HYBRID FINITE ELEMENT AND MOMENT METHOD SOFTWARE FOR THE SERAT ARRAY

7th Quarterly Report

Sanders, A Lockheed Martin Co.
95 Canal Street NCA1-6268
P.O. Box 868
Nashua, NH 030601-0868

July 1998

PROJECT INFORMATION

PROJECT TITLE: Hybrid Finite Element Design Codes for the SERAT Array

REPORT TITLE: 7th Quarterly Report

U-M REPORT No.: 035067-10-T

CONTRACT
START DATE: October 1996
END DATE: September 1998

DATE: July 10, 1998
(7th Quarterly Report)

SPONSOR: Roland Gilbert
SANDERS, INC, A Lockheed Martin Co.
MER 24-1583
PO Box 868
Nashua, NH 030601-0868
Phone: (603) 885-5861
Email: RGILBERT@mailgw.sanders.lockheed.com

SPONSOR
CONTRACT No.: P.O. QP2047

U-M PRINCIPAL
INVESTIGATOR: John L. Volakis
EECS Dept.
University of Michigan
1301 Beal Ave
Ann Arbor, MI 48109-2122
Phone: (313) 764-0500 FAX: (313) 747-2106
volakis@umich.edu
http://www-personal.engin.umich.edu/~volakis/

CONTRIBUTORS
TO THIS REPORT: *T.* Eibert(UM), Y. Erdemli (UM), D. Giszczak(UM), K. Sertel(UM), , D. Jackson(UH), J. Volakis(UM) and D. Wilton(UH)

TABLE OF CONTENTS

CHRONOLOGY of Events (Updated Every Quarter)

- April 1996 — Proposal Submission
- July 1996 — Answers to Proposal Questions
- August 1996 — Began Contract Negotiations
- 20 Sept. 1996 — Kickoff meeting at Ann Arbor (attended by Sanders, UM and UH)
- October 1996 — Contract Signed between U-M and Sanders in Mid October
- October 1996 — Subcontract to the Univ of Houston (formalized in early November)
- 15 Nov. 1996 — **SERAT Review meeting** (at Nashua)
- 9 January 1997 — **Submitted First Quarterly Report**
  Report Described Code Plan and Progress on the Moment Method FSS Code. Specifically, a new scheme was developed to accelerate the convergence of the periodic Green's function
- 28 February 1997 — **Prepared viewgraphs on the project's progress review**.
  Showed first validation results for the moment method FSS code with the new accelerated Green's function; showed results for a new algorithm to accelerate the boundary integral truncation of the planar and curved FSS hybrid FEM code using the Adaptive Integral Method(AIM) and CVSS, the new LU solver specialized to sparse matrices
- 5 April 1997 — **Submission of Second Quarterly Report.**
  Report included the first validation results for the stand alone small array FEM code (with dipole FSS elements and dipole antenna elements). A similar validation was done for the moment method FSS developed at Houston. The fast AIM algorithm was described for boundary truncation and the TRIANGLE surface mesher was introduced to generate the aperture mesh, subsequently grown down to the FSS.
- 30 May 1997 — **Semi-annual review at the Univ. of Michigan**
  (attended by all parties)
  Review covered progress up-to-date. At this meeting, emphasis was on the validation of the three codes which took 'shape and form' between March-May 1997 in accordance with the proposed schedule. Theory, validations and comparisons among the codes were presented.
- 25 June 1997 — **SERAT review Meeting** (Nashua)
- 5 July 1997 — **Submission of Third Quarterly Report**
  The major component of this report was the description and validation of the periodic hybrid FEM code, FSS-PRISM. Comparisons among the FSS-EIGER, FSS-BRICK and FSS-PRISM were given for the first time.

Also, the geometry drivers for the FSS-BRICK and FSS-EIGER were given.

- 6 Oct 1997 — Submission of Fourth Quarterly Report
  Delivered FSS-EIGER and FSS-BRICK, both with manuals and preprocessors. Primitives are used to specify antenna and FSS elements in each code. Report gives example calculations for non-commensurate FSS panels for the first time using the scaling approach implemented in FSS-PRISM. Another first is the hybridization of the fast multipole method with the finite element code PRISM as a step toward the curved FSS modeling.
- 24 Oct. 1997 — **Review at the Univ. of Michigan**
  Attended by U-M and Houston project people, Gibert, Pirrung and Asvestas.
  Presented status of FSS-BRICK, FSS-EIGER, FSS-PRISM and FSS-CURVE.
  Delivered manuals for FSS-EIGER and FSS-BRICK (and codes); Successes for non-commensurate array modeling were presented (5 layer example); Update on FSS_EIGER for non-commensurate was given; Initial implementation of finite curved array code was presented with the fast multipole method for mesh truncation.
- Dec 1997 — **5th Quarterly Report**
  The major highlights of this progress report are
  - Implementation of the fast integral method into FSS-PRISM, making prism a practical analysis code, even when the sampling requirements are very high
  - Completion of FSS/Antenna geometry Driver for FSS-PRISM
  - First implementation of curved arrays with the FMM for mesh truncation.
  - Additional validations for non-commensurate FSS
- April 1998 — **6th Quarterly Report**
  The major highlights of this progress report are
  - Delivered FSS-PRISM for modeling SERAT antennas and FSS
  - Delivered Users Manual for FSS-PRISM. Manual included several test cases (input files and examples)
  - Delivered FSSBUILD, the preprocessor to FSSEIGER, the SERAT moment method code developed at the Univ. of Houston.
  - FSSEIGER
  - Curved SERAT code (FSS-CURVE) can now handle transmission and reflection coefficient computations. It has also been upgraded to include lumped loads
- July 1998 — **7th Quarterly Report**
  The major highlights of this progress report are
  - Extensive validation of FSS-CURVE for finite curved arrays

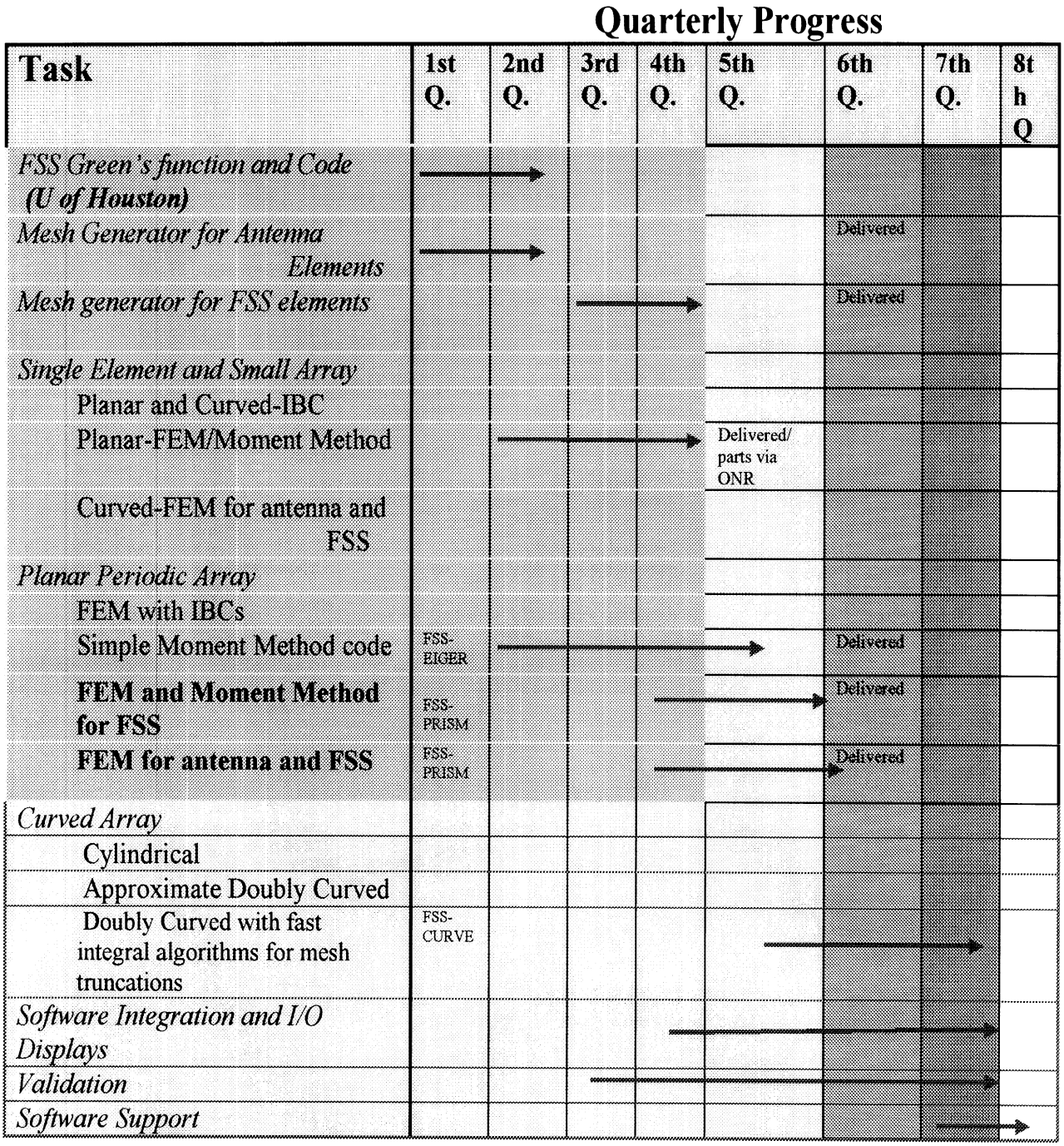
- Developed I/O interface and capabilities of FSS-CURVE to conform to those of FSS-PRISM
- Delivery of FSS-CURVE will be in July 1998
- Developed Windows tools to display fields and grids of the FSS-PRISM input and outputs at each layer, including possible animation as the field propagates through the FSS or antenna element.
- Improved FSS-PRISM manual and included description of the Windows/PC display tools
- Developed and tested a new fast method for mesh truncation which demonstrates a 100 fold speed-up even for small systems. This should remove the drawback of FE-BI systems for small and simple geometries.

MEETINGS

- Presentation of this work were given at an ONR/DARPA meeting in June 1998

- Three presentation/papers related to this work were given at the 1998 IEEE Antennas and Propagation Symposium held end of June in Atlanta, GA (these presentations are attached)

UPDATED MILESTONE CHART (page 7)



Executive Summary and Project Status

We continue to be on schedule with all activities and code development tasks (refer to page 7). Last quarter we delivered all code modules for FSS-PRISM and FSS-EIGER one quarter ahead of schedule. The only remaining code that must be delivered is the finite curved array simulator FSS-CURVE. The development of FSS-CURVE was recognized to have the highest risk among all other modules. Nevertheless, its development was carried out as expected and with no loss of generality. Last quarter, FSS-CURVE was extensively validated for single and multi-element array structures as illustrated in Figure 1 to Figure 3. Also, the I/O of FSS-CURVE was configured to be compatible with FSS-PRISM. Its delivery will take place during July 1998, and will be the last module to be delivered except for upgrades.

During the last quarter we also developed a Windows/PC viewer of the FSS-PRISM geometry and the field distributions across each layer along the depth of the FSS or substrate. The viewer reads the geometry and output files of FSS-PRISM and displays these at any desired orientation which can be changed using the usual mouse buttons (see Figure 4) . Various options of grid/color/polarization displays are alos available. This viewer is provided on the enclosed diskette and more detailed description of its capabilities is given later.

During the past three months we also worked with Sanders in support of porting and validating FSS-PRISM at Sanders's facilities. Several clarifications to the manual were done as a result of our interactions with Sanders. A more detailed account of the changes/recommendations to the manual are given later.

A second technical report and paper was written during the past 3 months on the fast integral method implementation incorporated within FSS-PRISM. This is technical report 035067-11-T to be submitted in July. This report demonstrates the CPU requirements of FSS-PRISM and the speed-up afforded by AIM, the employed fast integral method.

Perhaps the most important development during the past quarter was the introduction of a new fast integral method which led to the lower, O(N), CPU and memory requirements. The speed-up was 2 orders of magnitude even for very small systems and this is in contrast to all other fast methods which provide speed-ups only for large systems (see **Figure 5**). The technique, referred to as FSDA, is discussed at a later section and was validated during the past month for various geometries. We have no plans to fully incorporate FSDA in a deliverable module by the end of this contract. However, the demonstrated speed-up makes it very attractive and essential for future design applications.

We may actually state that FSDA removes the usual advantage of moment method codes for a class of simplistic FSS and array geometries.

Summary of Code Modules and Capabilities

CODE	DESCRIPTION AND STATUS
FSS-BRICK	**Small (finite) Array Code** • Finite element-Boundary Integral (FE-BI) code using brick elements • Completed and Validated • Geometry Driver described in Report 035067-5-T
FSS-EIGER	**Moment Method code** • Uses multilayered Green's function and triangular boundary elements (Rao-Wilton-Glisson formulation) • Uses Ewald acceleration for periodic Green's function • Validated (slot and dipole elements) and delivered for commensurate FSS • Geometry Driver Manual delivered in October 1997 • Geometry Driver Manual Updated April 1998 and Delivered • FSSBUILD (FSS-EIGER geometry Driver) delivered April 1998. • FSS-EIGER to be delivered through ONR by Univ of Houston • FSS-EIGER is capable of Non-commensurate FSS modeling • FSSBUILD is being upgraded for entering geometries of non-commensurate FSS/arrays
FSS-PRISM	**Finite Element-Boundary Integral Code** • Combines flexibility of finite elements for volume and boundary element for robust mesh truncation • Uses Ewald acceleration for periodic free space Green's function • Incorporates Adaptive Integral Method for fast Boundary element evaluation • Uses layer de-coupling to handle non-commensurate FSS. • Validated for commensurate and non-commensurate FSS and slot and printed antenna elements • Geometry Driver and code were delivered April 1998 on 2 disks (one

for commensurate and another for non-commensurate)
- Manual with test case report were delivered April 1998 (see UM Radiation Lab Report 035067-7-T)
- Theory and many test cases are described in UM Radiation Lab Report 035067-9-T

FSS-CURVE

Curved SERAT array code

- Based on the non-periodic version of PRISM
- Incorporates the fast multipole method (FMM) for fast non-planar boundary integral evaluation
- Tested for planar and cylindrical array simulations
- Transmission and Reflection coeff. can be extracted.
- Incorporates lumped loads (horizontal and vertical)
- FSS Geometry driver is in progress
- Resistive card models are not yet validated.

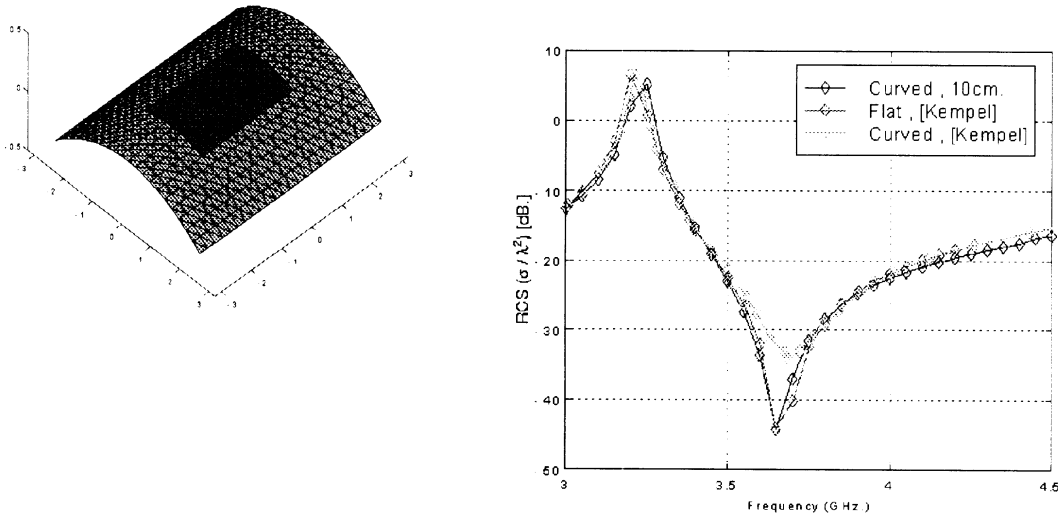


Figure 1. Validation of FSS-CURVE for a patch on a 10cm radius cylinder. The patch is 2cmx3cm and is situated in a cavity 5cmx6cm in aperture and 0.078cm deep, filled with a dielectric having ε_r=2.17.

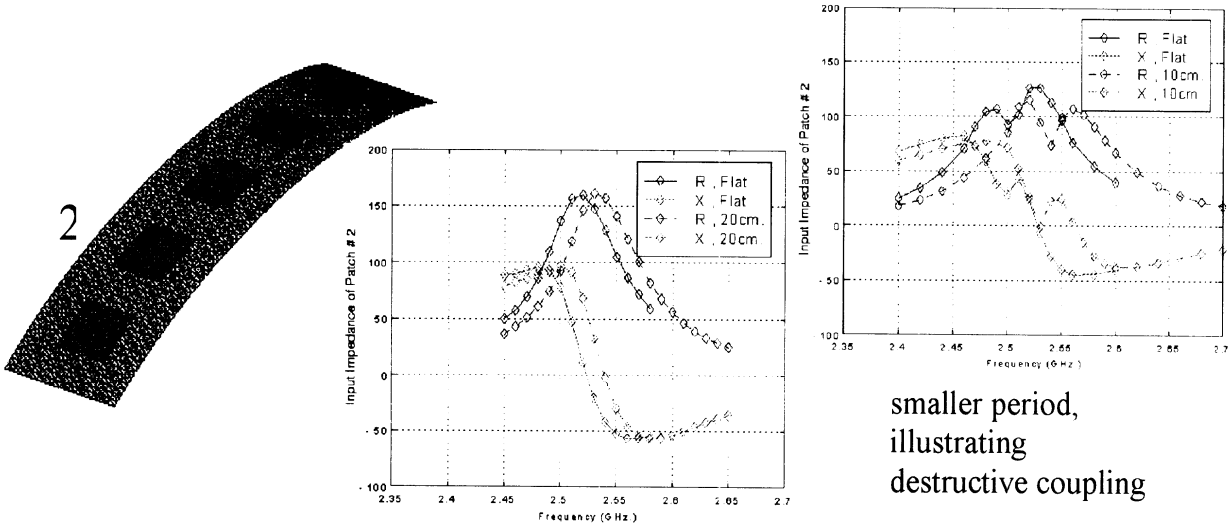


Figure 2. Input impedance of a 5-element patch array (real and reactive) on a 20cm radius cylinder as illustrated. The array patches were 3.5cm. by 2.625cm and the period was 6.125cm. The overall aperture on the cylinder was 33.25cm x 6.125cm resulting in 3300 unknowns of which 2200 were associated with the boundary integral (moment method portion) on the boundary. Note that a smaller period results in substantially coupling which causes major changes to the input impedance of the no. 2 antenna element (as shown).

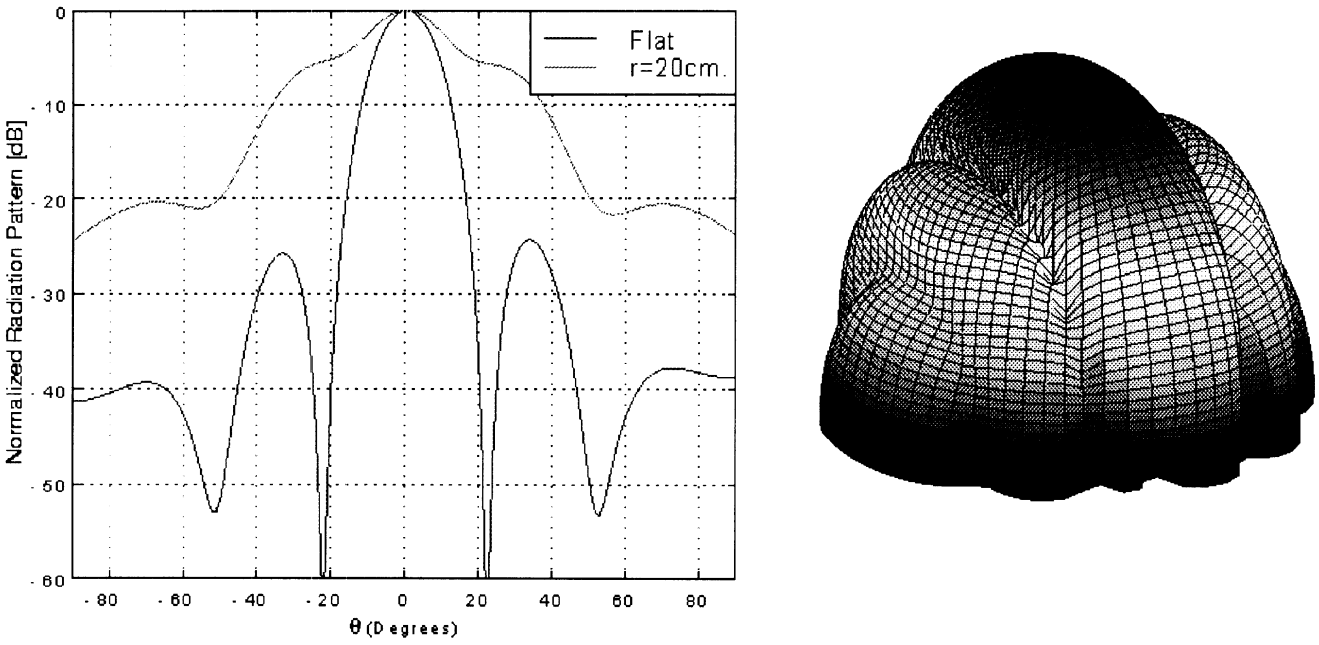


Figure 3. Comparison of radiation patterns of the 5-element array when situated on a flat and curved platform. As expected with the expect the pattern broadening for the curved case.

