT

[COARSE MESH]

FIGURE 15. SCALE SPECIFICATION COMMAND

The example shown in Figure 15 demonstrates the use of this command. Assume that the fine mesh of Figure 12 is dimensioned in centimeters, while the coarse mesh has units of feet. (Keep in mind that this is not the case for this problem. All dimensions are in meters in the problem.) Then, the card "SC CM" would be placed before the geometry input relating to the fine mesh part of the structure. Later, when the coarse mesh model is being input, the card "SC FT" would appear just before the data generating this coarse-mesh wire grid.

As an example of using the default option suppose that the fine mesh data were in units of centimeters and the coarse mesh data were in units of meters. The first scale card would read "SC CM" while the second would just read "SC". The second card would be needed, since, if it were not present, then GEMACS would consider the coarse mesh data to be in centimeters also.

The final general card is the last card to appear in the geometry data input deck. This is the END command. It signals the geometry processor to stop reading input since no more geometry commands exist.

There is one further point to note regarding this card. The user does not need the dollar sign to delineate the presence of a comment in the command. Figure 16 shows the general format of the command as well as a specific example showing how the end of the geometry input data is typically identified.

B. Subsection Generation

One of the key points in the MOM technique is the interaction between subsections or segments. In GEMACS the physical structure is divided

```
END

END OF GEOMETRY HUTTOP
```

FIGURE 16. END OF GEOMETRY COMMAND

up into a number of these subsections. There are two basic ways of generating these subsections. One way is by a combination of point generation (PT), connect point (CP) and multiple point connection (MP) commands. The second way is to generate subsections individually or as wires with a specified number of equal-length subsections using the wire input (WR) command. Each of these will be discussed in turn, using examples that could appear in the input deck for the example problem.

The Point (PT) specification command is probably the most fundamental geometry card, although it is seldom used since the other commands are more efficient for generating the geometry. The general form is shown in Figure 17. The code PT identifies the type of the command. The parameter NPT is an integer identifier of the point. Note that this must be an unique identifier in the global sense. That is, when a point is given a number, no other point, even if it is in a different coordinate system, can have that same identifier. The reason is that GEMACS eventually translates all points into the global coordinate system. The parameters X,Y,Z locate the point within the coordinate system specified by the parameters NCS.

Care must be exercised when using the default option if more than one coordinate system is specified. As pointed out previously, GEMACS will default to the last coordinate system specified on any command and not to the global coordinate system. This further implies that the first command in a sequence that references a particular coordinate system must not have a default for the NCS field. All subsequent commands referring to

PT	NPT	X	Y	Z	[NCS]	
PT	1	0.0	0.0	0.0	1	(FICTITIOUS)
PT	2	0.0	0.0	0.0	2	(FICTITIOUS)

FIGURE 17. POINT SPECIFICATION COMMAND

this same coordinate system may however exercise the default option until a point in some other system is generated. This new coordinate system must be identified in this latter command.

Two examples are shown in Figure 17, each of which specifies a point at the origin of each coordinate system defined in Figure 13. Again, these examples are labeled as fictitious since in the sample input deck the wire specification command is used rather than the point command.

All the intersections of the wire grid model shown in Figure 12 could be specified by points using the point specification command. The subsections joining these points would then be generated using either a Connect Points (CP) command or a multiple point connection (MP) command. The connect points command is shown in Figure 18 along with a fictitious example. It is used to generate a specified number of subsections with a specified tag identifier and radius between two previously defined points. Note that these points need not be located in the same coordinate system.

The mnemonic CP identifies this as the connect points command. The parameters N1 and N2 are the integer identifiers of the first and second points, respectively. Note that the wire connecting these points is a directed line segment with a reference direction going from point 1 to point 2. The positive and negative signs on the computed currents (pages A-42 to A-47) indicate whether the current is flowing in the direction of the line segment or against it, respectively.

The last three parameters may be defaulted. Here again they will each default to the last corresponding value specified on any previous command. The value of the parameter NSEG is the number of subsections

CP	N1	N2	[NSEG]	[NTAG]	[NRAD]
RA	0.001	0.05	0.015	0.0001	
PT	1	0.0	0.0	1.0	
PT	2	0.0	0.0	0.0	
CP	1 2	2	1	3	

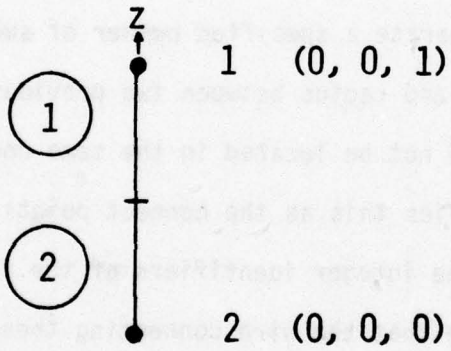


FIGURE 18. CONNECT POINTS COMMAND

or segments into which the wire connecting N1 and N2 is to be divided. NTAG would be the tag identifier of each of the subsections on the wire. The value of NRAD is the location number of the appropriate radius for this wire in the list of radii given in the radius (RA) command.

The example shown in Figure 18 illustrates what could be the first four cards of a geometry input deck. This is again a fictitious example with respect to the sample problem. The radius table contains four entries. The second and third cards would define the two points that are to be connected. The fourth card is the connect points command. It specifies that points one and two are to be connected by a wire that is to be subdivided into two subsections, each subsection having a tag identifier of one. The radius of the wire is found as the third entry in the list given in the radius (RA) command, or fifteen-thousandths of a meter. The result of this operation is shown in the diagram in Figure 18.

The figure also illustrates how GEMACS will number the subsections that are generated by the various geometry commands. As pointed out previously, the line is directed from point N1 to point N2 by the order of the points given in the connect points command. Therefore, the two subsections that are generated will be numbered in sequence along the wire from one to two. That is, segment one will extend from point one to the midpoint of the wire. Then segment two will extend from the midpoint of the wire to point two. These segment numbers are shown in the circles in the diagram in Figure 18.

A generalization of the connect points operation is to connect a series of points. This is accomplished by the Multiple Point (MP) connection command shown in Figure 19. In this case the points to be connected are joined by a single unbroken line, but the line may have any number of bends. The command requires, however, that each wire between each pair of points be subdivided into an equal number of subsections. That is, if three points are to be connected by two wires, the wire between pair one to have three segments and the wire between pair two to have four segments, then two separate commands would be needed since one of the parameters in this command is a single entry for the number of subsections into which each wire is divided.

Looking at the command in detail, it is seen that MP is the two-letter mnemonic for this operation. The value for NPTS is the number of points that are to be connected. In the command there must be present this number of point integer identifiers. For example, if points one through four are to be connected, then NPTS would be four; and this would be followed by the integer identifiers of each of those four points, given in the order in which the points are to be connected together. As explained previously and will be shown in an example, the order in which the pairs of points are connected will determine the numbering of the subsections.

The values for NP1 through NPNPTS would be the integer identifiers of the points that are to be connected together during the operation initiated by this command.

MP	NPTS	NP1	NP2	···	NPNPTS	[NSEG]	[NTAG]	[NRAD]
MP	5	1	2	3	4	6	1	1	1

(FICTITIOUS)

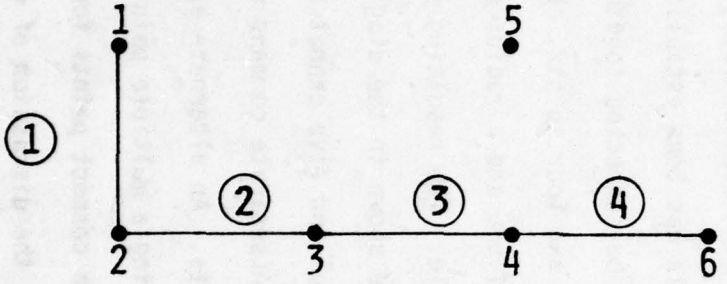


FIGURE 19. MULTIPLE POINTS CONNECTION COMMAND

The final three parameters in this command are the number of subsections between each pair of points, the tag identifier associated with each subsection and the radius of the wires connecting the points. Note, again, that the wires between each pair of points will be divided into identical numbers of subsections; and each subsection will have the same tag identifier and the same radius.

As in the connect points command each of the last three parameters may be defaulted. The default value would be the last previous entry of the respective parameter on any previous command.

Figure 19 shows an example of the use of this command. Assume that six points have already been defined using the point specification command and that the radius table has been established. The example command states that five points are to be connected together in the order one to two, two to three, three to four and four to six. Each connecting wire will have one subsection, a tag of one and a radius equal to the value of the first entry in the radius table. The resulting subsections and their subsection numbers (in circles) are shown in the diagram in Figure 19.

Note that points four and five cannot be connected together using this command as it stands. A separate command would be needed to generate a wire between these points. An alternate approach would be to connect points one to five together using a multiple points connect command, followed by a separate command to connect points four and six.

This then completes the discussion of one way of generating wires and subsections within GEMACS. It is a two-step process, first defining

the points and then connecting the points. It is not the recommended procedure, and it should be used only sparingly. What is perhaps the most fundamental command contained in the GEMACS geometry language will now be described.

The format for the Wire (WR) input command is shown in Figure 20. The use of this command is the preferred method to input wire segment data into GEMACS. Study of the input data deck for the solution of the sample problem (pages A-12 to A-13) will show that it is the only method used to initially define wire subsections.

The mnemonic WR signifies that this is the wire input command. The values of X1, Y1 and Z1 are the coordinates of the first end of the wire. This is what has been called point 1 in the connect points (CP) command. The values of X2, Y2 and Z2 are the coordinates of the second end of the wire, or point 2.

The next three parameters again specify the number of equal length subsections into which the wire will be divided, the tag identifier of each of those subsections and the location in the radius table of the radius of each of the subsections. As before, each of these parameters may be defaulted, the default being to the last previous entry of the respective parameter on any earlier command.

The final parameter in this command is the specification of the coordinate system in which the points and the wire exist. In this case both of the points must lie in the same coordinate system. It should be noted that this is not necessary when the connect points (CP) command

WR	X1	Y1	Z1	X2	Y2	Z2	[NSEG]	[NTAG]	[NRAD]	[NCS]
WR	0.0	0.0	0.05	0.0	0.0	0.0	1	2	1	2
WR	0.0	0.0	0.0	0.0	0.05	0.0				
WR	0.0	0.0	0.1	0.0	0.0	0.0	1	1	1	1
WR	0.0	0.0	0.0	0.0	0.1	0.0				

FIGURE 20. WIRE INPUT COMMAND

is used. The difference is that the latter command references point numbers (which are globally unique). In the wire input command only the (x,y,z) coordinates of the endpoints of the wire are specified.

The coordinate system specification may be defaulted. Here again, as previously pointed out, care must be exercised when more than one coordinate system is used in an analysis. As usual, the default will be to the last coordinate system specified on any previous command.

Two examples are also shown in Figure 20. Each example consists of two cards. Each pair of cards will eventually be used to develop the fine-mesh grid and the coarse-mesh grid, respectively. The results are shown as the dark lines in Figure 21.

A note of explanation regarding Figure 21 is appropriate at this point. Recall from Figure 12 that the coarse-mesh model is defined in coordinate system number one. The coordinate designations at the left and at the bottom of figure 21 refer to this coordinate system. The fine-mesh model is defined in coordinate system number two. The coordinates on the top and right in figure 21 refer to this coordinate system.

Study of the first example in Figure 20 shows that the first card generates a wire extending in the Z-direction from 0.05 to 0.0 meters. This has one segment and a tag of two. The radius is located in the first entry of the radius table, and the wire is located in the second coordinate system.

The second card generates a wire extending along the local Y-axis from 0.0 to 0.05. The rest of the values in this command are defaulted to those specified on the previous card. Note that all subsections lying

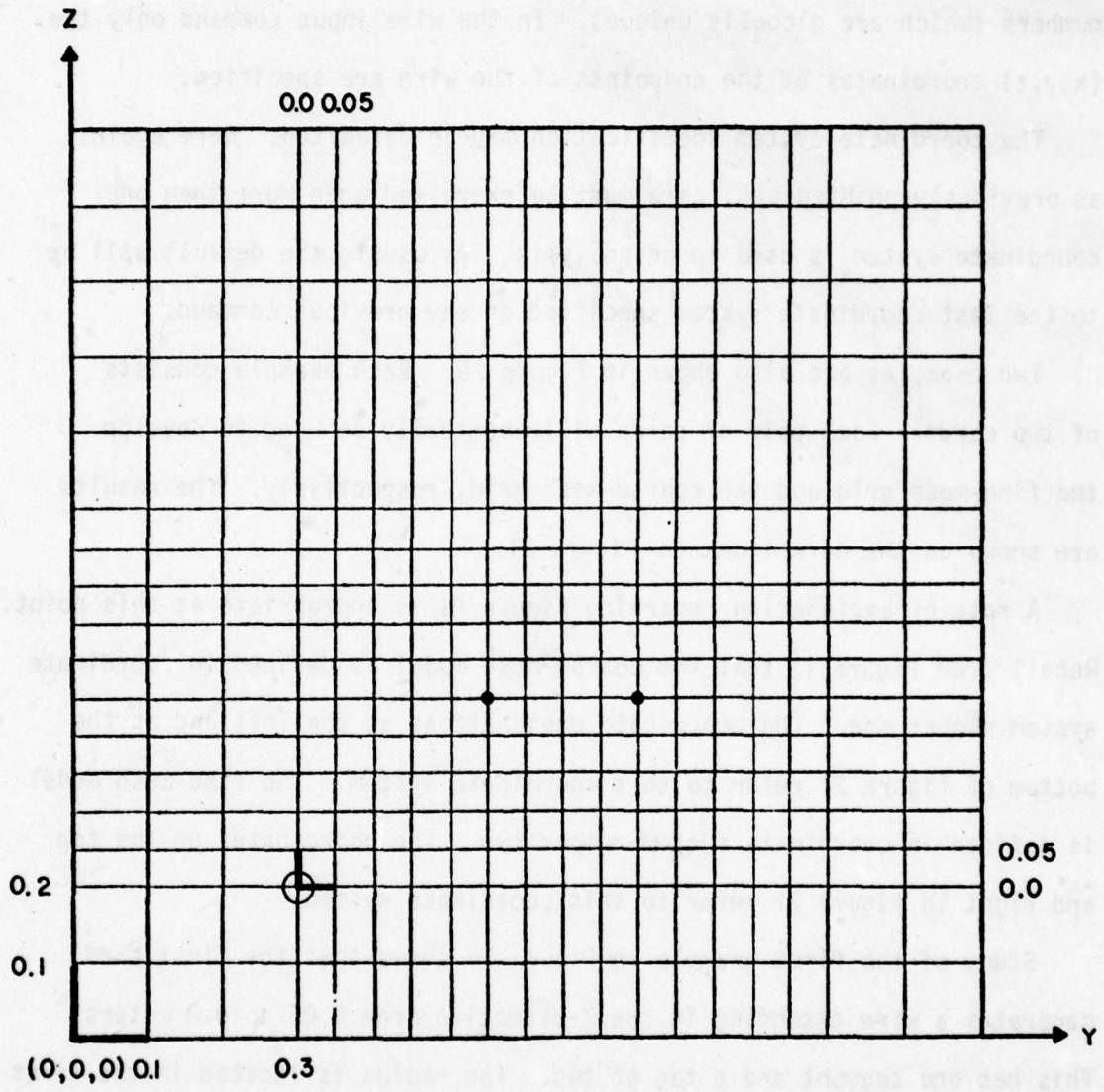


FIGURE 21. WIRES GENERATED BY WIRE INPUT COMMANDS OF FIGURE 20.

in coordinate system number two will in the sample problem be given a tag of two. This will aid the user later on when he reviews the geometry stored in GEMACS.

The second example also consists of two cards. The first generates a wire along the global Z-axis extending from 0.1 to 0.0. It has one subsection, a tag of one and a radius located in the first entry of the radius table. It is located in coordinate system number one. Again, the tag number is being used to indicate in what coordinate system the wire lies.

The second card of this example generates a wire along the global Y-axis extending from 0.0 to 0.1. Its remaining parameters are identical to those of the first segment in this example.

This method of subsection generation is a one-step process. Four commands were used to generate the four subsections. Specifying points and then connecting them would have required eight commands.

An entire geometry could be generated by specifying each and every subsection. However, for the example this would require more than 700 such commands. The geometry language provides other commands which allow one to use these basic subsections as building blocks to generate larger structures using many fewer commands than 700. In the discussion of these commands it will be seen that some are used quite extensively in the generation of a wire grid model for the top of the hut.

C. Element Generation

It was shown that one of the ways of grouping subsections having some particular characteristic in common is by giving them all the same tag

number. A second technique of grouping is by identifying all of the subsections as part of some "defined element." That is, a specified group of subsections is identified as "WING" or "FINE" or COARSE" or any other legal name chosen to identify that set. A legal name is an identifier composed of six characters or less, a character being one of the letters of the alphabet (A-Z) or one of the numerals (0-9). However, the first character must be one of the letters of the alphabet (A-Z).

The generation of an element begins with a Define Element (DF) command, shown in Figure 22. As can be seen, it is a simple command consisting of the mnemonic DF and the name of the element. Figure 22 also shows two examples which are used in the solution of the sample problem.

The define element command is then followed by a deck of geometry commands which may include point specification and connection, wire input and any of the macrogenerators to be discussed very shortly. All subsections and points defined by these commands will become associated with this element and as such may be referenced by calling out the name of the element. That is, if any operation is performed on the element, then that operation is performed on all the points and wires specified as being a part of that element.

When all of the commands that are needed to define the geometry of the element have been input into GEMACS, the set is then closed by placing a Define End (DE) command after the last geometry command of the element. This command is shown in Figure 23. As can be seen it simply consists of the mnemonic DE.

```
DF  NAME

DF  GRID1
DF  GRID2
```

FIGURE 22. DEFINE ELEMENT COMMAND

```
DE

RA   0.000776
DF   GRID1
WR   0.0  0.0  0.05  0.0  0.0  0.0  1  2  1  2
WR   0.0  0.0  0.0  0.0  0.05  0.0
DE   $GRID1
```

FIGURE 23. DEFINE END COMMAND

Note however, that a comment should be included to indicate what element is being closed. In this case the dollar sign ($) is needed as well as the blank after the mnemonic. Figure 23 shows an example of the definition of an element. The result is the pair of lines shown darkened in the fine mesh of Figure 21. The first card sets up the radius table. The second card defines the element and gives it the name GRID1. The third and fourth cards are the same as those used in the first example in Figure 20. The last card in Figure 23 is the define end command with the comment that this closes the group to be identified as GRID1.

Two points should be kept in mind when using the define operation. First, the define element operation may be nested. This is easiest to explain by using an example. The fictitious example in Figure 24 starts off with a define element command, labeling the element as "PLANE". This is followed by a series of wire input cards, which may define some part of the aircraft. These cards are followed by another define element card with a label of "WING". The following wire input cards would then define the wing. Note that no define end command for the plane has as yet appeared. The geometry for "WING" thus also becomes part of the geometry for "PLANE". Therefore, any reference to the label "PLANE" will affect all the subsections belonging to "WING" also. Any reference to "WING" will only affect the second set of wire input cards, but not the first set, which are a part of "PLANE".

The first define end card applies to "WING", and it closes that group only. It does not close the group belonging to "PLANE". Thus, the group "WING" is a nested defined element. Nesting can be carried to ten deep.

```
DF  PLANE
WR
 :
WR
DF  WING                    (FICTITIOUS)
WR
 :
WR
DE $ WING
 :
DE $ PLANE
```

FIGURE 24. NESTED DEFINED ELEMENTS

This is an example of nesting two-deep.

Other elements could be defined describing the tail, the vertical stabilizer and so forth in the same manner. These would again be nested two-deep with the group "PLANE". A reference to "TAIL" would not in any way affect the group "WING". Once a group is closed, no new elements may be added to that group.

The final define end command would close the group "PLANE". Any reference to "PLANE" would therefore affect the groups "WING", "TAIL", and any other subelements that were defined. The point is that closing nested defined elements works from the innermost set outward. Each subelement is closed before the element to which it belongs is closed.

The second point is illustrated by the fictitious example shown in Figure 25. The example again starts out by defining an element called "PLANE", described by WR commands. Then, an operation called translation is performed on these wires. This command is explained in Section IV.D. Briefly, what it does is generate a duplicate set of wires using the first set of wire input cards as a model. Notice, though, that up to this point the group "PLANE" has not been closed by a define end command. This example shows that an element need not be completely defined before it can be referenced on one of the macrogenerator commands. Also, keep in mind that the group of subsections generated by the translate command will be affected by any reference to the element "PLANE".

Once elements have been defined, various operations may be performed on the subsections comprising those elements by simply referring to the name of the element. Two of these commands belong to the class of element

```
DF  PLANE
WR
 .
 .
 .
WR
XL  DF  PLANE  . . .     (FICTITIOUS)
WR
 .
 .
 .
WR
DE  $ PLANE
```

FIGURE 25. EXAMPLE OF REFERENCE TO INCOMPLETELY DEFINED ELEMENT

generation commands. The first is the Attach (AT) command, Figure 26. This command results in a group of points and/or subsections being relocated at the origin of the coordinate system specified in this command. This command is useful when an element is defined in some local coordinate system; and it will be used on several different elements, each of which is defined in its own individual coordinate system. Suppose, for example, that the shape of an antenna has been defined in coordinate system one. It is to be located on the left and right wings and left and right horizontal stabilizers of an aircraft. Suppose further that each of the four aircraft sections was defined in its own coordinate system labeled two to five, respectively. The implementing commands are shown in the example in Figure 26.

From the figure it can be seen that AT is the mnemonic for the attach command. Either point identifiers (PT), tag identifiers (TG) or defined elements (DF) may be referenced by specifying the number of the point or tag or the name of the element. The final piece of data is the number of the coordinate system into which the group of points or wires will be transferred. It cannot be defaulted in this command.

However, it must be remembered that the origin of the coordinate system in which "ANTENA" is defined will be located at the origin of the coordinate system specified in the attach command. Depending on the relative positions of the respective coordinate axes, there may result both translation and rotation. Therefore, the origins of coordinate systems two through five must be at the locations on the wings and stabilizers at which

AT { PT / TG / DF } { N / N / NAME } NCS

```
AT      DF      ANTENA      2
AT      DF      ANTENA      3
AT      DF      ANTENA      4      (FICTITIOUS)
AT      DF      ANTENA      5
```

FIGURE 26. ATTACH COMMAND

the antenna is to be located.

The final command in this group is the Combine Elements (CE) command in Figure 27. The sole purpose of this command is to make referencing elements easier for the user. When several elements such as "BODY", "WING" and "TAIL" have been defined, then this command may be used to lump their groups under the element name "PLANE". This would be an alternative to what was done in the example in Figure 24. The advantage of this technique over that shown in the example in Figure 24 is that in the latter technique all the elements and subelements must be defined in the same coordinate system, which may not be convenient due to the complexity of the geometry. The elements combined in the example in Figure 27 may be defined in their own individual coordinate systems.

However, once the elements have been combined under one name, then the individual elements can no longer be accessed under their individual names. In the example just cited, once the element "BODY" has been incorporated under the name "PLANE", then one cannot any longer refer to the individual element "BODY". However, using the method shown in Figure 24, the individual elements can still be individually accessed.

