031307-6-T

USER'S MANUAL FOR **FEMA-PRISM**

T. Ozdemir
J. L. Volakis

Mission Research Corp.
3975 Research Blvd.
Dayton, Ohio 45340

Rome Laboratories/ERPT
U.S. Air Force
Griffis AFB, New York 13441

MARCH 1996

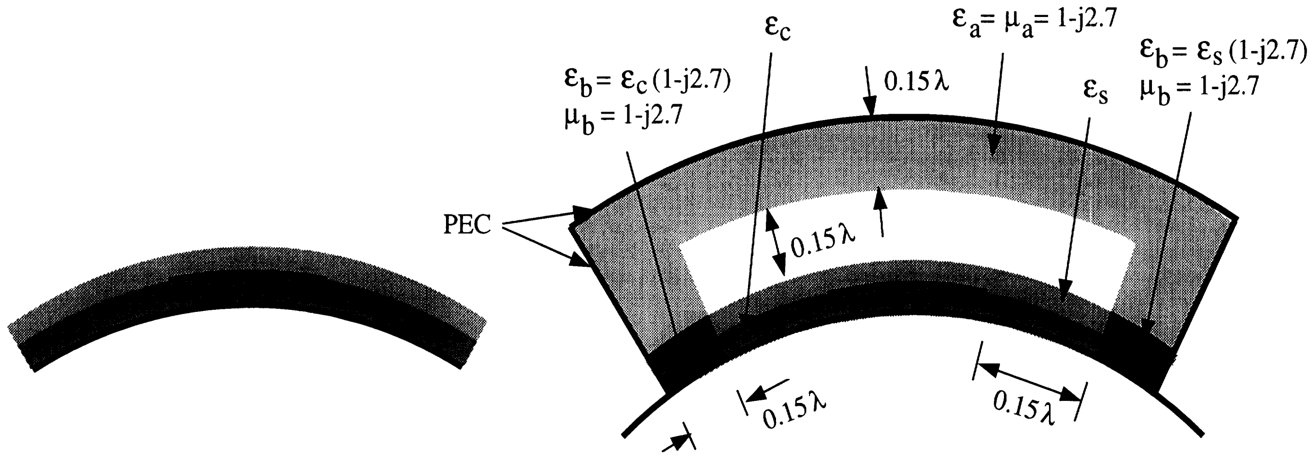
31307-6-T = RL-2442

User's Manual for FEMA-PRISM (Version 1)

Tayfun Özdemir
John L. Volakis

Radiation Laboratory
Department of Electrical Engineering and Computer Science
1301 Beal Ave.
University of Michigan
Ann Arbor, MI 48109-2122
TEL: (313) 764-0502, (313) 747-1797
FAX: (313) 747-2106
E-MAIL: tayfun@umich.edu, volakis@umich.edu
HOME PAGE: http://www-personal.engin.umich.edu/~volakis/

CONTENTS



ε_c
$\varepsilon_a = \mu_a = 1\text{-}j2.7$
$\varepsilon_b = \varepsilon_s(1\text{-}j2.7)$
$\varepsilon_b = \varepsilon_c(1\text{-}j2.7)$
$\mu_b = 1\text{-}j2.7$
0.15λ
ε_s
$\mu_b = 1\text{-}j2.7$
PEC
0.15λ
0.15λ
0.15λ

(a) (b)

Figure 1: Antenna Modeling

1 General code description

FEMA-PRISM is written in Fortran 77 computer language and has been verified to run on Unix platforms for HP, Sun, and SGI workstations. It is currently a serial code with potential for parallellization. It can be run with limited memory allocation at the expense of speed.

FEMA-PRISM is used to analyze printed antennas on doubly curved surfaces. It employs the finite element method (FEM) in conjunction with artificial absorbers to truncate the mesh (see Figure 1). The resulting FEM system of linear equations are solved interatively using BiCG technique. For details of the analysis and the theory of the formulation, the user is referred to [1] and [2].

2 Types of antennas that can be modeled

Microstrip as well as cavity-backed antennas with or without coatings can be modeled (see Figure 1). Currently, only probe feed can be specified at any arbitrary location.

3 Specifying antenna geometry

Because the antennas conform to the platform, only a surface mesh is needed. A built-in surface mesh generator exists for rectangular and circular patch antennas (see Section 11). Once the surface mesh is created, it can be viewed very easily using MatLab tools (see Section 12). The volume mesh is simply grown along the surface normal and the distorted prism is the building block of the resulting mesh [1].