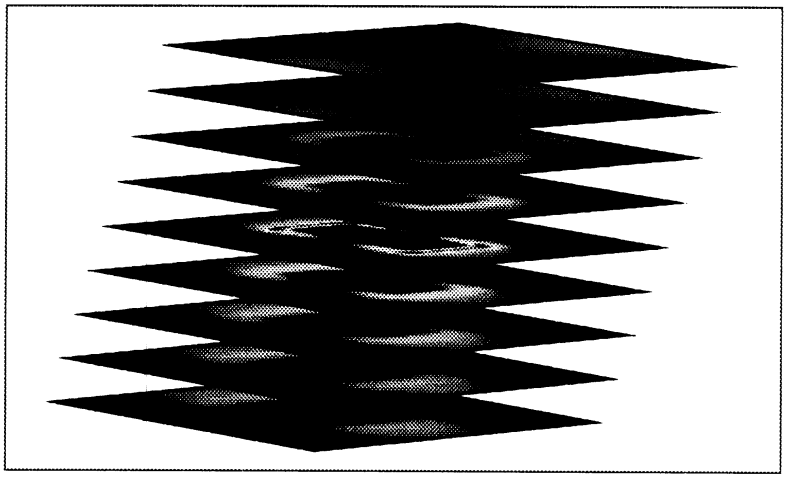


Figure 4. Fields in the FSS layers as displayed by the FSS-PRISM viewer

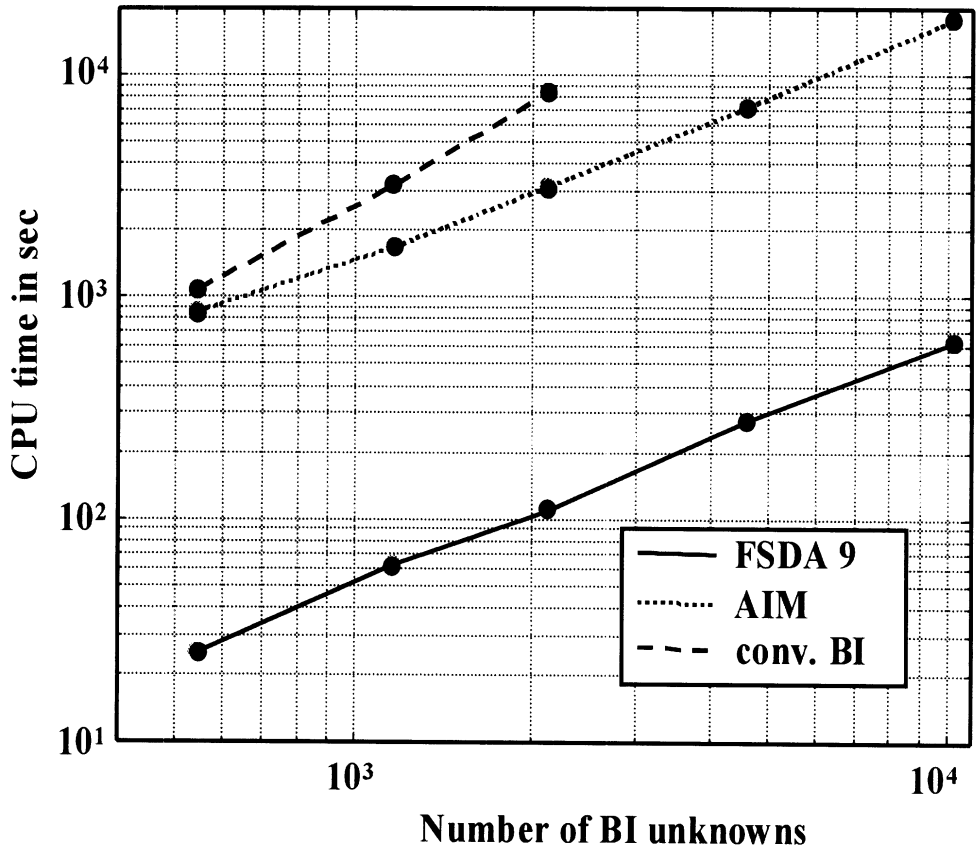


Figure 5. Speed-up curves of FSDA when compared to the traditional BI approach and AIM

Project Goals

The goal of the SERAT project at the University of Michigan (with subcontract to Univ. of Houston) is to develop a suite of software for the analysis of strip and slot dipoles on multilayered substrates backed by a frequency selective surface. The dipoles are equipped with photonic switches permitting variable electrical dipole lengths for broadband performance and the FSS is suitably designed to simulate a variable substrate thickness for optimal operation. A general view of the geometry is given in Figure 1.

The UM/UH team proposed to construct a code which combines various computational modules interfaced with appropriate pre-processors and post-processors. The computational modules include:

- Stand-alone moment method simulation of the FSS with up to 10 layers with commensurate and non-commensurate periodicities.
- Simple moment method simulation of the antenna elements on the FSS panels
- Hybrid FEM simulation modules for small arrays, planar periodic arrays and curved arrays on FSS panels.

Various options for modeling the FSS and for mesh truncation were proposed to provide a compromise between speed and accuracy. These are outlined in the proposal and summarized in the included milestone chart (repeated from the proposal).

As called for in the milestone chart, we are proceeding in accordance with the schedule in our proposal. In most cases we are ahead of schedule by one quarter or so. Our many examples and validations are testaments to the practical utility and speed of the codes. The FE-BI codes include algorithm speed-ups based on fast algorithms. These can deliver as much are 2 orders of magnitude in CPU speed-up and memory reduction.

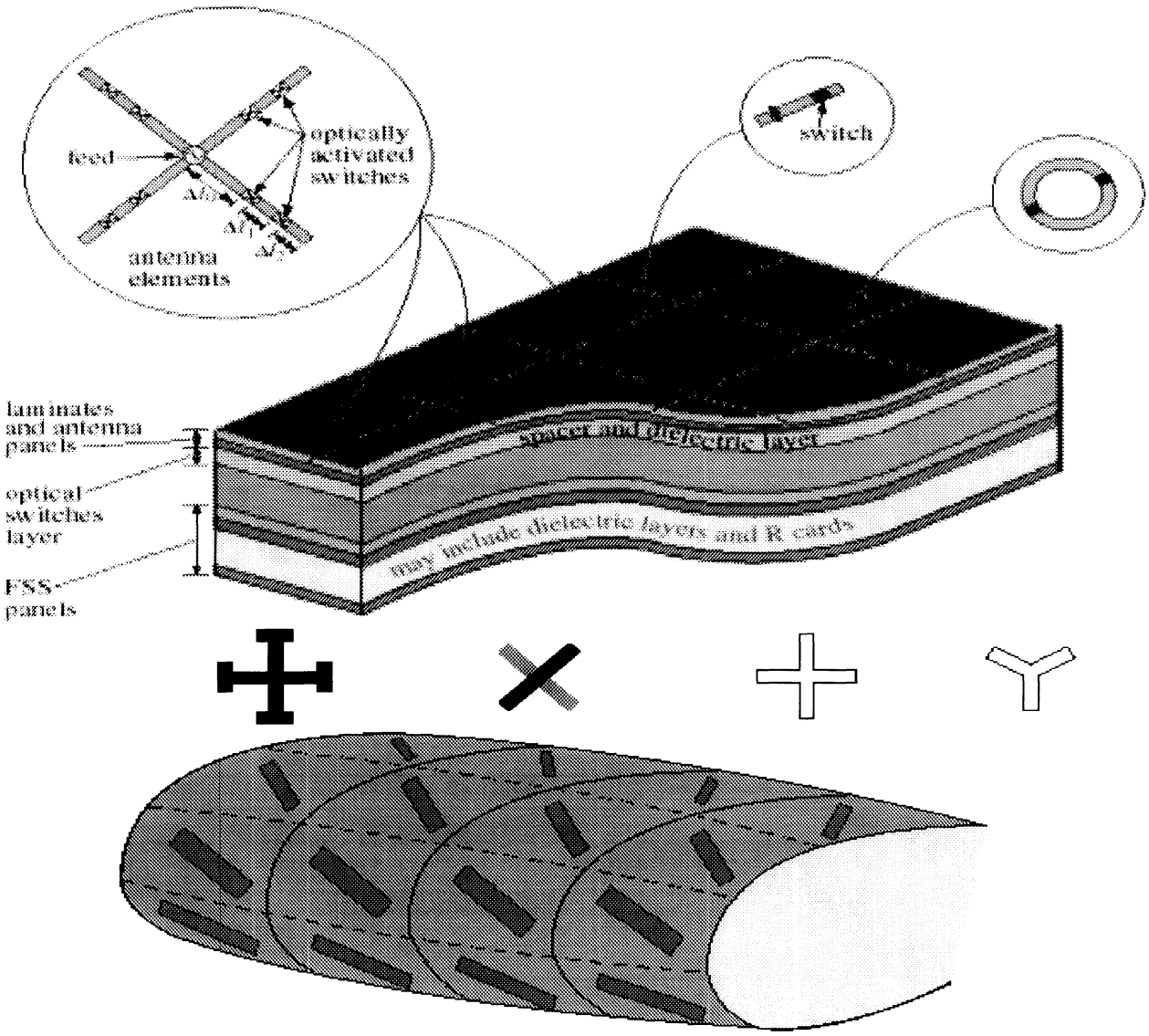


Figure 1. Illustration of the SERAT panel (Planar and Curved)

Update on FSS-EIGER

Current focus in the University of Houston FSS/EIGER modeling effort for SERAT is to correct errors in computing terms of the Green's dyad relating electric fields to magnetic currents. The terms, which are actually infinite series whose convergence behavior must be considered, appear only when slot and conducting elements are both present in a SERAT configuration; i.e., the terms in error do not affect problems involving apertures alone or conducting elements alone. However, because problems with coupling slots and conducting elements are of practical interest, correction of the error is receiving highest priority. It is anticipated that the error will be corrected within the next few weeks and that a corrected version of FSS/EIGER will be available immediately thereafter.

FSS/EIGER's capability for handling noncommensurate periodic structures is perhaps superfluous since the same capability exists in the prism-based hybrid periodic FEM/MoM code. Nevertheless, a low level, no-cost effort is proceeding to incorporate this noncommensurate modeling capability into FSS/EIGER's preprocessor, FSS/BUILD. We are also currently looking into incorporating a full three dimensional modeling capability into FSS/EIGER. This would enable, for example, nonplanar elements or feed lines to be modeled. Any such unsupported efforts which are finished before September 1, 1998 will be included in FSS/EIGER at no cost. The University of Houston continues its commitment to the development of FSS/EIGER, and continues to support the University of Michigan in its use of the code and its validation.

Update on FSS-PRISM

FSS-PRISM was delivered April 1998

The geometry driver (users manual) along with test cases and capabilities are described in the UM Radiation Laboratory Report 035067-7-T

The theory of the FSS-PRISM code is described in the UM Radiation Laboratory Reports 035067-9-T and 035067-11-T. The first describes the general finite element formulation and the latter gives a description of the speed-ups achieved using fast integral methods.

Also, the appendices of this report include copies of the slides presented in June at the 1998 IEEE Int. Antennas and Propagation Symposium.

Below we provide an account of the improvements that took place over the past three months:

During this quarter, FSS_PRISM was tested at Sanders and The University of Michigan gave technical support in running several examples. In addition to this, The University of Michigan worked on a new boundary integral termination which can give considerable speed-up as compared to the current implementation. First results with the new boundary integral were obtained and will be presented in this progress report.

Technical Support for FSS_PRISM

During the technical support phase with Sanders it became obvious that the appropriate selection of the array parameters within the code may cause difficulties. Therefore, in the following some explanations are included which can facilitate a proper choice for the array parameters.
The need for choosing the array parameters prior to compilation of the code arises from the use of Fortran 77. Fortran 77 does not know dynamic memory allocation and therefore all arrays must be dimensioned sufficiently large before the code is compiled. The goal should be to dimension the arrays as small as possible (to save memory) and as large as necessary to run the code for the given example.

For FSS_PRISM all array dimensions are defined in the file *fema.dm1* which is included in the following:

```
C ARRAY DIMENSIONS IN THE PROGRAMM
C ******************************

c number of surface nodes
      INTEGER NdmSNo
c number of nodes in the volume
      INTEGER NdmVNo
c max number of periodic nodes in a layer
```

```
      INTEGER NdmPNo
c number of triangles
      INTEGER NdmTri
c number of prisms
      INTEGER NdmPri
c number of surface edges
      INTEGER NdmSEd
c number of nonzero surface edges (larger than number in bottom or surface)
      INTEGER NdmNZS
c number of edges in the volume
      INTEGER NdmVEd
c number of nonzero edges in the volume
      INTEGER NdmNZE
c max number of periodic edges in a layer
      INTEGER NdmPEd
c number of prism layers
      INTEGER NdmLay
c number of resistive edges
      INTEGER NdmREd
c number of resisitvie triangles
      INTEGER NdmRTri
c number of z directed short circuit pins
      INTEGER NdmZPin
c number of z directed resisitve edges
      INTEGER NdmZEd
c number of matrix elements in the sparse matrix
      INTEGER NdmRow
c number of matrix elements in AIPC preconditioner matrix
c (must at least be larger than NNZE if GMRES is used)
      INTEGER NdmRowP
c size of FFTPad
      INTEGER NdmFFTPad
c number of local iterations for GMRES solver
      INTEGER NdmLoIt


      PARAMETER(
     &NdmSNo=4100,
     &NdmVNo=20420,
     &NdmPNo=72,
     &NdmTri=8000,
     &NdmPri=40000,
     &NdmSEd=10000,
     &NdmNZS=9000,
     &NdmVEd=100000,
     &NdmNZE=80000,
```

```
&NdmPEd=72,
&NdmLay=20,
&NdmREd=50,
&NdmRTri=50,
&NdmZPin=10,
&NdmZEd=10,
&NdmRow=5000000,
&NdmRowP=80000,
&NdmFFTPad=800,
&NdmLoIt=1)
```

In the Driver code, which is usually used to generate the mesh for the given geometry only some of the parameters in *fema.dm1* are used. Also, the code has a relatively low memory demand. Therefore, it is recommended to set the parameter values for the compilation of the Driver code to very large values which should not be exceeded for most of the problems to be considered.

For the compilation of FSS_PRISM, the necessary array dimensions should be approximately determined by some simple estimations.
The required value for *NdmSNo* is the very first number in the file *Data2*.
Also, the minimum value for *NdmTri* is given by the second number in the file *Data2*.
The number of layers is defined in the file *Data1*, *NdmLay* should be set to a value 1 larger than the actual number of layers. The necessay number of volume nodes (*NdmVNo*) is given by the number of layers plus one multiplied by the number of nodes in one layer. Also, the number of prisms in the volume mesh is the number of triangles in one layer times the number of layers. *NdmPNo* should be larger than the number of nodes along the longest side of the triangular surface mesh. *NdmPEd* can be set to the same value. The number of edges in the surface mesh (*NdmSEd*) is approximately 1.5 times the number of triangles in the surface mesh. The number of edges in the volume mesh (*NdmVEd*) is given by the number of surface edge multiplied by the number of layers+1 and the number of surface nodes multiplied by the number of layers.
For a more efficient implementation of the code, metallic edges (zero electric field) are removed during the course of the code. Therefore, memory can be saved by setting the number of non-zero edges (*NdmNZE*, *NdmNZS*) to smaller values than the corresponding total numbers of edges (*NdmVEd*, *NdmSEd*). Of course, this makes only sense if the number of non-zero edges is really smaller than the number of total edges. The values for (*NdmREd, NdmRTri, NdmZPin, NdmZEd*) need only be larger than zero if the corresponding mesh elements are defined for the given problem. *NdmRow* defines the maximum number of matrix elements which is usually a couple of millions. The value depends on several parameters and cannot be estimated very easily. The code starts printing messages of the form *IS* >=..... if the value of *NdmRow* is too small. In this case, the code should by stopped and recompiled with adjusted parameters. Also, for most of the other paramters error messages are given during run-time of the code if the corresponding maximum values are exceeded. In addition to the built-in checks, most compilers have a compiler mode which allows array checking during run-time. Running this makes sure that no array dimensions are violated. The value of *NdmRowP* in the file *fema.dm1* should be larger than *NdmNZE* if the GMRES solver is used. Also, *NdmLoIt* is the number of search vectors for the GMRES solver and *NdmFFTPad* is the one-dimensional size of the two-dimensional FFTPad. The actual size of the FFTPad is defined in the file *Data1*.

Another issue that was encountered during the technical support phase is related to the series representation of the periodic Green's function. The FSS_PRISM code employs a series representation of the periodic Green's function which is accelerated by the Ewald transformation. Based on this acceleration, for most practical applications summation limits from –1 to +1 are appropriate for an accurate representation of the Green's function. However, if the unit cell size becomes very large with respect to the applied wavelength the number of series terms must be increased. Typically, if the sidelength of the unit cell is larger than two wavelengths the number of series terms must be increased otherwise the code will fail to converge and the results cannot be used. Since most applications are based on unit cells with sidelengths less than one wavelength the series limits for the Ewald series are fixed to –1 to +1 within the code. If larger series limits are required the variables LIM1 and LIM2 in the EWALD subroutine in the file *bint.f* must be set to larger values (1: limits –1 to +1, 2: limits –2 to +2, ...). LIM1 and LIM2 corresponding to the spectral and spatial series contributions of the Ewald series should have the same values.

Finally, the issue of properly meshing the unit cell is of some importance. The sample rate of the finite element and boundary integral mesh should be chosen that at least 10 samples per wavelength in the considered material are reached. For the FSS_PRSIM code, this means that the sample rate in the surface mesh and also the sampling along the height of the unit cell must be adjusted according to this criterion.

Speed-up improvements using new fast integral implementation (fast spectral domain algorithm (FSDA))

Thomas F. Eibert and John L. Volakis
Radiation Laboratory, EECS Department
The University of Michigan, Ann Arbor, MI 48109-2122

1 Introduction

Application of hybrid finite element (FE) / boundary integral (BI) methods to infinite periodic structures (antennas or frequency selective surfaces) [1, 2, 3, 4] provides full 3D modeling flexibility. Basically, the FE method is employed to model a unit cell representing the array, whereas the BI provides for a rigorous mesh truncation at the upper and/or lower surfaces of the discretized unit cell. Practical problems can usually be analyzed using planar BI surfaces and the corresponding half-space periodic Green's functions. However, for large unit cell apertures, the resulting fully populated BI matrix leads to CPU intensive solutions. In contrast to pure method of moments (MoM) solutions, this is true even for relatively simple geometries. Therefore, considerable speed-up of the method can only be achieved by accelerating the BI termination. In recent years, considerable work has been spent on the development of fast integral algorithms such as the fast multipole method (FMM) [5, 6, 7, 8, 9] or the adaptive integral method (AIM) [10, 11]. Both the FMM and AIM are employed in conjunction with iterative solvers and accelerate the execution of the pertinent matrix-vector products. The FMM is based on a multipole and plane wave expansion of the free-space Green's function which allows to calculate the far-zone interactions for clusters of basis functions. AIM utilizes the Toeplitz properties of the underlying Green's function for uniform meshes and calculates the far-zone interactions in the discrete Fourier domain. That is, in both techniques the far-zone matrix elements are not explicitly generated. This is possible because in the discrete Fourier domain source and observation points are decoupled and in the FMM the coupling between source and observation points is given via the centers of the expansion function clusters. The same decoupling as given in the discrete Fourier domain is inherent to the conventional spectral domain formulation of planar surface intergral equations which is equivalent to a Floquet mode representation for array problems. In the FSDA, we take advantage of this mode decoupling to derive a fast alternative for the evaluation of the BI.

The method starts with the conventional Floquet mode representation of the BI termination for hybrid FE/BI methods; however, the MoM BI matrix elements are not explicitly calculated. Instead, within each iteration step, the Fourier transforms of the basis functions multiplied

for hybrid FE/BI methods; however, the MoM BI matrix elements are not explicitly calculated. Instead, within each iteration step, the Fourier transforms of the basis functions multiplied with their actual expansion coefficients are summed up. The spectral integral (Floquet mode series) is then calculated only once for every testing function to evaluate the matrix-vector products of the iterative solver. In contrast to AIM or FMM, this procedure does not even require the explicit calculation of the near coupling BI matrix elements. The FSDA is completely free of system matrix. Therefore, the memory demand is mostly determined by the storage of the Fourier transforms of the basis functions which are typically calculated before the iteration process is started. Thus, for fixed numbers of Floquet series terms, the memory demand is of $\mathcal{O}(n_S)$ and the complexity for the calculation of the matrix-vector products is also of $\mathcal{O}(n_S)$. More importantly, in contrast to FMM and AIM the FSDA results in considerable speed-ups of about two orders of magnitude or more even for relatively small system sizes.