Looking at the command itself in Figure 27, it is seen that CE is the symbol for this command. Then, NAME1 is the name by which the resultant group will be known. The remaining names given in the command are the names of the individual defined elements which are to be combined. In the example, "PLANE" will be the name of the resultant structure. "BODY", "WING" and "TAIL" will be the individually defined elements making up this structure.

The definition of elements finds its greatest utilization when the geometry of the structure is being generated by the so-called macrogener-

```
CE  NAME1   NAME2  NAME3 . . .

CE  PLANE   BODY   WING   TAIL     (FICTITIOUS)
```

FIGURE 27. COMBINE ELEMENTS COMMAND

ators, the operations of reflection, rotation and translation. This last command is extremely useful when a wire grid model of a structure is being generated.

D. Macrogenerators

A macrogenerator is a command which uses as input a specified group of already defined subsections and generates one or more identical groups of subsections by either reflection, rotation or translation.

Before beginning the discussion of these three commands the numbering sequence of the points and lines should be described. Remember that when points are defined by the point specification (PT) command, the user specifies what the point number is to be. When one of the macrogenerator commands is used, the generation of additional points occurs. These new points will be ordered in the same sequence in which the original points were specified. That is, suppose points one through five were defined and then reflected through some plane. Point six will be the image of point one, point seven will result from point two, and so on; point ten being the image of point five. If further individual points are going to be defined, then the user must remember to start the numbering at eleven, since point numbers must be globally unique.

The same thing will happen with the numbering of subsections. Subsections are automatically ordered in the sequence in which they are generated. This ordering was illustrated in the example of Figure 18. If the points in the previous example were subsections, then subsection one would image as subsection six with the same relative positions of negative and positive ends.

Subsections and points in the plane of reflection will also generate colocated subsections and points with their individual identifiers. Thus, in the example if point or subsection three were in the plane of reflection, it would generate point or subsection eight, respectively. Since GEMACS is generating these identically numbered points, it will flag these points and ignore their existence. The identical segments will be zeroed out in all calculations involving the interaction matrix.

The first macrogenerator command to be discussed is the Reflect (RF) command, which is shown in Figure 28. This command executes the symmetry operation of reflecting through a plane normal to the axis specified. For example, if the X-axis is specified, then the plane of reflection is the Y-Z plane; and all (x,y,z) coordinates become (-x,y,z) coordinates. Note that since segments may be located in the plane of reflection, when a wire is located in the Y-Z plane, then a wire would be generated in the Y-Z plane and be superimposed on the original wire. However, its existence would be flagged as a duplicate wire; and it would be ignored in all future calculations. The same is of course true for points located in the plane of reflection.

As can be seen from Figure 28, the mnemonic for this command is RF. The points, subsections or defined elements to be reflected are referenced by their identifier for points or by the tag for subsections or by their name for defined elements. The symbol A1 represents either the X-axis, Y-axis or Z-axis depending on which plane the reflection is to occur through first. If the X-axis is specified, then reflection will be through the Y-Z plane. If the Y-axis is specified, reflection will occur through

RF $\left\{\begin{matrix}\text{PT}\\\text{TG}\\\text{DF}\end{matrix}\right\}$ $\left\{\begin{matrix}\text{N}\\\text{N}\\\text{NAME}\end{matrix}\right\}$ A1 A2 A3 [INCTAG] [NCS]

RF TG 2 X Y 2 (FICTITIOUS)

FIGURE 28. REFLECT COMMAND

the X-Z plane. If the Z-axis is specified, reflection will occur through the X-Y plane.

The symbol A2 represents the axis normal to the plane through which the second reflection will occur. Similarly, A3 represents the axis normal to the plane through which the third reflection will occur. For any one of these symbols the input would be X, Y or Z. No axis designation shall occur more than once. The order in which the axis designations appear will determine the order of the planes through which reflection will occur. Finally, if the reflection operation is being performed on an element being defined (that is, a reference to an incompletely defined element - Figure 25), all subsections generated by the command will be associated with the element being defined.

It should be noted that although the number of planes of reflection may vary from one to three, none of the symbols A1, A2 or A3 is shown as one that can be defaulted. The code is set up such that only one or two axes can be input without having to worry about including two or one default values, respectively. This is shown in the example given in Figure 28.

The symbol INCTAG represents the tag increment parameter. The integer input here will be added to the tag numbers specified on the subsections that are to be reflected. Subsections generated by the first reflection will have tags incremented by the value of INCTAG. Subsections generated by the second reflection will have tags incremented by twice the value of INCTAG over the tag numbers of the original subsections. Subsections generated by the third reflection will have tag identifiers incremented by four times the value of INCTAG over the tags of the original subsections.

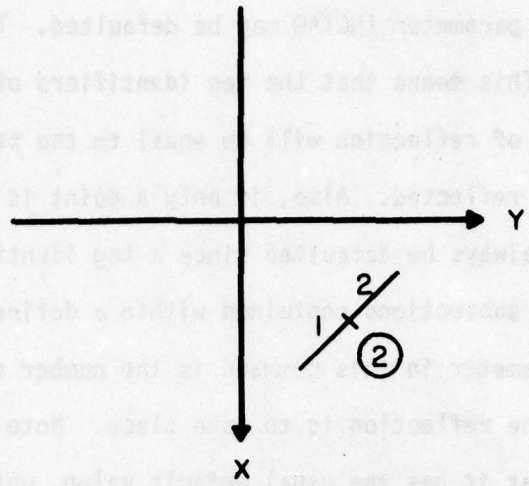
Note that the parameter INCTAG may be defaulted. The default value is always zero. This means that the tag identifiers of subsections generated by any order of reflection will be equal to the tags of the original subsections being reflected. Also, if only a point is being reflected, then INCTAG will always be defaulted since a tag identifier only applies to subsections or to subsections contained within a defined element.

The final parameter in this command is the number of the coordinate system in which the reflection is to take place. Note that it can be defaulted; and that it has the usual default value, which is the last identifier for a coordinate system specified on any previous command.

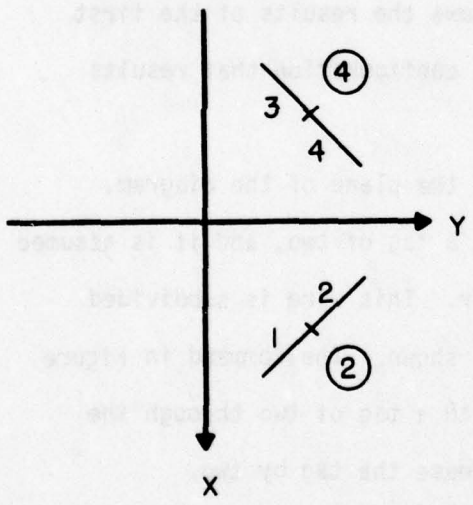
An example of this command is shown in Figure 28. Figure 29 (a) shows the original configuration. Figure 29(b) shows the results of the first reflection, and Figure 29(c) shows the final configuration that results from the command.

In Figure 29 the Z-axis is coming out of the plane of the diagram. There is a single wire in the X-Y plane with a tag of two, and it is assumed that it is the only wire with this tag number. This wire is subdivided into two subsections, which are sequenced as shown. The command in Figure 28 tells GEMACS to first reflect the wire with a tag of two through the Y-Z plane (normal to the X-axis) and to increase the tag by two.

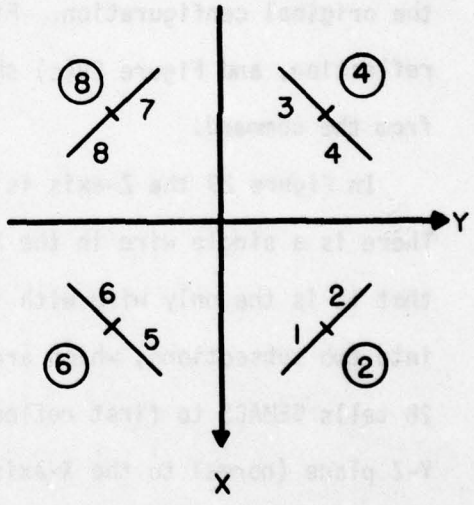
Figure 29(b) shows the results of this operation. The original wire is left unaffected. A second wire has been generated in the X-Y plane. The wire has a tag of four and is subsectioned into two segments, subsection three being the image of subsection one and segment four being generated by segment two.



(a) ORIGINAL CONFIGURATION



(b) FIRST REFLECTION



(c) SECOND REFLECTION

FIGURE 29. RESULTS OF THE EXAMPLE SHOWN IN FIGURE 28.

The example in Figure 28 also specifies that a second reflection is to be performed. This is to be through the X-Z plane; that is, normal to the Y-axis. The tag in this case is to be incremented by two times two, resulting from the fact that this is the second reflection incrementing the tag identifier by two.

Figure 29(c) shows the results of this operation. First of all, note that not only were the original subsections reflected but that the subsections generated as a result of the first reflection were also reflected. The wire with the tag of two generated a wire that has a tag of six (two times two plus two), which has two segments. Subsections five and six were generated by segments one and two, respectively. The wire with a tag of eight (two times two plus four) is the image of the wire with the tag of four. Correspondingly, segments seven and eight are the images of subsections three and four, respectively. This shows how the subsection numbers are sequenced as new subsections are generated and how tag identifiers are incremented.

The second macrogenerator command is the Rotation (RT) command. The general form of the command is shown in Figure 30. This command is extremely useful to generate geometries that have one or more axes of revolution, such as a cylinder or a sphere. The mnemonic for this operation is RT. Points, line segments or defined elements may be rotated, identifying them by their number, tag or name, respectively. The value of IADD indicates the number of additional sets of points or subsections that are to be generated.

RT $\left\{\begin{matrix}\text{PT}\\\text{TG}\\\text{DF}\end{matrix}\right\}$ $\left\{\begin{matrix}\text{N}\\\text{N}\\\text{NAME}\end{matrix}\right\}$ [ADD] RX RY RZ [INCTAG] [NCS]

RT PT 1 5 0.0 0.0 300.0 (FICTITIOUS)

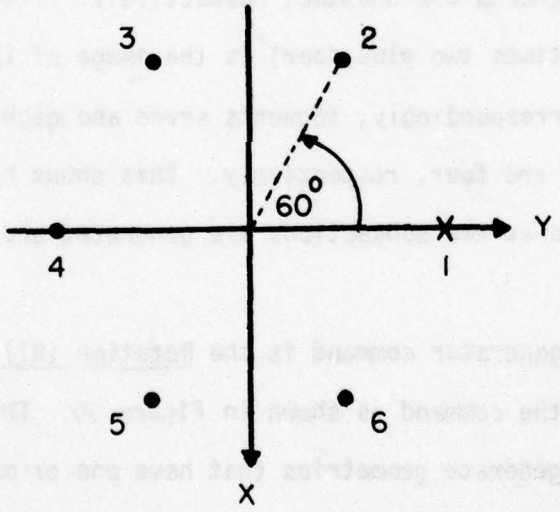


FIGURE 30. ROTATE COMMAND

The parameters RX, RY, RZ are the total angles through which the original points and wires are to be rotated. Keep in mind that these do not represent incremental changes around the axes. For example, if IADD were equal to three and RX equal to fifteen, then each original point or subsection would generate a new point or subsection at angles of five, ten and fifteen degrees of rotation. The result would not be points and subsections located at angles of 15, 30 and 45 degrees.

One other point to keep in mind is that none of these parameters may be defaulted or left out as was the case with the reflect command. Also, the order in which the parameters appear is important. The rotation around the X-axis (RX) must be first, followed by the rotation about the Y-axis (RY), which in turn is followed by the rotation about the Z-axis (RZ). Therefore, if rotation about the Y-axis only is desired, the value for RY must be preceded by a zero for RX and followed by a zero for RZ.

One final note on the direction of positive rotation for RX, RY and RZ. Starting at the original point or subsection, positive rotation is considered to be counterclockwise around the X-axis for RX, around the Y-axis for RY and around the Z-axis for RZ. For example, if a point exists on the positive X-axis and is to be rotated around the Z-axis, then the direction of positive rotation is from the positive X-axis to the positive Y-axis.

The final two parameters are the same as those that appear in the reflect command, namely the tag increment and the number of the coordinate system in which the rotation is to take place. The default values are also the same; zero for the tag increment and the last coordinate system specified on any previous command for NCS.

Figure 30 also illustrates the use of this command. There is originally a point located on the Y-axis. It is denoted by the "X" and has a number one. The sample command says that this point is to be rotated three hundred degrees around the Z-axis, which is directed out of the figure. During this rotation five new points are to be generated. The diagram shows the result. The result is six points spaced around a circle, each separated from its neighbors by sixty degrees. The direction of positive rotation is indicated by the arrow and determines the order in which the points are finally numbered.

As is the case with the reflect command, if an element is being defined when the rotate command is encountered, all the new points and subsections will be associated with that particular element being defined.

The last macrogenerator is the Translation (XL) command, whose general format is shown in Figure 31. In form it is almost identical to the rotation command that has just been discussed. The result of this operation is the generation of additional points and subsections that are identical to those specified but are displaced by a specified distance.

The symbol for this command is XL. Again the operation can be performed on points, subsections and defined elements by specifying the number, the tag or the name, respectively. The number of new sets of points or segments to be generated is given by the value of IADD. If IADD is zero, then the original set is translated to the new location.

The parameters DX, DY and DZ are the incremental changes in the X, Y and Z directions, respectively, that will be added to each succeeding new

```
XL  { PT }  { N    }   IADD  DX  DY  DZ  [INCTAG]  [NCS]
    { TG }  { N    }
    { DF }  { NAME }
```

```
RA    0.000776
DF    GRID1
WR    0.0  0.0  0.05  0.0  0.0  0.0  1  2  1  2
WR    0.0  0.0  0.0   0.0  0.05 0.0
DE    $GRID1
XL    DF   GRID1   13 0.0  0.05  0.0
.
.
.
```

FIGURE 31. TRANSLATE COMMAND

set of points or subsections that have been generated. For example, if point one is located at X equals five and if three new points are to be generated at incremental spacings of three along the X-axis, then point two will be located at X equals eight, point three will be located at X equals eleven; and point four will be located at X equals fourteen. Thus, (DX,DY,DZ) forms an incremental translation vector.

The last two parameters, INCTAG and NCS, are the same ones that have been discussed for the reflect and rotate commands. They have the same default values of zero and the last specified coordinate system.

As is the case with the first two macrogenerator commands, if an element is in the process of being defined when the translate command is encountered, all the new points and wires will be associated with the element being defined.

This command finds a tremendous amount of use in the generation of a constant-mesh wire grid. The example shown in Figure 31 is taken from the input deck for the analysis of the sample problem. The results of the translation are shown in Figure 32 as the darkened lines between Y' equals 0.05 and 0.7 meters in the second coordinate system. The first five cards generate subsections one and two and are the same as those in Figure 23. The defined element GRID1 is shown in Figure 21 as the darkened pair of lines in the fine-mesh wire grid. The translate command then says to take the defined element GRID1 and generate 13 additional elements, each translated from the preceding one by 0.05 meters in the Y'-direction. The value for INCTAG has been defaulted so that all of the new subsections will have a tag of two, which is the tag of the original subsections. The value for

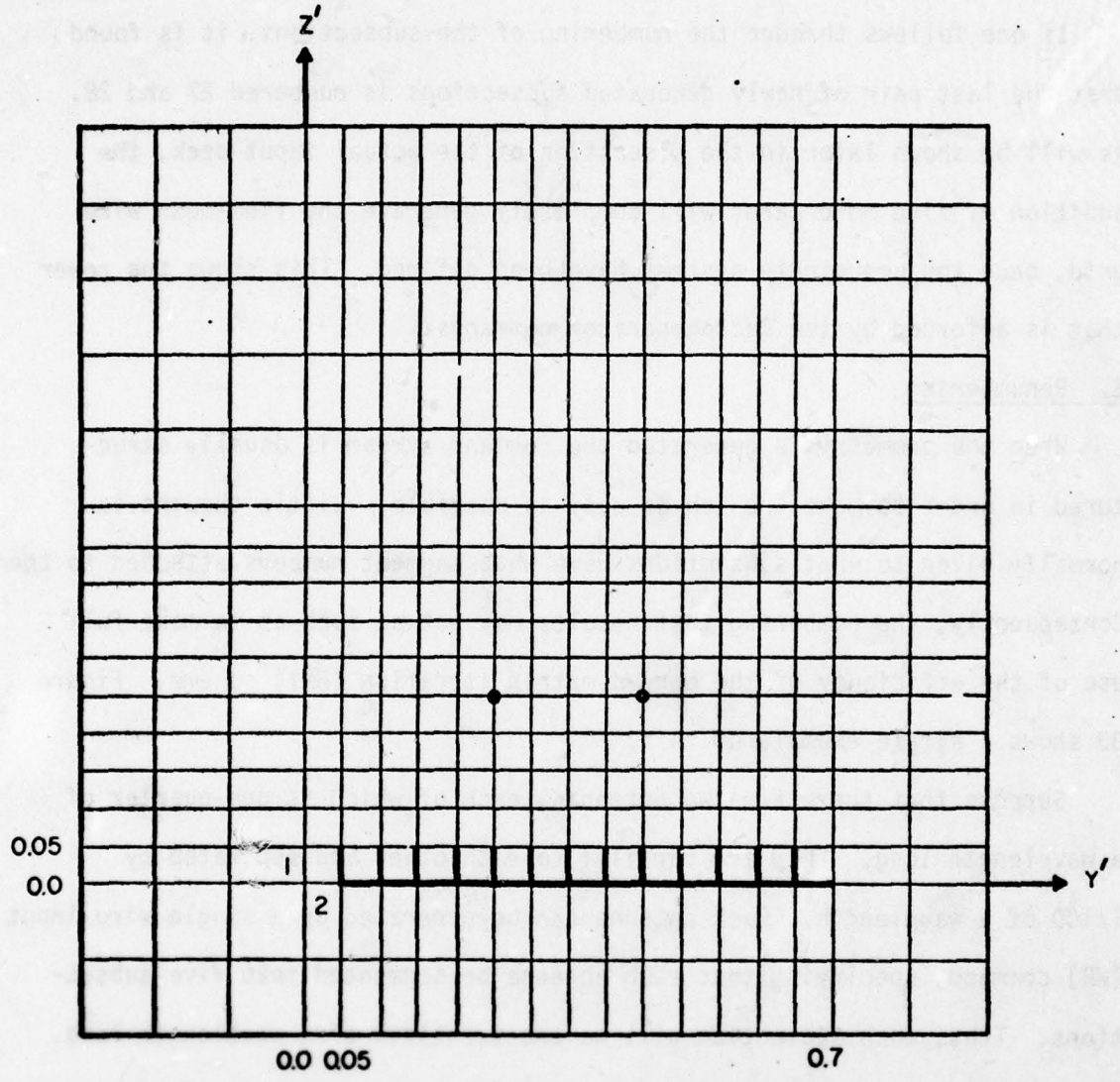


FIGURE 32. RESULTS OF EXAMPLE GIVEN IN FIGURE 31.

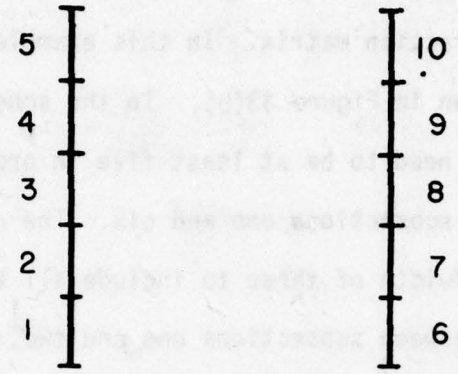
the number of the coordinate system is also defaulted since the new elements are in coordinate system number two, which is specified in the first wire input (WR) command.

If one follows through the numbering of the subsections, it is found that the last pair of newly generated subsections is numbered 27 and 28. As will be shown later in the discussion of the actual input deck, the addition of five more cards will completely generate the fine-mesh wire grid, once the coordinate systems have been defined. This shows the power that is afforded by the macrogenerator commands.

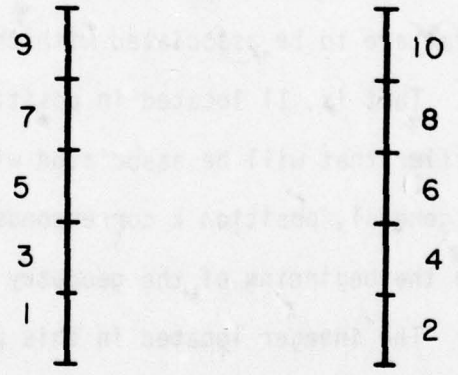
E. Renumbering

When the geometry is generated the command stream is usually structured in order to make the job as easy as possible. Little thought is normally given to what subsections have what segment numbers attached to them. Consequently, the numbering that results may not be such as to make full use of the efficiency of the banded matrix iteration (BMI) scheme. Figure 33 shows a simple example of this.

Suppose that there are two antennas, each of which is one-quarter of a wavelength long. They are parallel to each other and separated by 1/100 of a wavelength. Each antenna can be generated by a single wire input (WR) command, specifying that each antenna be segmented into five subsections. Thus, each subsection will be one-twentieth of a wavelength long. This is shown in figure 33(a). This means that the center-to-center separations between parallel subsections is five times smaller than the center-to-center separation between adjacent subsections on the same antenna. The interactions between subsections on separate antennas are greater than the



(A) ORIGINAL NUMBERING



(FICTITIOUS)

(B) RENUMBERED

NOTES: 1) ANTENNA SEPARATION = 0.01λ
2) SEGMENT LENGTHS = 0.05λ

FIGURE 33. EXAMPLE OF RENUMBERING (NOT TO SCALE)

interactions between subsections on the same antenna.

When the BMI technique is used, its efficiency is increased by having as many of the largest interactions as possible clustered near the main diagonal of the interaction matrix. In this example, the subsections should be numbered as shown in Figure 33(b). In the scheme shown in Figure 33(a) the bandwidth would need to be at least five in order to include the large interaction between subsections one and six. The numbering in Figure 33(b) would require a bandwidth of three to include all the major interactions; that is, those between subsections one and two, one and three, and one and four.

Figure 34 shows the forms of the Renumber (RN) command. The symbol RN denotes that this is the renumber command. The symbols I1 through In are the new segment numbers that are to be associated with the segment in that position in the list. That is, I1 located in position one of the list is the integer identifier that will be associated with what is originally subsection one. In general, position k corresponds to the kth subsection generated since the beginning of the geometry input or since the last renumber command. The integer located in this position is the new segment number that will be associated with the kth subsection generated.

Three things must be kept in mind when using this command. First, no two numbers in this list may ever be the same. This will give the same subsection identifier to two different subsections. Second, the list may not exceed 256 items due to the structure of the code. This does not

RN I1 I2 I3 . . . IN

-OR-

RN I1 I2 . . . -IM IP IQ . . . IN

RN 1 3 5 7 9 2 4 6 8 10 (FICTITIOUS)

FIGURE 34. RENUMBER COMMAND

mean that all the numbers input be between one and 256 or that they follow any particular sequence. However, if there are N subsections and all of them are to be renumbered, then all of the integers one through N must appear in a renumber command. Finally, when a renumber command is encountered, all of the subsections generated since the last renumber command are renumbered according to the sequence specified by the integers in the list. If this is the first renumber command, then all of the subsections generated since the beginning of the geometry input will be resequenced. Therefore, if the first form of the command is used, no more than 256 subsections should be generated before a renumber command is encountered.