4 Input Files

MainInput :	Contains information about the geometry and other input data
SurfMesh :	Contains the surface mesh data (specifying antenna surface detail)

5 Output Files

Imp :	Stores the input impedance
EqvCur :	Equivalent magnetic current over the surface of the antenna
EdgeUnk :	Complex amplitudes of edge unknowns
MeshDsply:	Contains surface mesh data for plotting using MatLab
ElmMat :	Contains element matrices for each prism in the mesh. It is only computed when a new mesh is generated and can be reused as long as the mesh does not change. Whether it is generated or read in is controlled by an entry in the input file *MainInput.*

6 Running the code

Compile the code by typing

f77 FEMA-PRISM.f -o *executable-filename*

To run the code, type in *executable-filename.* The code will first read the integer entry of the first row of the input file *MainInput*, which will tell the code whether the user supplied surface is alreay in the file *SurfMesh* or the surfmesh is to be generated by the code according to the second row of *MainInput.* If the user supplies the surface mesh, the code proceeds to read in the contents of the file *SurfMesh* and carries out the analysis according to the subsequent rows of information in the file *MainInput.* If the mesh is to be generated by the code's built-in mesh generator, the code also reads in the second row of the file *MainInput*, stores the surface mesh data in the file *SurfMesh* and terminates. The code has to be rerun with user-supplied mesh option. For more detailed explanation, see the following three sections.

7 Input files *MainInput* and *SurfMesh*

MainInput contains information describing the antenna geometry, substrate/superstrate materials, frequency of operation, etc. Figure 2 shows the data format. As shown there, the first row tells the code whether the user supplies the surface mesh or whether the mesh will be generated by the built-in mesh generator. The second row has the antenna geometry info if built-in mesh generator is used. Otherwise this information is skipped. Starting with the third row (second row if user-supplied surface mesh option is chosen), the rest of the information is concerned with the geometry and the run. **All length quantities are in units of Centimeter, frequencies are in GHz, currents are in Amperes, electric filed is in Volts/cm, impedance is in Ohms and material parameters are always relative quantities with respect to those of the free-space.**

SurfMesh contains surface mesh data and must be ready prior to running the code if the user-supplied mesh option is chosen. It can be created for rectangular and circular patches by the code itself. The first row of the file *SurfMesh* contains a series of numbers specifying how many triangles and nodes are contained in the surface mesh, the number of triangles within the absorbing layer, etc.

8 Running with user-supplied surface mesh option

In this operation mode, the surface mesh data has to be ready in the file called *SurfMesh.* Figure 2 shows the general set up of the *MainInput* along with a description for each entry. Each filled circle indicates a row. All rows are read by the code with free format. Letters R,I, or C refer to a real, integer or a complex number entry. All entries on the same row must be seperated at least by a single space.

9 Running with built-in surface mesh generator

In this operation mode, the code has to be run twice. In the first run, the first row of the file *MainInput* has the entry value "2" or "3". The second row provides the information the built-in mesh generator needs to generate the mesh. The code stores the mesh data in *SurfMesh* and terminates. The code must then be rerun with the user-supplied mesh option.

Note that the built-in surface mesh generator generates planar surface meshes (located in the plane $z = 0$).

Caution: **In creating a surface mesh for a rectangular patch, if the patch is cavity-backed, care must be taken to leave at least one cell between the cavity wall and the air-absorber interface. If the patch is a microstrip, at least two cells must be left between the patch boundary and the air-absorber interface.**

- I → 1 = User-supplied surface mesh, 2 = built-in mesh generator for circular patch, 3 = built-in mesh generator for rectangular patch

Present only if the first row is "2"

- R I I I I
  - → Radius of the surface mesh (all the way to the termination boundary)
  - → Number of rings from the center to the edge of the antenna
  - → Number of rings from the center to the edge of the cavity
  - → Total number of rings in the surface mesh
  - → Number of absorber rings

Present only if the first row is "3"

- R R I I I I I I I I
  - → Incremental length in x-direction (uniform sampling)
  - → " y-direction (uniform sampling)
  - → Number of cells along x-dimension of the patch
  - → " y-dimension of the patch
  - → Number of cells between patch and cavity wall in x-direction
  - → " y-direction
  - → Number of cells between the cavity wall and the termination boundary in x-dir.
  - → " y-dir.
  - → Number of absorber cells in x-direction
  - → " y-direction

- I I I I
  - → # of substrate layers
  - → 1 = all substrate layers are identical, 0 = otherwise
  - → # of superstrate layers (enter zero for no superstrate)
  - → 1 = all superstrate layers have the same thickness and material parameters, 0 = otherwise

- R C C
- R C C
- : : :
- R C C

} Ordered from the bottom of the cavity up, each row corresponds to a substrate layer. Only one row is needed if all layers are identical (row has the info for a single layer).
  - → Thickness of the layer
  - → Relative permittivity of the layer
  - → Relative permeability of the layer

- R C C
- R C C
- : : :
- R C C

} Same as above but for the superstrate. Ordered from the antenna surface up (first row corresponds to the layer just above the antenna surface).