2 Formulation

2.1 Basic Hybrid FE/BI Formulation

The conventional implementation of the hybrid FE/BI method for doubly periodic arrays as illustrated in Fig. 1 ($e^{j\omega t}$ time convention) leads to a linear algebraic system of the form

$$\begin{bmatrix} A^{int} & A^{cross}_{1,top} & A^{cross}_{1,bot} \\ A^{cross}_{2,top} & A^{bound}_{top} & 0 \\ A^{cross}_{2,bot} & 0 & A^{bound}_{bot} \end{bmatrix} \begin{Bmatrix} E^{int} \\ E^{bound}_{top} \\ E^{bound}_{bot} \end{Bmatrix} + \begin{bmatrix} 0 & 0 & 0 \\ 0 & Z_{top} & 0 \\ 0 & 0 & Z_{bot} \end{bmatrix} \begin{Bmatrix} E^{int} \\ E^{bound}_{top} \\ E^{bound}_{bot} \end{Bmatrix} = \begin{Bmatrix} f^{int} \\ f^{bound}_{top} \\ f^{bound}_{bottom} \end{Bmatrix}, \quad (1)$$

where only a unit cell of the array is discretized. The A–matrices are sparse (20 to 40 non-zero

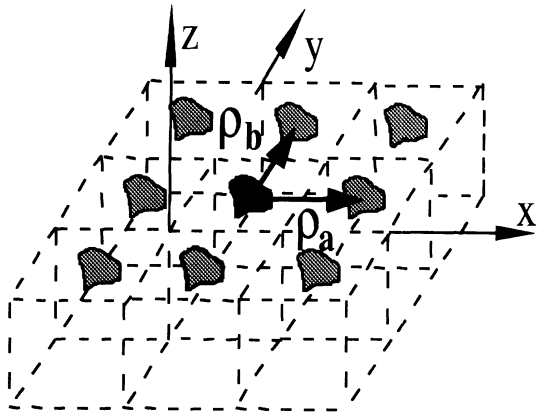


Figure 1: Doubly periodic array configuration.

elements per row) and result from the FE portion of the hybrid method. The BI Z–matrices

are fully populated and are associated with the edges on the top and bottom boundaries of the discretized unit cell. The right-hand side vector elements f represent excitations in the FE and BI portions of the method. At the side walls, the unit cell mesh is terminated by imposing phase boundary conditions (PBC) according to the Floquet theorem. In our implementation, volume tessellation is based on triangular prismatic finite elements [13]. This results in triangular surface meshes with Rao-Wilton-Glisson basis functions [14] for the magnetic currents on the planar BI surfaces. For arbitrary scan angles of the array, all matrices in (1) can be nonsymmetric due to the PBCs and the periodic Green's function. For the solution of the system, biconjugate gradient (BiCG) or generalized minimal residual (GMRES) solvers are used, both of which rely on an efficient evaluation of the matrix-vector products associated with (1) [15]. The computational complexity per matrix-vector product of the total A–matrix is of $\mathcal{O}(n_V)$, where n_V denotes the number of volume edges. However, the complexity in calculating the matrix-vector products $[Z]\{x\}$ is $\mathcal{O}(n_S^2)$ and storage requirements are of the same order. For a more efficient implementation of the method, it is therefore crucial to reduce the complexity of the BI matrix-vector products. Also, for array problems it is essential to reduce the number of the explicitly calculated BI matrix elements as far as possible since the numerical cost for generating the periodic Green's function is relatively high. That is, if a fast integral method like AIM or FMM is applied to the periodic problem the total solution time for most practical problems will mostly be determined by the calculation of the near-zone coupling elements which are needed to eliminate the approximations of the fast algorithm. Therefore, the FSDA is especially well-suited for array calculations because it is completely free of BI system matrix and it reduces the effort for computing the matrix-vector products within the iterative solver.

2.2 Fast Spectral Domain Algorithm

The FSDA replaces the conventional discretized representation of the BI in form of the Z–matrices in (1). Its derivation starts with the appropriate boundary integral for planar BI surfaces to be evaluated for the magnetic field intensity. It is given by

$$\boldsymbol{H}(\boldsymbol{r}) = -2j\frac{k_0}{Z_0}\iint_S \bar{\boldsymbol{G}}_p(\boldsymbol{r},\boldsymbol{r}_s)\cdot(\boldsymbol{E}(\boldsymbol{r}_s)\times\hat{n})\,ds_s + \boldsymbol{H}^{inc}(\boldsymbol{r})\,, \tag{2}$$

where $\boldsymbol{H}^{inc}(\boldsymbol{r})$ is an incident wave in the presence of a metallic interface in S and the periodic Green's function $\bar{\boldsymbol{G}}_p(\boldsymbol{r},\boldsymbol{r}_s)$ in the spatial domain is given by

$$\bar{\boldsymbol{G}}_p(\boldsymbol{r},\boldsymbol{r}_s) = (\bar{\boldsymbol{I}} + \frac{1}{k_0^2}\nabla\nabla)\sum_{p=-\infty}^{\infty}\sum_{q=-\infty}^{\infty} \mathrm{e}^{-j\boldsymbol{k}_{t00}\cdot\boldsymbol{\rho}_{pq}}\,\frac{\mathrm{e}^{-jk_0R_{pq}}}{4\pi R_{pq}}\,, \tag{3}$$

where

$$R_{pq} = |\boldsymbol{r} - \boldsymbol{r}_s - \boldsymbol{\rho}_{pq}|\,, \tag{4}$$

and

$$\boldsymbol{\rho}_{pq} = p\,\boldsymbol{\rho}_a + q\,\boldsymbol{\rho}_b\,. \tag{5}$$

Here, $\boldsymbol{\rho}_a, \boldsymbol{\rho}_b$ are the lattice vectors parallel to the xy–plane (see Fig. 1) and in (3) $\bar{\boldsymbol{I}}$ denotes the unit dyad. $\boldsymbol{r}$ and $\boldsymbol{r}_s$ are observation and source points and k_0 and Z_0 are the wavenumber and characteristic impedance of free space, respectively. Also, $\boldsymbol{k}_{t00}$ in (3) is given by

$$\boldsymbol{k}_{t00} = k_{x00}\,\hat{x} + k_{y00}\,\hat{y} = \pm(k_0 \sin\vartheta_0 \cos\varphi_0\,\hat{x} + k_0 \sin\vartheta_0 \sin\varphi_0\,\hat{y})\,, \tag{6}$$

where ϑ_0, φ_0 are the spherical coordinates corresponding to the scan angles of a phased array (positive sign) or the arrival angles of an incident plane wave (negative sign). In the spectral domain, (2) can be written as

$$\boldsymbol{H}(\boldsymbol{r}) = -2j\frac{k_0}{Z_0}\iint_{k_x k_y} \tilde{\bar{\boldsymbol{G}}}_p(k_x, k_y) \cdot (\boldsymbol{E}(k_x, k_y) \times \hat{n})\, dk_x dk_y + \boldsymbol{H}^{inc}(\boldsymbol{r})\,, \tag{7}$$

where the periodic Green's function spectral representation is given by

$$\tilde{\bar{\boldsymbol{G}}}_p(k_x, k_y) = \frac{1}{2jAk_0^2} \sum_{p=-\infty}^{\infty} \sum_{q=-\infty}^{\infty} \frac{1}{k_z} \begin{bmatrix} k_0^2 - k_y^2 & k_x k_y \\ k_x k_y & k_0^2 - k_x^2 \end{bmatrix} \delta(\boldsymbol{k}_t - \boldsymbol{k}_{tpq})\,. \tag{8}$$

$A = |\boldsymbol{\rho}_a \times \boldsymbol{\rho}_b|$ is the cross sectional area of the unit cell,

$$\boldsymbol{k}_t = k_x\hat{x} + k_y\hat{y}\,, \tag{9}$$

$$\boldsymbol{k}_{tpq} = \boldsymbol{k}_{t00} + \frac{2\pi}{A}\left[m(\boldsymbol{\rho}_b \times \hat{\boldsymbol{z}}) + n(\hat{\boldsymbol{z}} \times \boldsymbol{\rho}_a)\right] \tag{10}$$

is the so-called reciprocal lattice vector, and

$$k_z = \sqrt{k_0^2 - \boldsymbol{k}_t \cdot \boldsymbol{k}_t}\,, \tag{11}$$

where $\mathrm{Re}(k_z) \geq 0, \mathrm{Im}(k_z) \leq 0$. Also, " ˜ " denotes two-dimensional Fourier transforms. Expanding $\boldsymbol{E}(\boldsymbol{r})$ in terms of triangular Whitney edge elements

$$\boldsymbol{E}(\boldsymbol{r}) = \sum_{n=1}^{N} E_n \boldsymbol{w}_n(\boldsymbol{r}) \tag{12}$$

and applying MoM testing to (7) with weighting functions

$$\boldsymbol{b}_m(\boldsymbol{r}) = \hat{n} \times \boldsymbol{w}_m(\boldsymbol{r}) \tag{13}$$

which are equivalent to the well-known Rao-Wilton-Glisson basis functions [14], gives the weak form of the magnetic field intensity in the BI surface

$$\begin{aligned} \langle \boldsymbol{H}(\boldsymbol{r}), \boldsymbol{b}_m(\boldsymbol{r}) \rangle \;=\; & \sum_{p=-\infty}^{\infty} \sum_{q=-\infty}^{\infty} \tilde{\boldsymbol{b}}_m^*(k_{xpq}, k_{ypq}) \cdot \frac{1}{k_0 Z_0 A k_{zpq}} \begin{bmatrix} k_0^2 - k_{ypq}^2 & k_{xpq}k_{ypq} \\ k_{xpq}k_{ypq} & k_0^2 - k_{xpq}^2 \end{bmatrix} \cdot \\ & \sum_{n=1}^{N} E_n \tilde{\boldsymbol{b}}_n(k_{xqp}, k_{ypq}) + \left\langle \boldsymbol{H}^{inc}(\boldsymbol{r}), \boldsymbol{b}_m(\boldsymbol{r}) \right\rangle \end{aligned} \tag{14}$$

to be evaluated for $m = 1, ..., N$ and where " * " denotes complex conjugation.
Equation (14) is equivalent to one row of the matrix-vector products involving the BI submatrices in (1) and certain expansion coefficients E_n. In a conventional spectral domain BI formulation, the summation over n together with the expansion coefficients E_n are moved in front of the spectral sums (p and q) and the spectral series are evaluated for all m, n combinations to obtain the individual matrix elements of the BI submatrices in (1). That is, the spectral series must be evaluated $N_{bot}^2 + N_{top}^2$ times if N_{bot} and N_{top} are the numbers of BI unknowns on the bottom and top BI surfaces, respectively. Also, in an iterative solver the pertinent matrix-vector products require N_{bot}^2 and N_{top}^2 multiplications. Compared to this, FSDA leads to substantial speed-up because the BI matrix elements are not explicitly calculated. In accordance with FSDA, within each iteration of the iterative solver, first the summation over n is computed with consideration of expansion coefficients E_n belonging to the actual search vector within the iteration process. Consequently, the spectral representation of the whole search vector as distributed over the BI surface is obtained. Then the spectral representation of this search vector is multiplied with the dyadic Green's function in the spectral domain and finally the double spectral series is evaluated for each of the testing functions $\boldsymbol{b}_m$. Thus, whithin each iteration the spectral series is evaluated $n_{bot} + n_{top}$ times and the FSDA leads to a complexity of $\mathcal{O}(n_s)$ if it is assumed that the number of terms in the spectral series is constant. Of course, for increasing unit cell sizes this assumption is not valid and the number of series terms must be increased, but this is also the case for conventional BI implementations even when series acceleration techniques like the Ewald transformation are employed [16, 17]. The convergence of the spectral series of the FSDA is determined by the spectral distributions of the whole search vectors within the iteration process rather than the spectral distributions of the individual basis functions. As also pointed out in [2], the number of required Floquet modes can often be considerably decreased if the BI surfaces are shifted away from the inhomogeneities within the FE solution domain. The Fourier transforms of the Rao-Wilton-Glisson basis functions required for the evaluation of (14) can be calculated analytically as for instance presented in [18].

3 Results

As a first example, we calculated the power reflection coefficient of the strip-dipole array shown in Fig. 2 which was presented in [19]. FSDA results for different summation limits of the Floquet mode series are compared with results obtained by a MoM code involving a multilayered media Green's function [20]. FSDA 0 which refers to only 1 Floquet mode is unacceptable; however, FSDA 1 (summation limits from -1 to 1, 9 terms of the modal representation) is

already converged. The small frequency shift between the FE/BI results with FSDA acceleration and the MoM results was also observed for the conventional FE/BI implementation. For this problem, the unit cell was meshed using triangular prismatic elements resulting in a total number of 55370 volume and 4880 BI unknowns in each of the top and bottom BI surfaces. The CPU time requirements on a Pentium II PC/266 MHz were 244 sec, 281 sec, and 330 sec per frequency point corresponding to Floquet mode series limits of (0,0), (-1,1), and (-2,2), respectively. For the illustration of the CPU time dependence on the number of BI

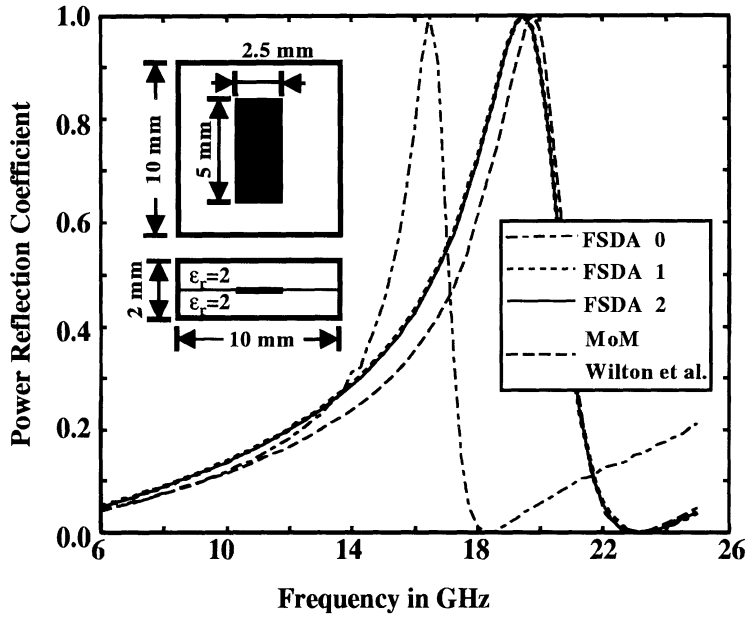


Figure 2: Power reflection coefficient of strip dipole array as presented in [19], FSDA 0, 1, 2: Floquet mode series limits of (0,0), (-1,1), (-2,2), MoM results according to [20].

unknowns, we analyzed a strip dipole array with metallic backing (BI only on top of the mesh) for plane wave incidence. The operational frequency was close to the resonance frequency of the dipoles and several dipole elements were grouped in the unit cell mesh to generate meshes with increasing numbers of BI unknowns. The largest investigated unit cell contained a 3×3 array. One prism layer was used in the height of the mesh. Again, the calculations were performed on a Pentium II PC/266 MHz and the FSDA results are compared to results obtained by a conventional and an AIM accelerated BI implementation [10, 11]. The Green's function Floquet mode series for the conventional spatial domain mixed potential BI formulation were accelerated by the Ewald transformation [16, 17]. The number of Floquet modes in the FSDA as well as the number of terms in the Ewald series were chosen appropriately for the largest analyzed unit cell (series limits -9 to 9 in the FSDA). Fig. 3 illustrates the total CPU times dependent on the number of BI unknowns. As seen, the FSDA is even for small numbers of unknowns almost two orders of magnitude faster than the conventional BI. In contrast to

this, the AIM accelerated implementation has almost the same CPU time requirements for small BI systems since it keeps the near-zone elements of the BI matrix. As it is the case for every fast algorithm, the speed advantage of the FSDA compared to the conventional BI gets increasingly more pronounced for larger problems. The total solution time factor between AIM and the conventional BI is, of course, also increasing, but the factor between FSDA and AIM keeps almost constant for increasing problem sizes.

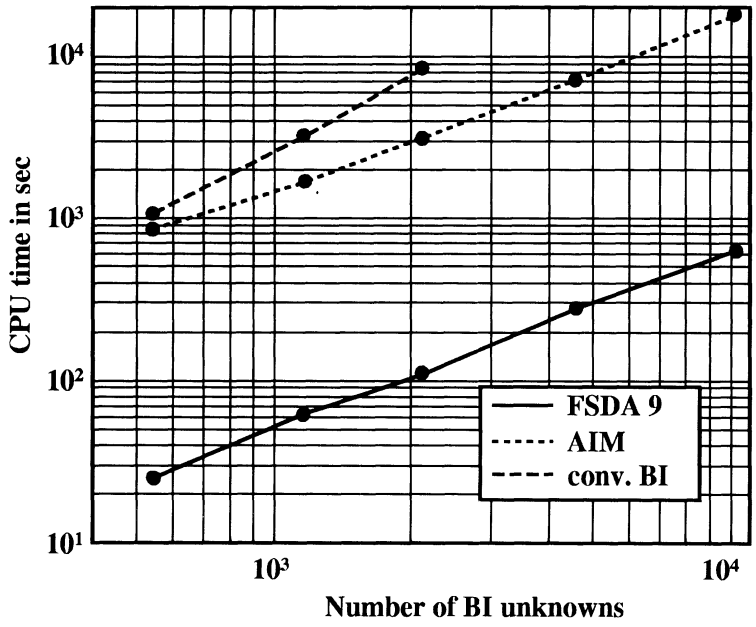


Figure 3: Total solution times dependent on the number of BI unknowns for the FSDA, conventional BI, and conventional BI accelerated by AIM (obtained for test problem of microstrip dipole array with one prism layer on Pentium II PC/266 MHz).

References

[1] D. T. McGrath and V. P. Pyati, "Phased Array Antenna Analysis with the Hybrid Finite Element Method," *IEEE Trans. Antennas Propagat.*, vol. 42, no. 12, pp. 1625–1630, Dec. 1994.

[2] E. W. Lucas and T. W. Fontana, "A 3-D Hybrid Finite Element/Boundary Element Method for the Unified Radiation and Scattering Analysis of General Infinite Periodic Arrays," *IEEE Trans. Antennas Propagat.*, vol. 43, no. 2, pp. 145–153, Feb. 1995.

[3] D. T. McGrath and V. P. Pyati, "Periodic Structure Analysis Using a Hybrid Finite Element Method," *Radio Science*, vol. 31, no. 5, pp. 1173–1179, Sep/Oct. 1996.

[4] J. D'Angelo and I. Mayergoyz, "Pased Array Antenna Analysis," in *Finite Element Software for Microwave Engineering*, Edited by T. Itoh, G. Pelosi, and P. P. Silvester, John Wiley & Sons, Inc., 1996.

[5] N. Engheta, W. D. Murphy, V. Rohklin, and M. S. Vassiliou, "The Fast Multipole Method (FMM) for Electromagnetic Scattering Problems," *IEEE Trans. Antennas Propagat.*, vol. 40, no. 6, pp. 634–641, Jun. 1992.

[6] R. Coifman, V. Rohklin, and S. Wandzura, "The Fast Multipole Method for the Wave Equation: A Pedestrian Prescription," *IEEE Antennas Propagat. Mag.*, vol. 35, pp. 7-12, Jun. 1993.

[7] N. Lu and J.–M. Jin, "Application of the Fast Multipole Method to Finite–Element Boundary–Integral Solution of Scattering Problems," *IEEE Trans. Antennas Propagat.*, vol. 44, no. 6, pp. 781–786, Jun. 1996.

[8] S. S. Bindiganavale and J. L. Volakis, "Comparison of Three FMM Techniques for Solving Hybrid FE–BI Systems," *IEEE Antennas Propagat. Mag.*, vol. 39, no. 4, pp. 47–60, Aug. 97.

[9] J. M. Song, C. C. Lu, and W. C. Chew, "Multilevel Fast Multipole Algorithm for Electromagnetic Scattering by Large Complex Objects," *IEEE Trans. Antennas Propagat.*, vol. 45, no. 10, pp. 1488–1493, Oct. 1997.

[10] E. Bleszynski, M. Bleszynski, and T. Jaroszewicz, "AIM: Adaptive Integral Method Compression Algorithm for Solving Large–Scale Electromagnetic Scattering and Radiation Problems," *Radio Science*, vol. 31, no. 5, pp. 1225–1251, Sep./Oct. 1996.

[11] H. T. Anastassiu, M. Smelyanskiy, S. Bindiganavale, and J. L. Volakis, "Scattering from Relatively Flat Surfaces Using the Adaptive Integral Method," *Radio Science*, vol. 33, no. 1, pp. 7–16, Jan.–Feb. 1998.

[12] T. F. Eibert and J. L. Volakis, Fast Spectral Domain Algorithm for Rapid Solution of Integral Equations, *Electronic Letters*, vol. 34, no. 13, pp. 1297–1299, Jun. 25, 1998.

[13] T. Özdemir and J. L. Volakis, "Triangular Prisms for Edge–Based Vector Finite Element Analysis of Conformal Antennas," *IEEE Trans. Antennas Propagat.*, vol. 45, no. 5, pp. 788–797, May 1997.

[14] S. M. Rao, D. R. Wilton, and A. W. Glisson, "Electromagnetic Scattering by Surfaces of Arbitrary Shape," *IEEE Trans. Antennas Propagat.*, vol. 30, no. 3, pp. 409–418, May 1982.

[15] Y. Saad, *Iterative Methods for Sparse Linear Systems*, Boston: PWS Pub. Co., 1996.

[16] P. P. Ewald, Die Berechnung optischer und elektrostatischer Gitterpotentiale, *Ann. Phys. 64*, pp. 253–287, 1921.

[17] K. E. Jordan, G. R. Richter, P. Sheng, An Efficient Numerical Evaluation of the Green's Function for the Helmholtz Operator on Periodic Structures, *J. of Comp. Phy.*, 63, pp. 222–235, 1986.

[18] B. Houshmand, W. C. Chew, and S.-W. Lee, "Fourier Transform of a Linear Distribution with Triangular Support and Its Applications in Electromagnetics," *IEEE Trans. Antennas Propagat.*, vol. 39, pp. 252–254, Feb. 1991.

[19] R. Mittra, C. H. Chan, and T. Cwik, "Techniques for Analyzing Frequency Selective Surfaces — A Review," *Proc. IEEE*, vol. 76, no. 12, pp. 1593–1615, Dec. 1988.

[20] D. R. Wilton and D. R. Jackson, Personal Communications, 1997.