A shortcut is available and is shown as the second form in Figure 34. When a negative integer is encountered in the list such as -Im in Figure 34, the new sequence number of the subsection corresponding to that position in the list will be the absolute value of the negative integer; and the new sequence numbers will be incremented by one until the number of segments less one identified by the next integer in the list (Ip in Figure 34) has been counted. Figure 35 shows an example of this. Suppose that subsections one through ten have been generated and that they are to be renumbered. These are represented by the line labeled "OLD" in Figure 35. By employing the command shown in the figure one ends up with the sequence shown in the line labeled "NEW".

Item one in the list has the value two. Thus, what was subsection one is now subsection two. Four is the value of item two in the list, and the result is that subsection two now becomes subsection four. Item three is a

```
RN  2  4  -5  4  1  9  3  10

OLD 1  2  3  4  5  6  7  8  9  10
NEW 2  4  5  6  7  8  1  9  3  10
```

(FICTITIOUS)

FIGURE 35. RENUMBER EXAMPLE

negative five. Subsection three becomes subsection five. The next item in the list is four. That means that the next three subsections are sequentially numbered, starting with six since the preceding subsection is numbered five. Therefore, segments four, five and six become subsections six, seven and eight, respectively. The last four subsections are renumbered in the same manner as the first two subsections.

The example shown in Figure 34 is the renumber command that would result in the segment numbering shown in Figure 33.

This completes the discussion of the GEMACS geometry language. It is seen to be a very simple language and yet it is sufficiently general so that one can develop any geometry to any degree of accuracy. For example, the model of the hut and antennas requires 714 subsections; but only 51 commands are needed. How this is accomplished is shown in the next subsection.

F. Sample Problem Geometry Input Deck

The first thing that must be done is to set up the global parameters for the problem. This is done by using the general commands discussed in section IV-A. This part of the deck is shown in Figure 36. Line one sets up all the radii that will be used in the analysis of the coupling. In this case all the subsections will have the same radius, that is, the radius of the physical antenna wires.

Figure 12 shows the total wire grid that will be generated. It also shows the two coordinate systems that will be employed. In both cases the X-axis comes out of the figure. Coordinate system one is identical to the global system and has axes Y and Z. Coordinate system two is the system in the darkened box and has axes Y' and Z'. Its origin is displaced from the

```
1   RA  0.000776
2   CS  1  0.0  0.0  0.0  0.0  0.0  0.0
3   CS  2  0.0  0.3  0.2  0.0  0.0  0.0
```

FIGURE 36. GENERAL CARDS OF INPUT DECK

global origin by being located in the global system at Y equals 0.3 and Z equals 0.2 meters. Lines two and three in Figure 36 provide this information to GEMACS.

Figure 37 shows the cards needed to generate the fine-mesh wire grid. Much of this has already been discussed in connection with Figure 31. Card one specifies an element which is one row of the fine-mesh wire grid. Cards two through five generate segments one and two in Figure 32 and have been discussed with reference to Figures 20 and 21. Card six generates the darkened lines in Figure 32. Card seven generates subsection 29 in Figure 38.

Figure 38 shows the results of the first eight cards shown in Figure 37. A basic building block has been generated which is translated by the command given in card nine a total of nine times. The result of this translation is shown in Figure 39 as subsections 30 to 290. Card ten in Figure 37 closes the grid by drawing a line across the top of the grid which is segmented into 14 subsections. This generates segments 291 to 304. This completes the generation of the fine-mesh wire grid model of the top of the hut in the vicinity of the two antennas of interest.

Figure 40 shows the cards that are needed to generate the models of the antennas. The diagram in the figure shows the results of the operations along with the subsection numbering. The first card generates a model of the transmitting antenna. It has a length of one-tenth of a meter and is divided up into two subsections. It has been given a tag identifier of three to flag these segments in the segment table output by GEMACS. The second card generates a model of the receiving antenna, which has a length of three-

```
1     DF  ROW1
2     DF  GRID1
3     WR  0.0  0.0  0.05  0.0  0.0  0.0  1  2  1  2
4     WR  0.0  0.0  0.0  0.0  0.05  0.0
5     DE  $GRID1
6     XL  DF  GRID1  13  0.0  0.05  0.0
7     WR  0.0  0.7  0.0  0.0  0.7  0.05
8     DE  $ROW1
9     XL  DF  ROW1  9  0.0  0.0  0.05
10    WR  0.0  0.0  0.5  0.0  0.7  0.5  14
```

FIGURE 37. GENERATION OF FINE-MESH GRID

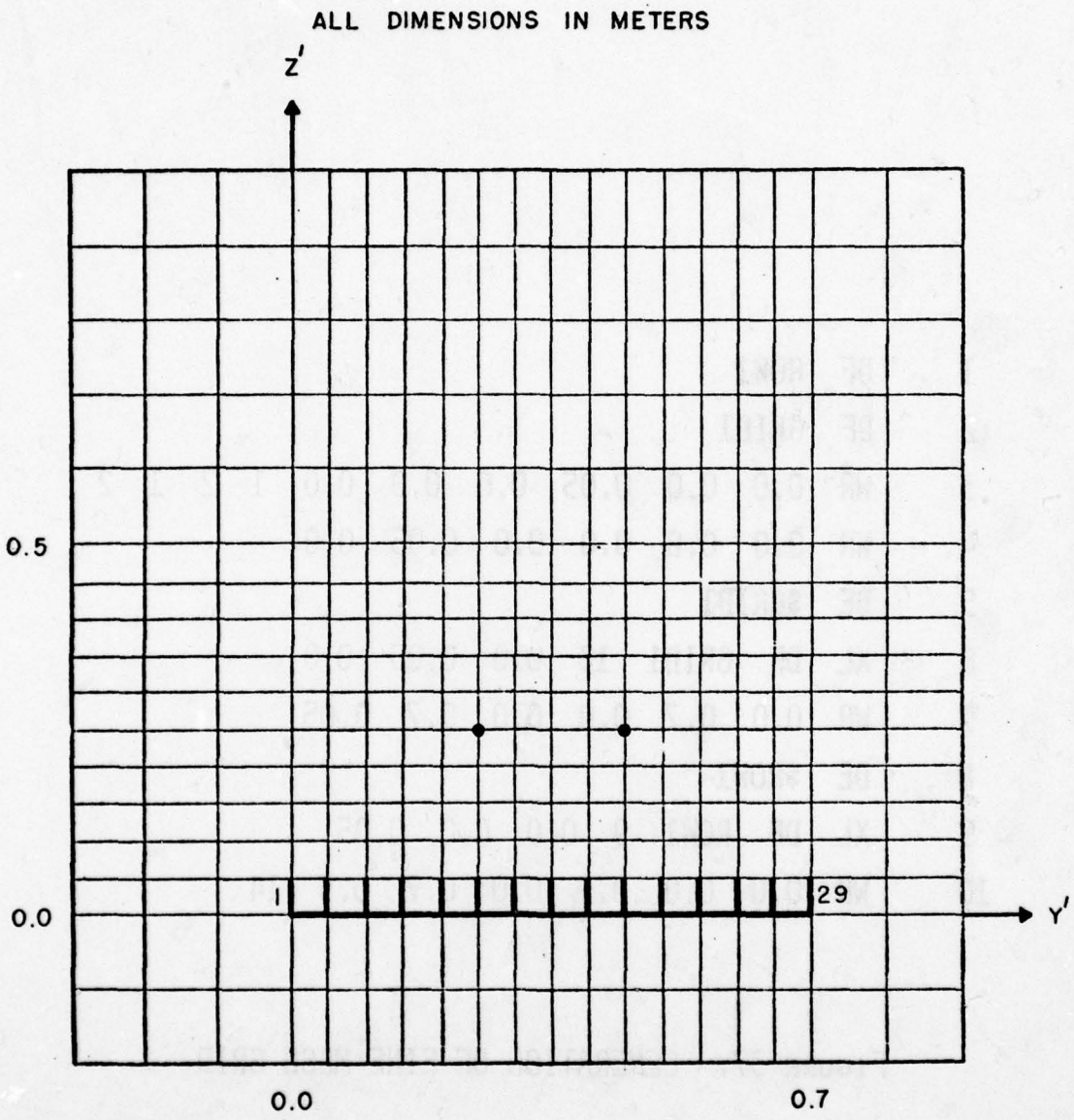


FIGURE 38. DEFINITION OF ROW1

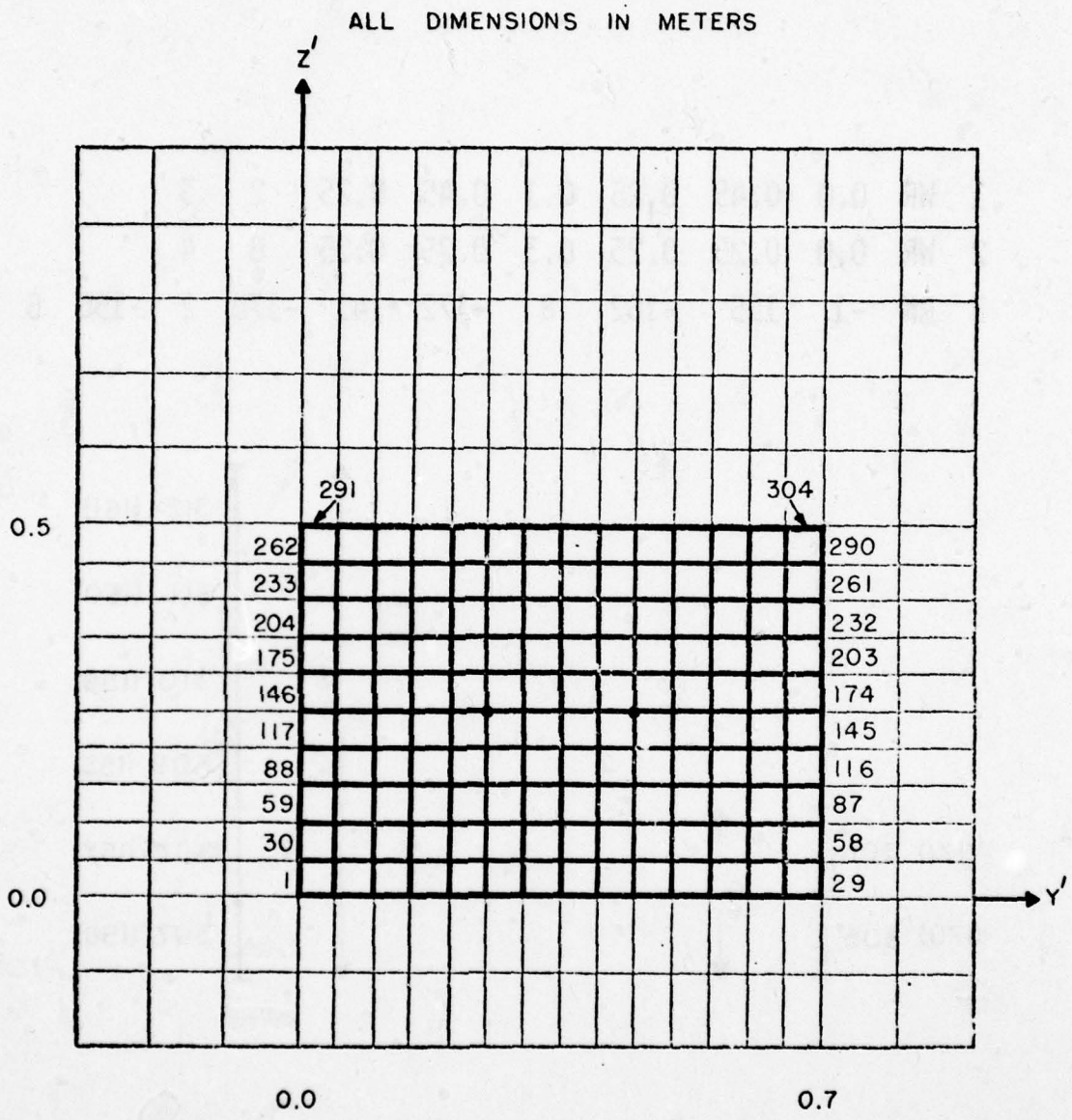
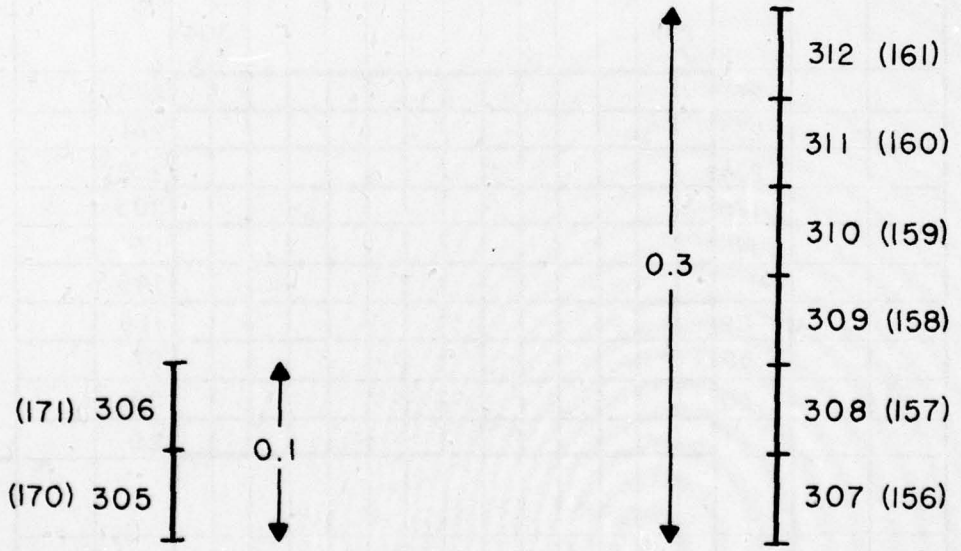


FIGURE 39. FINE-MESH WIRE GRID

```
1  WR  0.0  0.45  0.25  0.1  0.45  0.25   2   3
2  WR  0.0  0.25  0.25  0.3  0.25  0.25   6   4
3  RN  -1   155   -162   8   -172   141  -170  2  -156  6
```

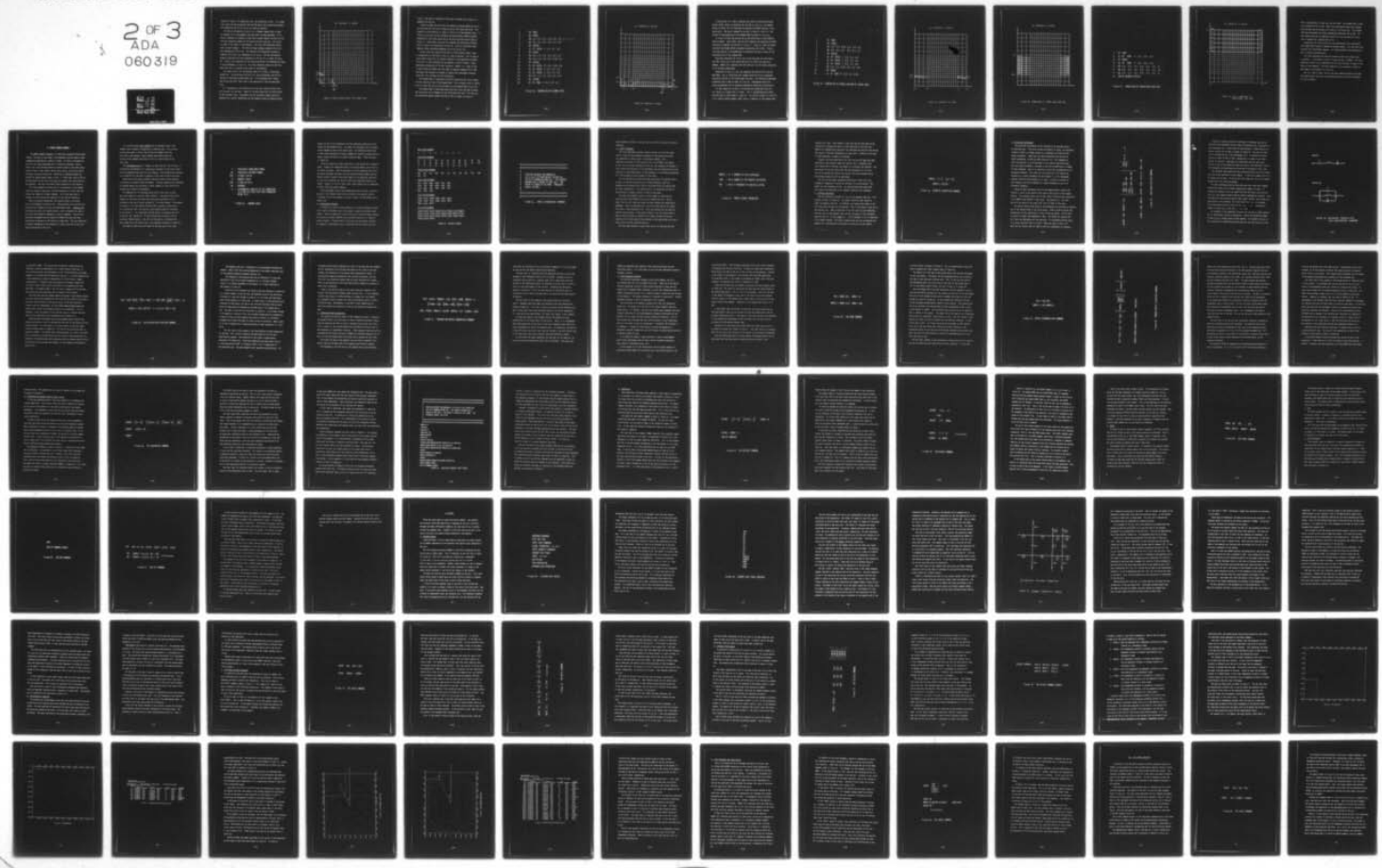


ALL DIMENSIONS IN METERS

FIGURE 40. WIRE ANTENNA MODELS

AD-A060 319 ROME AIR DEVELOPMENT CENTER GRIFFISS AFB N Y F/G 9/5
AN INTRODUCTION TO THE GENERAL ELECTROMAGNETIC MODEL FOR THE AN--ETC(U)
SEP 78 K R SIARKIEWICZ
UNCLASSIFIED RADC-TR-78-181 NL

2 OF 3
ADA
060319



tenths of a meter, six subsections and a tag identifier of four. It is seen from figure 40 that subsections 305 and 306 are on the transmitting antenna and subsections 307 to 312 are on the receiving antenna.

The third card shown in Figure 40 is a renumber command that is used to renumber all of the segments that have thus far been generated. It is used to renumber the antennas so that their segment numbers are more in line with the subsection numbers of the nearby wire mesh segments. The result is shown in the figure in the Appendix. The first 155 subsections retain their original numbers. Then the wire mesh segments numbered 156 to 163 are renumbered as 162 to 169. The remaining 141 subsections originally numbered 164 to 304, are renumbered as 172 to 312. The two transmitting antenna subsections are then renumbered as 170 and 171, as shown in Figure 40. Finally, the subsections on the receiving antenna are renumbered as shown in the parentheses in Figure 40. The final numbering scheme for the fine-mesh wire grid is as shown in the figure in the Appendix.

This takes care of all the modeling that will be done in coordinate system two. The coarse-mesh wire grid will now be generated, and will be defined in coordinate system number one. This development will pretty much follow the basic lines used in the generation of the fine-mesh wire grid.

It is necessary to first define one of the basic building blocks that will be used over and over. Figure 41 outlines these basic building blocks by the darkened lines. The numbers are the subsection numbers that will eventually be used as identifiers for the segments making up these building

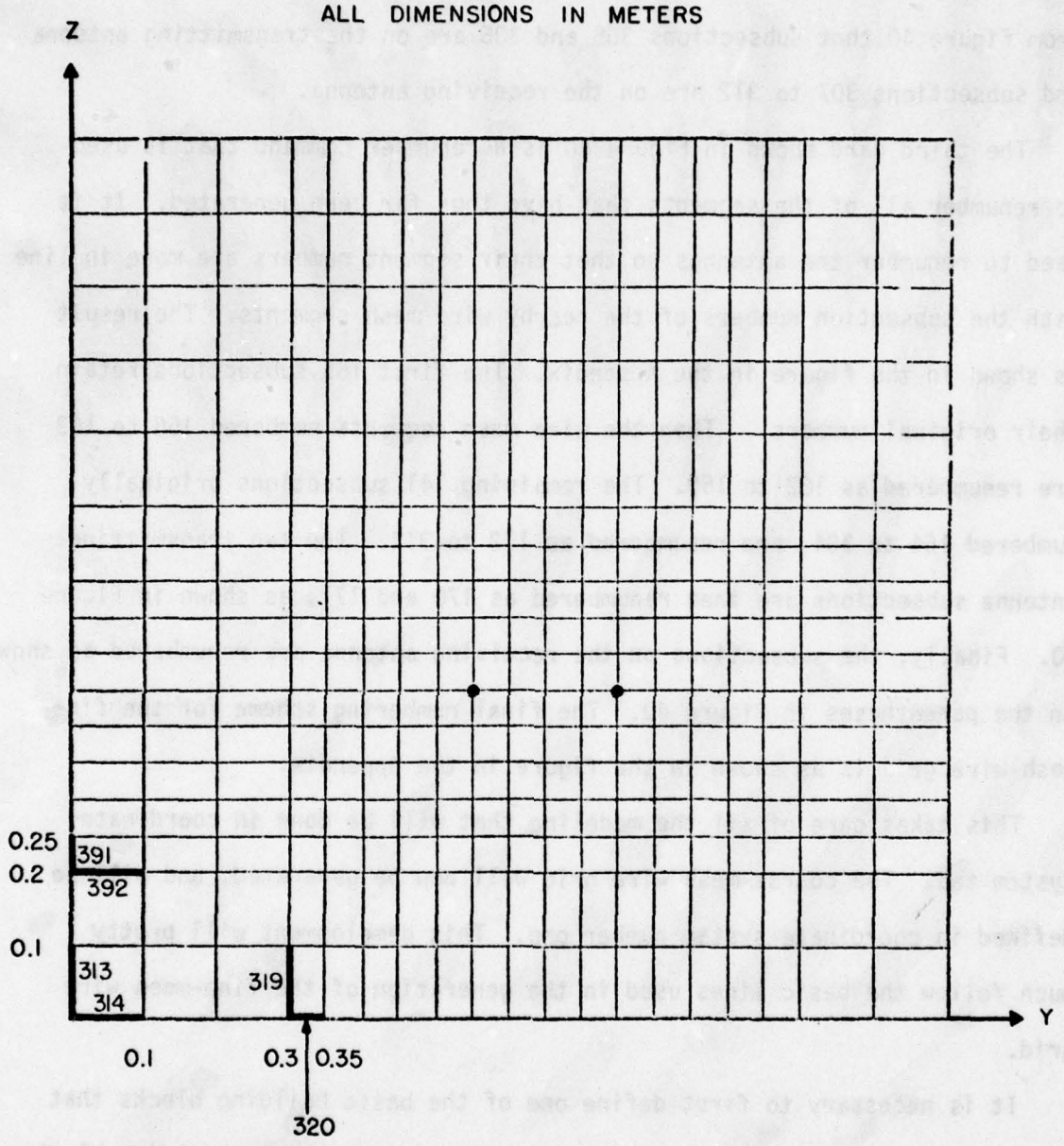


FIGURE 41. BASIC BUILDING BLOCKS FOR COARSE GRID

blocks. The block of interest at this point is GRID2 and is made up of segments 313 and 314.