- I → # of probe feeds

- I I I C
- I I I C
- : : :
- I I I C

} Each row corresponds to a probe feed
  - → Surface node number #1
  - → Surface node number #2 } Probe current flows from node #1 to node #2.
  - → Layer # (layer within which the normally oriented probe is located, or the layer at top of which the laterally oriented probe is located). Entry can be positive or negative and increase away from the surface of the antenna with zero corresponding to the layer immediately below the antenna.
  - → Complex amplitude of the probe current

- R R R I R I I
  - → Starting frequency in GHz
  - → Final frequency in GHz
  - → Increment frequency in GHz
  - → Frequency run to save (1 = save the first freq. run, 2 = next frequency, etc.)
  - → Tolerance (~ 0.01)
  - → 1 = monitor convergence (dump residual error at each iter.), 0 = otherwise
  - → 1 = compute element matrices, 0 = read in element matrices

- I → 1 = Read in user specified termination parameters (given in the following row), 0 = code will figure out the optimum parameters (this is the safe course if one is not familiar with the artificial absorber termination).

- R I I C
  - → Thickness of one layer (all layers have the same thickness)
  - → Total number of layers from the top of the outer-most superstrate layer to the termination boundary
  - → Number of absorber layers
  - → Relative permittivity of the absorbing layers.

Figure 2: Input file *MainInput*

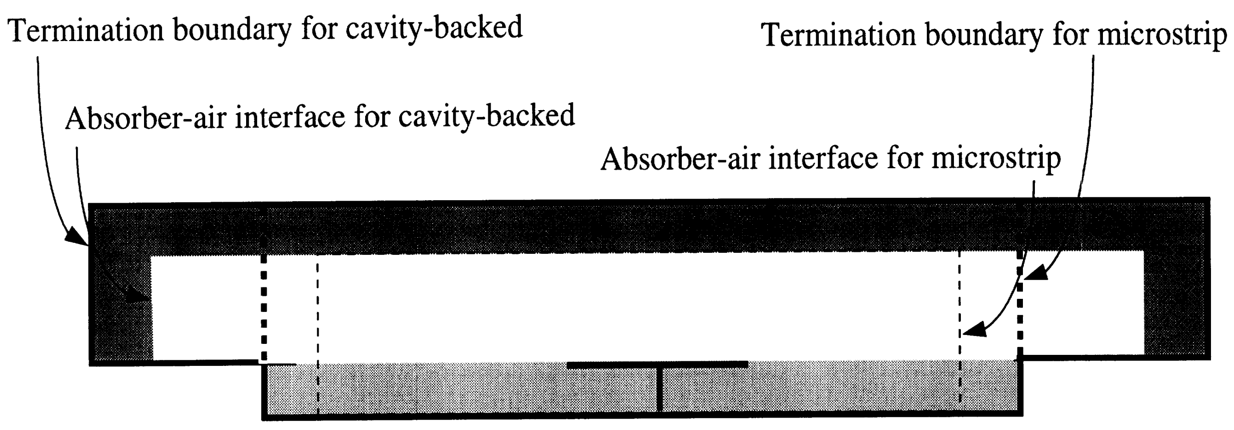

Termination boundary for cavity-backed
Termination boundary for microstrip
Absorber-air interface for cavity-backed
Absorber-air interface for microstrip

Figure 3: Microstrip vs cavity-backed configuration

10 Microstrip *vs* cavity-backed

As shown in Figure 3, the microstrip configuration can be realized as a cavity-backed configuration with the mesh terminated at the cavity walls. Basically, at first, the code treats every configuration as cavity-backed and if the extent of the mesh is the same as the cavity, the code recognizes it as a microstrip. Consequently, in the first row of *SurfMesh*, the second and third entries are identical and also the fourth and the fifth entries are identical for microstrip geometry.

In specifying a rectangular microstrip patch for the built-in mesh generator, zero should be entered for the distance between the cavity wall and the termination boundary (entries #7 and 8). For a circular patch, the same quantity should be entered for the number of rings from the center to the cavity wall and for the total number of rings in the mesh (entries #3 and 4).