Electric field viewer for FSS-Prism

This program was written for Windows 95 or Windows NT using OpenGL.

This program will view EdgeUnk files when a Data2 file is available. Large files may take a while to load. The status bar will show progress. The DOS window will prompt for the number of samples you wish per average edgelength. This fits a chosen number of samples across a distance equal to an average triangle edge length. For parts with few elements, say a hundred, a value like 10 will produce good results. For 10000 elements, 1.0 is good. In general, take 100/sqrt(number of elements) to get a good looking plot. Fraction values are permitted. Values under 0.5 produce no display.

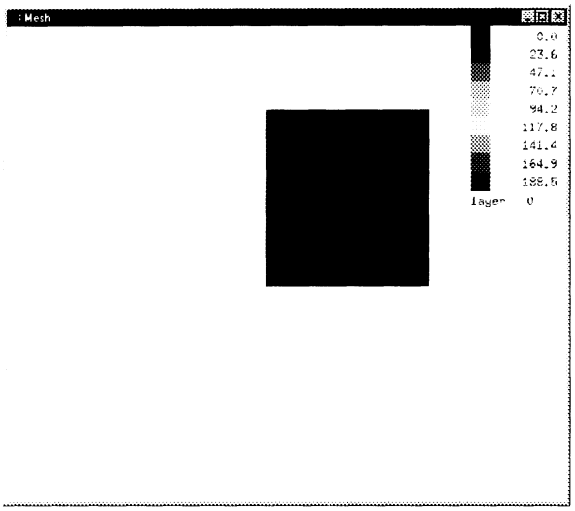
Once the view is loaded, press the first mouse button and drag the mouse on the display window to rotate to model. Press the right mouse button and drag to move the part around. To zoom in and out, either press both left and right buttons together or press the middle button and drag up or down.

T	T alternates between triangle and interpolated modes.
Q	Press q to display the quiver plot. This plot is a set of arrows showing the magnitude and direction of the real part of electric field.

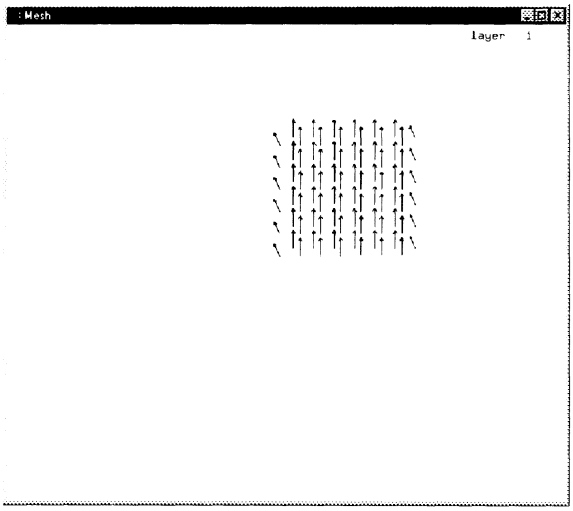
	While showing triangles, the following options are available
G	G will display the geometry. (T or Q returns to data display.)
X	Press x to display the x component of the electric field. Either the real part or the imaginary part may be shown depending upon which mode is current.
Y	Y displays the y component of the electric field. Both real and imaginary parts can be shown separately.
M	This displays the magnitude of the electric field, real or imaginary part.
R	Press r to display the real component x direction, y direction, or magnitude.
I	I will show the imaginary component x direction, y direction, or magnitude.
A	Animate the fields when overall magnitude is displayed.
	While showing interpolated data
R	R will bring up the real part magnitude.
I	I shows the imaginary magnitude.
	While in either mode
O	This displays the overall magnitude of electric field, combining both the real and imaginary magnitudes.
+	This key moves up a sample layer in the FSS.
-	- moves down a sample layer.
L	L displays all layers.
[	To decrease the distance between layers, press [.
]	To spread the layers apart more, press].

Use the Data1 and Data2 files from the Prism example with the generated EdgeUnk. Put them in a directory with FSSEfield.exe and run the executable. When questioned, use 8.8 samples.

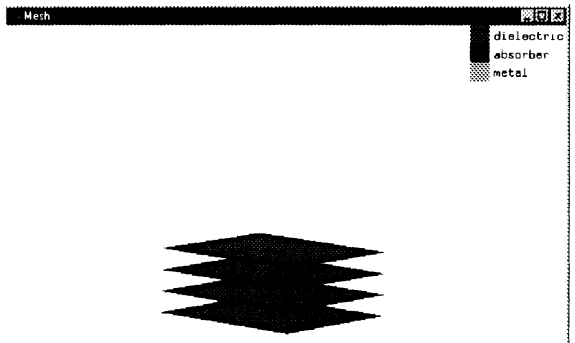
When it loads up, it should look



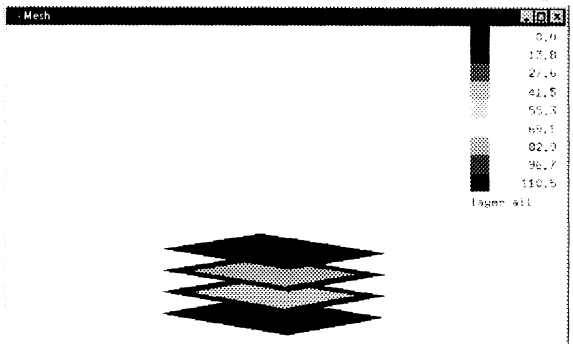
Hit 'Q' for the quiver



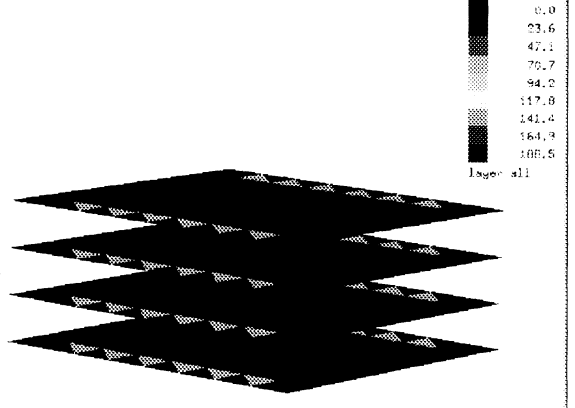
Hit 'G' then press the left mouse button on the window and rotate by dragging to see all



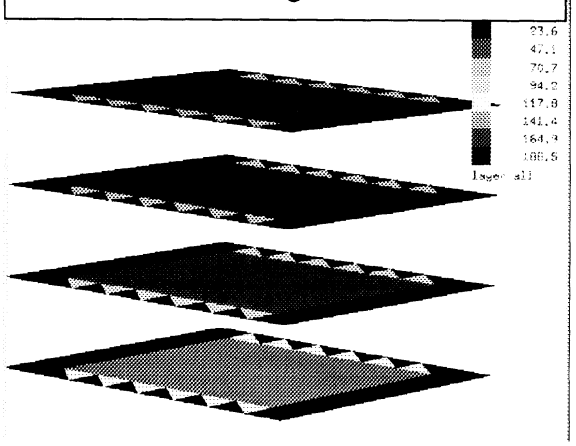
Hit 'T', 'I' to see the imaginary part of



Press 'T','O' to see the overall magnitude mesh data. Press the middle or left and right buttons and drag up to zoom in.



Press ']' several times to increase the distance between the layers so that you can see them all. Then press the right mouse button and drag up until you can see the whole drawing.



Magnetic current viewer for FSS-Prism

This program will view EqvCur and EqvCurB files when a Data2 file is available.

X	Show magnitude of the current in the x direction.
Y	Magnitude of the currents in the y direction.
Z	Z direction current magnitudes.
O	Show the overall <x,y,z> magnitude.
T	View magnitudes for the top layer.
B	Bottom layer magnitudes.

Curved Finite Array Code (FSS-CURVE) Update

K. Sertel

The development of the computer code FSS-CURVE for the solution of finite conformal FSS structures is being continued as scheduled. As of July 10th, the following features were added.

- The curved array code is integrated with the driver of FSS-PRISM, so the geometry description can be done with the same driver by just setting a flag for curvature.
- Several antenna and array geometries are tested for validation purposes. Coupling effects are demonstrated for an array geometry.

The option of defining curved surface meshes is being implemented into the driver. The transmission and reflection coefficient calculations have only been tested for perpendicular incidence. The code is being modified to calculate transmission and reflection coefficients for arbitrary incidences.

1 Examples

To validate the code, the patch antenna geometry given in [1] is analyzed using FSS-CURVE. The geometry is depicted in Fig. 1 (a). The results for the input impedance of the antenna show very good agreement with those given in [1].

Figures 2 (a) and 2 (b) show the RCS of the curved patch presented in [1] for two different polarizations.

Next, a 5×1 antenna array on a curved platform is examined. The effect of curvature on the input impedance and radiation pattern of the antenna can be predicted using FSS-CURVE. The surface mesh is given in Fig. 3. Fig. 4-5

shows the input impedance of the array elements for flat and curved array geometries (Input impedance of the 5th element shows a similar behavior as that of the 1st element and so do 2nd and 4th feeds as can be observed).

The effect of curvature on the radiation pattern is demonstrated by comparison the two radiation patterns plotted in Fig. 6 and Fig. 7.

The effect of coupling when the array elements are brought closer, is observed in Fig. 8 and Fig. 9.

The geometry descriptions of the test case antenna and arrays can be found in the presentation named "Fast Multipole Method (FMM) Solutions of Finite Element-Boundary Integral (FE-BI) Implementations for Antenna Modeling Involving Curved Geometries" given in APS'98 Atlanta, which is attached to this report.

References

[1] L. C. Kempel, J.L. Volakis and R. J. Silva, "Radiation by Cavity-Backed Antennas on a Circular Cylinder", IEE Proc.-Microw. Antennas Propag., Vol. 142, No. 3, pp. 233-239, June 1995

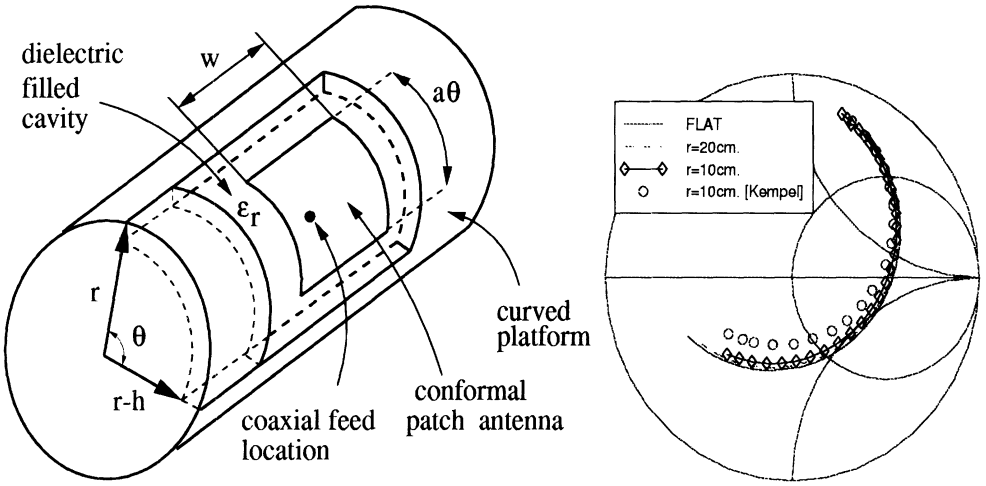


Figure 1: Curved Patch Geometry and Input Impedance Validation.

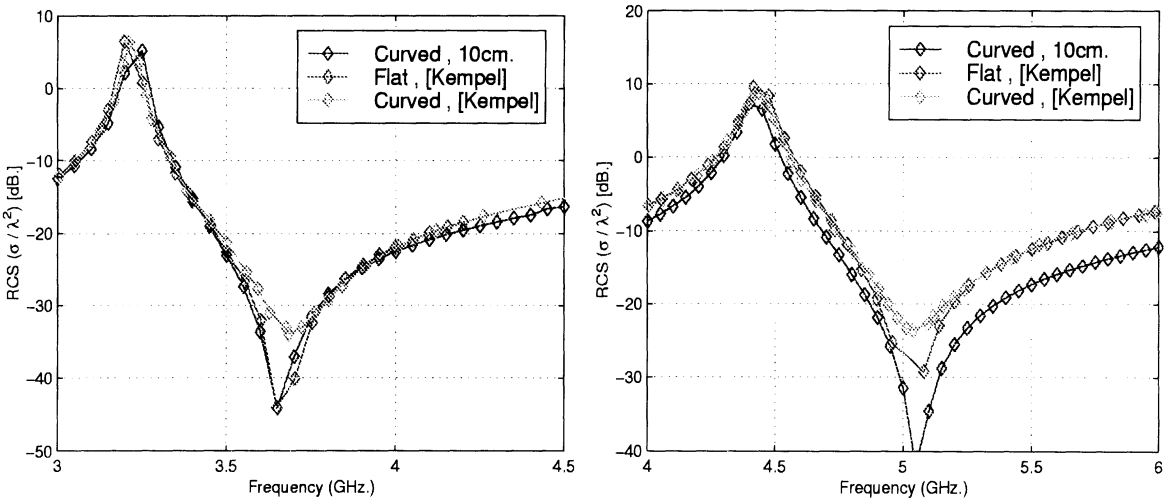


Figure 2: RCS (frequency sweep) of the curved patch for two polarizations.

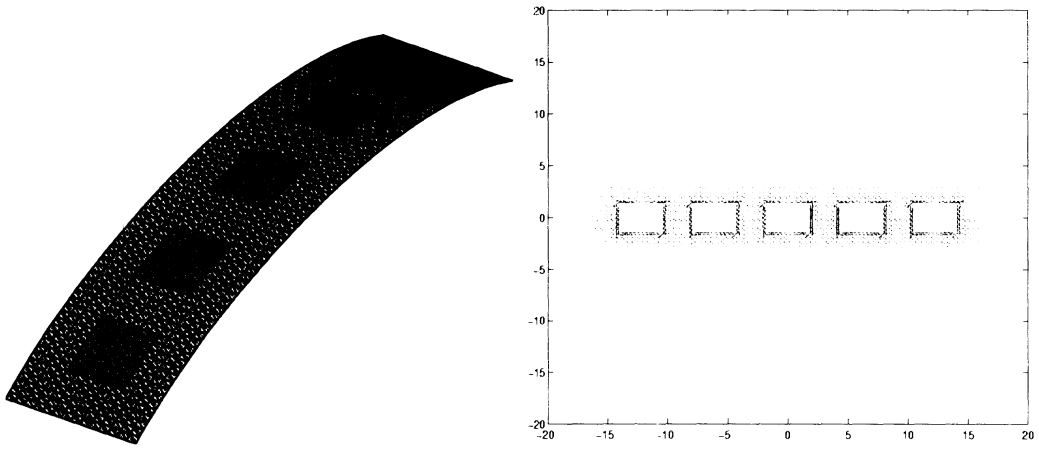


Figure 3: Array mesh and the surface magnetic current distribution at resonance.

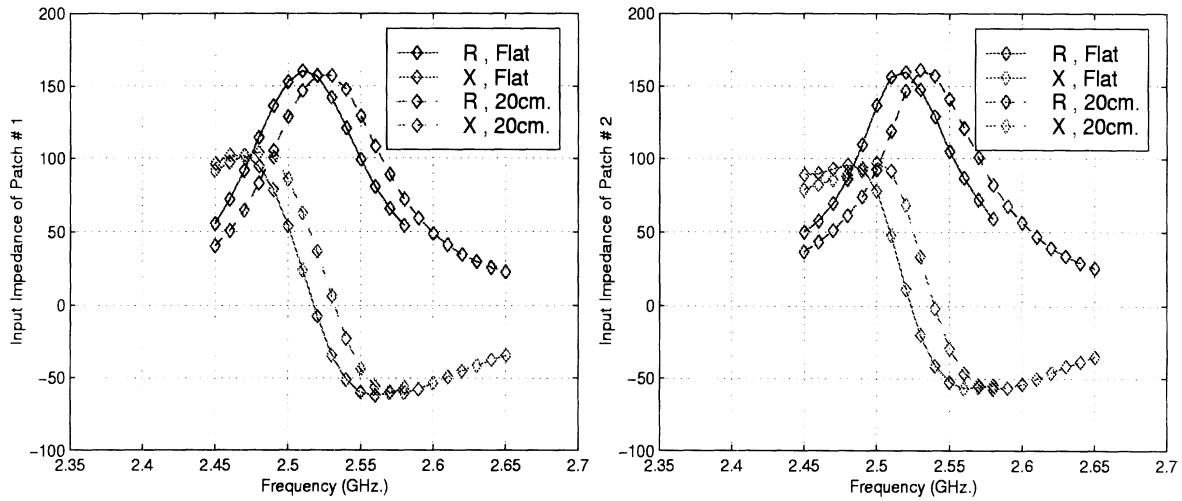


Figure 4: Input impedance of the 1st and the 2nd elements.

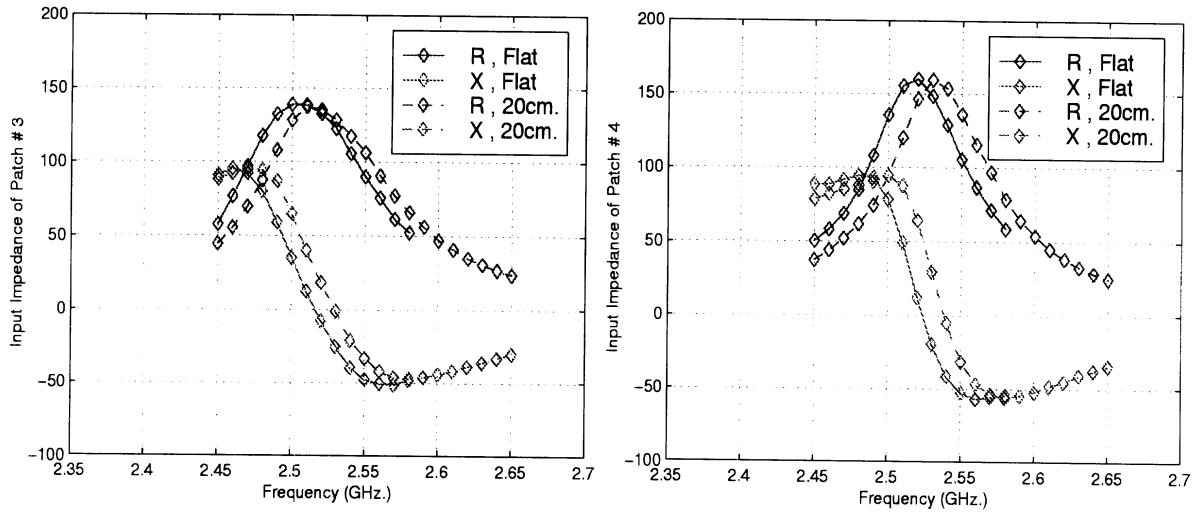


Figure 5: Input impedance of the 3rd and the 4th elements.

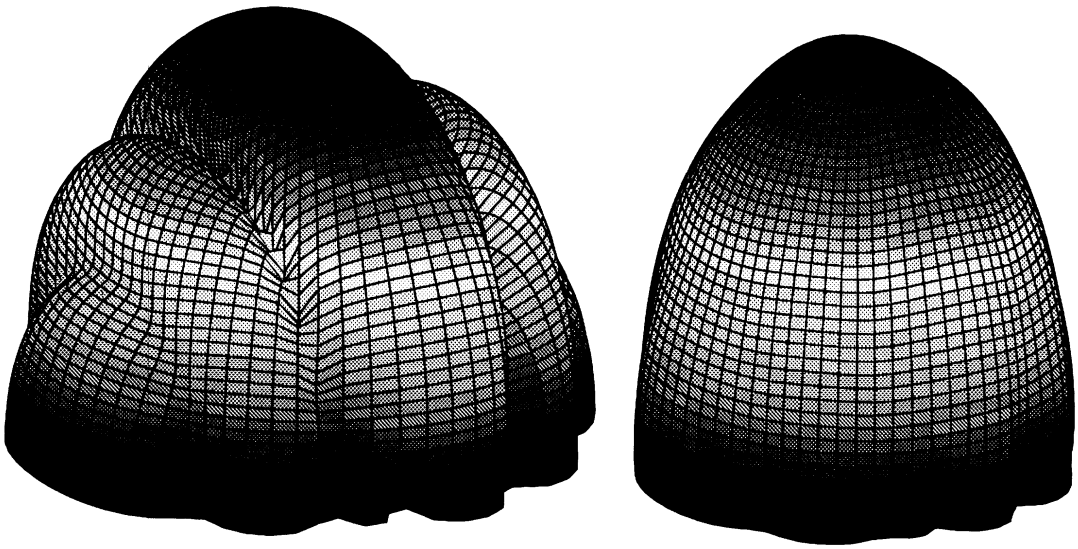


Figure 6: Radiation pattern of the flat and curved array.