Figure 42 shows the cards that are needed to generate GRID2 and one of the rows that will be used to develop most of the coarse-mesh grid. This element is called ROW2 and is shown in Figure 43 as the darkened lines. In Figure 42 cards one and two introduce the definitions of ROW2 and GRID2, respectively, the latter being one of the basic building blocks shown in figure 41. Cards three, four and five complete the definition of GRID2, which is made up of subsections 313 and 314. Card six translates these segments twice, generating segments 315 to 318 (Figure 43).

At this point the second of the basic building blocks, GRID3, shown as subsections 319 and 320 in Figure 41, must be defined. This is done in cards seven through ten in a manner identical to the generation of GRID2. This block is then translated by the command in card 11 thirteen times, generating subsections 321 to 346, ending at Y equals 1.0. Cards 12 and 13 again translate GRID2, each one time, to generate segments 347 to 350. The reason two commands are needed is because the displacement distance along the Y-axis changes for each translation.

The element ROW2 is completed by defining subsection 351, which closes off the right-hand side of the element ROW2. This is done in card 14, while card 15 closes the set of cards belonging to the element ROW2 (Figure 43).

The element ROW2 is then translated along the Z-axis one time to generate the second row of elements in the coarse-mesh wire grid. For this row the subsection numbers extend from 352 to 390, as shown in Figure 43.

```
 1        DF  ROW2
 2        DF  GRID2
 3        WR  0.0  0.0  0.1  0.0  0.0  0.0  1  1  1  1
 4        WR  0.0  0.0  0.0  0.0  0.1  0.0
 5        DE  $GRID2
 6        XL  DF  GRID2  2  0.0  0.1  0.0
 7        DF  GRID3
 8        WR  0.0  0.3  0.1  0.0  0.3  0.0
 9        WR  0.0  0.3  0.0  0.0  0.35  0.0
10        DE  $GRID3
11        XL  DF  GRID3  13  0.0  0.05  0.0
12        XL  DF  GRID2  1  0.0  1.0  0.0
13        XL  DF  GRID2  1  0.0  1.1  0.0
14        WR  0.0  1.2  0.1  0.0  1.2  0.0
15        DE  $ ROW2
16        XL  DF  ROW2  1  0.0  0.0  0.1
```

FIGURE 42. GENERATION OF ELEMENT ROW2

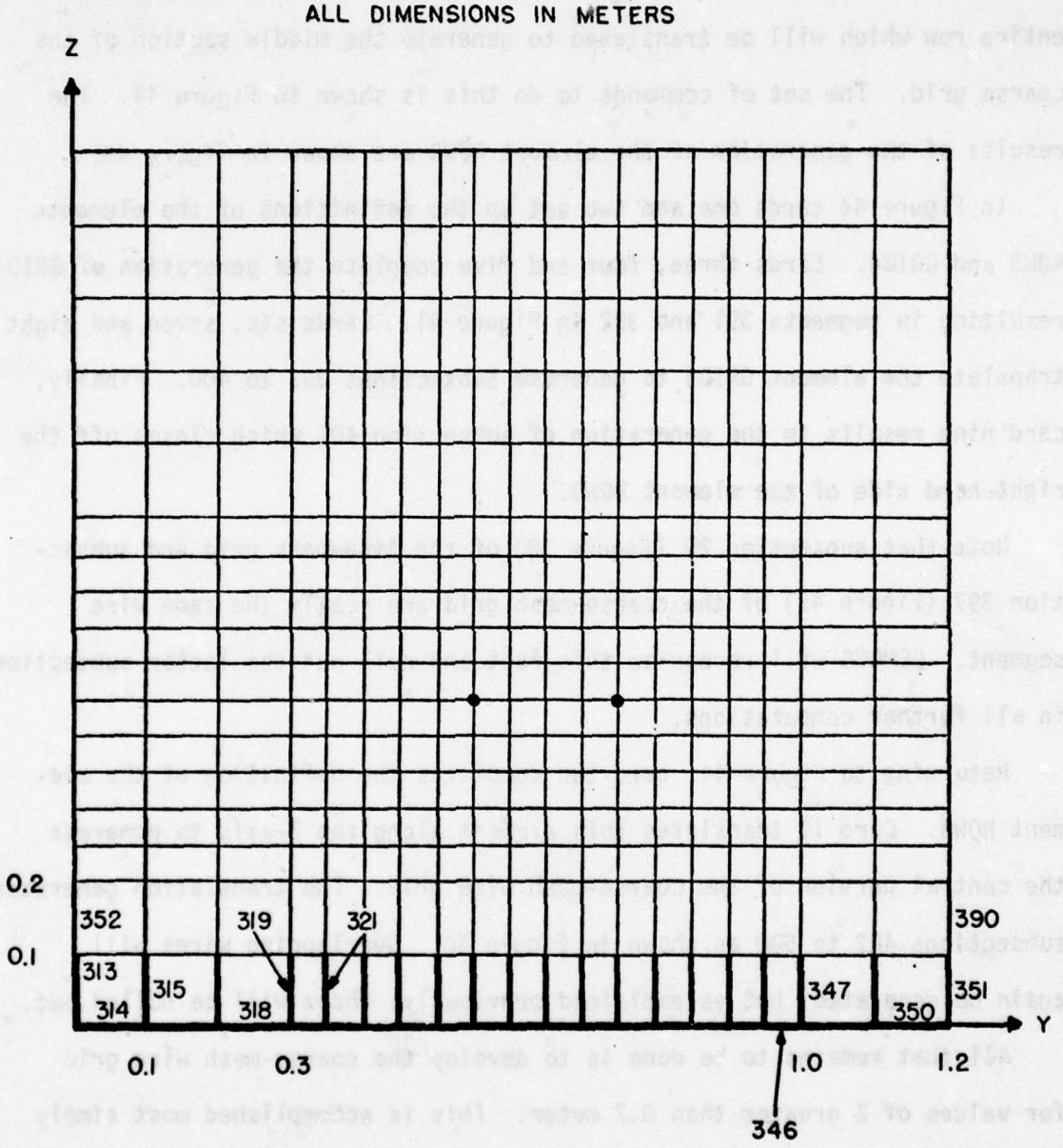


FIGURE 43. DEFINITION OF ROW2

At this point it is time to generate the third of the basic building blocks, GRID4, shown as subsection 391 and 392 in Figure 41, and another entire row which will be translated to generate the middle section of the coarse grid. The set of commands to do this is shown in Figure 44. The results of the generation of the element ROW3 are shown in Figure 45.

In Figure 44 cards one and two set up the definitions of the elements ROW3 and GRID4. Cards three, four and five complete the generation of GRID4 resulting in segments 391 and 392 in Figure 41. Cards six, seven and eight translate the element GRID4 to generate subsections 393 to 400. Finally, card nine results in the generation of subsection 401 which closes off the right-hand side of the element ROW3.

Note that subsection 29 (Figure 38) of the fine-mesh grid and subsection 397 (Figure 45) of the coarse-mesh grid are really the same wire segment. GEMACS will recognize this fact and null out the latter subsection in all further computations.

Returning to Figure 44, card ten completes the definition of the element ROW3. Card 11 translates this element along the Z-axis to generate the central portion of the coarse-mesh wire grid. The translation generates subsections 402 to 500 as shown in Figure 46. Overlapping wires will again be generated; but as explained previously, these will be nulled out.

All that remains to be done is to develop the coarse-mesh wire grid for values of Z greater than 0.7 meter. This is accomplished most simply with the deck of cards shown in figure 47. The result is shown in Figure 48. First, define another element, ROW4, which is identical to the element ROW2.

```
 1        DF  ROW3
 2        DF  GRID4
 3        WR  0.0  0.0  0.25  0.0  0.0  0.2
 4        WR  0.0  0.0  0.2   0.0  0.1  0.2
 5        DE  $GRID4
 6        XL  DF  GRID4  2  0.0  0.1  0.0
 7        XL  DF  GRID4  1  0.0  1.0  0.0
 8        XL  DF  GRID4  1  0.0  1.1  0.0
 9        WR  0.0  1.2  0.25  0.0  1.2  0.2
10        DE  $ROW3
11        XL  DF  ROW3  9  0.0  0.0  0.05
```

FIGURE 44. GENERATION OF MIDDLE SECTION OF COARSE GRID

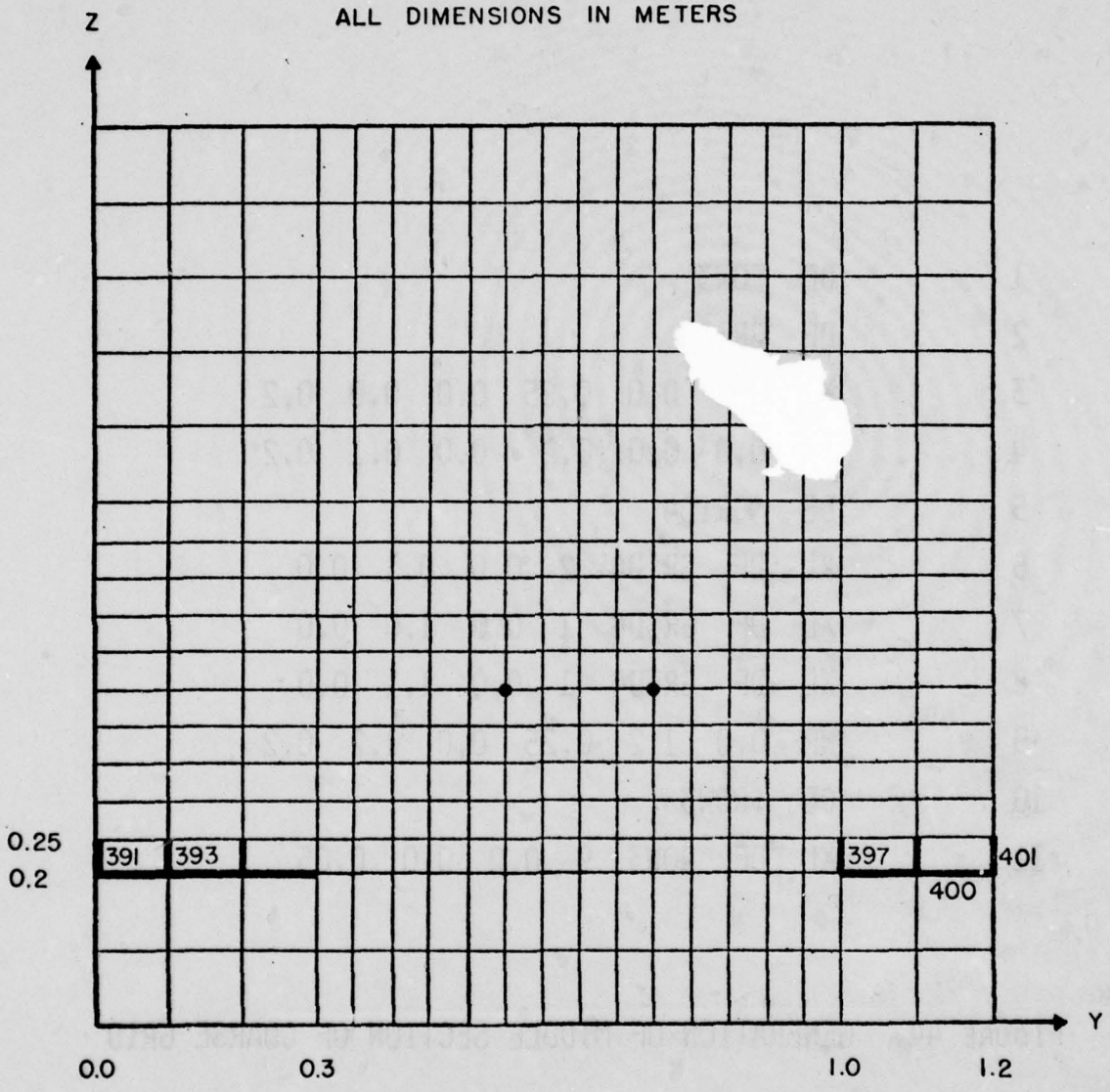


FIGURE 45. DEFINITION OF ROW3

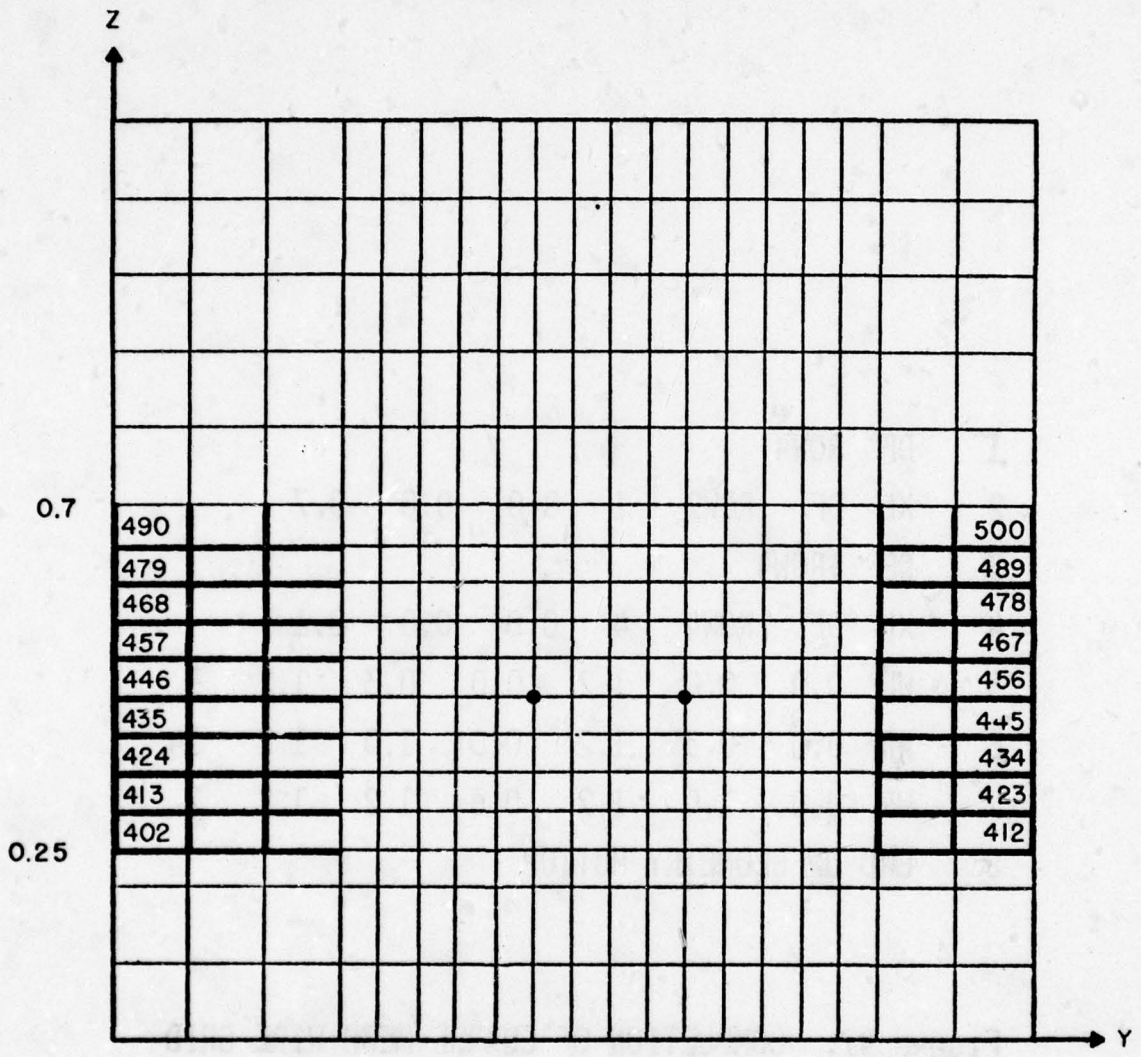


FIGURE 46. SUBSECTIONS OF COARSE MESH WIRE GRID

```
1    DF  ROW4
2    XL  DF   ROW2   1   0.0   0.0   0.7
3    DE  $ROW4
4    XL  DF   ROW4   4   0.0   0.0   0.1
5    WR  0.0   0.0   1.2   0.0   0.3   1.2   3
6    WR  0.0   0.3   1.2   0.0   1.0   1.2   14
7    WR  0.0   1.0   1.2   0.0   1.2   1.2   2
8    END OF GEOMETRY H!JTTOP
```

FIGURE 47. COMPLETION OF COARSE-MESH WIRE GRID

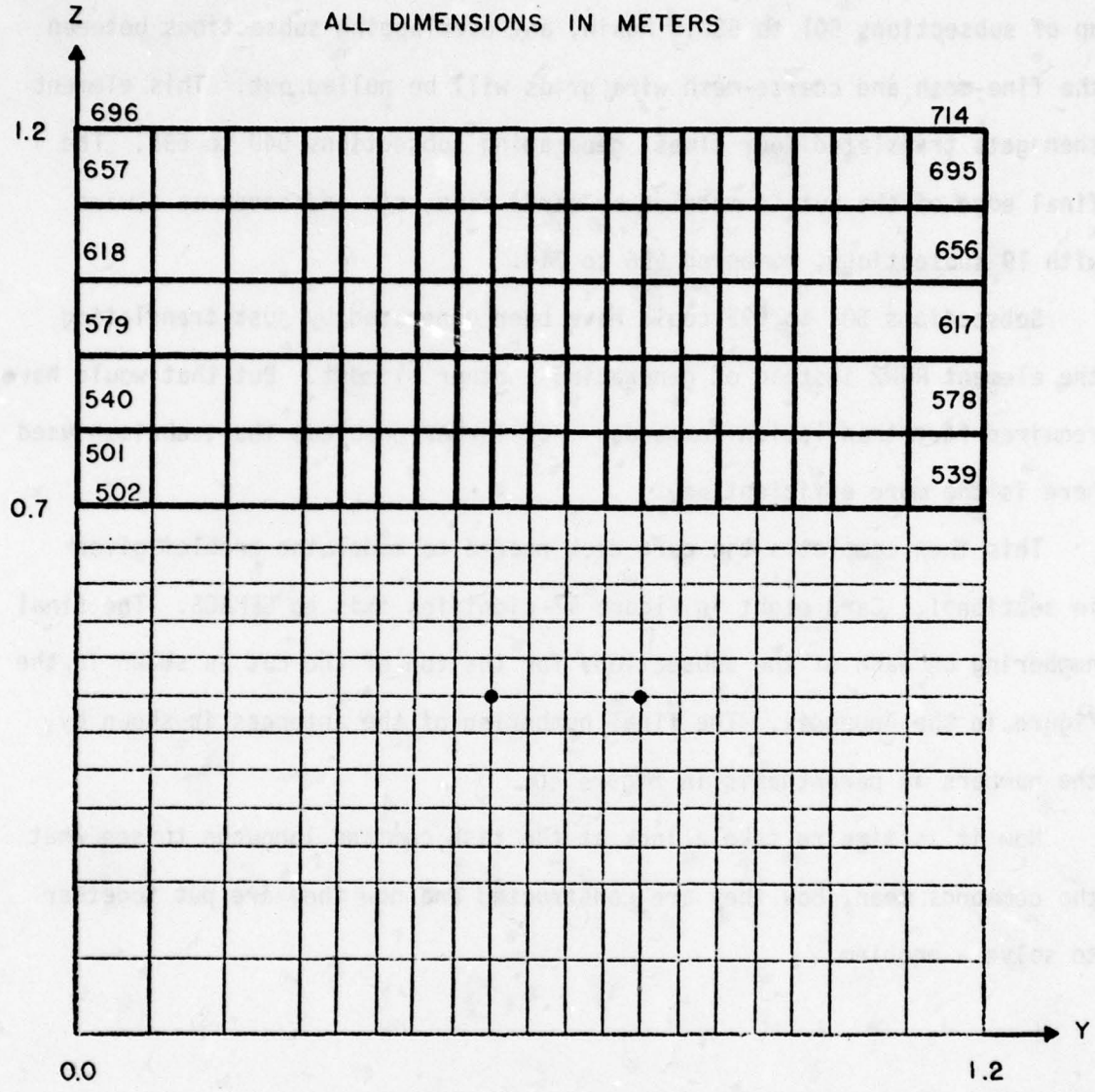


FIGURE 48. FINAL SUBSECTIONS OF COARSE-MESH WIRE GRID

This is accomplished in cards one, two and three. The element ROW4 is made up of subsections 501 to 539. Again, any overlapping subsections between the fine-mesh and coarse-mesh wire grids will be nulled out. This element then gets translated four times, generating subsections 540 to 695. The final edge of the hut is modeled by cards five, six and seven as a wire with 19 subsections, numbered 696 to 714.

Subsections 501 to 695 could have been generated by just translating the element ROW2 instead of generating another element. But that would have required four translation commands. For larger problems the technique used here is the more efficient way.

This then completes the card deck needed to model the problem given in section I. Card eight in Figure 47 signifies this to GEMACS. The final numbering on each of the subsections for the top of the hut is shown in the figure in the Appendix. The final numbering of the antennas is shown by the numbers in parenthesis in Figure 40.

Now it is time to take a look at the task command language to see what the commands mean, how they are constructed and how they are put together to solve a problem.

V. GEMACS COMMAND LANGUAGE

The GEMACS command language is a free-field, keyword-oriented input stream. The order of the items is not important, and the items in each command are delimited by a blank or a comma. An item is considered to be all of the input associated with a particular parameter, such as THETA = 90. Each item may consist of several entries, each entry being called a field. Thus, THETA = 90 has three fields, the variable THETA, the equal sign and the value 90. Blanks may be imbedded between the fields, but cannot appear within a field. If THETA were input as THE TA, the code would set the error flag since two symbols were found when one was expected. The error flag would inhibit execution of the analysis.

All of the commands are read prior to the execution of any command. Errors are noted as they occur, and the existence of an error does not prevent the reading of further commands. All of the input deck is scanned for errors during the submittal of the code for execution.

There are several conventions that should be kept in mind when any of the commands are being set up. These basically are involved with comment cards, comments on cards and continuation cards. Comment cards are those cards that are inserted within a deck to identify the deck or to explain the purpose of commands or items on commands. They are non-executable statements and are ignored by GEMACS once they have been output back to the user in the listing of input deck. They are identified as comment statements by the presence of a dollar sign ($) as the first nonblank character on the card.

In a similar manner short comments may be appended right in the command itself instead of appearing as a separate card. This is done by placing a space, a dollar sign ($) and the comment after the last item in the command. Again, GEMACS does nothing with this portion of the command other than to list it in the listing of the input deck.

The continuation card of a command, or more than one line of input, is signified by the presence of a continuation character in the first column of each succeeding input card for this command. This continuation character is an asterisk (*), and again it must be in the first column of the card.

These are the only basic, general conventions that must be observed in setting up a command stream for GEMACS. Cautions regarding the occurrence of commands before the occurrence of other commands do exist and will be pointed out as they are encountered.