11 How to create a user-defined surface mesh

The file *SurfMesh* contains surface mesh data. The format is given in Figure 4. As in Figure 2, here also the filled circles represent rows, and the letters I, R, or C imply real, integer, or complex number entries, respectively. The first row has information about the number of surface triangles and surface nodes. Starting from the second row is the information about the relation between the local and the global indexing of surface nodes. **It is important to note that the local numbering of surface nodes increase counter-clock wise. Also, the triangular patches are numbered starting from the antenna region, continuing with the region between the antenna boundary and the cavity boundary (if cavity-backed) and finishing**

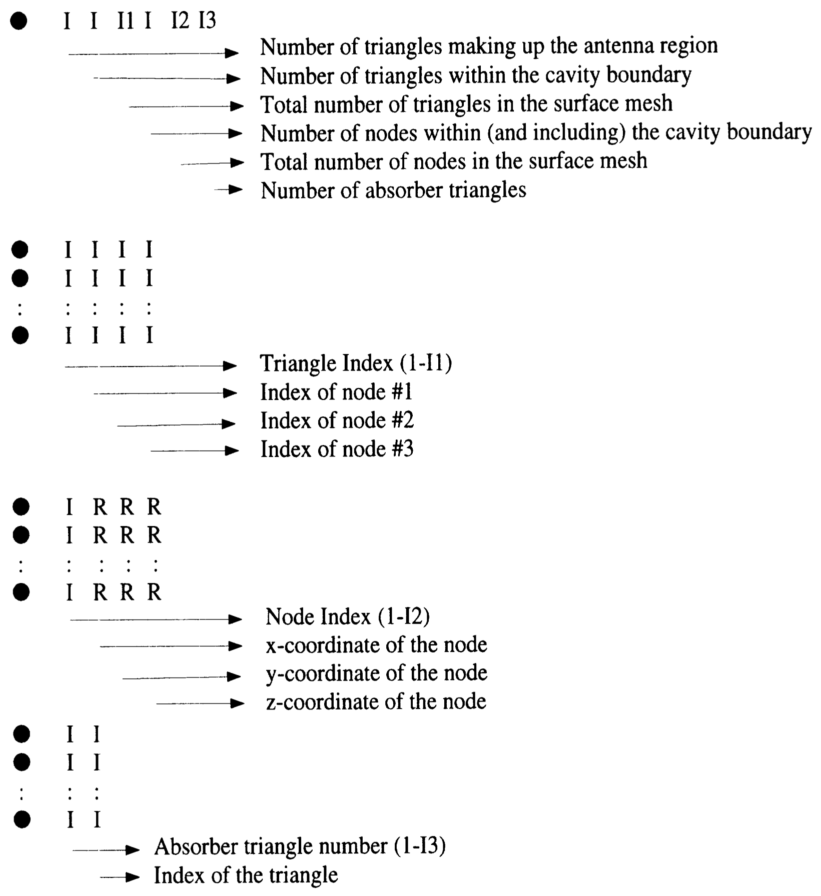

I I I1 I I2 I3
Number of triangles making up the antenna region
Number of triangles within the cavity boundary
Total number of triangles in the surface mesh
Number of nodes within (and including) the cavity boundary
Total number of nodes in the surface mesh
Number of absorber triangles
I I I I
I I I I
I I I I
Triangle Index (1-I1)
Index of node #1
Index of node #2
Index of node #3
I R R R
I R R R
I R R R
Node Index (1-I2)
x-coordinate of the node
y-coordinate of the node
z-coordinate of the node
I I
I I
I I
Absorber triangle number (1-I3)
Index of the triangle

Figure 4: Input file *SurfMesh*

with the region between the cavity and the termination boundaries. This provides a simple way of identifying the antenna and cavity patches. For example, if N is the number of triangles making up the antenna region, triangles numbered 1 through N are the antenna triangles, etc.

The information about the coordinates of the nodes follows with each row corresponding to a node. Coordinates are in units of Centimeter. **It is also important to note that the nodes over the cavity aperture are numbered first.** This simplifies the volume indexing for cavity-backed antennas.

The last section of the file identifies the triangular patches making up the absorber section which is the outer skirt of the surface mesh.

12 Viewing the surface mesh using MatLab

For determining the node location of feeds and to inspect the mesh quality, it is useful to view the surface mesh. In fact, this is a must if the built-in mesh generator is used to create the mesh unless one is familiar with the workings of the mesh generator.