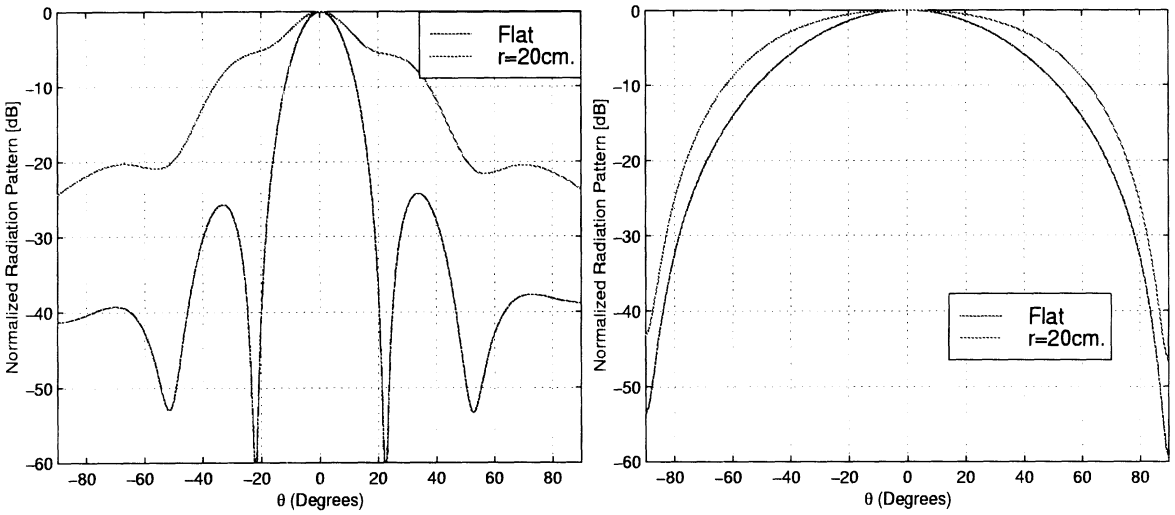


Figure 7: Radiation pattern at two principle cuts for flat and curved arrays.

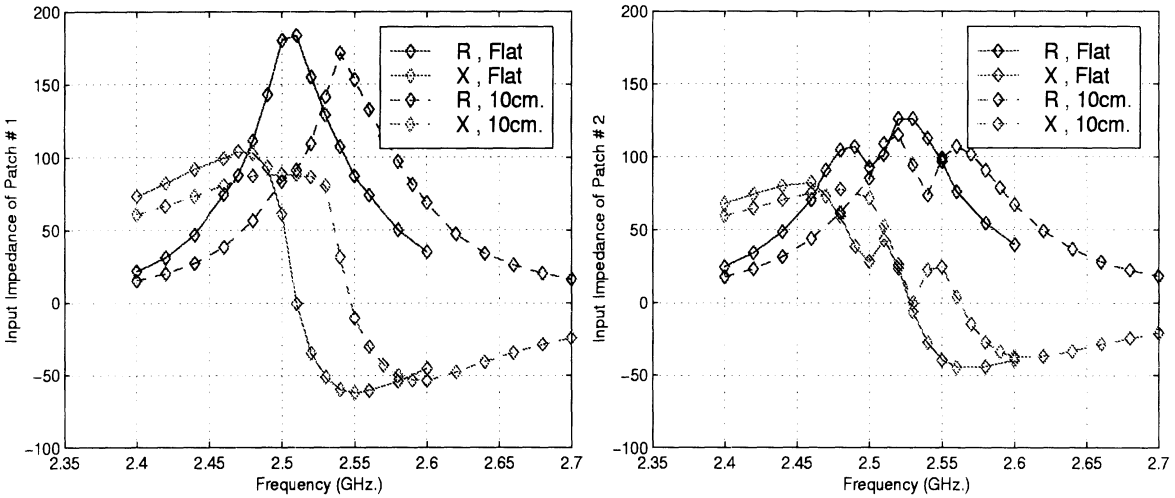


Figure 8: Coupling effects on input impedance for closely spaced array elements. Feed 1 and feed 2.

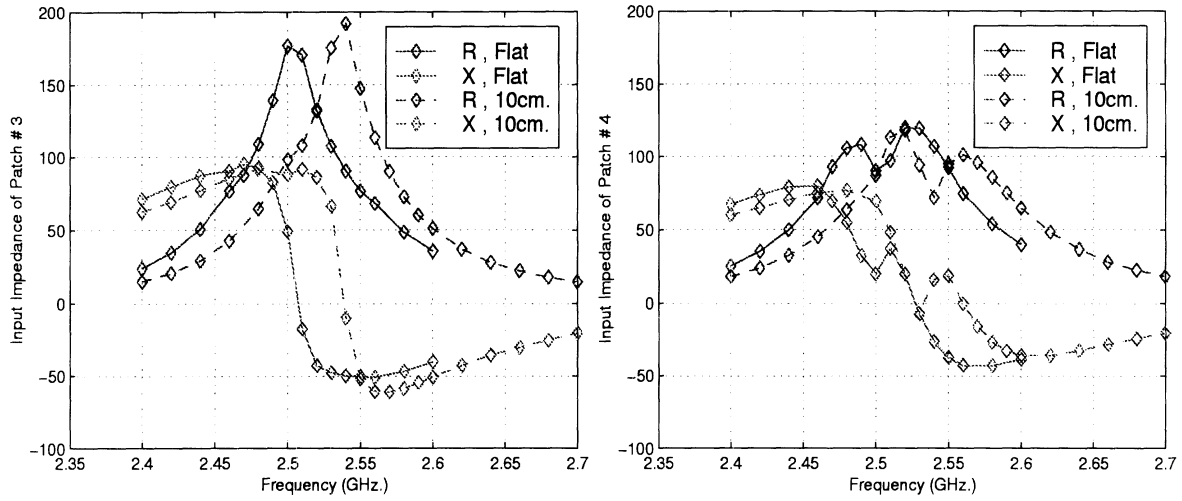


Figure 9: Coupling effects on input impedance for closely spaced array elements. Feed 3 and feed 4.

APPENDICES: Conference Presentation Slides

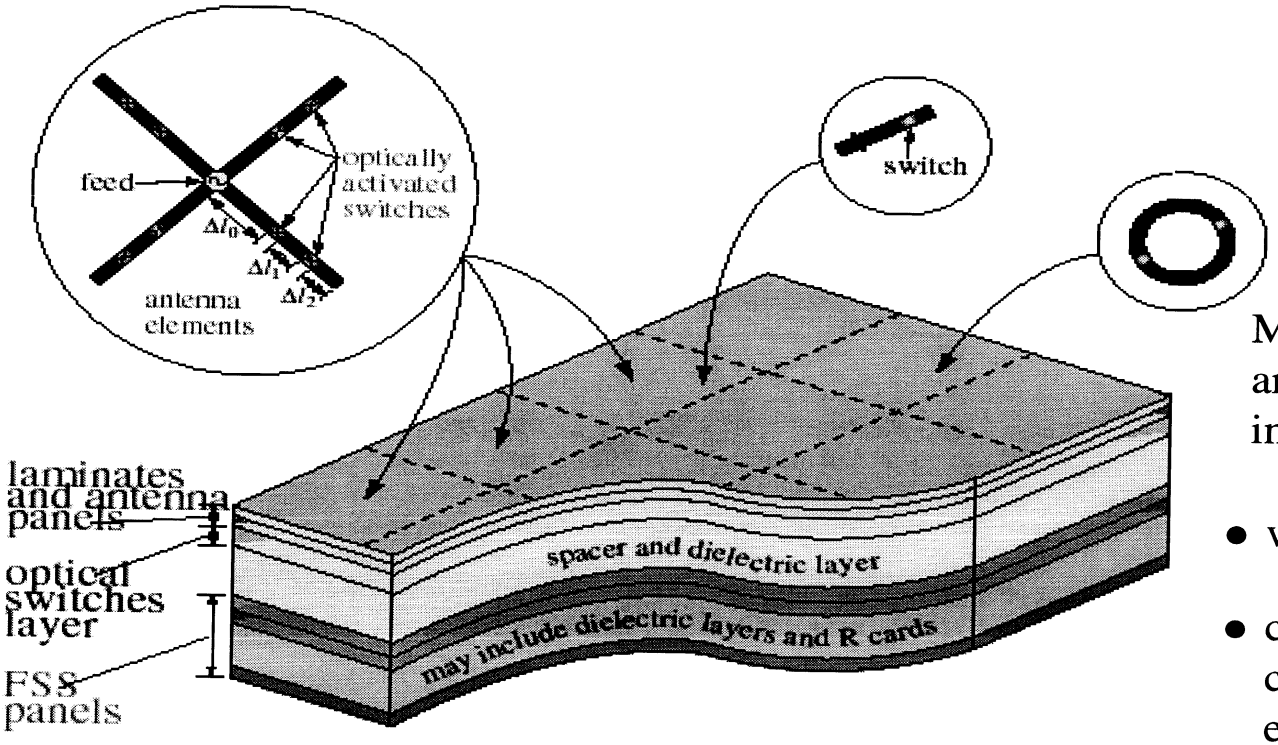
Hybrid FE-BI Modeling of Commensurate and Non-Commensurate 3D Doubly Periodic Structures *by T. F. Eibert and J.L. Volakis*

Hybrid FE-BI Modeling of Commensurate and Non-Commensurate 3D Doubly Periodic Structures

Thomas F. Eibert and John L. Volakis
Radiation Laboratory, EECS Department
The University of Michigan
Ann Arbor, MI 48109-2122

- Motivation and problem definition
- Finite Element (FE) - Boundary Integral (BI) formulation
- Periodic phase boundary conditions and Green's function
- Commensurate applications
- Extension to non-commensurate problems
- Non-commensurate applications
- Conclusions

Problem Definition: Analysis of Doubly Periodic Structures (Antenna Arrays and Frequency Selective Surfaces (FSS))

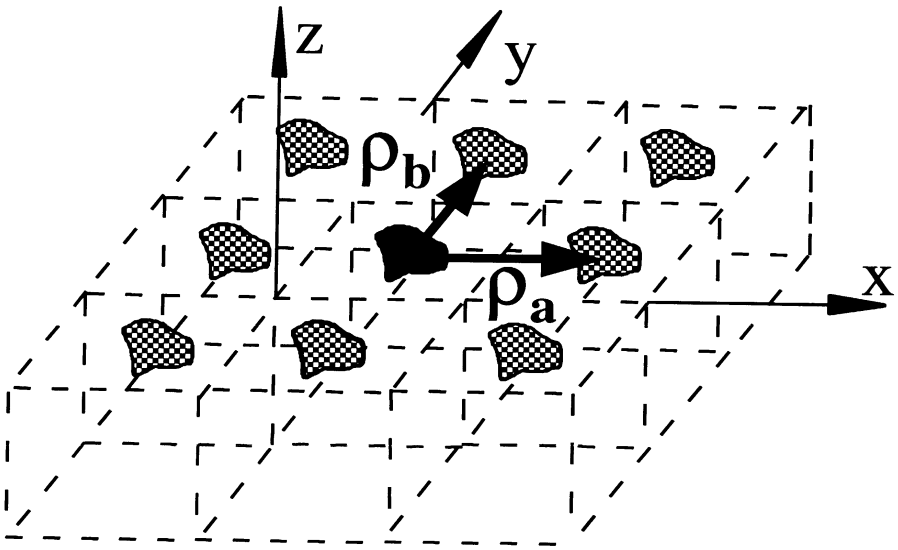


Most computational approaches are based on spectral domain integral equation formulations.

- well-suited for planar structures
- computationally intensive for complicated radiation/scattering elements

Need for three-dimensional modeling capabilities can be fulfilled with Hybrid Finite Element (FE) - Boundary Integral (BI) formulation.

Finite Element-Boundary Integral Formulation of the Array Problem



Finite Element functional

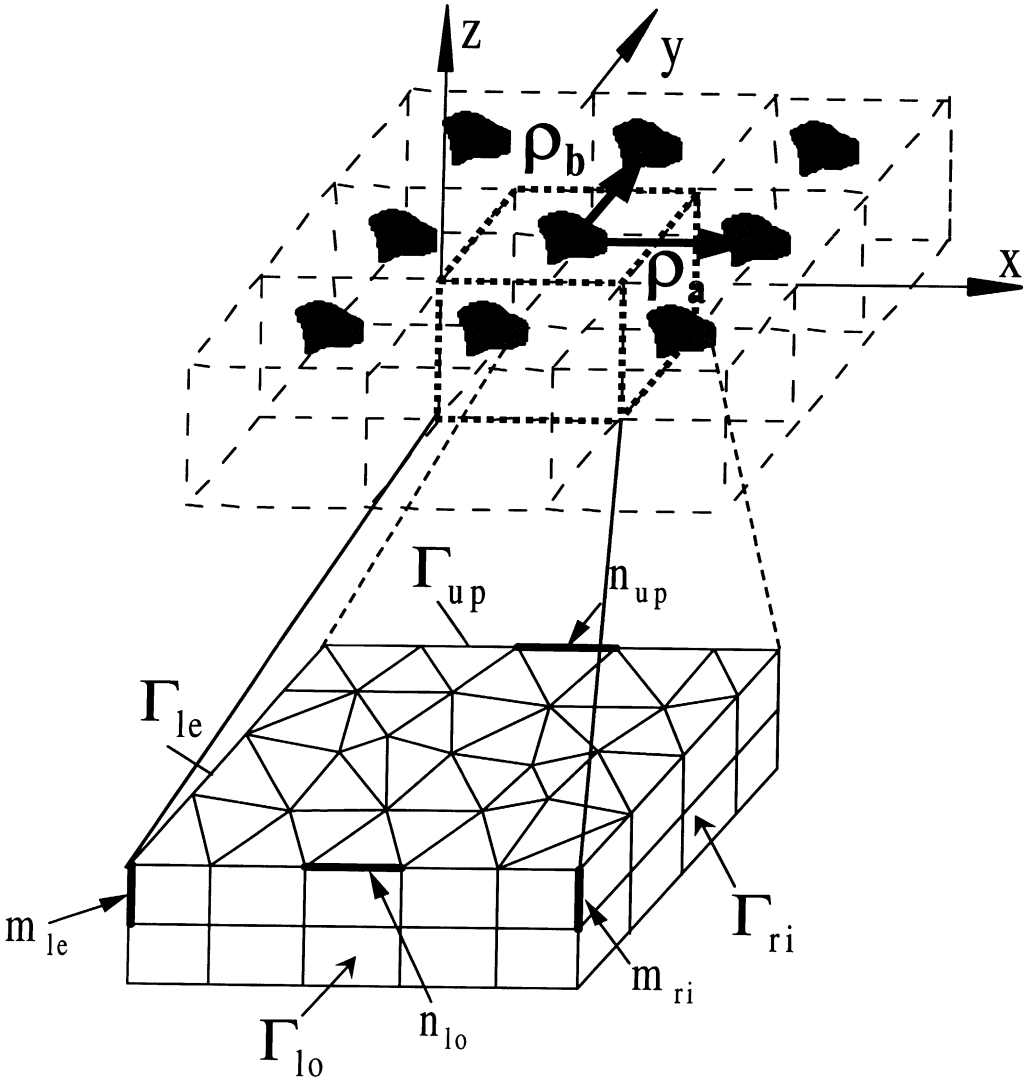
$$F(\vec{E}) = \frac{1}{2}\iiint_V \left[\frac{1}{\mu_r}(\nabla\times\vec{E})\cdot(\nabla\times\vec{E}) - k_0^2\varepsilon_r\vec{E}\cdot\vec{E}\right]dv + jk_0Z_0\iint_S \vec{E}\cdot(\vec{H}\times\hat{n})ds$$

Boundary Integral

$$\vec{H} = -2j\frac{k_0}{Z_0}\left[\iint_S G(\vec{r},\vec{r}^{\,'})(\vec{E}\times\hat{n})\,ds + \frac{1}{k_0^2}\nabla\iint_S G(\vec{r},\vec{r}^{\,'})\,\nabla_s\cdot(\vec{E}\times\hat{n})\,ds\right] + \vec{H}^{inc}$$

$$G(\vec{r},\vec{r}^{\,'}) = \frac{1}{4\pi}\frac{e^{-jk_0|\vec{r}-\vec{r}^{\,'}|}}{|\vec{r}-\vec{r}^{\,'}|}$$

Reduction of Array Problem to one Unit Cell



Periodicity conditions:

$$\vec{E}(\vec{r}+m\vec{\rho}_a+n\vec{\rho}_b)=\vec{E}(\vec{r})\,\mathrm{e}^{-\mathrm{j}\vec{k}_{t00}\cdot(m\vec{\rho}_a+n\vec{\rho}_b)}$$

$$\vec{H}(\vec{r}+m\vec{\rho}_a+n\vec{\rho}_b)=\vec{H}(\vec{r})\,\mathrm{e}^{-\mathrm{j}\vec{k}_{t00}\cdot(m\vec{\rho}_a+n\vec{\rho}_b)}$$

with

$$\vec{k}_{t00}=\pm(k_0\sin\vartheta_0\cos\varphi_0\;\hat{x}+k_0\sin\vartheta_0\sin\varphi_0\;\hat{y})$$

ϑ_0,φ_0 : scan angle of array

Phase boundary condition for FE mesh:

$$e_{m_{le}}=e_{m_{ri}}\,\mathrm{e}^{-j\vec{k}_{t00}\cdot\vec{\rho}_a}$$

$$e_{n_{up}}=e_{n_{lo}}\,\mathrm{e}^{-j\vec{k}_{t00}\cdot\vec{\rho}_b}$$

Periodic Green's Function Series Acceleration

Periodic Green's function

in the spatial domain

$$G_p(\vec{r},\vec{r}^{\,'}) = \sum_{m=-\infty}^{\infty}\sum_{n=-\infty}^{\infty} \mathrm{e}^{-j\vec{k}_{t00}\cdot\vec{\rho}_{mn}} \frac{\mathrm{e}^{-jk_0R_{mn}}}{4\pi\, R_{mn}}$$

in the spectral domain

$$G_p(\vec{r},\vec{r}^{\,'}) = \sum_{m=-\infty}^{\infty}\sum_{n=-\infty}^{\infty} \frac{\mathrm{e}^{-j\vec{k}_{tmn}\cdot(\vec{\rho}-\vec{\rho}^{\,'})}}{2Ajk_{zmn}} \mathrm{e}^{-jk_{zmn}|z-z'|}$$

with

$$R_{mn} = |\,\vec{r}-\vec{r}^{\,'}-\vec{\rho}_{mn}\,|$$

$$A = |\,\vec{\rho}_a \times \vec{\rho}_b\,| \quad \text{:area of unit cell}$$

$$\vec{k}_{tmn} = \vec{k}_{t00} + \frac{2\pi}{A}\left[n\,(\hat{z}\times\vec{\rho}_a) + m\,(\vec{\rho}_b\times\hat{z})\right]$$

$$k_{zmn} = \sqrt{k_0^2 - \vec{k}_{tmn}\cdot\vec{k}_{tmn}}$$

Ewald Transformation

(in collaboration with D. R. Wilton and D. R. Jackson)

Idea

$$G_p(\vec{r},\vec{r}^{\,'}) = G_{p1}(\vec{r},\vec{r}^{\,'}) + G_{p2}(\vec{r},\vec{r}^{\,'})$$

$$G_{p1}(\vec{r},\vec{r}^{\,'}) = \sum_{m=-\infty}^{\infty}\sum_{n=-\infty}^{\infty} \frac{\mathrm{e}^{-j\vec{k}_{t00}\cdot\vec{\rho}_{mn}}}{4\pi} \frac{2}{\sqrt{\pi}} \int_0^E \mathrm{e}^{-R_{mn}^2 s^2 + \frac{k_0^2}{4s^2}}\, ds$$

$$G_{p2}(\vec{r},\vec{r}^{\,'}) = \sum_{m=-\infty}^{\infty}\sum_{n=-\infty}^{\infty} \frac{\mathrm{e}^{-j\vec{k}_{t00}\cdot\vec{\rho}_{mn}}}{4\pi} \frac{2}{\sqrt{\pi}} \int_E^\infty \mathrm{e}^{-R_{mn}^2 s^2 + \frac{k_0^2}{4s^2}}\, ds$$

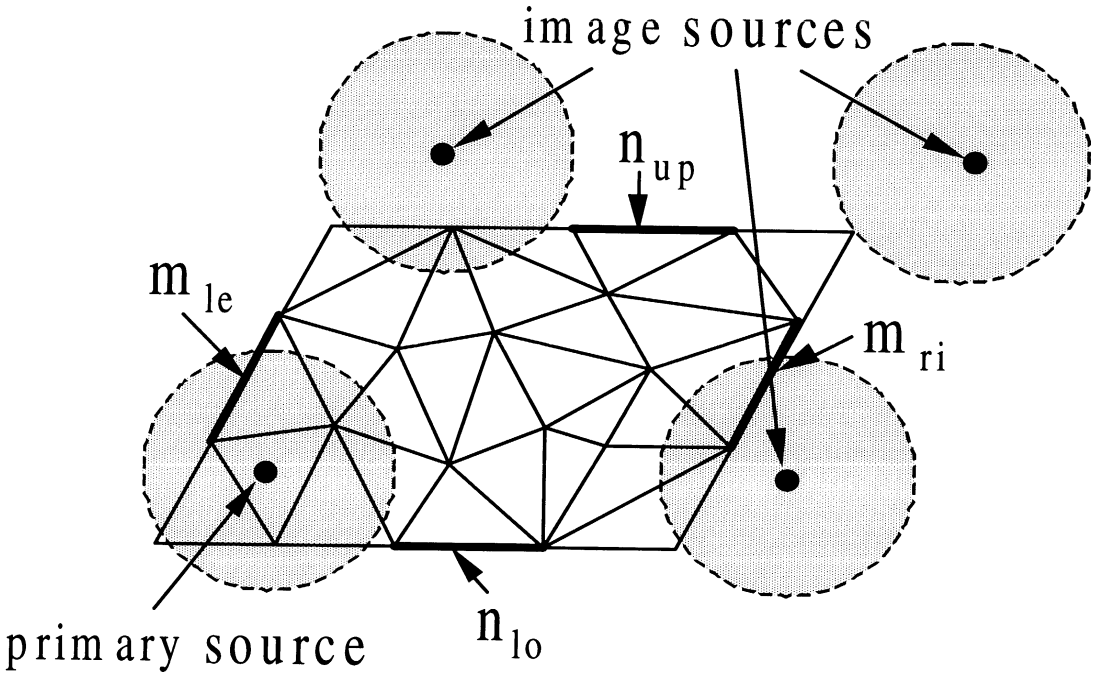
Result

$$G_{p1}(\vec{r},\vec{r}^{\,'}) = \sum_{m=-\infty}^{\infty}\sum_{n=-\infty}^{\infty} \frac{\mathrm{e}^{-j\vec{k}_{tmn}\cdot(\vec{\rho}-\vec{\rho}^{\,'})}}{2Ajk_{zmn}} \mathrm{erfc}\left(\frac{jk_{zmn}}{2E}\right), \quad z = z^{'} = 0$$

$$G_{p2}(\vec{r},\vec{r}^{\,'}) = \sum_{m=-\infty}^{\infty}\sum_{n=-\infty}^{\infty} \frac{\mathrm{e}^{-j\vec{k}_{t00}\cdot\vec{\rho}_{mn}}}{8\pi\, R_{mn}} \left[\mathrm{e}^{-jk_0R_{mn}}\,\mathrm{erfc}\left(R_{mn} - \frac{jk}{2E}\right) + \mathrm{e}^{jk_0R_{mn}}\,\mathrm{erfc}\left(R_{mn} + \frac{jk}{2E}\right)\right]$$

$$\mathrm{erfc}(x) = \frac{2}{\sqrt{\pi}} \int_x^\infty \mathrm{e}^{-t^2}\, dt \quad \text{:complementary error function}$$