The commands will be discussed pretty much in the order in which they would appear in a typical GEMACS analysis. The general form of the command will be given and those items that may be defaulted will be indicated by the use of square brackets ([ ]) in the figures. The presence of braces ({ }) in the figures indicates that a multiple choice exists for a particular item. The need for a user-defined symbol is indicated by the letter S. If a previously defined symbol is required, this will be shown by the letters DS. If the variable can use either a new or previously defined symbol, this will be indicated by the letters SDS. These codes along with others used in the commands are shown in Figure 49.

The symbolic names that are defined by the user must in all cases

S - PREVIOUSLY UNDEFINED SYMBOL

DS - PREVIOUSLY DEFINED SYMBOL

SDS - EITHER S OR DS

N - NUMERIC VALUE

DSN - EITHER DS OR N

KW - KEYWORD

A - ALPHANUMERIC WORD UP TO SIX CHARACTERS (A-Z AND 0-9) WITH THE FIRST CHARACTER BEING (A-Z)

FIGURE 49. COMMAND CODES

consist of one to six characters, the first character being one of the letters of the alphabet (A-Z). The other five characters may be letters of the alphabet or one of the digits (0-9). An additional restriction is that certain keywords are used by GEMACS for specific purposes (such as default names) and may not be used as symbolic names. These are shown in Figure 50.

There yet remains one final restriction on the definition of symbolic names. Suppose the interaction matrix is labelled XXXXXX, where the X's are legal characters. Then if the matrix is to be decomposed into lower and upper triangular matrices, the user must not define symbols such as XXXLWR, XXXUPR OR XXXPVT since these names will be generated internally by the code to identify the files in which the decomposed matrices are stored. However, as will be shown later, these symbols may be referenced in PRINT, WRITE and PURGE commands.

After the general form of the command there will be shown a specific example of the command geared to the solution of the example problem. A complete list of the commands is given in Table 1 of the GEMACS user's manual [10].

A. Introductory Comments

In setting up a deck of input commands it is best to preface the deck with a series of comment cards that somewhat identify the problem being worked. Since the input deck is the first part of the hard copy printout, this serves to readily identify this particular analysis in a stack of similar outputs. A sample that is being used for the example is shown in Figure 51. Note that for ease of identification the dollar sign has

ONE LETTER KEYWORDS

C D N O R V X Z

TWO LETTER KEYWORDS

CW C1 C2 DM DP DR DT DW DX DY DZ
IS LU NP NR ON P1 P2 R1 R2 SC SW
T1 T2 VS X1 X2 Y1 Y2 Z1 Z2

THREE LETTER KEYWORDS

ABS CDP ECC END FRQ ILP INV LUD OFF PHI RDP
SEQ SET

FOUR LETTER KEYWORDS

AXIS BAND BNDW COND EPSR ESRC
LOOP PLOT PRLC READ SCDP SEGS
SIZE SRDP SRLC TAGS TIME TYPE
VSRC ZGEN ZIMP

FIVE LETTER KEYWORDS

CONJG CPINC CPNUM DEBUG LABEL PARTN
PIVOT PRINT PULSE PURGE SOLVE THETA
TRACE VALUE WRITE

SIX LETTER KEYWORDS

BACSUB CHKPNT COLPSE CONVRG EFIELD EXPAND FILEID GMDATA
LINLIN LINLOG LINPLR LOGLIN LOGLOG LOGPLR MAXITR PCESIN
REDUCE REFLCT REPLCE RSTART SINCOS SYMDEF TRANSP WIPOUT
ZCODES ZLOADS ZMATRX

FIGURE 50. RESERVED SYMBOLS

```
$**************************************************
$**************************************************
$
$    COUPLING BETWEEN TWO MONOPOLES
$    (0.25 AND 0.75 WAVELENGTH) LOCATED ON
$    TOP OF A COMMUNICATIONS HUT.  THE SHORTER
$    ANTENNA HAS A SOURCE OF ONE VOLT.  THE
$    OTHER IS LOADED BY 50 OHMS.  FREQUENCY
$    EQUALS 750 MHZ.
$
$**************************************************
$**************************************************
```

FIGURE 51. SAMPLE INTRODUCTORY COMMENTS

been indented to column 3, and the text is set off by the use of lines of asterisks.

B. Global Parameters

The first executable commands usually consist of up to five cards which fix some of the permanent parameters of the analysis and avoid the repetition of these values on subsequent commands. This specifies the number of files available for use by GEMACS, the amount of CPU time in minutes allocated for the job, the frequency of interest in MHz at which the analysis is to be performed, and the conductivity and permittivity of the ground plane, if the structure is not perfectly conducting or if the antennas are not located in free space.

For this particular problem the three cards shown in Figure 52 would be used. If the top of the hut were a finite conductor, then the conductivity and permittivity would be specified using the symbols COND and EPSR, respectively. The conductivity is in mhos/meter and EPSR is the dielectric constant relative to that of free space.

Note that when NUMFIL is set to 17 this number of files must be made available since GEMACS has no way of checking on this. Two of these files are for input and output and their numbers are installation dependent. They are presently set as five for input and six for output. File seven is the checkpoint file. Files one through four are used for internal scratch space. Files eight through 17 (in this case) would be used to store data sets as identified by symbols defined by the user. Examples of such sets would be the currents on the subsections, the interaction matrix and the excitation matrix.

The TIME specification is useful when one has no idea how long the

```
NUMFIL = 17  $ NUMBER OF FILES AVAILABLE

TIME   = 570 $ NUMBER OF CPU MINUTES ALLOCATED

FRQ    = 750.0 $ FREQUENCY OF ANALYSIS IN MHz
```

FIGURE 52. SAMPLE GLOBAL PARAMETERS

analysis will take. What happens is that when the CPU time used by the job equals or exceeds the amount of time specified on the TIME card, GEMACS will automatically checkpoint and terminate the analysis thus saving the results of the work performed up to that point. Nothing will be lost or duplicated when a restart is performed.

However, the time specified on the TIME card must be less than that specified on the computer system job control card. Otherwise, when this latter time is reached, the system will terminate the analysis; and the results of the work done since the last checkpoint will be lost. It will be possible to restart, but some of the work performed during the previous execution will be duplicated.

The last of the global parameters specified in Figure 52 is the frequency in megahertz at which the analysis is to be performed. The symbol for this parameter is FRQ. Its specification here means that this item can be defaulted on all commands requiring this parameter.

C. Geometry Generation

The next executable command will read in the geometry data set. This command is shown in Figure 53. The symbol HUTTOP has been chosen to label the geometry data set. In addition, the system card reader is the file from which the data may be obtained. This is the default value for LU, the logical unit. Keep in mind that if the file name had been defaulted, then the item for the geometry data set may not appear in any subsequent command; that is, it must always be a default parameter if it is defaulted in the GMDATA command. The actual commands that are used to generate the geometry are explained and illustrated in section IV of this manual.

GMDATA [= S] [LU = N]

GMDATA = HUTTOP

FIGURE 53. GEOMETRY GENERATION COMMAND

D. Electrical Environment

The electrical environment of the structure to be analyzed should be specified next. This would include the effects of loads, the excitation (electric fields or voltage sources or a combination of both) and the ground parameters. These last are usually specified with the set of global parameters, as COND and EPSR (Section V.B). The frequency at which the analysis is to be performed is also usually specified along with the global parameters, although it can also be specified on a number of other commands. Once it is specified, it may then be defaulted on all subsequent commands. The surest way to avoid error is to specify the frequency as a global parameter and then default it in all subsequent commands requiring the frequency as an input. As shown in Section IX, the frequency may also be updated as a global parameter by use of an arithmetic statement.

Figure 54 shows the general form of the command used to specify the loading, as well as the specific form used in the analysis of the sample problem. This command allows one to place loads on the structure identified in the GMDATA item (HUTTOP in this case). See Section V.C. The load data will be stored in the symbol SDS, which is LOAD in this case.

The type of load and the value of the parameters are specified by choosing one of the possibilities in the large braces. COND is used to specify the conductivity of the subsections in units of mhos per meter. ZIMP is used to specify a lumped load impedance in ohms. Two values are required for this item, the resistance and the reactance. In the example problem there is a purely resistive load of 50 ohms. Hence the input is 50.0, 0.0. Note that the values given for COND or ZIMP are independent of frequency.

$$\text{ZLOADS} = \text{SDS} \quad [\text{GMDATA} = \text{DS}] \quad \left\{ \begin{array}{l} \text{COND} = \text{DSN} \\ \text{ZIMP} = \text{DSN, DSN} \\ \left\{ \begin{array}{l} \text{PRLC} \\ \text{SRLC} \end{array} \right\} = \text{DSN, DSN, DSN} \end{array} \right\} \quad \left\{ \begin{array}{l} \text{TAGS} \\ \text{SEGS} \end{array} \right\} = \text{N, N, ..., N}$$


```
ZLOADS = LOAD  GMDATA = HUTTOP  ZIMP = 50.0,0.0  SEGS = 156
```

FIGURE 54. LOAD SPECIFICATION COMMAND

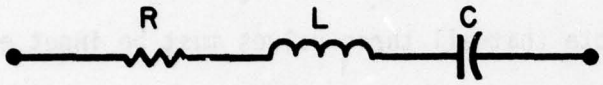
Individual circuits, which are functions of frequency, may also be specified with equivalent overall values of resistance (R), inductance (L) and capacitance (C), in that order of specification. Either parallel or series circuits may be used, as shown in Figure 55. The units are ohms for R, millihenries for L and microfarads for C. A parallel or series circuit is input as PRLC or SRLC, respectively, as shown in the inner braces in Figure 54. Note that all three values must be input even if one or more of the quantities does not exist in the equivalent circuit. The inductance and capacitance are functions of frequency in this case.

The TAGS/SEGS item identifies those subsections which are to be loaded with the values specified in this command. In this case segment 156 is to be loaded with 50 ohms. As can be seen from Figure 40, this is the bottom subsection of the receiving antenna.

If other subsections were to have this same load, then their segment identifiers would also be listed, separated by commas or blanks. If consecutive segments are to be loaded with the same value, then only the first and last segment numbers need be listed, separated by a hyphen. Thus, the TAGS/SEGS item could be input as SEGS equals 156-161, since there are 6 subsections on this antenna, if all the subsections were to be loaded. The alternative would be to list all the segment numbers separated by commas or blanks (e.g., 156, 157, ..., 160, 161).

As mentioned in the beginning of Section IV, the tag is simply another way of identifying a group of subsections. Unlike the subsection number it need not be a number unique to each segment. As a matter of fact, all 714 subsections in HUTTOP may have the same tag, and that tag may be zero

SERIES



PARALLEL

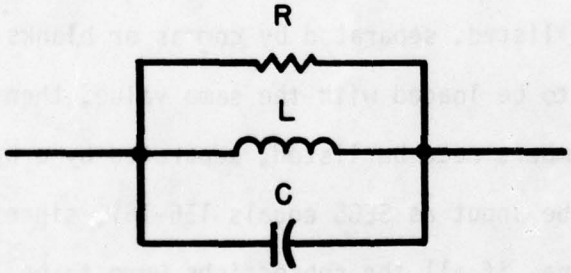


FIGURE 55. EQUIVALENT CIRCUITS FOR LOAD SPECIFICATION COMMAND

or any other number. The tag may thus be used as a convenient way to reference a group of subsections, all of which have the same load, or all of which have the same excitation, or all of which belong to the same antenna; or any other division devised by the user. It can be thought of as a flag identifying a group of subsections all of which have a common characteristic. Therefore, when the geometry was modeled, segment 156 was given a tag of four; and the identifier in the command could have been TAGS = 4 instead of SEGS = 156, but only if segments 157 to 161 were given some other tag since they are not loaded in this problem.

Note that more than one ZLOADS command may appear in the command stream, but each command must reference the same load data set (LOAD in this case). Also more than one type of load may be placed on the same subsection. Finally, and very importantly, all the ZLOADS commands must precede the command which causes the interaction matrix to be generated. This is necessary since the symbol for the load data set is a required item for input on the interaction matrix generation card (Section V.E).

Figure 56 shows the general form of the command used to specify the antenna excitation, as well as the specific form used in the analysis of the example problem. As can be seen, it is quite similar to the load specification command shown in Figure 54. The file for the excitation data is labeled as SOURCE. The keyword VSRC (voltage source) is a flag to the code that an excitation is being placed directly on one or more subsections, as opposed to illuminating the entire structure with an incident electric field. This latter can also be done with GEMACS, and its command is explained in Section VIII.

SDS = VSRC [(DS)] [FRQ = DSN] V = DSN, DSN $\left\{\begin{matrix}\text{TAGS}\\\text{SEGS}\end{matrix}\right\}$ = N,N,...,N

SOURCE = VSRC (HUTTOP) V = 1.0,0.0 SEGS = 170

FIGURE 56. EXCITATION SPECIFICATION COMMAND

The geometry data set is referenced in the parentheses following the keyword. Recall that this would be defaulted if the default name were used in the geometry generation command (Section V.C).

The frequency of the analysis may also be defaulted if it has been specified in the list of global parameters or on a previous command. Since it is a global parameter in the example, it is shown defaulted in the example in Figure 56.

The value of the voltage to be placed across the terminals is specified as a complex number, giving the real and imaginary parts of the excitation. In almost all cases the voltage is given as 1.0, 0.0 since the electromagnetic fields problem is a linear one. In other words, if the excitation were increased by some factor, say two or ten, then the resultant field at some point in space would be increased by the same factor, for example two or ten. The same is true for the calculation of coupling. If the input voltage is increased by a factor of five, the received voltage would be increased by a factor of five; but the coupling (the ratio of received to excitation voltages) would remain the same for either value of excitation. Hence, there is no loss in generality by always specifying an input voltage of 1.0 + j0.0 volts.

The last item in this command is the specification of which subsections are excited. This is done in exactly the same manner as for the load specification command. The excitation in this case is placed across subsection 170 (Figure 40). Since this subsection has been given a tag of 3, the item could be TAGS = 3 instead of SEGS = 170, if subsection 171 had some other tag. The same comments apply regarding multiple cards. All

excitation specification commands must refer to the same data set (SOURCE); and all subsections with the same excitation can be listed on the same command, the subsection or tag numbers being separated by commas. If consecutively numbered subsections have the same excitation, only the first and last subsection numbers need be given, separated by a hyphen. Refer to the discussion of the load specification command for examples of these input conventions.

There is one point to keep in mind when using this command or the electric field excitation command (ESRC, Section VIII). If the frequency at which the analysis is being performed is changed (as, for example, in a parametric study, Section IX), then the excitation matrix is reinitialized to zero. Consequently, a new VSRC or ESRC command must be input even though the excitation on each of the subsections remains the same.

E. Interaction Matrix Generation

The general and specific forms for this command are shown in Figure 57. The keyword SINCOS is at this point in time an inflexible quantity indicating the expansion set that is used to represent the current, much like a Taylor series is used to approximate some mathematical function such as the exponential or logarithmic functions. Other expansion functions will be incorporated into future versions of GEMACS and different keywords will be used, but for the present time only SINCOS may be entered for this item.

Once again the name of the geometry data set must be entered if the default name has not been used in the geometry specification command.

The frequency is also an item in this command that may be defaulted.

```
ZGEN  SINCOS  [GMDATA = DS]  [FRQ = DSN]  ZMATRX = S
              [ZLOADS = DS]  [COND = DSN]  [EPSR = DSN]

ZGEN  SINCOS  GMDATA = HUTTOP  ZMATRX = ZIJ  ZLOADS = LOAD
```

FIGURE 57. INTERACTION MATRIX GENERATION COMMAND

Since this was included as one of the global parameters, it is not included on the card for the sample problem input data deck.

The next item is a required item and identifies the file in which the elements of the interaction matrix will be stored. Although up to six characters may be used for a symbolic name, ZIJ is being used since the entries in the interaction matrix are referred to as the "zij's", which is short for the ijth element of the Z-matrix. Reviewing the MOM matrix equation, (Figure 5), it will be seen that this is the matrix which is multiplied by the I-matrix (current matrix) to yield the excitation matrix (E-matrix).

The next item of this command is the identification of the load matrix. Remember that this symbol was defined on the load specification command (Figure 54). This item is shown as one which may be defaulted, but this is true only if no loads have been placed on any of the subsections. If a load matrix has been defined, its symbol must appear in this command.

The next two items of this command specify the conductivity and permittivity of the ground plane, if one exists. Again, it is recommended that these quantities be specified in the set of commands which define the global parameters and should be defaulted here. The quantities then need to be specified only once; and, since they are located in one well-defined area of the input deck, they can be quickly found and modified as needed.

At this point the global parameters and the data for the geometry, for the load and the excitation have been input into GEMACS. From these data

GEMACS has generated the elements of the interaction matrix and the excitation matrix. It is now ready to solve the basic MOM matrix equation (Figures 7 and 8).

F. Matrix Equation Solution

Two solution techniques presently exist within GEMACS, and it is anticipated that more will be added to the code. These are a full-matrix solution technique, in which a Gauss-Jordan algorithm is used; and the banded matrix iteration (BMI) technique, in which the interaction matrix is banded, decomposed and used in an iterative scheme to solve for the currents on the geometry. The former technique is discussed in Section VII. The BMI technique (Figures 7 and 8) is discussed in this section.

The first step is to develop a banded matrix from the interaction matrix [Z]. The object is to get all of the larger elements into this banded matrix. In addition, one must also choose those elements that significantly contribute to the electromagnetic phenomena associated with the problem, such as edges and other nearby perturbing influences. This is not an easy task; and about the only guide available is past experience, which at this point is next to nonexistent. A bandwidth (M in Figure 7) is selected. It should be large enough to assure convergence to a reasonably accurate solution and yet small enough to obtain that solution in a practical amount of time.

It is better to choose a large bandwidth in order to have GEMACS yield faster convergence than to have a smaller bandwidth requiring a large number of iterations [5,pp. 3-5].

In the example all of the interactions involving those elements in coordinate system number two are placed within the banded portion of the

interaction matrix. This includes subsections one to 312, which represent the antennas and the top of the hut. To these are added those subsections which border the edge of the hut in the vicinity of the antennas. Looking at the figure in the Appendix, this includes the first 500 subsections. The bandwidth (that is, the number of diagonals on either side of the main diagonal) is therefore 499, which is rounded off to 500.

These are the data that are needed to construct the BAND command, shown in Figure 58. The symbol DS represents the name that has been assigned to the interaction matrix. This is "ZIJ" in the example. The value for BNDW is the number of diagonals on one side or the other of the main diagonal that will appear in the banded matrix, whose symbol is given by SDS in the general form of the command. Therefore, in the example the value of N is 500.

A restriction needs to be kept in mind when naming the banded matrix. The same symbolic name may not be used for both the banded matrix and the original interaction matrix. The reason is that both matrices are referenced in the command that implements the BMI technique and some means must be available to differentiate between them.

Therefore, for the example the symbol BNDZIJ has been chosen since it is almost an acronym for "banded ZIJ matrix." The final form of the command that will be used in the solution of the problem is also shown in Figure 58.

Once the interaction matrix has been banded, the banded portion must be decomposed into lower and upper triangular matrices as shown in the

```
SDS = BAND (DS)    BNDW = N

BNDZIJ = BAND (ZIJ)  BNDW = 500
```

FIGURE 58. THE BAND COMMAND

solution process indicated in Figure 8. This is accomplished by using the matrix decomposition (LUD) command shown in Figure 59.

The symbol DS is the name of the banded matrix that has been developed, in this case BNDZIJ. The symbol for the decomposed matrix may be either a new name or some previously defined name. Also, in the case the name of the decomposed matrix may be the same as the name for the banded matrix. This is the method usually chosen, and that is what is in Figure 59.

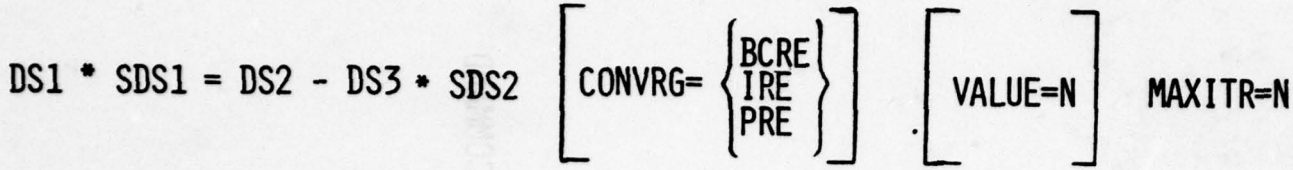
Once the banded matrix has been decomposed, the BMI command is used to solve the MOM matrix equation for the current on the wires in the grid model. The form of this command and the specific command to be used to solve the example are shown in Figure 60. In this command DS1 represents the name of the decomposed banded matrix, BNDZIJ in this example. The currents that are to be calculated are stored under the name represented by SDS1, which in the example is CURENT. The excitation that has been set up is stored in DS2, or SOURCE in the example. The symbol DS3 is the name of the interaction matrix that was calculated. Recall that it is ZIJ. The final symbol in the equation is SDS2, the set of currents from the previous iteration. It may be the same as SDS1. An initial value may be pre-loaded in SDS2 (Section V.K), but it has been found that leaving the initial currents on the subsections equal to zero does not significantly decrease the solution time. Thus, the array for the currents is usually not preloaded and SDS2 is normally the same as SDS1.

The next item, CONVRG, is the convergence criterion that will be used to test the currents that have been derived at each iteration. At this time

```
SDS = LUD (DS)

BNDZIJ = LUD (BNDZIJ)
```

FIGURE 59. MATRIX DECOMPOSITION COMMAND



BNDZIJ*CURENT=SOURCE-ZIJ*CURENT MAXITR=10

FIGURE 60. BANDED MATRIX ITERATION COMMAND

GEMACS has three convergence tests built into it. The pros and cons of each of these tests have been discussed in the BDM quarterly reports [6,8] and are touched on briefly in the engineering manual [9]. Basically they are the Boundary Condition Relative Error (BCRE), the Iterative Relative Error (IRE) and the Predicted Relative Error (PRE). The BCRE compares the boundary condition obtained using the calculated currents to the original excitation. The IRE compares the currents at one iteration to those obtained from the previous iteration, convergence being attained when there is a very small change in the value of the new set of currents. The PRE is a function of the IRE and is based on the fact that the latter can be represented in an approximate fashion by a exponential function, especially when the convergence rate is fairly rapid. The PRE was found to yield more useful data relating to convergence; and, as such, was felt to be the best of the three available convergence tests. As a consequence, the default value for this item is the PRE. This is the one used in the solution of the example problem.

The next item is the value of the convergence criterion in percent at which the iteration process will terminate. The values of each of the convergence criteria are computed at each step of the iteration process according to equations 73, 74 or 76 of the engineering manual [9]. For the predicted relative error (PRE) it is basically a function of the ratio of the current vector at one iteration to the current vector at the preceding iteration.

The choice of VALUE is dependent on the problem and what quantity is being investigated. If it is the electric field far from the antenna, a

value of ten percent will yield good results. When working close to the antennas, as in the example, the VALUE item should be given its default value, which is one percent. This requires more iterations, but the nature of the problem necessitates the acceptance of the increased cost.

The final item in this command is the maximum number of iterations that will be allowed. It may happen that the way the grid is set up or the choice of the bandwidth casues the solution to converge very slowly. In that case an answer will be obtained eventually, but at a high cost. The MAXITR item provides a check which will not allow GEMACS to grind away forever. Suppose, for example, that the value of MAXITR is ten. If convergence is not reached after the tenth iteration, GEMACS will terminate the analysis and print out the values of current at that point. It will also output the value of the convergence criterion in percent and predict how many more iterations are required. Based on this information the value of MAXITR can be changed if not too many more iterations are required or the bandwidth may be increased in an attempt to obtain a faster rate of convergence; or, as a last resort, a new wire grid model may be set up. The analysis is then restarted from the last checkpoint Section V.H).