The file *MeshPlot* contains a MatLab program for viewing the surface mesh. To run the code, execute the line commands in the given order. There are five seperate sections. The first section consists of six lines of commands, and it reads in the file *MeshDsply* (which has the mesh data) and sets up the screen. Before executing these commands, variables "nt" and "nn" (second and third lines) must be set to the number of triangles and number of nodes in the mesh, respectively. The following four sections fall under the titles *Display Mesh*, *Triangle Numbering*, *Global Node Numbering*, and *Local Node Numbering.* The functions of these sections are self-explanatory and sections can be executed in any order.

Index of each triangle is indicated at the center of the triangle and the index of each node is shown at the location of the node (the lines stop short of converging at the nodes in order not to cross over the text). The local indexing of three nodes of each triangle is indicated (as 1, 2, or 3) counter-clock wise just inside that node within the repective triangle. With all this information on the screen, the picture can look too crowded. In these cases, zooming on a particular section of the mesh is necessary for clarity. To do this just type *zoom* in the command window and click the part of the screen that interests you (with the left mouse botton).

From the first line of the program, one sees that it reads in the file *MeshDsply* which contains the node indexing and coordinates. *MeshDsply* is generated each time the code is run with user-supplied mesh option. If the built-in mesh generator is used, it is created at the same time as *SurfMesh.*

It should be noted that this plotting scheme is only useful for planar meshes. When the surface mesh defines a three dimensional curved surface , the MatLab program will display its projection onto the $x - y$ plane since it considers only the x and y coordinates of nodes.

13 Output file *EqvCur*

EqvCur is the file containing the equivalent magnetic currents (radiating in free-space) distributed across the platform surface. They need to be integrated to obtain the radiated field. They are the true currents in the case of planar antennas, and the Fortran code *FarField.f* can be used to integrate them for far field evaluation. In the case of non-planar platforms, the equivalent currents are local approximations to the true quantities.

The equivalent magnetic currents have been computed from the apeture electric field using $\mathbf{M} = 2\mathbf{E} \times \mathbf{n}$ where $\mathbf{n}$ is the surface normal pointing away from the surface. The factor of 2 comes from removing the ground plane (assuming locally flat), implying that without the factor of 2, the currents radiate in the presence of the ground plane. Hence, given the ability to radiate the currents in the presence of the ground plane, the quantities in the file *EqvCur* must be divided by 2. This is possible for planar, cylindrical, spherical and conical ground planes.

The actual computation of the currents is carried out by averaging the electric field vector over each non-zero triangular patch (using the basis functions). The resulting vector is then crossed with the surface normal and the outcome is the average value of the equivalent magnetic current vector over that triangular patch. The file *EqvCur* has as many lines (rows) as the number of non-matallic patches over the surface on which the antenna resides. Except for the first row which indicates the number of patches (on which **M** is given) and the free-space wave number, each successive row contains the patch number, the area of the patch in cm^2, the (x, y, z) coordinates of the center of the patch (computed by averaging the coordinates of the three patch nodes), and the complex amplitudes of the (x, y, z) components of the magnetic current multiplied by a factor of 2 as noted earlier. The Fortran code *FarField.f* reads the magnetic current data in this format. Thus to compute the radiated field *FarField.f* must be executed with *EqvCur* as the input file.

14 Output file *EdgeUnk*

The file *EdgeUnk* stores the values to all edge unknowns. Needless to say, it has as many rows of information as the total number of edge unknowns. The first column is the index of the edge. The next six columns are the (x, y, z) coordinates of the end nodes of the edge. The second set of coordinates belong to the node toward which the edge points. The next column is the magnitude of the electric field vector (the unknown) which is parallel to and constant along the edge. The last two columns are the real and imaginary parts of the complex amplitude of the electric field unknown, respectively. All field quantities are in units of $Volts/cm$.

The frequency for which this information is saved is determined by the first integer entry of the frequency information row of the input file *MainInput* (see Figure 2). If the entry is zero, the file is not stored. This is often the user choice as the file takes up a substantial amount of memory and should be saved only if needed. The non-zero value of the entry specifies which frequeny run to save. The number of frequency runs are determined by the first three real entries of the same row.

15 Output file *Imp*

The file *Imp* stores the input impedance measured at the locations of the probe feeds. Input impedance is calculated as $Z_{in} = -E\,l\,/\,I$ where E is the complex amplitude of electric field unknown along the edge coinciding with the probe (in $Volts/cm$), l is the probe length (in $Centimeters$) and I is the complex amplitude of the probe current (in $Amperes$). The resulting impedance value is in units of $Ohms$. Here it has been assumed that the current in the probe flows in the direction of that edge.