Boundary Integral Implementation



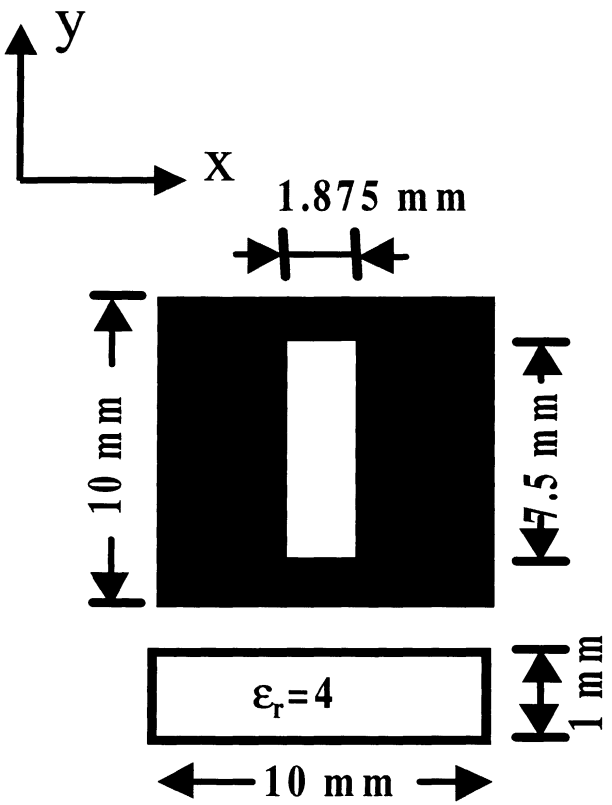
Phase boundary condition for BI mesh:

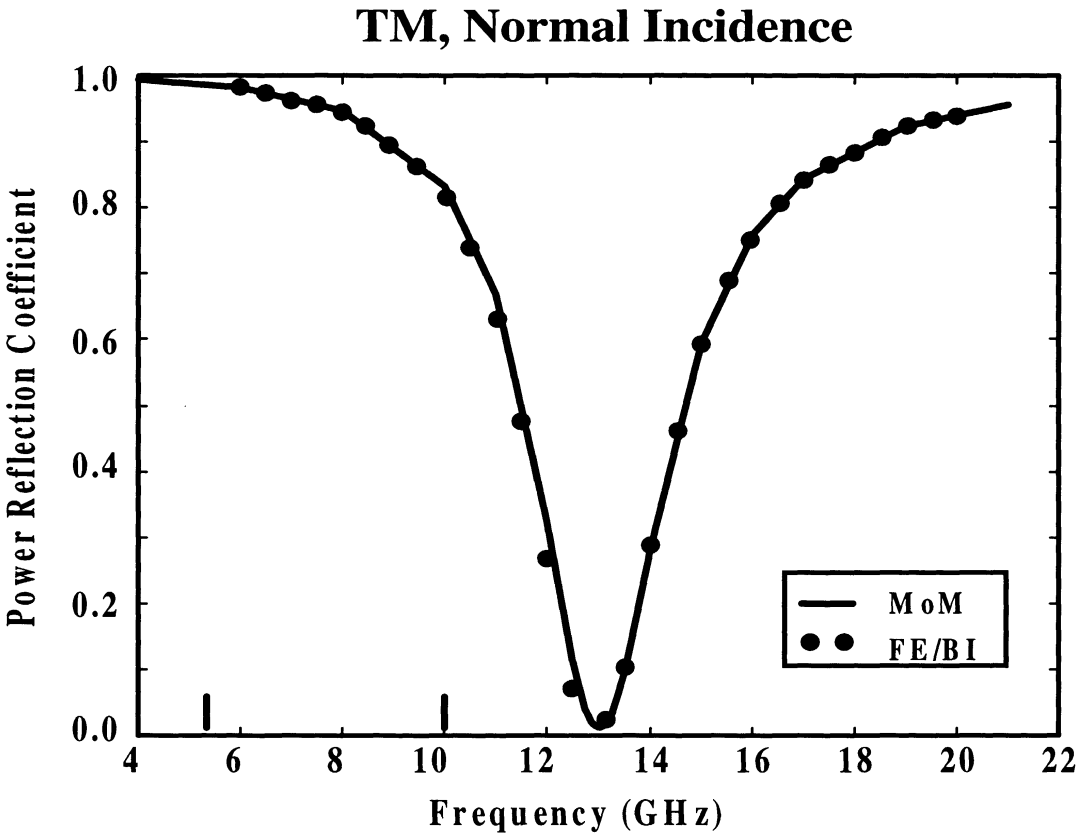
$$e_{m_{le}} = e_{m_{ri}} \, \mathrm{e}^{-j\vec{k}_{t00}\cdot\vec{\rho}_a}$$

$$e_{n_{up}} = e_{n_{lo}} \, \mathrm{e}^{-j\vec{k}_{t00}\cdot\vec{\rho}_b}$$

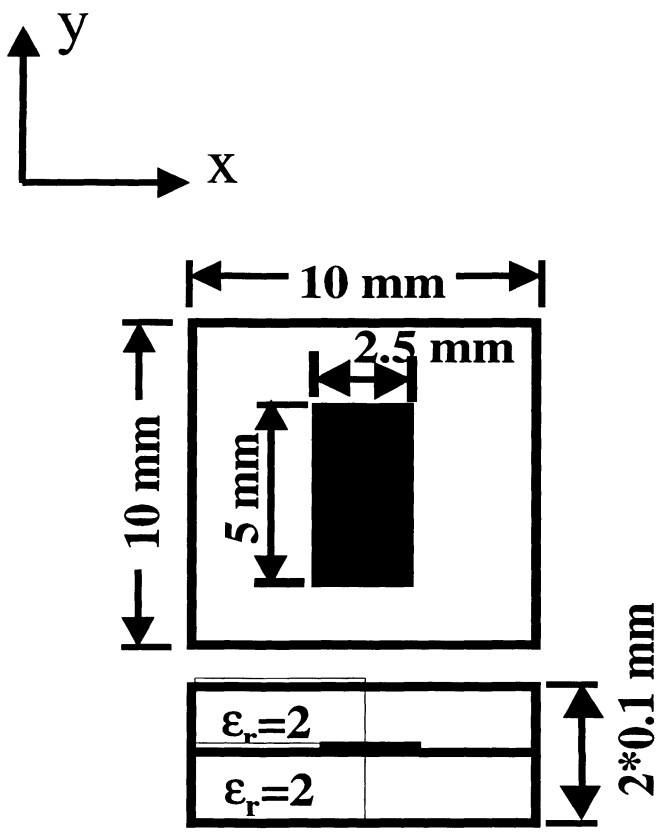
Proper handling of the singular periodic image sources

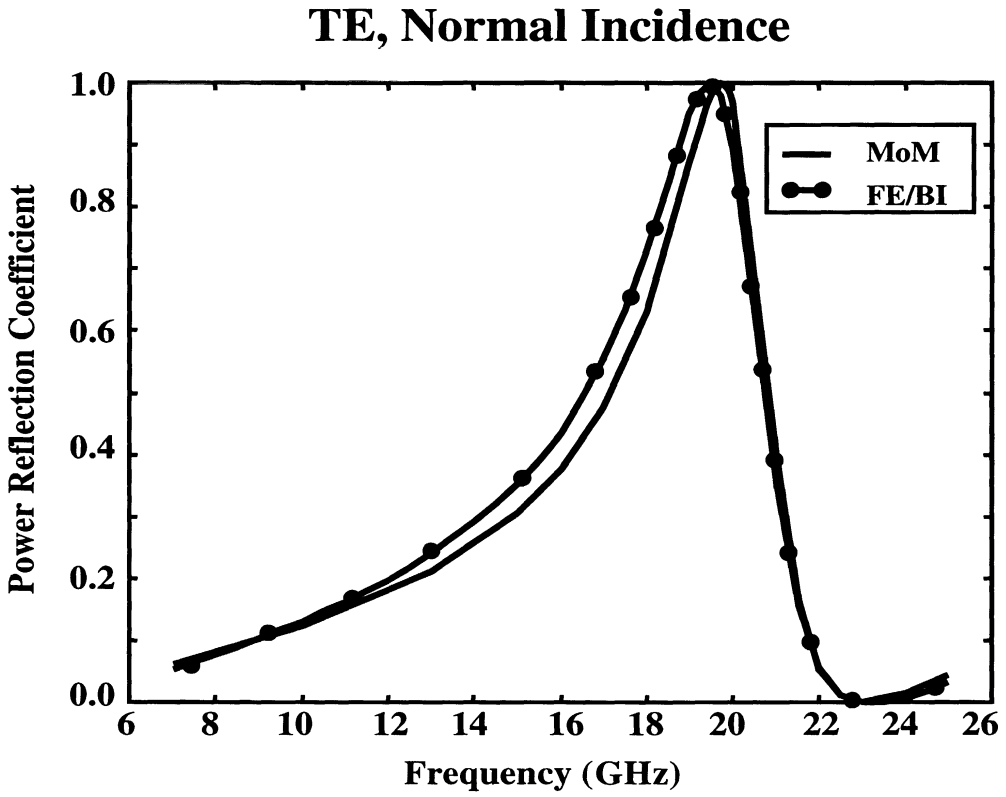
Power Reflection Coefficient of 1-Layer Slot Array



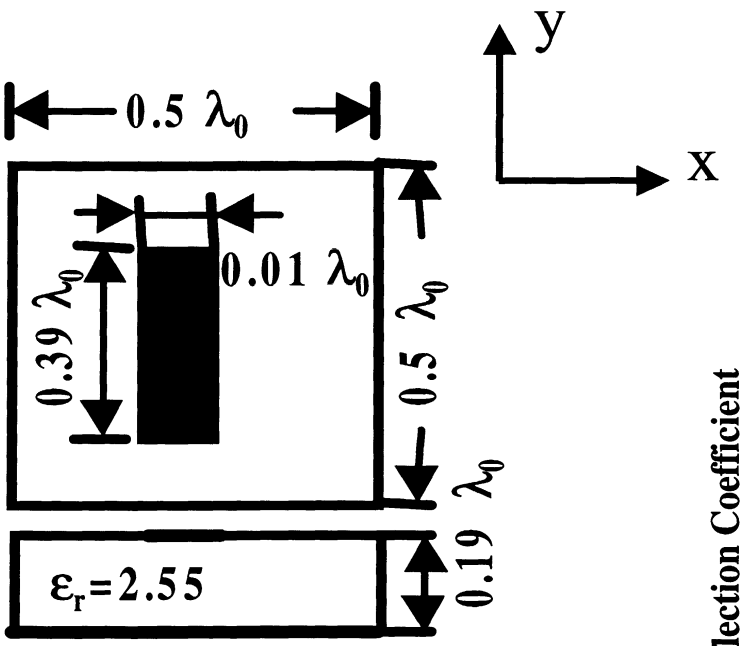


Power Reflection Coefficient of a 1-Layer Strip Array



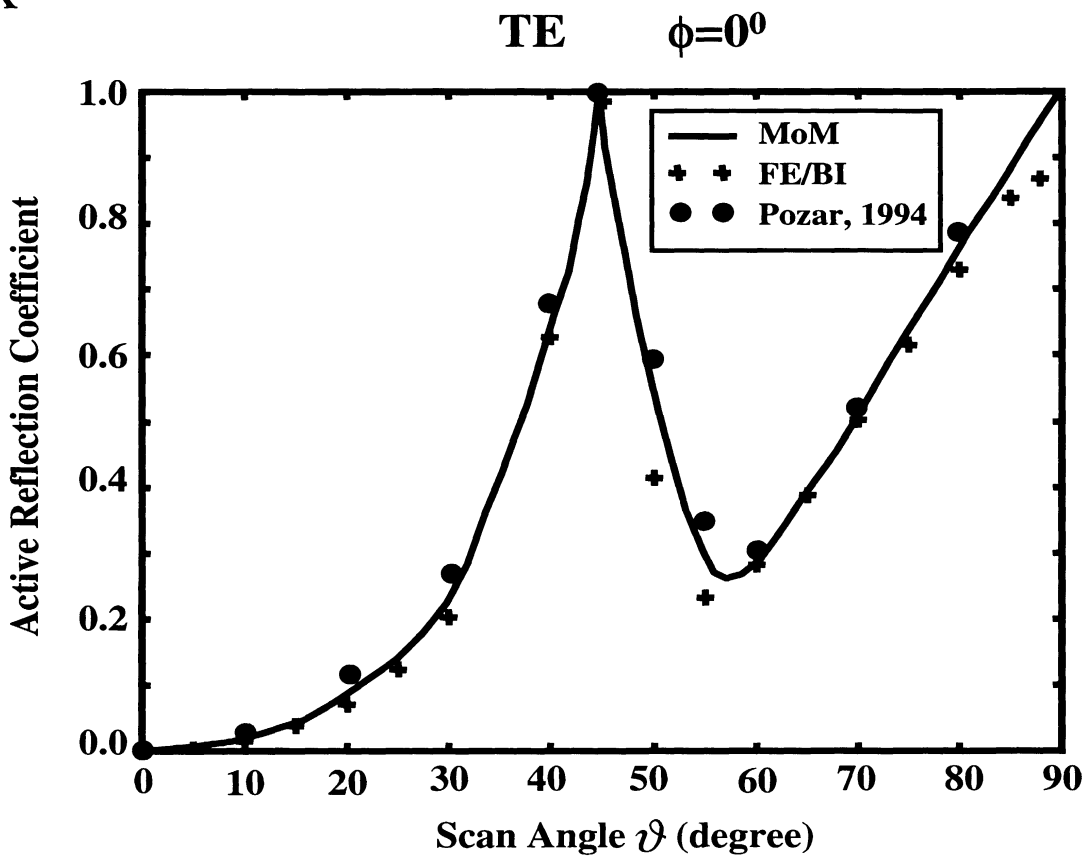


Active Reflection Coefficient of Microstrip Dipole Array

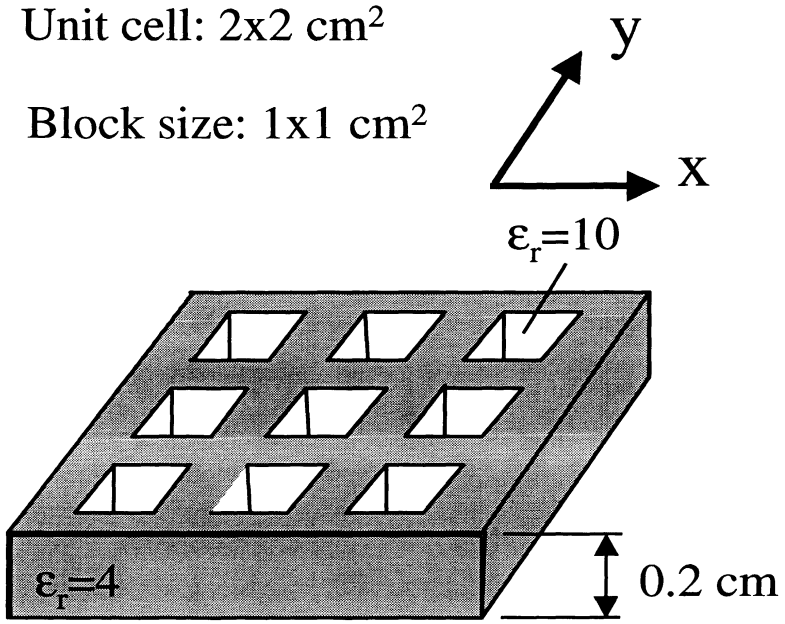


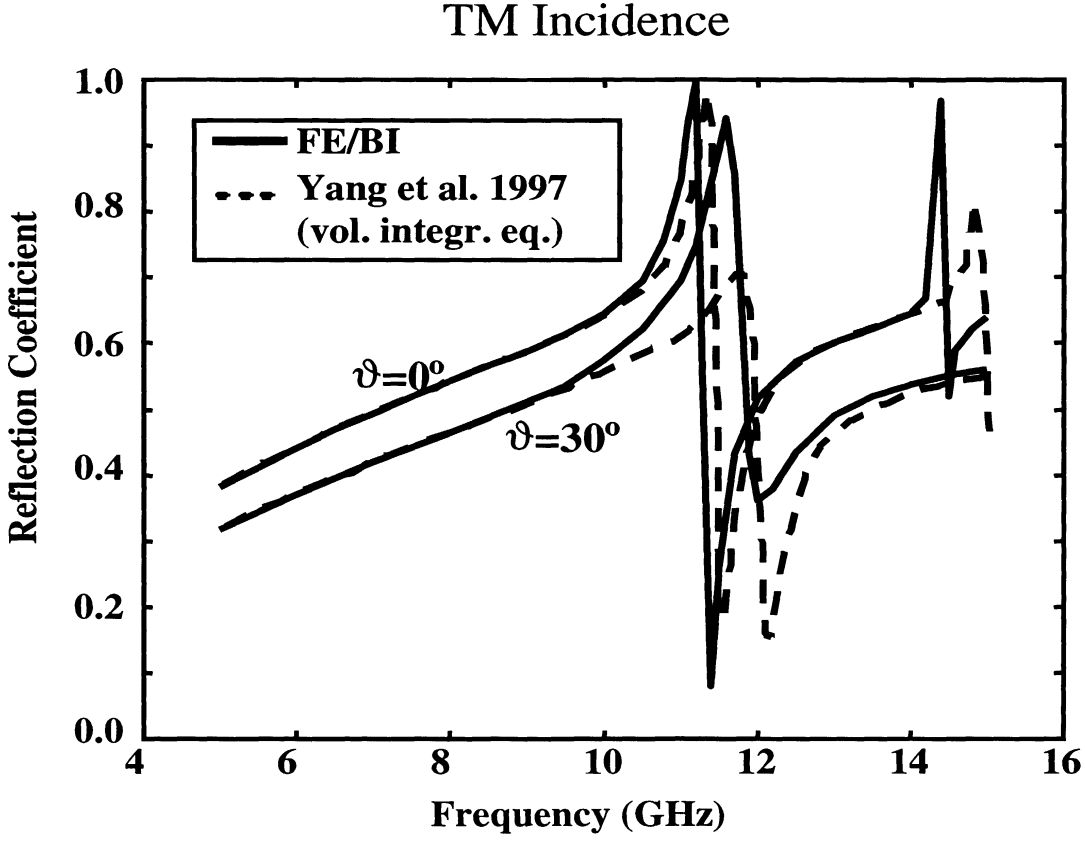
Active Reflection Coefficient:

$$r(\vartheta) = \frac{Z^{inp}(\vartheta) - Z^{inp}(\vartheta = 0^0)}{Z^{inp}(\vartheta) + Z^{inp}(\vartheta = 0^0)}$$