Note that this item cannot be defaulted. A nominal value to use as a starting point is ten. The form of this command that is used in the analysis of the problem is shown in Figure 60.

At this point GEMACS has now calculated the current on each of the subsections. These data are all that are needed to solve the coupling problem. Therefore, the next operation is to have GEMACS print out these

current values. The commands that are used for control of the output are discussed in Section VI.

G. Checkpoint and Example Problem Input Listing

It has been mentioned several times that GEMACS has a checkpoint and restart capability. This feature allows the user to restart his analysis at any point in the stream if it has been interrupted for any reason whatsoever. The importance is that the user may restart from very nearly the point at which the analysis had left off, thus eliminating a lot of costly duplication.

In order to make use of the restart option one or more checkpoints must have been taken during the analysis by using the checkpoint command. When a checkpoint is taken, a snapshot of the computer status is stored away on some peripheral device or file, such as a magnetic tape. All of the data input to or generated by GEMACS are stored in an orderly manner on this file. In this way the status of the analysis at that point in time is saved; and it is available for later use as a starting point from which a modified analysis stream may be begun.

The checkpoint command is shown in Figure 61. Since most of the items can be defaulted, it is generally a very simple input card to generate. The symbol CHKPNT signifies the type of command. The first item (LU) designates what file number the checkpoint tape or disc is loaded on. This is one of those files that must be accounted for when specifying the number of files as a global parameter (NUMFIL in Figure 52). As a rule one uses the default value which puts the checkpoint storage device on logical unit seven.

```
CHKPNT  [LU = N]  [FILEID = A]  [CPINC = N]  [NR]

CHKPNT  CPINC = 60

CHKPNT
```

FIGURE 61. THE CHECKPOINT COMMAND

The FILEID item can be used to label the checkpoint file with an identifier significant to the job. This is useful when several checkpoint files are floating around. GEMACS compares this name with the one that is given in the restart command (Section V.H); and if the two differ, a warning is printed out advising the user to double check to make certain that the correct checkpoint data is being used. The default name for the file in this and the restart command is CHKPNT.

The next item, CPINC, specifies how often the checkpoint is to be taken, counting in minutes of computer processing time (not wall clock time). This is the actual time that the computer is working on the problem. For longer problems it is recommended that a checkpoint be taken every 60 minutes. Taking a checkpoint on a large problem may require five minutes; and if taken too often, a large percentage of the time could be spent just transferring data to the checkpoint file. Separating the checkpoints by too great a time runs the risk of termination a long time after the last checkpoint, which may cause expensive duplication of all the operations since that last checkpoint was taken.

If this item is defaulted, an immediate checkpoint is taken regardless of what task has just been completed. For reasons to be explained shortly, a checkpoint should be taken just after the interaction matrix has been completely generated. In this way a record of the completed matrix will be available if a change in the bandwidth is desired. Remember, the generation of the interaction matrix is an expensive process.

The final item, NR, provides control over whether or not an historical record of the checkpoints taken is kept. When the symbol "NR" is typed

on the card, GEMACS will not rewind the checkpoint file. The next checkpoint will start right after the last record of the previous checkpoint. This is true whether the checkpoints are taken at specified intervals of time or taken immediately. When restarting, any checkpoint taken during the course of the analysis will be available for access.

If this item is defaulted, then after each checkpoint is taken the file is rewound and the subsequent checkpoint will then start at the beginning of the file. In effect one is wiping out checkpoints as new data are stored on the file. The disadvantage of this course of action is that when restarting one can access only the last checkpoint written. Rewinding and overwriting save storage space, but they limit the flexibility when restarting.

Sample checkpoint commands are also shown in Figure 61. The examples are identical to the commands used in the analysis of the sample problem. In the first example, it is specified that a checkpoint will be taken after every 60 minutes of computer processing time. Note that the file is to be rewound before each new checkpoint is taken. The historical record will be lost, but storage space will be saved; this will be significant since there are so many elements in the interaction matrix. Such a timed checkpoint command should appear early in the task command input stream. In the example it is just after the read geometry (GMDATA) command, as shown in Figure 62.

The second example in Figure 61 shows what an immediate checkpoint command would look like. The default values for the file name and device number have again been used, as well as a default for the "NR" parameter.

```
$*******************************************************************************
$*******************************************************************************
$
$   COUPLING BETWEEN TWO MONOPOLES (0.25 and 0.75) LOCATED ON
$   TOP OF A COMMUNICATIONS HUT.  THE SHORTER ANTENNA HAS A
$   SOURCE OF ONE VOLT.  THE OTHER IS LOADED BY 50.0 OHMS.  THE
$   FREQUENCY EQUALS 750.0 MHZ.
$
$*******************************************************************************
$*******************************************************************************
NUMFIL=17
TIME=570
FRQ=750.0
DEBUG ON ILP
GMDATA=HUTTOP
DEBUG OFF
CHKPNT CPINC=60
ZLOADS=LOAD GMDATA=HUTTOP ZIMP=50.0,0.0 SEGS=156
SOURCE=VSRC(HUTTOP) V=1.0,0.0 SEGS=170
ZGEN SINCOS GMDATA=HUTTOP ZMATRX=ZIJ ZLOADS=LOAD
CHKPNT
BNDZIJ=BAND(ZIJ) BNDW=500
BNDZIJ=LUD(BNDZIJ)
CHKPNT
BNDZIJ*CURENT=SOURCE-ZIJ*CURENT MAXITR=10
PRINT SOURCE CURENT
END OF COMMAND STREAM
```

FIGURE 62. EXECUTION LANGUAGE INPUT STREAM

Finally, no input is specified for the increment parameter. Therefore, the checkpoint will be taken as soon as this command is encountered in the execution sequence. As can be seen from the sample problem input in Figure 62, checkpoints are to be taken as soon as the interaction matrix has been generated and just after the banded matrix has been decomposed.

Whenever a checkpoint is taken during the analysis, a standard message is printed on the GEMACS output. An example of this is to be seen on page 20 of the Appendix. The first line gives the unique numerical identifier of the checkpoint and the CPU time at which the checkpoint is being started. There then follows a listing of the names of the common areas being stored on the file as well as the number, name and size of all peripheral files that have been generated during the analysis. The last part of the printout gives the CPU time at which the checkpoint was completed, the length of time it took to take the checkpoint and the size of the checkpoint file in number of words written. All times are given in minutes of computer processor time (CPU).

With the exception of the output commands (Section VI) a discussion of all the execution commands that are included in the original input to perform the coupling analysis using the BMI technique has been completed. The complete execution command input stream is shown in Figure 62. This is pretty much the way that the commands would be sequenced in order to obtain the currents that are flowing on any structure. Still other commands are available, and they are discussed in the following four subsections and sections VI through IX.

H. Restarting

When analyzing a structure which requires a large number of subsections, it is necessary to divide up the process into several sittings or "runs". This means that the job must be submitted more than once possibly because other jobs require the computer resources being used. The restart command initiates a run someplace in the middle of the analysis instead of starting over from the beginning each time. This is made possible by using the data that have been stored away on the checkpoint tape.

The format for the restart command is shown in Figure 63. The symbol identifying the type of the command is RSTART. The next two items, LU and FILEID, are the same as those in the checkpoint command (Section V.G). If not, then the entries for these two items must be identical in both types of commands.

The last item in this command, CPNUM, specifies the checkpoint number from which the restart is to begin. As explained in Section V.G., each checkpoint is given an unique numerical identifier. This is true whether the file is rewound between checkpoints or not and whether it is a timed checkpoint or not. It is this identifier that is input for the CPNUM item. As can be seen in the sample problem output on page 22 of the Appendix, the checkpoint number immediately precedes the description of the data that are being stored on the file. The rest of the description includes a list of the common blocks read from the checkpoint file together with their sizes, followed by a list of the names and sizes of all the peripheral files. If a name were given to the checkpoint file, it would

```
RSTART    [LU = N]    [FILEID = A]      CPNUM = N
```

```
RSTART  CPNUM = 4
END OF COMMANDS
```

FIGURE 63. THE RESTART COMMAND

appear before the number of this file and the number of the checkpoint.

The example shown in Figure 63 shows what the total execute command list would look like if one were simply restarting from some point in the analysis after an execution was prematurely terminated. It would contain only the restart and end (Section V.J) commands.

It should be pointed out that one logical unit can be used to restart from and a second unit used to take subsequent checkpoints on. In this way the integrity of the data on the original checkpoint file can be maintained. To do this some logical unit (LU) other than seven must be specified in the restart command and a clean storage file loaded on seven to accumulate later checkpoint data. A good alternate to seven would be three or four, since at present these are unused.

In addition to simply restarting one can modify the original input stream; for example to increase or decrease the bandwidth with which the BAND and BMI commands are to work. This is done by using the WIPOUT command, whose format is shown in Figure 64. The symbol WIPOUT indicates the type of command, and the N's are the command sequence numbers to be wiped out. Note that there is a difference between card numbers and command sequence numbers. The comment cards shown in Figure 62 are cards one through ten, but they are not commands. Card 11 contains command one, and card 25 is command 15. Also if a command requires one or more continuation cards, the continuation cards do not increase the command sequence number.

The first example in Figure 64 illustrates the process of eliminating particular commands from the original task list. Any number of individual tasks may be deleted in this manner.

```
WIPOUT    N N ... N

    —OR—

WIPOUT    N N   99999


WIPOUT    3  6  12        (FICTITIOUS)

WIPOUT    12 99999
```

FIGURE 64. THE WIPOUT COMMAND

There is a shortcut for the WIPOUT command, and is also shown in Figure 64. The number 99999 has the significance that all commands, starting with the command whose sequence number is equal to the value of the N preceding the number 99999 down to, and including, the WIPOUT command are all deleted. To see this, think of the list of commands as generating a task list. Such a list can be seen on pages A-10 and A-11 of the Appendix. An explanation of this list is to be found in Section VI.E. The commands in the restart also generate a list which is appended to the bottom of the original list. If the N were 12 then all the tasks from 12 through the wipe out command would be eliminated. The second example in Figure 64 shows such a command.

The use of this second example in the input stream for the sample problem (Figure 62) would eliminate all the commands below the first immediate checkpoint; that is, from the BAND command down. The WIPOUT command could then be followed by a new BAND command, specifying a different bandwidth. This new command would then need to be followed by a sequence of commands identical to the original sequence since that original sequence was deleted. One command cannot be replaced by a new command and not disturb the sequence following the command that was replaced. The original command would be deleted and the replacing command put out of place at the end of the modified task list. This is further explained in Section VI.E.

In the second part of the sample problem shown in the Appendix, the analysis is restarted after the interaction matrix has been generated. This is shown on page A-48 of the Appendix. A full matrix solution process (Section VII) is then implemented to solve for the subsection currents.

There is one more point to keep in mind. If the operation has already begun and not been completed, its command cannot be wiped out. As can be seen from the sample output, the first checkpoint occurred while the interaction matrix generation command (ZGEN) was being executed. It could not have been deleted in the restart. This is one advantage of the immediate checkpoint as used in the sample input stream. The full matrix has been generated and stored , and the actual solution process has not started. Thus, a solution process different from the original can be used on a restart. As a matter of fact, this is what was done starting on page 49 of the Appendix. Also the bandwidth can be easily changed on a restart since the original BAND command has not yet begun to be executed.

I. PURGE

In order to save as much storage space as possible, all files and data that will no longer be used in the analysis should be eliminated. This is accomplished by the use of the PURGE command, shown in Figure 65. This also has the advantage of making the checkpoint file shorter and reducing the time it takes to transfer the data to this file.

The command itself is quite simple. It consists of the mnemonic PURGE, followed by the list of symbols whose files are to be purged. One example that is often used is to purge the interaction matrix after it has been decomposed. This is done only when using the full matrix solution. It cannot be done when using the BMI solution process until after the currents have been obtained, since the original interaction matrix is referenced by the BMI command.

```
PURGE  SDS  SDS  ...  SDS

PURGE  BNDZIJ  BNDUPR  BNDLWR
```

FIGURE 65. THE PURGE COMMAND

The example shown in Figure 65 is taken from the command language stream used in the second part of the sample problem. In this part a full matrix solution is used to find the currents on the wire grid model. Therefore, the banded matrix and its upper and lower derivatives are no longer needed. The command shown in Figure 65 will eliminate these files from storage and from all future checkpoints.

J. END

The final command that will appear in the task execution command stream is the END command. As shown in Figure 66, it consists simply of the three-letter symbol "END". It is used to indicate to GEMACS that all the commands have been input and processed.

As is the case with the END command in the geometry input (Section IV.A), there is no need to separate the comment by a dollar sign ($). All that is needed is a blank space or a comma. The message shown in the example in Figure 66 is a typical one used in all input decks.

K. The SET Command

This command, shown in Figure 67, is used to initialize or change the data associated with some set. One use of this command was mentioned in connection with the banded matrix iteration command (Section V.F). There it was stated that an initial value could be used for the currents to possibly speed up the iteration process. Use of this command could also be an alternate way to develop the excitation matrix, instead of using the voltage source command (VSRC) or to develop the load matrix (ZLOADS command), both discussed in section V.D.

```
END

END OF COMMAND STREAM
```

FIGURE 66. THE END COMMAND

```
SET   SDS = N, [N]  [R1=N]  [R2=N]  [C1=N]  [C2=N]

 SET   SOURCE = 0.0,0.0  R2 = 100
 SET   SOURCE = 1.0,0.0  R1 =  33      (FICTITIOUS)
```

FIGURE 67. THE SET COMMAND

As can be seen in Figure 67, the mnemonic for this command is SET. The symbol SDS represents the name of the file being generated. The data are input in either real or complex form, indicated by "N,[N]". If the data are real, the second value is defaulted. If the data are complex, then two values must be input, even if one or the other is zero. Note that only one real or one complex value may be input per command. This value will then be used for all the elements that exist within the row and column limits specified in the command.

The last four items shown in the command in Figure 67 specify these row and column limits. Any or all of these items may be defaulted. R1 and R2 are the row limits of the elements to which the values will be applied. C1 and C2 are the lowest and highest column limits, respectively. The default value is one for R1 and C1; and R2 and C2 are defaulted to R1 and C1, respectively. This latter holds true even if R1 and C1 have been defaulted.

An example of the use of this command is also shown in Figure 67. It assumes that an excitation matrix, labeled SOURCE, already exists and that it consists of 100 elements. It is desired to change the boundary condition such that only one subsection is excited by one volt. The first command will set all of the elements to zero. Since the excitation is a complex number, two values are input, one real and one imaginary. The default value for R1 has been used, while R2 has been set to the number of rows in the matrix. Since there is only one column in the matrix, C1 has been defaulted to one; and C2 has been defaulted to C1.

The second command then sets element 33 to one volt. In this command R2 has been defaulted to R1. Again, C1 and C2 have each assumed their default values.

Note that if element 66 was also to have been set to one volt, then a separate command would have been needed. Setting R2 to 66 in the second command would have resulted in elements 33 to 66 each having a value of one volt.

VI. OUTPUT

There are three types of output provided by GEMACS: the standard boiler-plate, those data specifically requested by the user; and error messages and debug information needed by the user when he has a problem running the computer code. Examples of the first two may be seen in the output listing of the sample problem contained in the Appendix.

A. Standard Output

Figure 68 shows a list of those types of data that are always printed out by GEMACS regardless of the problem or the specific data requested by the user.

The first output provided by GEMACS is the list of keywords that may not be used as symbol names. This is identical to the list that is shown in Figure 50. This list is printed out strictly as a convenience in case there is an error message stating that one of the names that is used is not acceptable. However, other reasons for such a message would be a name that is greater than seven characters in length or one whose initial character is not one of the letters of the alphabet.

The date and time are next provided by GEMACS to the user. This information really comes in handy when one starts piling up stacks of computer paper and loses track of the order in which things were done.

A list of the task commands input by the user is next printed out. Normally it is just a straight listing of this part of the input deck. However, if any errors were detected on any of the commands, the error will be printed out immediately after the erroneous card. The subsequent commands will also be processed and errors searched for, but the analysis will be

RESERVED KEYWORDS
DATE AND TIME
INPUT TASK COMMANDS
GLOBAL PARAMETERS (IF ANY)
INPUT GEOMETRY COMMANDS
SEGMENT DATA TABLE
LOADS (IF ANY)
EXCITATION
TASK PROCESSING
ANTENNA/LOAD PARAMETERS

FIGURE 68. STANDARD DATA OUTPUT

terminated after the last card of the geometry input has been scanned.

The global parameters will be printed out next, if any have been specified. These data include the number of files available, the time allowed for execution, the frequency in megahertz at which the analysis is being performed, and the conductivity and permittivity of any imperfect ground.

The global parameters are followed by a listing of the geometry input deck. As in the case of the command language deck this will be a straight listing unless an error was detected in the format. Typographical errors that result in the erroneous location of one or more subsections will not cause an error message. These will have to be ferreted out by the user.

These first five items are printed out mainly as a check for the user to satisfy himself that what he thought he put into GEMACS is actually what did get into GEMACS. The first analyzed data that are printed out are the specifics relating to each of the subsections. The data are contained in the segment data table, whose headings are listed in Figure 69. Each line in the table contains the data for one particular subsection.

The first column represents the tag number assigned to each subsection as it is generated. Looking at the output, it is seen that most of the first 312 subsections have a tag of two. This indicates subsections of the fine-mesh grid which was defined in coordinate system number two. Two subsections each have a tag of three, indicating the transmitting antenna. Six subsections have a tag of four indicating the receiving antenna. The rest of the subsections belong to the coarse-mesh grid and have a tag of one.

TAG
XN
XC
XP
YN
YC
YP
ZN
ZC
ZP
RADIUS
LENGTH
END1
ISEG
END2

FIGURE 69. SEGMENT DATA TABLE HEADINGS

The next nine columns list the (x,y,z) coordinates of the ends and center points of the subsection. The letter "N" stands for the first coordinate point on the wire input (WR) card; the letter "P" stands for the second coordinate point on the wire card. The letter "C" indicates the center coordinate of the subsection. Therefore, reading the table from left to right, the start, center and end points, respectively, for each coordinate are given. By reading this table carefully one can then tell whether or not each subsection is properly positioned in the wire model. Note that these values are in meters in the global coordinate system only.

The next two columns of the segment table list the radius and length in meters, respectively, of each subsection of the wire model. By glancing through the table it is seen that each subsection has a radius of 0.000776 meters and that the length of each of the first 312 subsections is 0.05 meters. For the coarse-mesh part of the grid the length of each subsection is either 0.05 or 0.1 meters. These data are to be expected based on the figures in Section IV showing the generation of the wire grid.

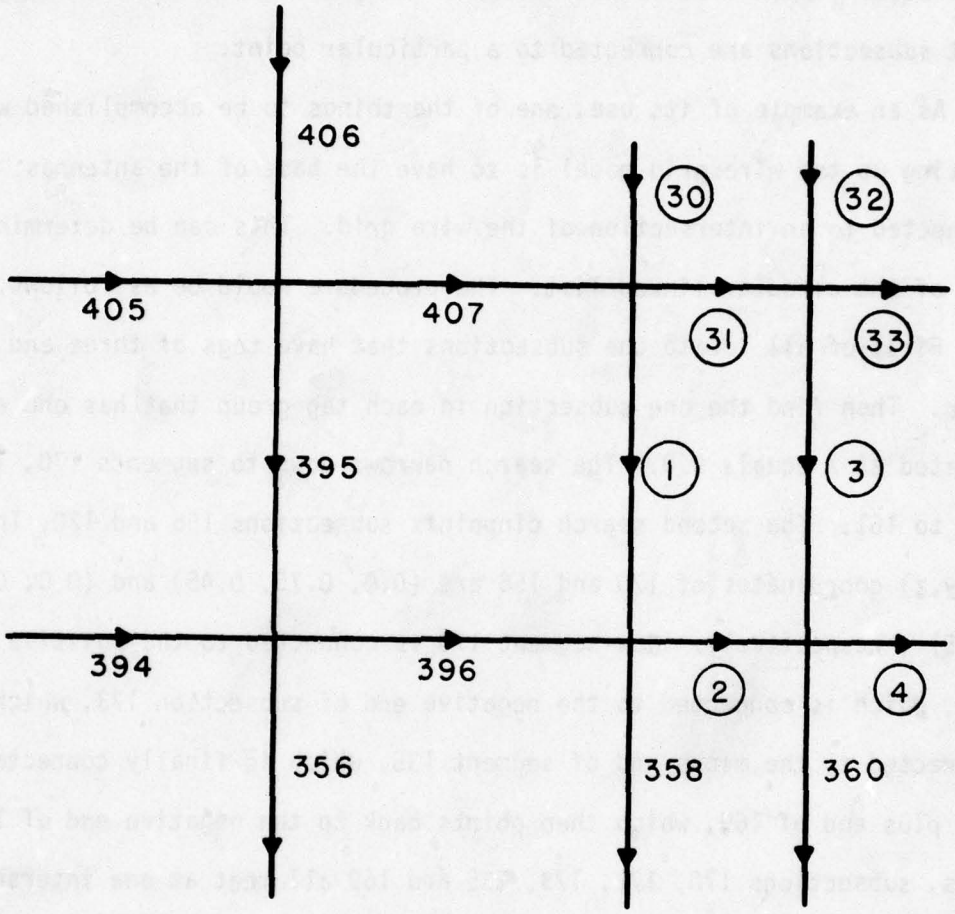
The next column, labeled "END1", specifies what is the lowest numbered segment attached to the negative end of the subsection. The word "negative" is used in the sense that one travels along the subsection from this end (END1) or point to the other end (END2) or point. There is also a significance attached to the sign associated with segment numbers listed in this column. The number in the first row of the sample output has no sign, while the number in the second row has a negative sign. The absence of a sign indicates a connection with the positive end of that subsection, and the presence of the negative sign shows a connection to the negative end of the

subsection indicated. Therefore, the negative end of segment one is connected to the positive end of subsection 30; and the negative end of subsection two is connected to the negative end of segment 358. This is shown in Figure 70, which is an expanded view of part of the wire grid model. The arrows indicate the reference direction of current flow. The reason why plus one is not in row two instead of -358 will be explained shortly.

The next to last column, ISEG, simply contains the subsection number for which the data in that row apply. With one exception, the numbers in this column appear only once. That case is illustrated in the row just below ISEG equals 396. The segment number as well as the tag for what should be subsection 397 is zeroed out. This indicates that subsection 397 is a duplicate of a preceding segment. That this duplicate subsection is numbered 29 can be determined by comparing Figures 38 and 45. This can also be verified by checking the list of duplicate subsections which occurs just before the segment data table. In the case of the sample problem there are 24 such duplicate pairs of subsections.

The final column in the segment data table gives the lowest numbered subsection and its end which is connected to the positive end of the subsection whose row is being studied.

GEMACS, in determining the data for the columns labeled "END1" and "END2", always looks forward from the subsection number under consideration. The first larger number it finds is the number that would be entered in the appropriate column. When the count reaches the highest segment number, GEMACS then cycles back to segment one and starts counting upward looking



○ INDICATES FINE-MESH SUBSECTION

FIGURE 70. SEGMENT CONNECTIVITY SAMPLE

for a subsection connected to the point. When it reaches the number of the subsection listed under ISEG, then the searching ceases. In this manner a circular linked list is established. This will aid in determining what subsections are connected to a particular point.