The first column of the file is the frequency (in GHz), the second column is the probe number (in the order specified in the input file *MainInput*), the next two columns are the real and imaginary parts of the impedance (in $Ohms$), respectively, and the last column is the number of iterations the BiCG solver had to carry out for convergence. The same information is also dumped on the screen while the code is running. This is a very useful product of the code as it can be used to predict the resonance frequency of the antenna (the frequency at which the input impedance is purely real).

16 Output file *ElmMat*

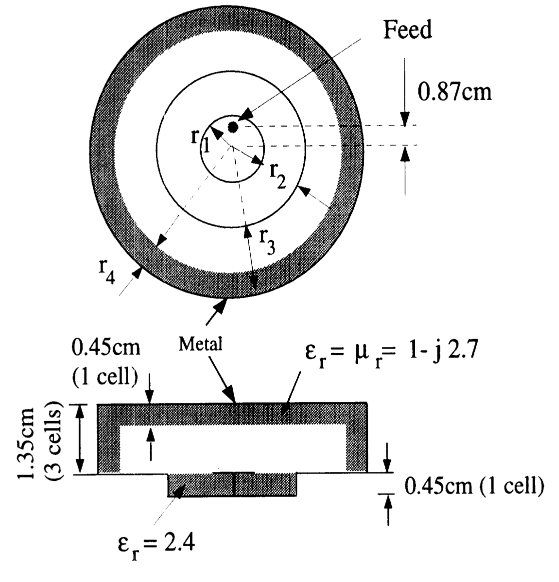
The file *ElmMat* stores the element matrices associated with the prisms making up the volume mesh. They are stored by the code every time a new volume mesh is introduced. Obviously, as long as the mesh stays the same so do the prisms and the element matrices for that matter. Therefore, if they are saved the first time the mesh is created, they are simply read. One disadvantage is that the file requires substantial memory space. The number of stored real entries (each with twelve decimals) is $162 \times Numberofprisms$ in the mesh. As shown in Figure 2, whether the data in *ElmMat* are computed or read in depends on the last entry of the frequency information row of the input file *MainInput.*

17 Demonstration runs

This section contains two demonstration runs which the user must carry out to insure that the code is working properly. For each run, the corresponding input/output files are provided in respective directories.

Demo #1: Cavity-backed circular patch

Before proceeding with the rest of the section, the reader must be advised that these demonstration runs are intended to show the operation of the code and should not be used as a measure of the code's accuracy. For example, both the thickness and the distance of the absorbing layer have been chosen half or one third of what they should



r_1 = 1.3cm (3 rings)
r_2 = 1.3cm (3 rings)
r_3 = 1.3cm (3 rings)
r_4 = 0.43cm (1 ring)

(a)

(b)

MainInput (for mesh generation)

```
1
3.9  3  6  9  1
```

MainInput (for run)

```
1
1  1  0  0
.45  (2.14, 0.)  (1., 0.)
1
8  8  0  (1., 0.)
3.  4.  .1  6  .01  0  1
1
.45  3  1  (1., -2.7)
```

(c)

Output on the screen:

```
SURFACE EDGE INDEXING ...
NUMBER OF ANTENNA EDGES= 90
NUMBER OF CAVITY EDGES= 342
TOTAL NUMBER OF EDGES= 756
VOLUME NODAL AND EDGE INDEXING ...
NUMBER OF PRISMS= 1674
NUMBER OF GLOBAL EDGES= 4306
NUMBER OF GLOBAL NODES= 1211
NUMBER OF BOUNDARY EDGES= 54
NUMBER OF BOUNDARY NODES= 54
NUMBER OF CAVITY BOUNDARY EDGES= 36
NUMBER OF CAVITY BOUNDARY NODES= 36
NUMBER OF GLOBAL METAL EDGES= 1944
NUMBER OF ABSORBER PRISMS= 690
NUMBER OF NON-ZERO EDGES= 2362
COMPUTING ELEMENT MATRICES ...
0 % Done
10 % Done
20 % Done
 :
80 % Done
90 % Done
100 % Done
Begin Frequency Sweep

 Freq(GHz)  Feed #  Re(Zin)    Im(Zin)   # of Iterat.

  3.0000     1       7.8512    59.1441    223
    :
  3.5000     1      46.6215   114.9450    222
  3.6000     1      81.9289   128.6968    221
  3.7000     1     142.2111   115.9399    228
  3.8000     1     181.5739    39.8753    227
  3.9000     1     141.1083   -28.9999    231
  4.0000     1      89.7684   -41.3362    225
                                    Resonance
```