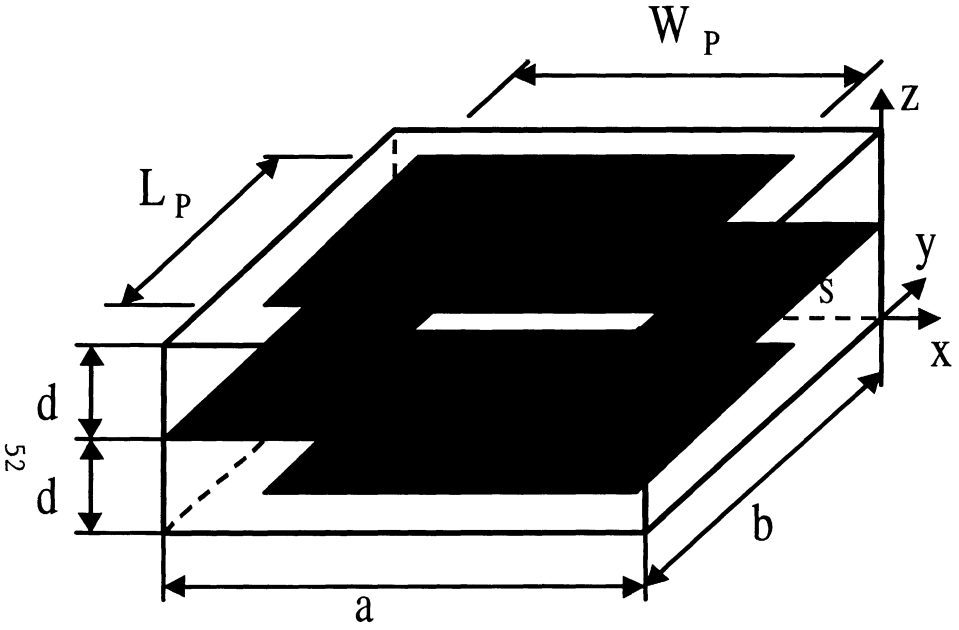


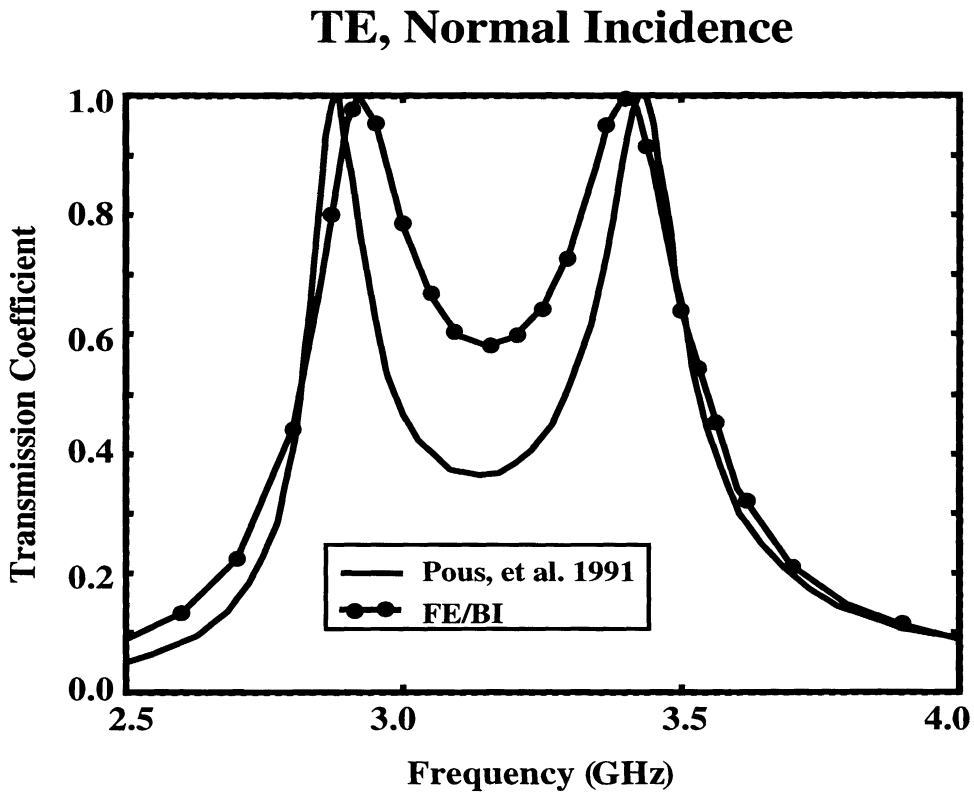
Plane Wave Reflection from a Dielectric Slab with Embedded Material Blocks ("Photonic Bandgap Material")





Aperture Coupled Microstrip Patches
Test Configuration

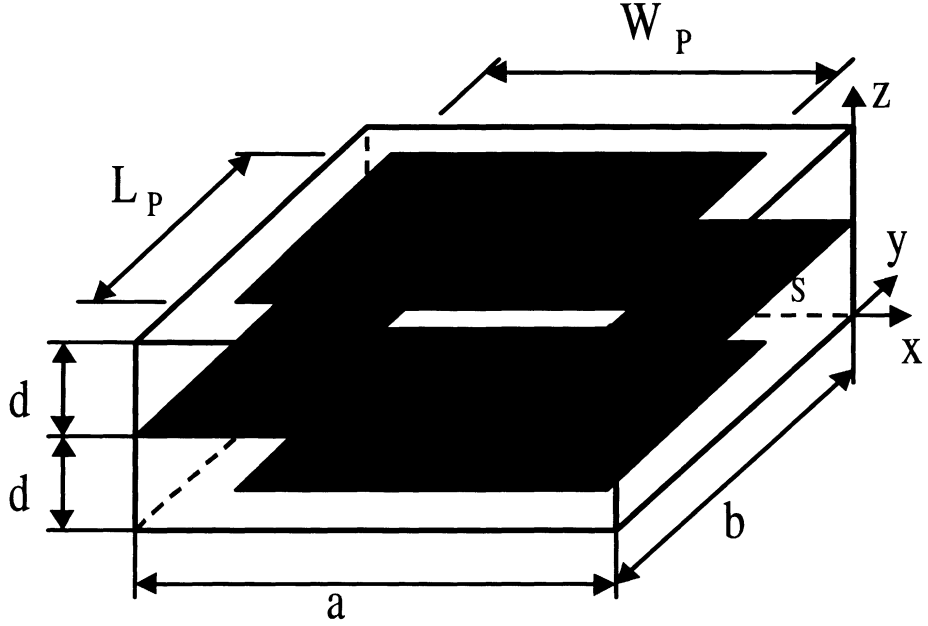


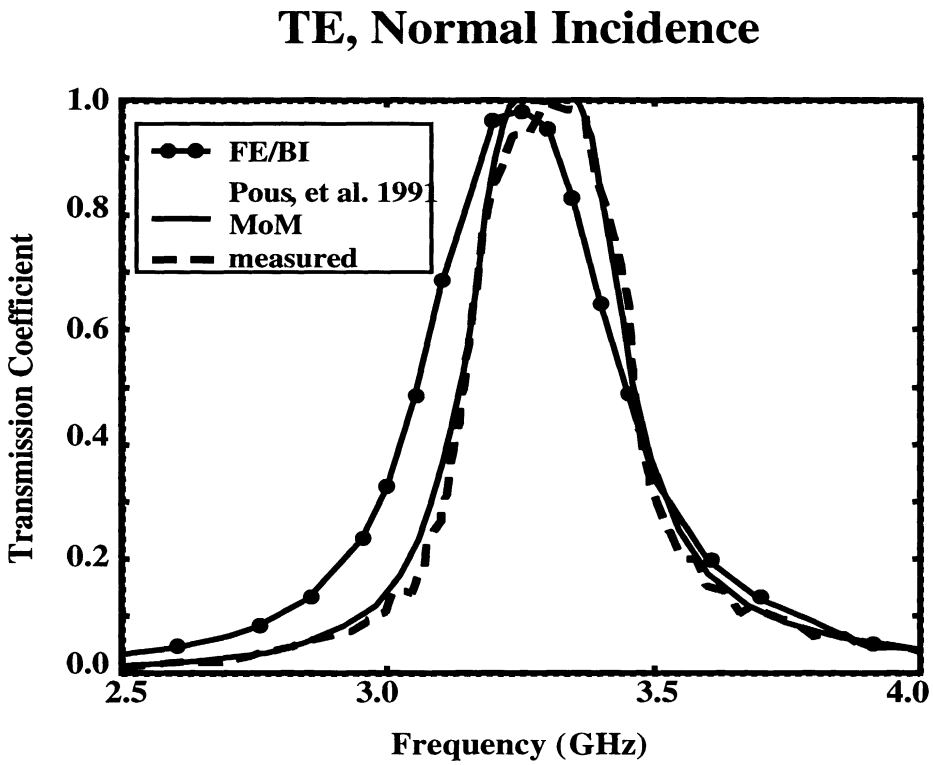


a=36.07 mm, b=34.04 mm, d=1.6 mm
W_s=2 mm, L_s=12 mm,
L_p=28 mm, W_p=18 mm
ε_r=2.2

2 x 8688 BI unknowns
105484 Total unknowns
about 3-5 hours per f (on IBM RS6000/590)

Aperture Coupled Microstrip Patches
Bandpass Configuration

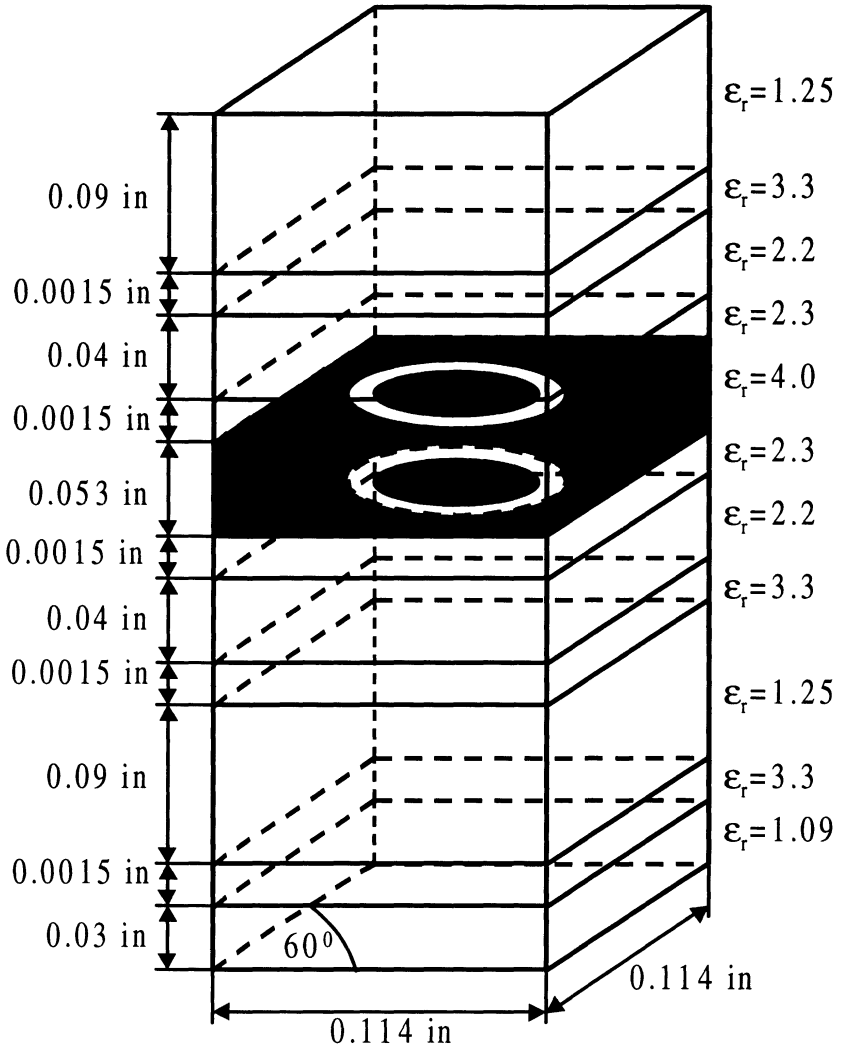


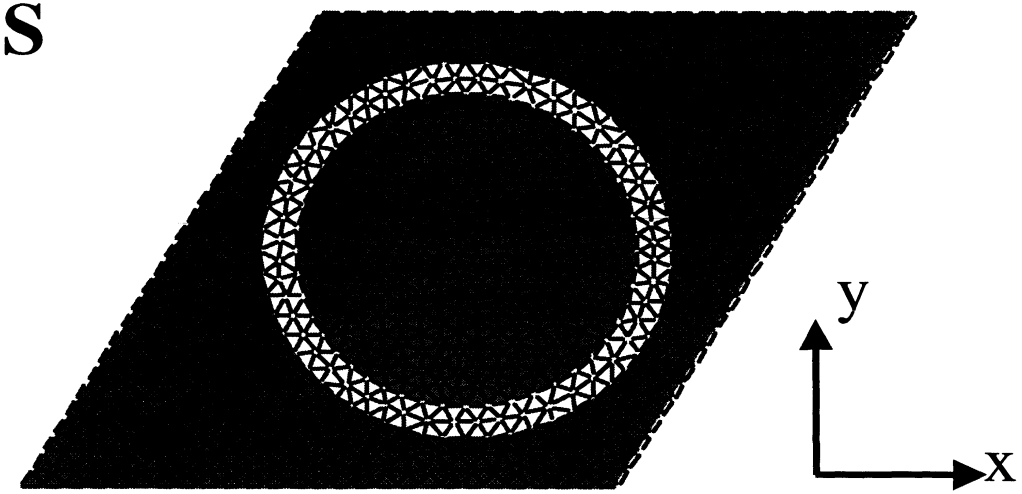


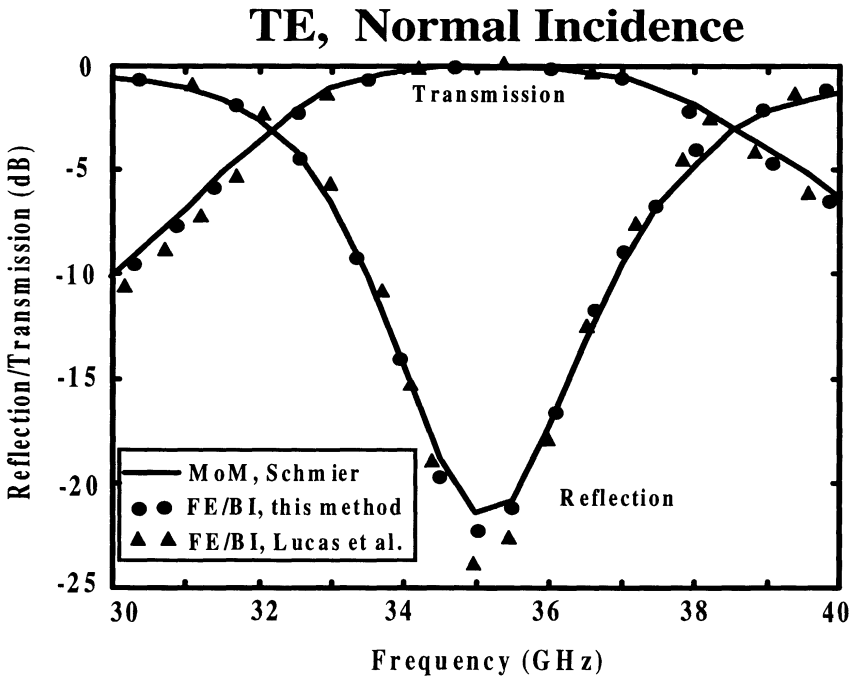
a=36.07 mm, b=34.04 mm, d=1.6 mm
W_s=2 mm, L_s=8/9 mm,
L_p=28 mm, W_p=28 mm
ε_r=2.2

2 x 5308 BI unknowns
98636 Total unknowns
about 3-4 hours per f (on IBM RS6000/590)

"Artificial Puck Plate" FSS

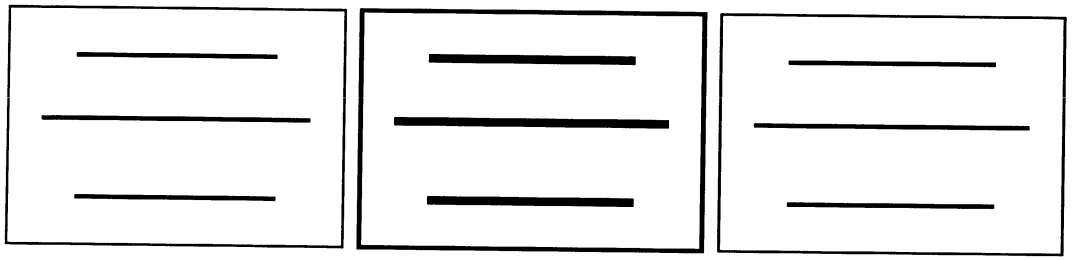






Commensurate and Non-Commensurate Periodicities

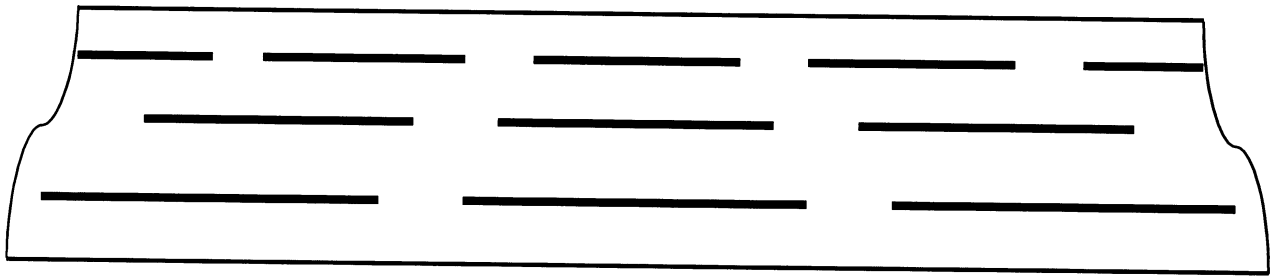
- **Commensurate Structures**



Unit cell

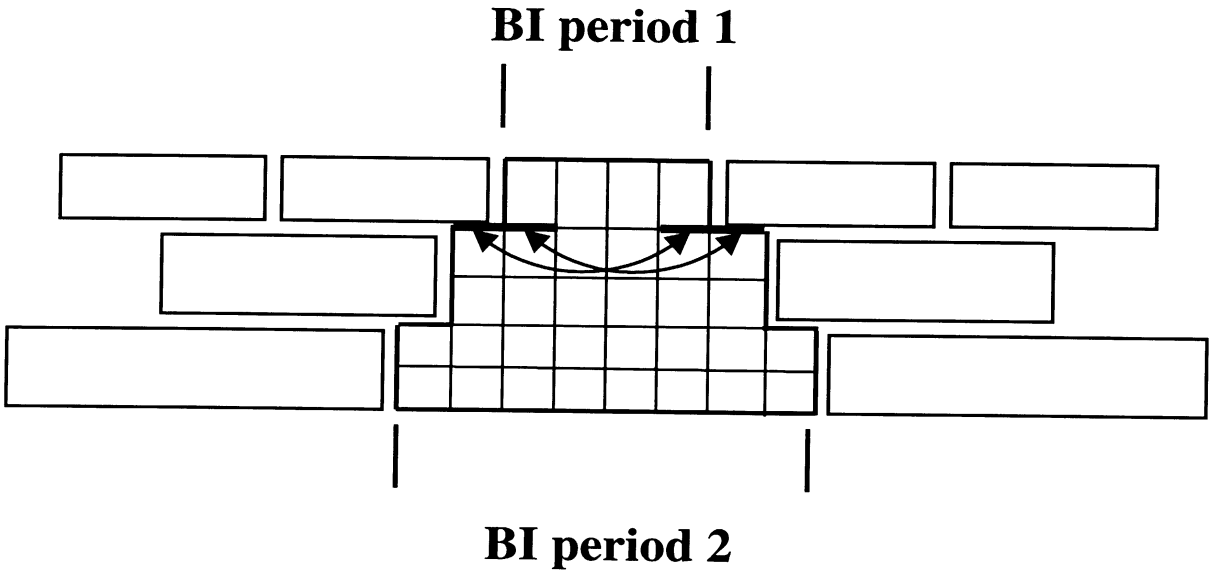
- **All layers have the same periodicity.**
- **For periodic excitation, exact modeling by considering one unit cell.**

- **Non-Commensurate Structures**



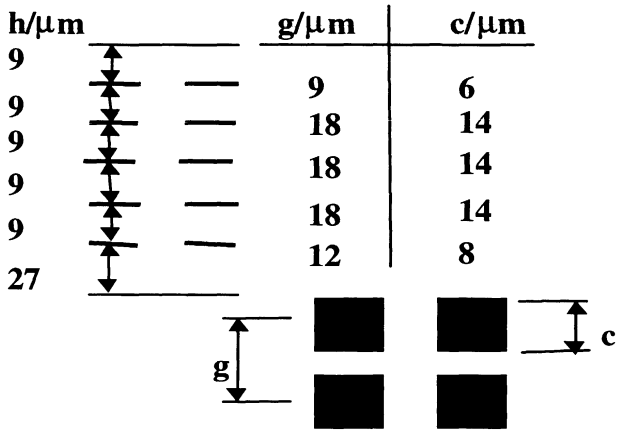
- **Different Periodicities for different Layers.**
- **In general, exact modeling not possible. (super-cell for special cases)**
- **Need for approximate model.**

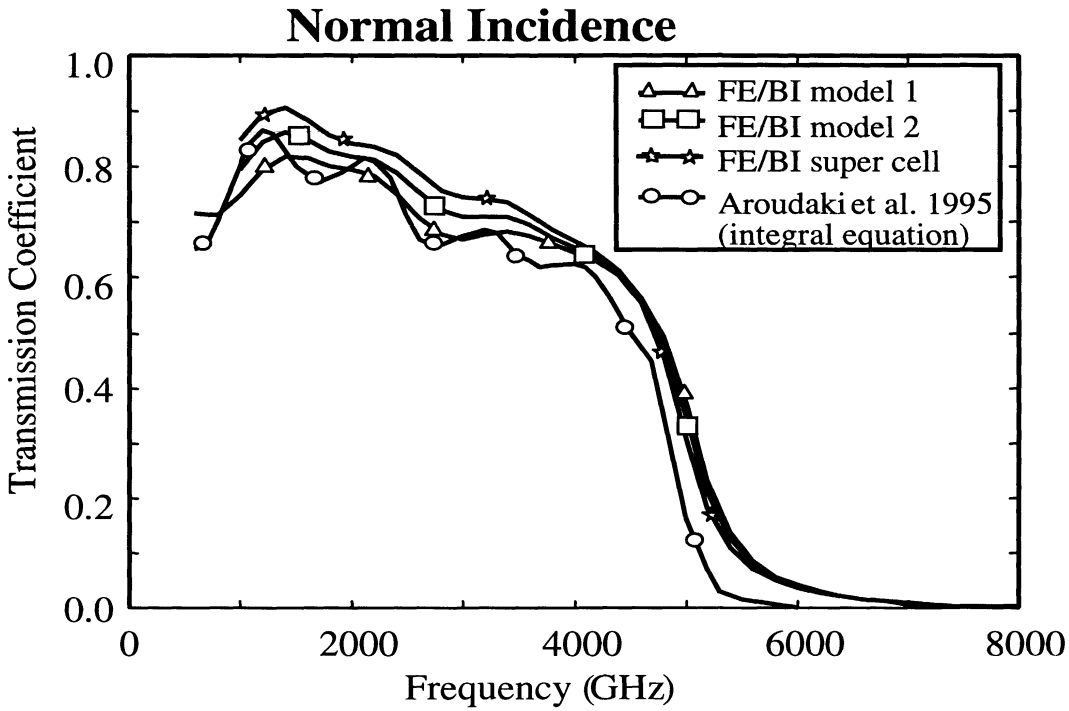
Modeling of Non-Commensurate Structures by Decoupling the Layer Periodicities