As an example of its use, one of the things to be accomplished when setting up the wire grid model is to have the base of the antennas connected to an intersection of the wire grid. This can be determined by use of the circular linked list. The procedure would be as follows.

First of all locate the subsections that have tags of three and four. Then find the one subsection in each tag group that has one end located at X equals 0.0. The search narrows down to segments 170, 171 and 156 to 161. The second search pinpoints subsections 156 and 170. The (x,y,z) coordinates of 170 and 156 are (0.0, 0.75, 0.45) and (0.0, 0.55, 0.45) , respectively. Now segment 170 is connected to the positive end of 172, which is connected to the negative end of subsection 173, which is connected to the minus end of segment 135, which is finally connected to the plus end of 169, which then points back to the negative end of 170. Thus, subsections 170, 172, 173, 135 and 169 all meet at one intersection. Similarly, it can be determined that 156, 162, 163, 127 and 155 all meet at one point. Thus, the two antennas are each located at an intersection of the wire grid.

Returning back to Figure 68, it is seen that the load data are next printed out, if any such data exist. The sample problem output shows the name of the data set (LOAD), the data showing the segment numbers on which loads are placed and the values of those loads.

The load type is "ZIMP", indicating a lumped load resistance and reactance is also shown.

These data are immediately followed by the data on the excitation. The geometry HUTTOP is excited by the values contained in SOURCE. The excitation type, location and strength are also shown.

The output that appears between the data for the excitation and the table of values of current traces the actual task execution. The tasks are accomplished in the order in which the task commands are sequenced. So, the first operation that occurs, as seen in the sample output, is that the impedance matrix (ZIJ) is being filled. Other auxiliary data such as the names of the geometry and load data sets are printed out, as well as the electrical parameters of the problem.

Then it is seen that GEMACS extracts the banded matrix from the original interaction matrix ZIJ with a bandwidth of 500. Also printed out are some statistics relating to the interaction matrix for the center column of that matrix. The band dominance factor is the ratio of the band norm to the difference between the column norm and the band norm, where the norm is the square root of the sum of the squares of each element in that column. The column norm is obtained by using all of the elements in the column, while the band norm is derived using only those elements that appear in the banded matrix. The higher this ratio the better, since a higher ratio says that more of the larger interactions are included in the banded matrix.

The next operation is the decomposition of the banded matrix. It is shown that diagonal pivoting is being used by the N after the "pivot equals"

identifier. This is the only pivoting allowed in the present version of GEMACS because, as was reported in one of the BDM quarterly reports [6], pivoting increases the bandwidth required and this would decrease the efficiency. Also, for the matrices being dealt with, pivoting has not been found necessary. As a matter of fact, this parameter is not even an input in the decomposition command (LUD).

Also printed out are the maximum and minimum diagonal values of the banded matrix, as well as the ratio of these two values. This ratio can be used as a measure of the ill-conditioness of the banded matrix. The higher the ratio, the more unstable is the matrix and the greater the effects of computer round-off error on the accuracy of the solution.

The next data printed out relate to the BMI solution technique. The equation as input in the command is printed out along with the maximum number of iterations allowed, the convergence criterion chosen and the value of that criterion to be reached before iteration is stopped. The iteration process can then be followed. The predicted number of iterations that will be needed and the values of each of the convergence criteria calculated at that iteration are also printed out.

Note that the first four iterations show that zero required iterations are necessary. GEMACS does not make such a prediction until it sets up a history of convergence, which requires four iterations to accomplish. After this, the figure for the number of predicted iterations is based on the algorithm derived in Section F of RADC-TR-75-272 [6].

When convergence or divergence is reached, a message is printed indicating this fact. The final values of the various convergence criteria are given, and it can be seen that the final value of the chosen criterion, the predicted relative error (PRE), is less than the value specified in the BMI command.

The final data that are considered part of the standard output, as listed in Figure 68, are those data relating to antenna input and output power and transmitting antenna input impedance. These are listed under the heading "Antenna/Load Parameters". For each subsection that is excited (170 in this case) the magnitude and phase of the input impedance is printed out as well as the power in watts delivered to the antenna. If there is also a load attached to this subsection, then the power loss in this load in watts is also given.

If the subsection is just simply loaded, then the first three items have a value of zero. The power loss in the load is the only non-zero entry for this subsection. This is done for subsection 156 in the output.

One final point to note is that interspersed throughout the execution trail are messages indicating that a checkpoint is being taken. The message and its explanation are discussed in section V.G.

B. Calculation of Coupling

The antenna and load parameters that have just been discussed allow the calculation of the coupling that exists between any pair of antennas in the system. The data supplied for subsection 156 specify the power delivered by the receiving antenna to the load. As can be seen, this value is 0.146 milliwatts. The power delivered to the transmitting antenna (subsection 170)

is given as 6.22 milliwatts. Calculation of ten times the log to the base ten of the ratio of these two powers gives the coupling between the two antennas as -16.5 dB.

Measurements were made on a similar structure [11]. Two monopoles were mounted on flat plates, one of which measured approximately 5.33 wavelengths on a side. The other plate measured approximately 11.4 wavelengths on a side. The transmitting antenna was located approximately 1.67 wavelengths from the nearest edge as shown in Figure 9A of reference [11]. The receiving antenna was located 1.67 and 3.17 wavelengths from the nearest edges and 0.5 wavelength from the transmitting antenna. The measured coupling was about -21 dB.

As discussed in the report several reasons can be cited to explain why a discrepancy can exist between the measured and predicted data. First, the measurements were not performed in a reflectionless area; and stray radiation could cause an interference effect to exist such that scattered fields from nearby surfaces could cause cancellation of the direct field from the transmitting to the receiving antenna.

Secondly, the plate in the example is considerably smaller than the one on which the measurements were made. Also, the antennas in this problem are much closer to the edges than they were in the experimental setup. The edge effects can cause unaccounted-for reflections.

Third, for the larger antennas it was difficult to keep the two monopoles perfectly parallel and both orthogonal to the ground plane. The mathematical model has both of these characteristics built in. Thus, a

polarization loss factor could cause a lower value of coupling to be measured in the experiment.

It should finally be noted that the measured data that are presented in the report are an average of several measurements taken on different days by different personnel. The measured data varied by one to two dB over the course of the experiment, especially when the longer antennas were involved.

Pending more careful experiments on a setup that more nearly approximates the mathematical model, it can be said that GEMACS provides a good engineering approximation to a very tricky problem in the near fields of collocated antennas.

C. The PRINT and WRITE Commands

In addition to the data that are automatically output by GEMACS, the data associated with any symbolic name can also be obtained. There are two commands available to obtain this information. The first of these is the "PRINT" command, shown in Figure 71. This command will result in the entire contents of the file being printed out. This command is most usually used to print out the current in amperes and the excitation in volts per meter on each of the subsections.

As can be seen in Figure 71 the mnemonic for this command is the word "PRINT". This is then followed by a list of the symbols, whose files are to be printed out. In the sample problem the voltage (excitation) and the current are to be printed out. Therefore, the symbols "SOURCE" and "CURRENT" are specified in the command.

PRINT SDS SDS...SDS

PRINT SOURCE CURENT

FIGURE 71. THE PRINT COMMAND

There are two formats in which the data are printed out. If the data are real, then ten values per line will be printed out. If the data are complex, only two values per line will be printed. In this case each value will have its real and imaginary components listed, as well as the magnitude and phase. Examples of the complex format can be seen in the output for the example problem.

Also included with the data is a heading that gives the type of data (real or complex) and the symbolic name of the file whose contents are being listed. The lineage that is given shows what other symbolic names are related to the one being considered. Thus, the current is derived from the banded matrix (BNDZIJ), which in turn is obtained from the interaction matrix (ZIJ), which is finally dependent on the geometry data set (HUTTOP). The excitation set, SOURCE, is only derived from the geometry (HUTTOP).

There are times, however, when the data set is too large to print all of the values; and yet some portion of the data needs to be displayed in order to check on the consistency of the data. One example of such a data set would be the elements of the interaction matrix. For the sample problem over 500,000 lines of data would be printed on 10,000 sheets of paper. The WRITE command, shown in Figure 72, can be used in this situation.

Using this command part of the data associated with the symbol DS is output on the file specified in the LU item in a binary format, which can be used as input by other programs. The default value of the LU item is the computer system high-speed printer. If the printer is used, then the data are output in the standard ASCII character set.

Also, if the output is being written on the system printer, then the

```
WRITE  DS   [LU = N]  [R1 = N]  [R2 = N]  [C1 = N]  [C2 = N]  [FILEID = A]

WRITE  ZIJ     R1 = 10     C2 = 650     (FICTITIOUS)
```

FIGURE 72. THE WRITE COMMAND

FILEID item is ignored, even if some value is input. If some storage file is used, such as a card or paper tape punch, then a FILEID, if specified, will be the first word output to this device. If no FILEID is specified, then no identifying word will be written on the output file. This word can, therefore, be used to insure that the proper data set has been accessed.

Since the data can be thought of as being stored in a matrix format, it can be said that the four remaining items specify the row and column limits of the data that are to be printed. Any combination of these items may be defaulted, the default values being dependent on what parameters have been defaulted. The values of R1 and C1 specify the first row and first column, respectively, of the data to be output. Their default values are each one.

The values of R2 and C2 are the last row and column, respectively, of the data to be printed out. Their default values are the values input for R1 and C1, respectively, if these latter values have been specified. If they have not been specified, then R2 and C2 will default to the number of rows and columns, respectively, in the matrix.

It then follows that if all four items have been defaulted, the entire matrix will be printed out, just as if the PRINT command had been used.

The example shown in Figure 72 will illustrate these statements. In this example it is requested that part of the interaction matrix be printed out on the system printer. The FILEID then is not needed, and it has been defaulted. The first row to be printed is row ten. Since the parameter R2 is defaulted, then only the data in the specified columns of row ten will be printed out since R2 will default to R1 in this case. The column limits

are one and 650, determined by the fact that C1 has been defaulted (and hence is one) and C2 has been given as 650. If both C1 and C2 had been defaulted, then the entire row would have been printed out.

D. Electric Field Output

In addition to determining the currents on the structure, GEMACS also calculates the electric field due to those currents. This can be done at any point in the near- or far-field region of the radiating structure. These data are then printed out in a tabular form and in an optional graphic form. The command that accomplishes these functions is shown in Figure 73.

The symbol represented by SDS is the name of the data file in which the electric field data are to be stored. This item may be defaulted, in which case the data are not stored and cannot be later referenced. Use of this option in no way affects the printing out of the tabular or graphic forms of the electric field. The advantage of the default option is that it reduces the number of data storage files required by GEMACS.

The defined symbol in parenthesis following the command mnemonic EFIELD is the name of the data set containing the subsection currents.

The U, V and W in the next nine items represent the geometric variables in the three commonly used coordinate systems: X,Y and Z in the rectangular system; R,θ and Z in the cylindrical system; and R,θ and ϕ in the spherical system. The symbols U1, DU and U2 represent the initial value, the increment and the final value of the variable U. The same is true of the three items relating to V and W.

Each of these three variables may represent any one of the geometric variables in any one of the three coordinate systems. That is, U1 may

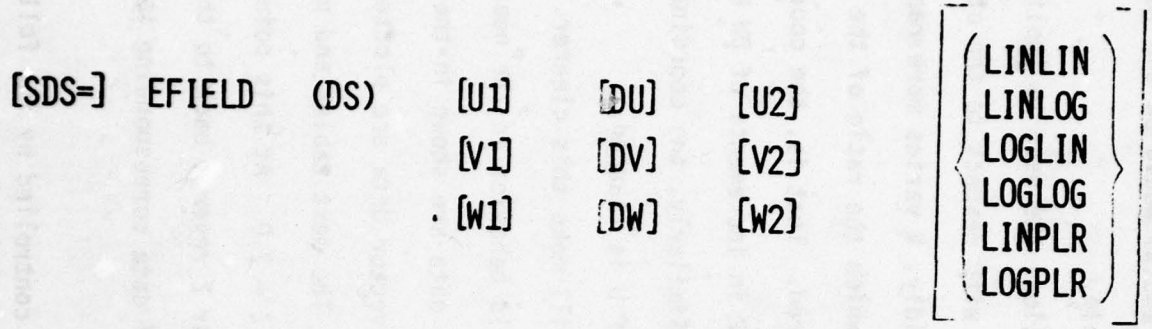


FIGURE 73. THE EFIELD COMMAND

represent either X1, Y1 or Z1 in the rectangular system; R1, θ1 or Z1 in the cylindrical system; or R1, θ1 or ϕ1 in the spherical system. Then, V1 would represent the initial value of one of the two remaining variables in that coordinate system; and W1 would be the initial value of the third coordinate system variable.

This freedom of representation allows the user to implicitly specify how rapidly each variable increments with respect to the other two coordinates. U varies the least rapidly, V varies more rapidly; and W is the independent variable against which the ratio of the electric field vector to the maximum field is measured. That is, the coordinate W is stepped through the range W1 to W2 in increments of DW before V1 is incremented by the value of DV. Similarly, the coordinate V is stepped through its range before the value of U is changed.

The example shown in Figure 74 will make this clearer. This command will print and plot the electric field behavior in the near field of the top of the hut. The complete output data are shown in the Appendix. It can be seen that the electric field vector data are plotted as a function of Y first for X = 0.0 and Z = 1.4. The next tables and plots are for X = 0.0 and Z = 1.7 and X = 0.0 and Z = 2.0. At this point the value of X is incremented to ten, and the value for Z reverts back to the original value. There then follow three more sets of data corresponding to Z = 1.4, 1.7 and 2.0, respectively.

The optional graphic display is controlled by the following six-choice item. If this item is defaulted, then only a tabular listing of the data will be output by GEMACS. If one of the six choices is present, then the plot will be in either a rectangular or polar form with axes

```
EFIELD (CURENT)   X1=0.0  DX=10.0  X2=10.0    LINLIN
                  Z1=1.4  DZ=0.3   Z2=2.0
                  Y1=0.0  DY=0.1   Y2=1.2
```

FIGURE 74. THE EFIELD COMMAND EXAMPLE

in either a linear or logarithmic progression. Each of the six choices is made up of two merged mnemonics as follows:

1. LINLIN - Both the dependent and independent variables are plotted linearly on a rectangular graph
2. LINLOG - The independent variable is plotted linearly and the dependent variable is plotted logarithmically on a rectangular graph
3. LOGLIN - The independent variable is plotted logarithmically and the dependent variable is plotted linearly on a rectangular graph
4. LOGLOG - Both the independent and dependent variables are plotted logarithmically on a rectangular graph
5. LINPLR - The independent variable is plotted as a function of angle from the reference, and the dependent variable is plotted linearly on a polar graph
6. LOGPLR - The independent variable is plotted as a function of angle from the reference, and the dependent variable is plotted logarithmically on a polar graph.

In each of these plots the independent variable will be one of the geometric variables: X,Y,Z in the Cartesian coordinate system; R,θ, Z in the cylindrical coordinate system; and R,θ,ϕ in the spherical coordinate system. The normalized magnitude of the electric field vector will be plotted as the dependent variable. This magnitude is not the absolute value of the electric field at the point being analyzed. It is the ratio of the field at that point to the maximum field calculated for all observation points generated by the command. Therefore, it is

normalized value, the normalization factor being printed out just before the tabulated values generated by the EFIELD command.

Note that if the LOG option is chosen, then the graphical and tabulated value is 20 times the common logarithm of the ratio of the point field strength to the maximum field strength. This represents the power in dB that the field strength at that observation point is down from the maximum value of field strength at all the observation points.

The simplest case is that of the LINLIN rectangular plot shown in Figure 75 in which both axes vary linearly. In this case the independent variable is plotted across the top of the page, and the dependent variable is plotted down the page. The origin of the plot is located in the upper left-hand corner of the plot. Figure 76 shows the same data plotted in a LINLOG format, in which the independent variable is plotted linearly across the top of the plot; and the dependent variable is plotted logarithmically down the side of the page.

The data for these plots are shown in Figure 77. The top line shows the normalization constant that is used to obtain the absolute value of the electric field vector at the observation points. The next line gives the values for the geometric coordinates which remain constant for these data, in this case X and Z. The left-hand column shows the variation of the independent variable, while the next six columns give the magnitude and phase of the three components of the electric field. The right-hand column gives the power ratio in dB between the total electric field at that observation point and the normalization factor.

For example, at Y = 2.0 meters, the total electric field vector is

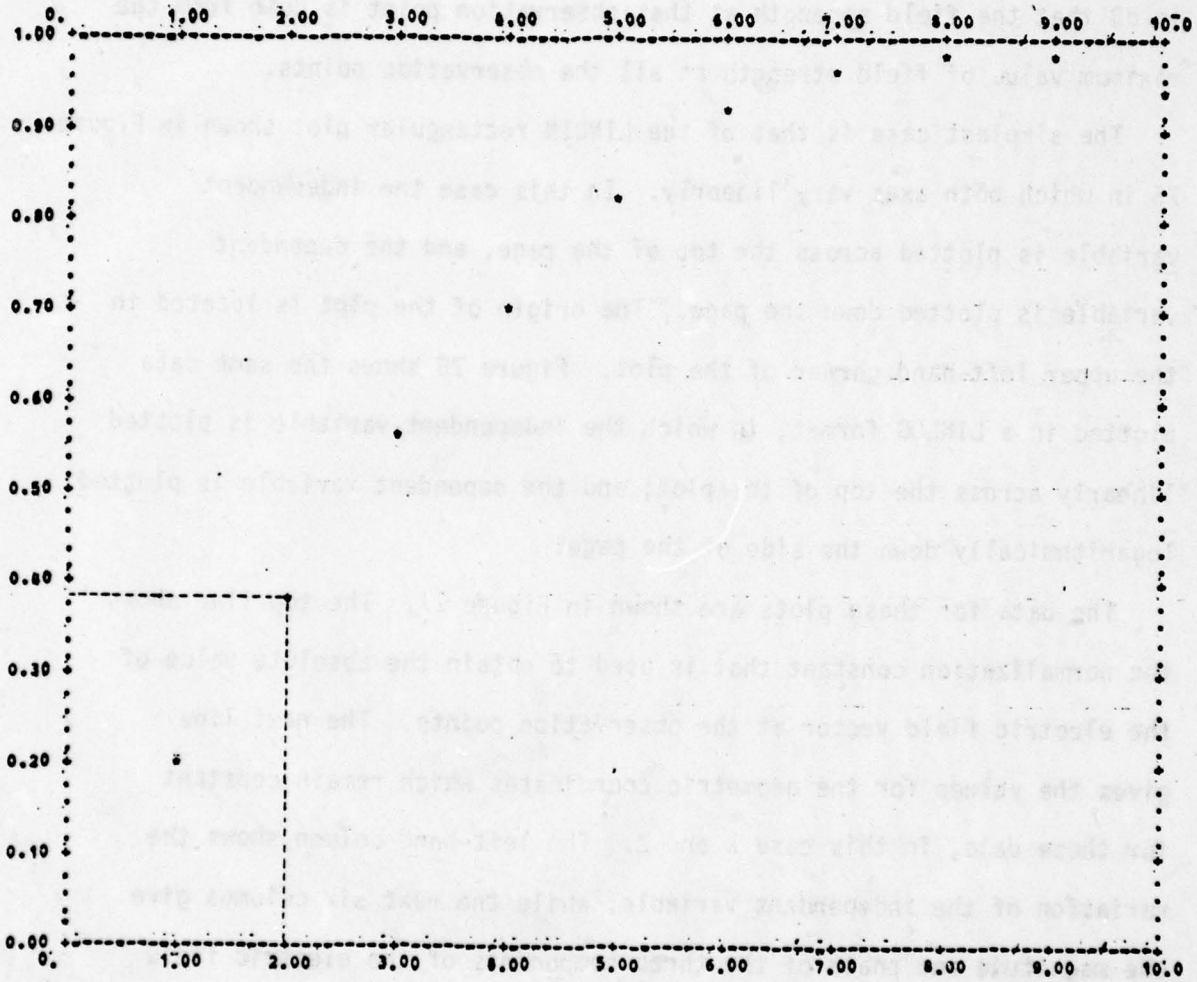


FIGURE 75. THE LINLIN PLOT

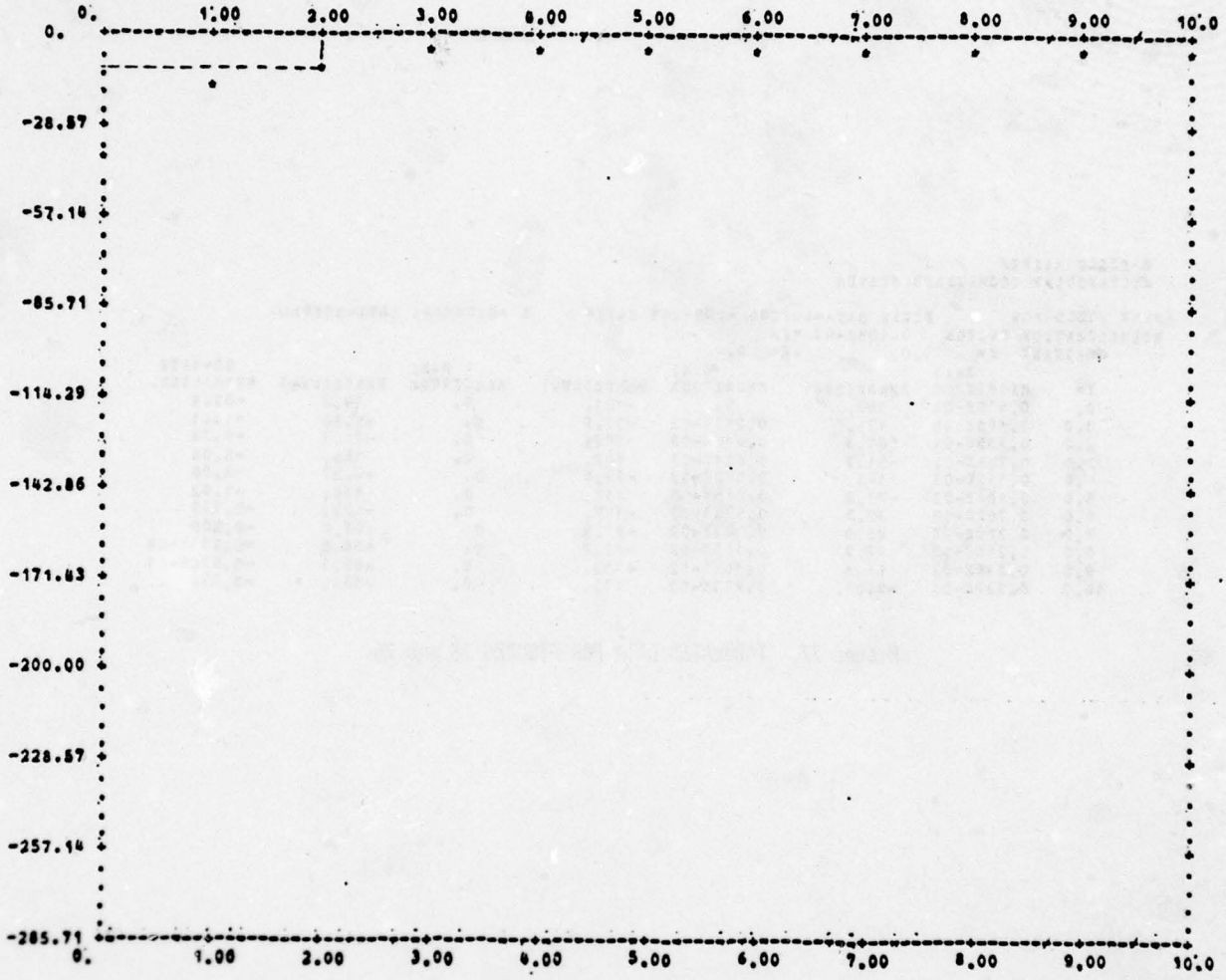


FIGURE 76. THE LINLOG PLOT

E-FIELD MATRIX I
RECTANGULAR COORDINATE SYSTEM

NEAR FIELD FOR FIELD DATA=NOPCOD =CURRENT DATA= I =GEOMETRY DATA=XDEPOL
NORMALIZATION FACTOR 0.108E-02 V/M
CONSTANT X= 30.0 Z= 0.