Figure 5: Cavity-backed circular patch: (a) Geometry of the patch and finite element-artificial absorber modeling, (b) contents of the input files, (c) screen dump produced by the code and the input impedance as a function of excitation frequency.

be to minimize the geometry and hence the CPU time required by each frequency run. For proper modeling of the antennas, both the thickness and the distance of the absorber must be at least $0.15\lambda_o$ at the operation frequency.
The first demonstration example is a cavity-backed circular patch. Figure 5(a) shows the patch and termination geometry. Note that in order to save CPU time, only one layer of absorber ($0.05\lambda_o$ thick) has been employed and placed about $0.1\lambda_o$ away from the cavity surface and walls. Figure 5(b) shows the contents of the input file for both the mesh generation and the actual run. Figure 5(c) shows the screen dump of the code after the run. An inspection of the frequency sweep shows that the antenna is resonant at $3.85GHz$. The input/output files are provided in electronic form in the directory "Demo1".

Demo #2: Microstrip rectangular patch with two layers of overlay

The second case is a microstrip rectangular patch with two layers of superstrate. Similar information for this patch is given in Figure 6. Note in Figure 6(a), the absorber sections that are in direct contact with the substrate and superstrate layers are colored differently to indicate that they have different material constants and the code automatically determines the permittivity and permeability of these sections in such a way that the waves normally incident on these sections of the absorber are totally absorbed, i.e., the wave impedances ($Z = \sqrt{\frac{\mu}{\epsilon}}$) on both sides of the interface are matched. This is clearly shown in Figure 1. Notice that the absorber section in contact with air have its relative permittivity and permeability equal to each other ($\epsilon_r = \mu_r$) resulting in the wave impedance inside the absorber section's being equal to that of the free-space (since $\sqrt{\frac{\mu}{\epsilon}} = \sqrt{\frac{\mu_r\mu_o}{\epsilon_r\epsilon_o}} = \sqrt{\frac{\mu_o}{\epsilon_o}} = Z_o$). As before, Figure 6(b) shows the contents of the input files for both mesh generation and actual run, and Figure 6(c) shows the screen dump created by the code after the run. The data show that the patch resonates at $5.05GHz$. The input/output files are provided in electronic form in the directory "Demo2".

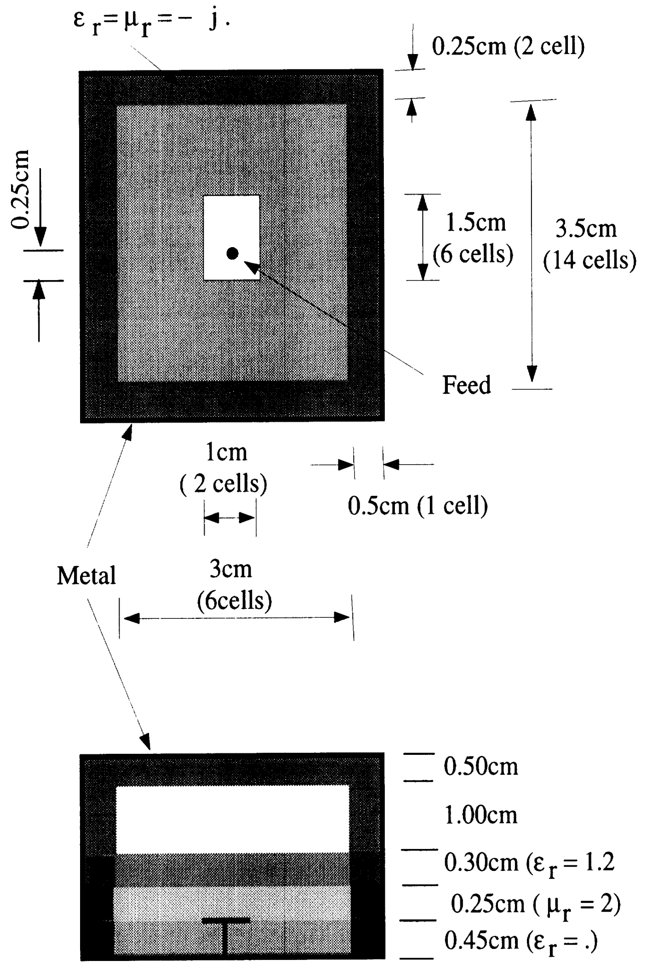
Note that, the input files for the first runs (for mesh generation) are not provided in electronic form since they can easily be copied from the figures. Also the files *ElmMat* and *EdgeUnk* are not provided due to memory restrictions.

18 Distribution disk and installation of the code

Below is the directory list of the distribution list:

README	: Text file containing brief information about the distribution disk
FEMA=PRISM.f	: Source code for FEMA-PRISM
FarField.f	: Source code for Far Field evaluation

(a)



(b)

MainInput (for mesh generation)

```
3
0.5 0.25 2 6 3 6 0 0 1 2
```

MainInput (for the actual run)

```
1
1 1 2 0
.45 (2.14,0.) (1.,0.)
0.25 (1.,0.) (2.,0.)
0.3 (1.2,0.) (1.,0.)
1
23 23 0 (1.,0.)
4.5 5.5 .1 6 .01 0 1
1
.45 3 1 (1.,-2.7)
```

(c)

Screen dump

```
SURFACE EDGE INDEXING ...
NUMBER OF ANTENNA EDGES= 44
NUMBER OF CAVITY EDGES= 458
TOTAL NUMBER OF EDGES= 458
VOLUME NODAL AND EDGE INDEXING ...
NUMBER OF PRISMS= 1728
NUMBER OF GLOBAL EDGES= 4232
NUMBER OF GLOBAL NODES= 1197
NUMBER OF BOUNDARY EDGES= 52
NUMBER OF BOUNDARY NODES= 52
NUMBER OF CAVITY BOUNDARY EDGES= 52
NUMBER OF CAVITY BOUNDARY NODES= 52
NUMBER OF GLOBAL METAL EDGES= 1532
NUMBER OF ABSORBER PRISMS= 888
NUMBER OF NON-ZERO EDGES= 2700
COMPUTING ELEMENT MATRICES ...
0 % Done
10 % Done
20 % Done
 :
90 % Done
100 % Done
Begin Frequency Sweep

            Freq(GHz) Feed #  Re(Zin)    Im(Zin)   # of Iterat.
             4.5000     1     32.6467   103.0583     411
             4.6000     1     46.5897   116.9454     451
             4.7000     1     70.2919   131.6365     426
             4.8000     1    110.8929   140.4605     436
Resonance    4.9000     1    170.3384   121.6465     505
             5.0000     1    211.6461    49.7454     504
             5.1000     1    184.2637   -28.5242     459
             5.2000     1    129.2563   -57.3716     428
             5.3000     1     88.1359   -55.3745     342
             5.4000     1     62.9073   -44.5909   . 328
             5.5000     1     47.5992   -32.9925     321
```

Figure 6: Microstrip rectangular patch with multiple superstrates: (a) Geometry of the patch and finite element-artificial absorber modeling, (b) contents of the input files, (c) screen dump produced by the code and the input impedance as a function of excitation frequency.

MeshPlot	: File containing the MatLab program
Demol	: Directory containing the files for Demo #1
Demo2	: Directory containing the files for Demo #2

Demol:	
MainInput	: File *MainInput* for Demo #1
SurfMesh	: File *SurfMesh* for Demo #1
MeshDsply	: File *MeshDsply* for Demo #1
EqvCur	: File *EqvCur* for Demo #1
Imp	: File *Imp* for Demo #1

Demo2:	
MainInput	: File *MainInput* for Demo #2
SurfMesh	: File *SurfMesh* for Demo #2
MeshDsply	: File *MeshDsply* for Demo #2
EqvCur	: File *EqvCur* for Demo #2
Imp	: File *Imp* for Demo #2

The disk is formatted on a Power Machintosh. The contents of the disk should be loaded into the working directory. No extra effort needed to install the code. Section 6 explains how to run the code.

References

[1] Özdemir, T. and J. L. Volakis, "Triangular prisms for edge-based vector finite element antenna analysis," *Radiation Laboratory Tech. Rep. No. 031307-4-T*, Radiation Laboratory, Dept. of Elect. Engr. Comp. Sci., Univ. of Michigan, Ann Arbor, Michigan 48109-2122, March 1995.

[2] Özdemir, T., J. Gong, S. Legault, J. Volakis, T. Senior, J. Berrie, R. Kipp and H. Wang, "Modeling of conformal antennas on doubly curved platforms and their interactions with aircraft platforms," *Annual Progress Report, Tech. Rep. No. 031307-5-T*, Radiation Laboratory, Dept. of Elect. Engr. Comp. Sci., Univ. of Michigan, Ann Arbor, Michigan 48109-2122, October 1995.