	E(X)		E(Y)		E(Z)		DB-GAIN
Y=	MAGNITUDE	PHASE(DEG)	MAGNITUDE	PHASE(DEG)	MAGNITUDE	PHASE(DEG)	NORMALIZED
0.	0.110E-04	153.	0.	-121.	0.	14.2	-39.9
1.0	0.180E-04	134.	0.212E-03	-37.9	0.	-9.80	-14.1
2.0	0.385E-04	66.3	0.416E-03	-109.	0.	-81.7	-8.28
3.0	0.708E-04	-51.3	0.601E-03	132.	0.	159.	-5.06
4.0	0.112E-03	143.	0.760E-03	-34.6	0.	-8.37	-3.00
5.0	0.158E-03	-70.3	0.866E-03	112.	0.	138.	-1.62
6.0	0.205E-03	30.5	0.976E-03	-147.	0.	-122.	-0.729
7.0	0.250E-03	86.0	0.103E-02	-91.8	0.	-67.0	-0.209
8.0	0.288E-03	97.2	0.105E-02	-80.7	0.	-56.5	-0.233E-04
9.0	0.318E-03	65.1	0.103E-02	-113.	0.	-89.2	-0.630E-01
10.0	0.337E-03	-9.01	0.983E-03	173.	0.	-164.	-0.376

FIGURE 77. TABULATED DATA FOR FIGURES 75 AND 76

approximately 0.4 mV/m. Dividing this by the normalization factor yields approximately 0.39, which is the value plotted on Figure 75. Taking the common logarithm of this value and multiplying by 20 yields -8.25 dB, the value that is plotted in Figure 76.

The same procedures for interpreting the plots and tabulated data can be used when studying the polar plots in the cylindrical and spherical coordinate systems. Figures 78, 79 and 80 show the LINPLR, LOGPLR and the tabulated values respectively, of a system being analyzed in the cylindrical coordinate system.

Again the first line of Figure 80 lists the normalization factor, and the second line gives the values of the constant geometrical coordinates. The rest of the table is read identically to that shown in Figure 77, except that the independent variable is now theta instead of Y.

In the plots in Figures 78 and 79, the origin is located in the center of the figure. The reference axis (the X-axis) is shown in each figure along with the direction of increasing theta, where theta is the angle around the Z-axis in the X-Y plane in the cylindrical system.

As an example, Figure 80 indicates that for theta equal to 30.0 degrees the magnitude of the electric field is approximately 0.35 mV/m, which is approximately 0.32 of the maximum electric field at all observation points. Multiplying this by the cosine of 30 degrees results in an X-axis value of 0.275. Multiplying 0.32 by the sine of 30 degrees gives a Y-axis value of 0.16. These results are shown by the dotted lines in figure 78.

Taking 20 times the common logarithm of 0.32 results in the normalized dB gain shown in the right-hand column of Figure 80. In order to

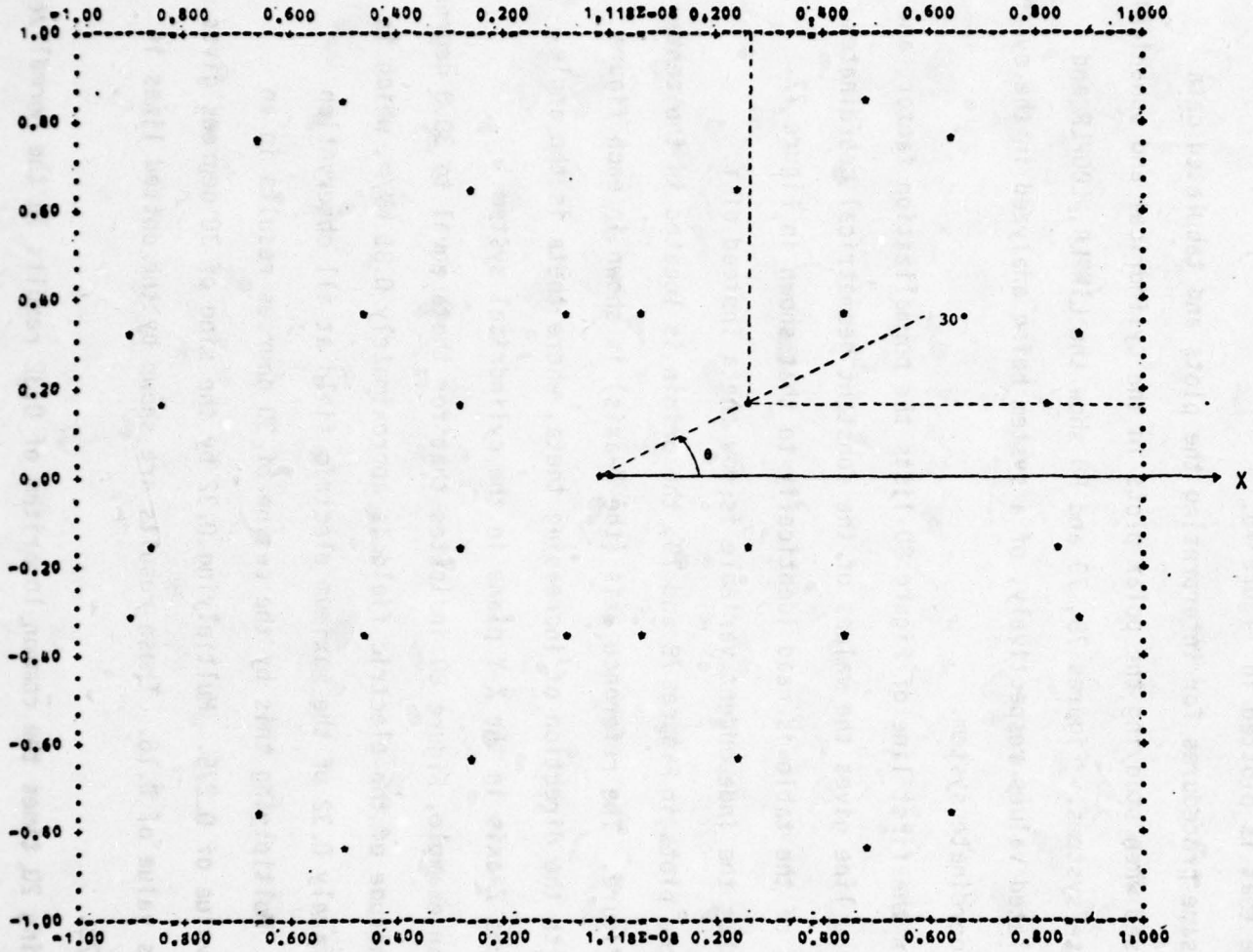


FIGURE 78. THE CYLINDRICAL LINPLR PLOT

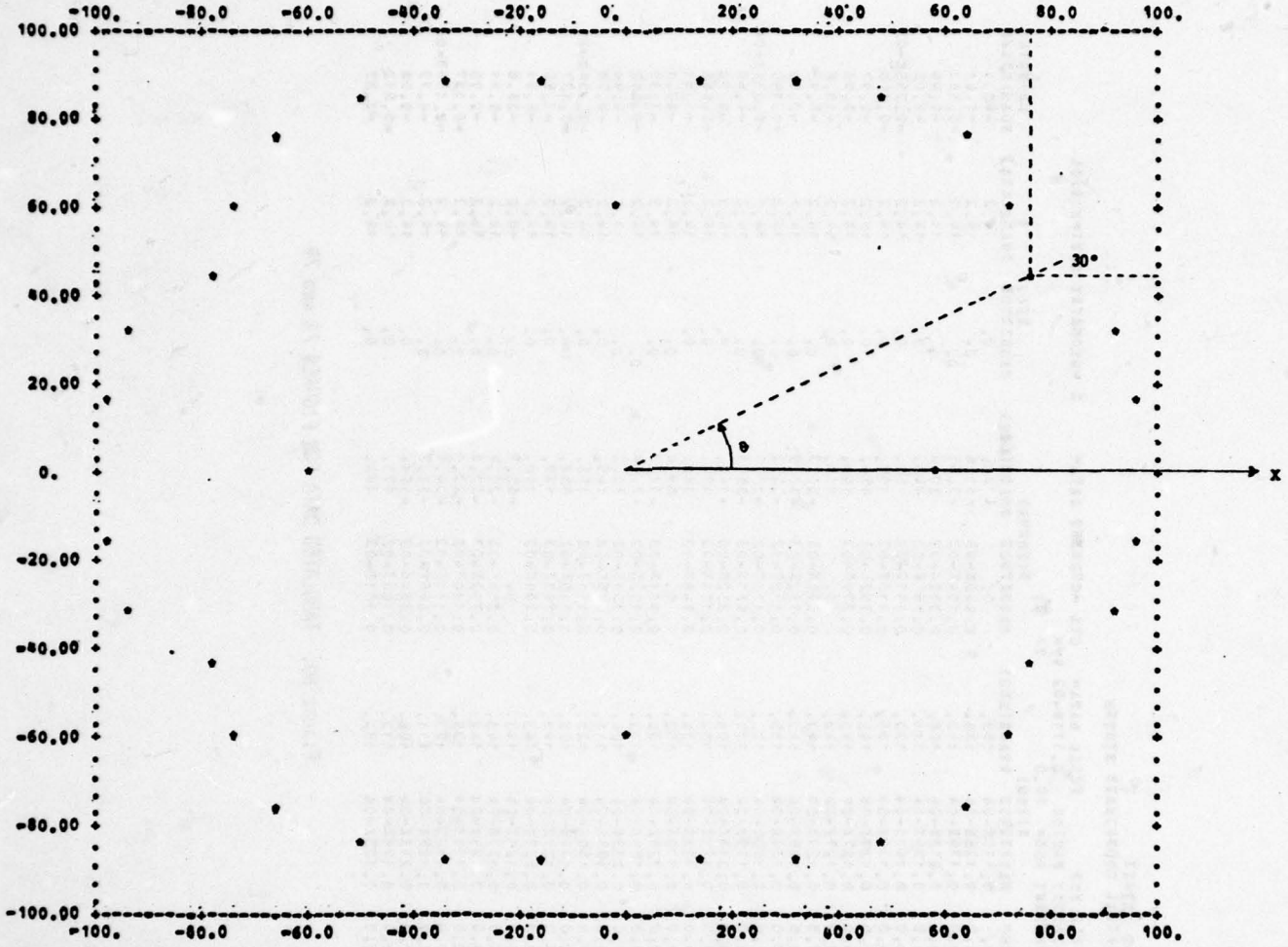


FIGURE 79. THE CYLINDRICAL LOGPLR PLOT

```
E-FIELD MATRIX      I
CYLINDRICAL COORDINATE SYSTEM

NEAR FIELD FOR        FIELD DATA=   CYL -CURRENT DATA=     I -GEOMETRY DATA=XDEPOL
NORMALIZATION FACTOR   0.111E-02 V/M
     CONSTANT RHO=  30.0          Z=  0.
                E(RHO)                  E(THETA)                E(Z)                  DB-GAIN
     TH=    MAGNITUDE   PHASE(DEG)   MAGNITUDE   PHASE(DEG)   MAGNITUDE   PHASE(DEG)   NORMALIZED
      0.    0.110E-04      153.          0.         -121.         0.          14.2        -40.1
     10.0   0.125E-04      134.       0.946E-03     -41.6        0.          14.2         -1.41
     20.0   0.179E-04      112.       0.104E-02      -7.51       0.          14.2         -0.563
     30.0   0.213E-04      106.       0.352E-03      20.4        0.          14.2         -9.99
     40.0   0.195E-04      111.       0.651E-03     145.          0.         14.2         -4.65
     50.0   0.160E-04      123.       0.111E-02     155.          0.         14.2         -0.255E-03
     60.0   0.138E-04      135.       0.110E-02     158.          0.         14.2         -0.140
     70.0   0.129E-04      142.       0.792E-03     159.          0.         14.2         -2.97
     80.0   0.127E-04      143.       0.398E-03     160.          0.         14.2         -8.93
     90.0   0.127E-04      143.          0.         118.          0.         14.2         -38.8
    100.0   0.127E-04      143.       0.398E-03     -20.3         0.         14.2         -8.93
    110.0   0.129E-04      142.       0.792E-03     -20.9         0.         14.2         -2.96
    120.0   0.138E-04      135.       0.110E-02     -22.2         0.         14.2         -0.140
    130.0   0.160E-04      123.       0.111E-02     -25.0         0.         14.2         -0.236E-04
    140.0   0.195E-04      111.       0.651E-03     -35.1         0.         14.2         -4.65
    150.0   0.213E-04      106.       0.352E-03     -160.         0.         14.2         -9.99
    160.0   0.179E-04      112.       0.104E-02      172.         0.         14.2         -0.563
    170.0   0.125E-04      134.       0.946E-03      468.        0.         14.2         -1.41
    180.0   0.110E-04      153.          0.          59.4        0.         14.2         -40.1
    190.0   0.125E-04      134.       0.951E-03     -31.0         0.         14.2         -1.37
    200.0   0.180E-04      112.       0.105E-02     -7.48         0.         14.2         -0.512
    210.0   0.213E-04      106.       0.354E-03      39.2         0.         14.2         -9.94
    220.0   0.195E-04      111.       0.646E-03     145.          0.         14.2         -4.73
    230.0   0.160E-04      123.       0.111E-02     155.          0.         14.2         -0.154E-01
    240.0   0.138E-04      135.       0.110E-02     158.          0.         14.2         -0.137
    250.0   0.129E-04      142.       0.793E-03     159.          0.         14.2         -2.95
    260.0   0.127E-04      143.       0.399E-03     160.          0.         14.2         -8.91
    270.0   0.127E-04      143.          0.         -62.7         0.         14.2         -38.8
    280.0   0.127E-04      143.       0.399E-03     -20.4         0.         14.2         -8.91
    290.0   0.129E-04      142.       0.793E-03     -20.9         0.         14.2         -2.95
    300.0   0.138E-04      135.       0.110E-02     -22.1         0.         14.2         -0.137
    310.0   0.160E-04      123.       0.111E-02     -24.8         0.         14.2         -0.156E-01
    320.0   0.195E-04      111.       0.646E-03     -34.7         0.         14.2         -4.73
    330.0   0.213E-04      106.       0.354E-03     -161.         0.         14.2         -9.94
    340.0   0.180E-04      112.       0.105E-02      173.         0.         14.2         -0.512
    350.0   0.125E-04      134.       0.951E-03      169.        0.         14.2         -1.37
```

FIGURE 80. TABULATED DATA FOR FIGURES 78 AND 79

correlate this number with the plotted values of Figure 79 this normalized value must be algebraically added to 100 dB, the dynamic range of the polar plots. The result for theta equal to 30 degrees is approxaimately 90 dB. Multiplying this first by the cosine of 30 degrees and then by the sine of 30 degrees yields 77.95 dB and 45 dB for the X- and Y-axis values, respectively.

There are two points to be noted about the polar plots. First, they are not square. Therefore, curves of constant power down are elliptical rather than circular. Second the sign on the axis values should be ignored. Some values are preceded by a negative sign only because of the limitations implicit in ANSI standard FORTRAN coding.

The analysis of the plots and tabulated data in the spherical coordinate system is identical to that just presented for the cylindrical coordinate system. The only point to note is that in the spherical coordinate system the independent variable may be either phi or theta. The angle phi is measured in the X-Y plane and is positive when rotating counterclockwise around the Z-axis. The reference (or zero-degree direction) is the X-axis. The theta angle is measured from the Z-axis and is positive when traveling from this axis to the X-Y plane. It has the same X-axis reference in the plots as was used for all the angular coordinates discussed.

Finally, when angular coordinates are used as the independent variable in a rectangular plot, they are treated the same as any of the linear independent variables. They are plotted on the X-axis and may vary linearly or logarithmically.

E. Error Messages and Debug Output

There is an extensive set of messages available to the user that is printed when GEMACS encounters an error during input processing or during the performance of an analysis. These are automatically printed out without the need for a user request. In addition, a dictionary will soon be available as a supplement to the user's manual which will alphabetically list the messages as they appear and provide information regarding the subroutine that generated the message, the cause of the error and the steps to be taken to eliminate the error.

As mentioned before, if an error is found during the reading of the input deck, GEMACS will print the appropriate error message and continue processing the rest of the input cards. If subsequent errors are found, further error messages are printed. However, execution of the analysis process will not be initiated. GEMACS will terminate after the input processing has been completed, and it will print out the contents of the input deck with the error message immediately following an improper command.

If the error should occur during the execution of the analysis, GEMACS will terminate the analysis at that point, print out an appropriate error message and take a checkpoint, if a checkpoint command (CHKPNT) has appeared in the command stream prior to the command that initiated the operation in which the error occurred. Since there is a checkpoint file available, it is possible to restart from the command at which the error occurred once the source of the error has been located and corrected. However, keep in mind that if a command is changed, the erroneous command and all subsequent commands must be wiped out when restarting and replaced by a new command stream (refer to the discussion in connection with Figure 64).

In addition to the error messages a wealth of information is available regarding the actual processing that goes on during the execution of an analysis. These data can be obtained through the use of the DEBUG command, shown in Figure 81. The mnemonic for this command is the word DEBUG. If the option chosen is "ON", then the data generated during the execution of the following command is printed out. Included in this printout will be the occurrence of any major operation, such as the storage or retrieval of major blocks of data. Examples of such output can be seen in example two of the GEMACS user's manual [10].

If the option "OFF" is chosen, the program and the output return to the normal mode of operation. All commands between DEBUG ON and DEBUG OFF will be affected and will have the diagnostic information printed out during the execution of their operations.

If the "TRACE" option is chosen, then the debug operation is turned on. However, in addition to the information normally printed out GEMACS will also provide the user with information regarding the entry into or the exit from various subroutines during the execution of an operation. This will aid in following the program flow and can be of use in tracking down "bugs" within the code.

If the "STATS" option if chosen, then statistics are collected and output, which describe what subroutines were accessed, how often, the amount of CPU time expended in each subroutine and the percentage of the total CPU time spent in each subroutine. These data are useful only when modifying GEMACS to increase efficiency. Also, the statistics are meaningful only when the entire analysis has been accomplished without the need for a restart, since in this case all statistics are reinitialized to zero.

```
DEBUG  { ON    }      [ILP]
       { OFF   }
       { TRACE }
       { STATS }
```

```
DEBUG ON
BNDZIJ*I=ANTSRC-ZIJXDP*I      MAXITR=10
DEBUG OFF
```

(FICTITIOUS)

FIGURE 81. THE DEBUG COMMAND

In addition, the use of this option significantly increases the cost of an analysis since a large number of subroutine calls is involved in the collection of these timing statistics.

In contrast to the other options available, once the STATS option is turned on it cannot be turned off. However, statistics are automatically collected whenever the TRACE option is in effect. In this case, when the TRACE option is turned off, the collection of statistics automatically ceases.

If option "ILP" is chosen, then the debug operation will be on during the reading of the input deck. All of the task tables, symbolic names and other useful data will be output, which are helpful in finding "bugs" in that part of the code which reads the commands and sets up the execution tables based on the data contained in the commands. This option is illustrated on pages A-3 to A-11 of the Appendix.

The example shown in Figure 81 is taken from example two of the GEMACS user's manual [10]. The only command that is affected is the one that initiates the BMI solution process. The first command would initiate the debug operation, while the last command would terminate the operation. If this latter card were not present, debug output would be printed out for all commands following the BMI command. Caution must be exercised when using the DEBUG command, since it may cause the printing of large volumes of data. This is especially true when the debug is turned on during the execution of the interaction matrix generation command (ZGEN).

VII. FULL MATRIX SOLUTION

In addition to the BMI matrix equation solution technique (Section V) GEMACS has the capability to solve the MOM matrix equation (Figure 5) using the entire interaction matrix by a Gauss-Jordan elimination process. The sequence of commands shown in Figure 62 is very nearly the same as would be used for the latter solution technique. The only difference is that the BAND, LUD and BMI commands would be replaced by the commands discussed in this section.

There are essentially two equivalent ways to implement the full matrix solution technique. The simpler of the two is to use the SOLVE command, shown in Figure 82. The three input quantities for this command correspond one-for-one to the quantities in the MOM matrix equation, shown in Figure 5. That is, DS1 represents the name of the interaction matrix, SDS is the symbolic name of the set of currents; and DS2 is the name of the excitation matrix. This can be seen directly by studying the example shown in the figure. Here the same symbols are used in the SOLVE command as were used in the BMI command (Figure 60).

Use of this command results in the lower/upper decomposition of the interaction matrix followed by the forward elimination/backward substitution process. As such it combines the LUD and BACSUB commands, a combination of which is the second method of implementing the full matrix solution process.

The decomposition command (LUD) is the same as is used in conjunction with the BMI solution process and is discussed in Section V (Figure 59).

SOLVE DS1 * SDS = DS2

SOLVE ZIJ * CURENT = SOURCE

FIGURE 82. THE SOLVE COMMAND

The forward elimination/backward substitution command (BACSUB), shown in Figure 83 finds the solution for an excitation, given a previously decomposed interaction matrix. Therefore, in Figure 83, DS1 represents the name of the interaction matrix after it has been decomposed, SDS is the name of the current vector; and DS2 is the symbolic name of the excitation matrix.

The example shown in Figure 83 lists the two commands as they would appear in a command stream after the interaction matrix generation command. The first command would decompose the interaction matrix, ZIJ, and store it in the file labeled LUDZIJ. Then, the second command would perform the elimination/substitution process using LUDZIJ and the excitation matrix, SOURCE, to calculate the subsection currents and store them in the matrix whose name is CURENT.

Although the commands shown in Figures 82 and 83 result in identical outputs, they each have their own advantages. When using the SOLVE command, the analyst needs to generate only one command to calculate the currents. Furthermore, the generation of the decomposed matrix set is automatically accomplished by GEMACS and need not be a concern of the user.

However, if the problem at hand requires the derivation of the subsection currents for a number of nonlinearly related excitation sets, then the commands shown in the example in Figure 83 should be used. The reason is that the interaction matrix and its decomposed triangular matrices are not dependent on the excitation specified. As a result, the interaction matrix needs to be decomposed only once using the LUD command, the resultant matrix then being used in a series of BACSUB commands, one such command