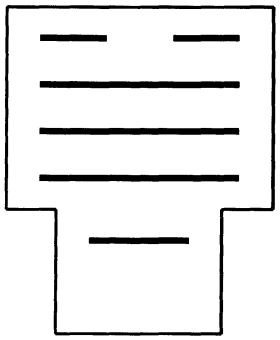
- assume fields to obey the geometric periodicities of the individual layers
- BI with different periodicities on top and bottom surfaces
- extended phase boundary conditions for horizontal boundary edges
- model improvement by grouping several periods in the individual layers

5-Layer FSS with Non-Commensurate Periodicities

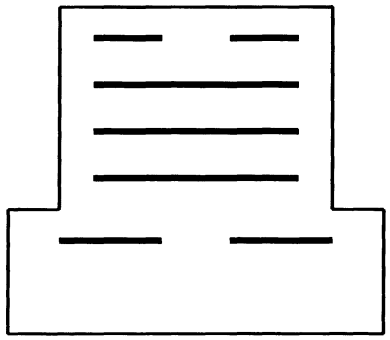




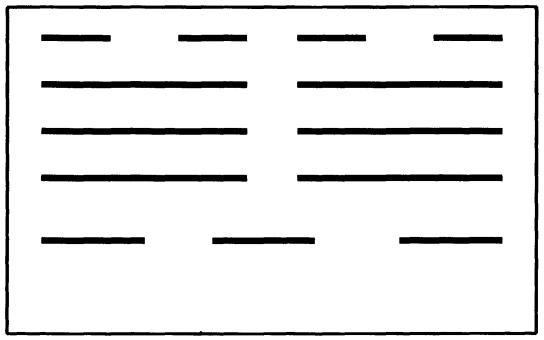
FE/BI model 1



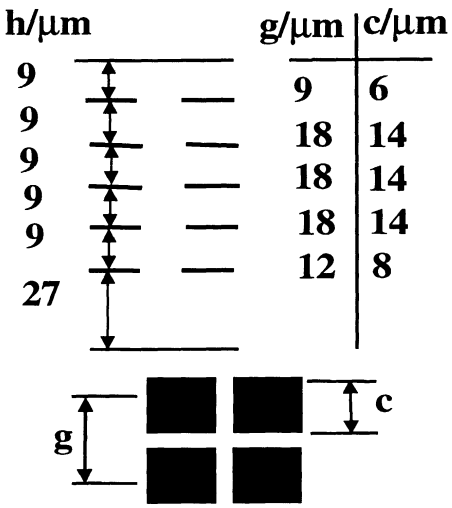
FE/BI model 2

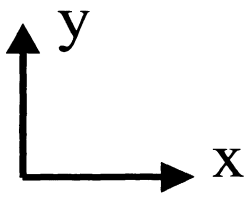


FE/BI super cell

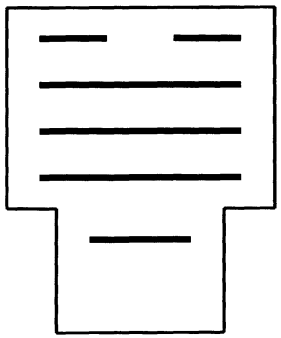


5-Layer FSS with Non-Commensurate Periodicities

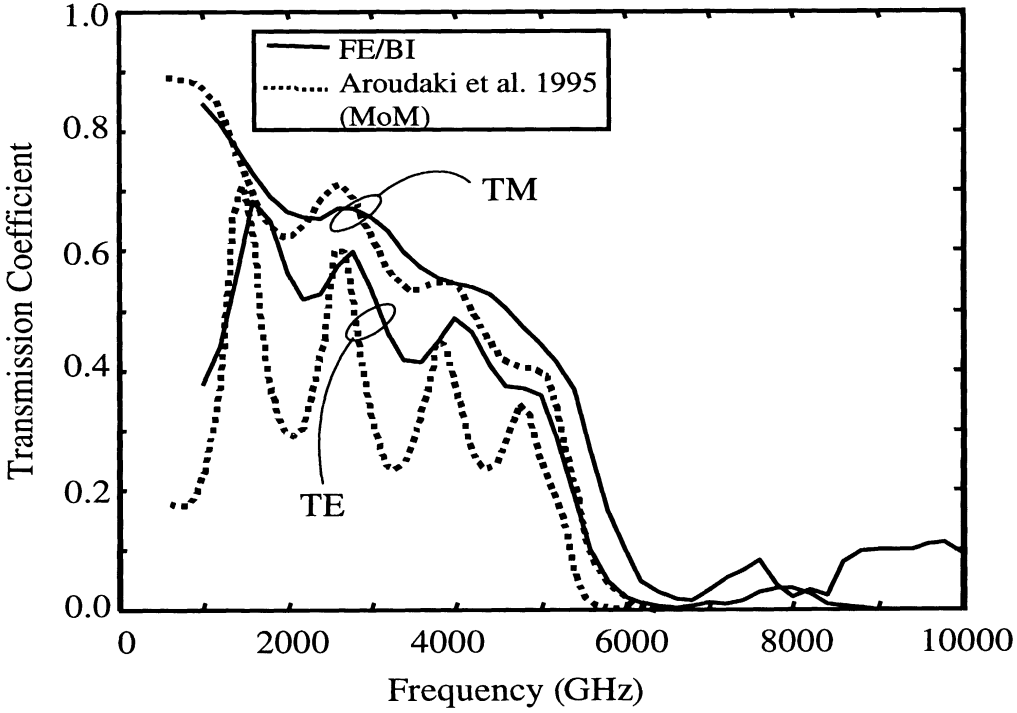




FE/BI model



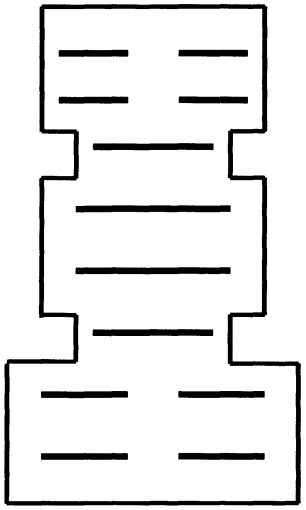
$\vartheta_0 = 70^o, \varphi_0 = 0^0$

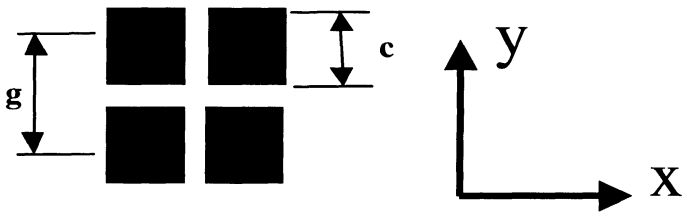


8-Layer FSS with Non-Commensurate Periodicities

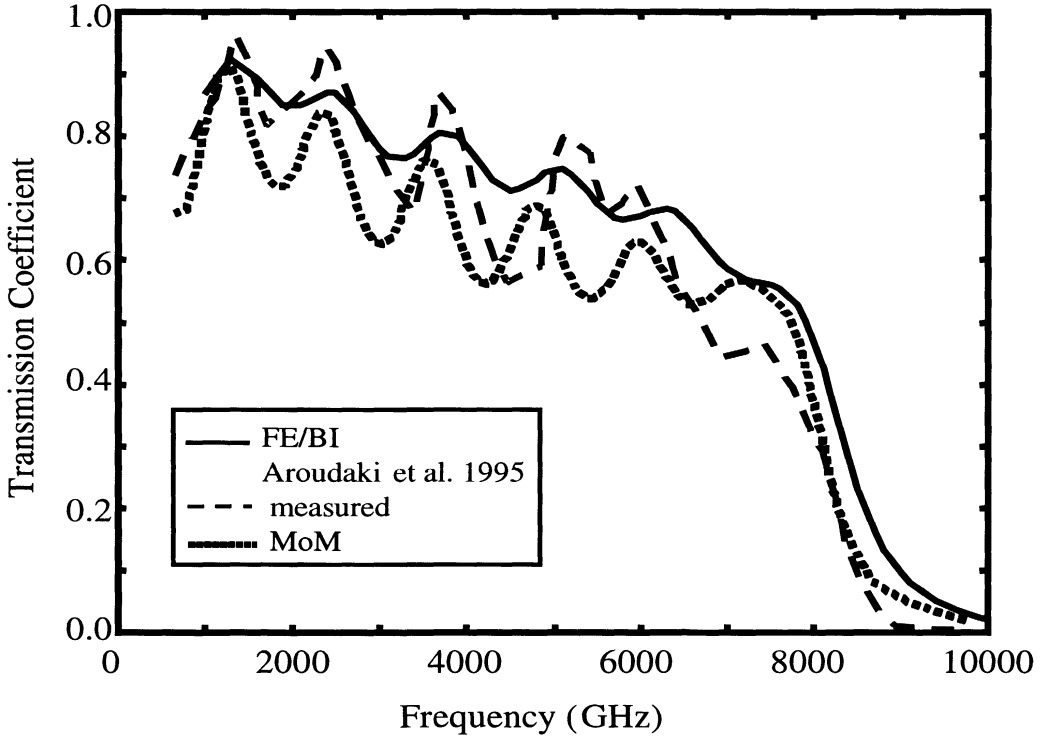
h/μm	g/μm	
7.8	5	3
7.8	5	3
7.8	9	6
7.8	10	7
7.8	10	7
7.8	9	6
7.8	6	4
7.8	6	4
7.8		

FE/BI model

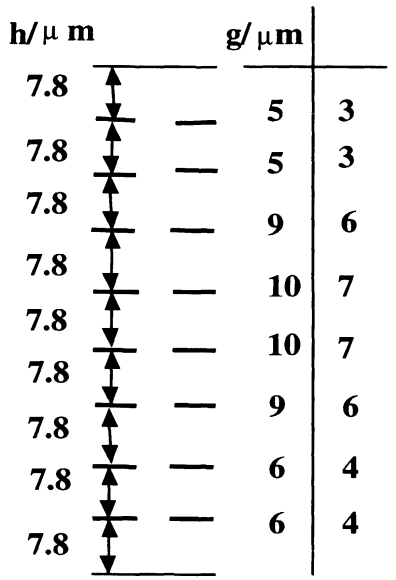




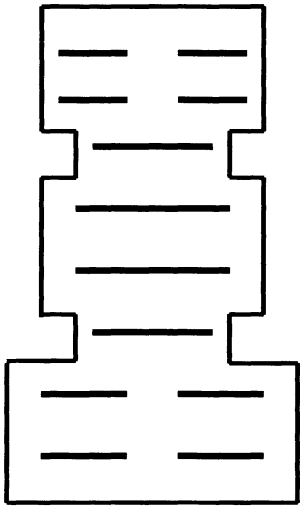
Normal Incidence

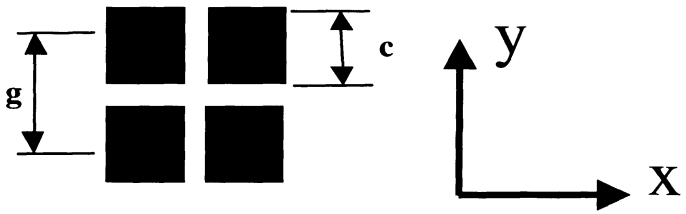


8-Layer FSS with Non-Commensurate Periodicities

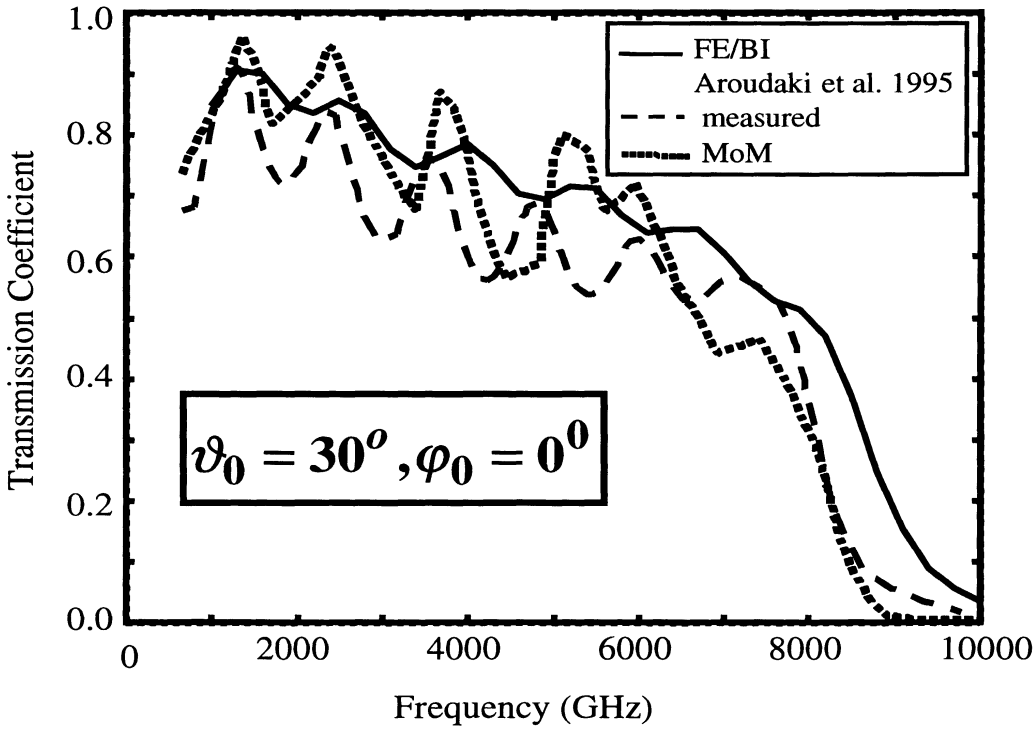


FE/BI model





Geometric mean of TE and TM



Conclusions

- Hybrid FE-BI modeling of doubly periodic structures
- FE with triangular prismatic elements
- MPIE formulation with Ewald acceleration of Green's function
- Approximate modeling of non-commensurate structures
- Results for commensurate and non-commensurate applications

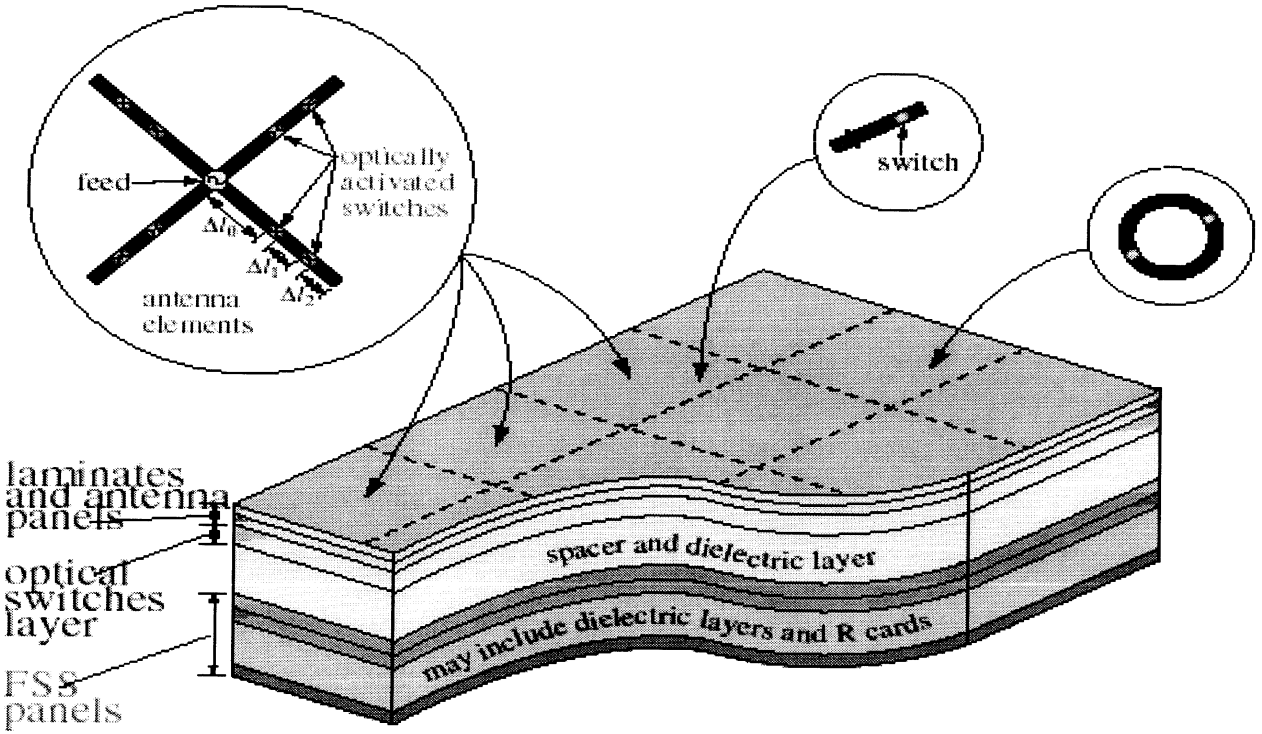
Adaptive Integral Method for Hybrid FE-BI Modeling of 3D Doubly Periodic Structures
by T.F. Eibert and J.L. Volakis

Adaptive Integral Method for Hybrid FE-BI Modeling of 3D Doubly Periodic Structures

Thomas F. Eibert and John L. Volakis
Radiation Laboratory, EECS Department
The University of Michigan
Ann Arbor, MI 48109-2122

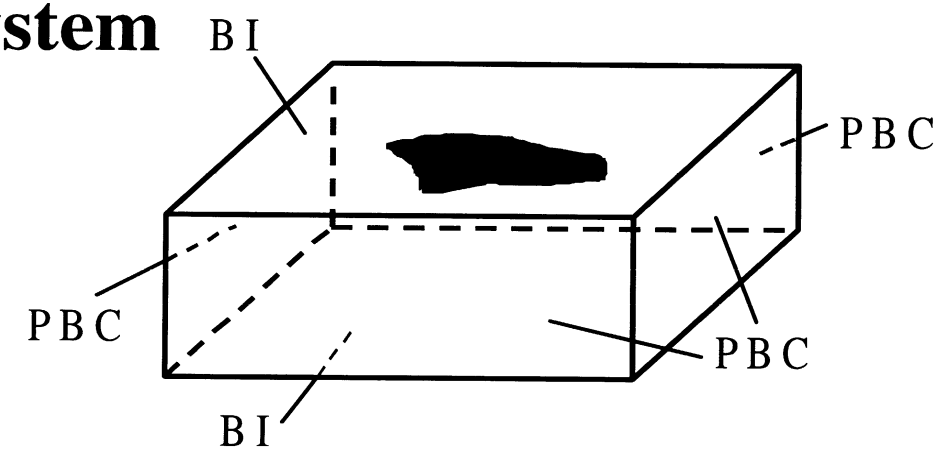
- Finite Element (FE) - Boundary Integral (BI) formulation
- Motivation and problem definition
- Some details of the AIM algorithm
- Validation and timing results
- Conclusions

Analysis of Doubly Periodic Structures
(Antenna Arrays and Frequency Selective Surfaces (FSS))



Need for three-dimensional modeling capabilities can be fulfilled with hybrid finite element (FE) - boundary integral (BI) formulation.

Formulation of the FE-BI System and Problem Definition



FE-BI hybrid system

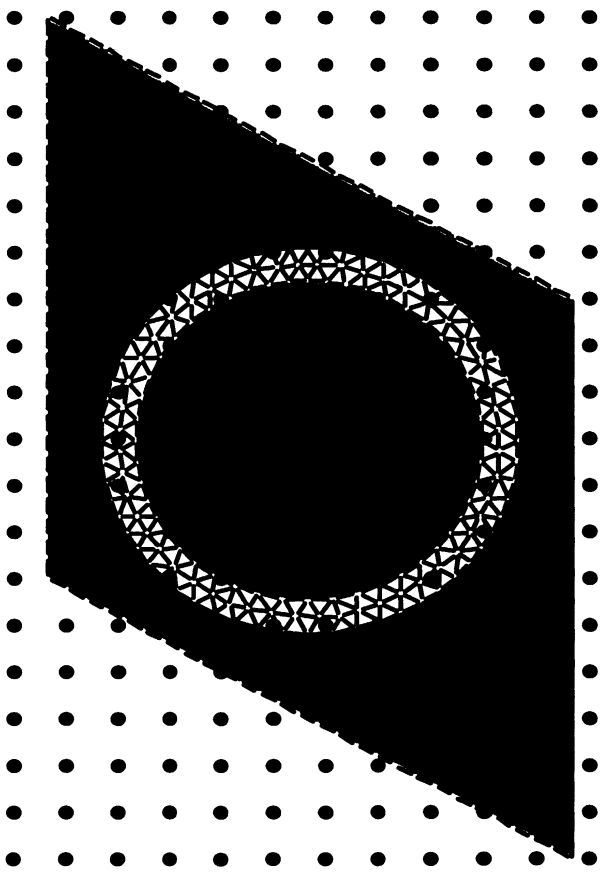
$$\begin{bmatrix} A^{\text{int}} & A^{cross}_{1,top} & A^{cross}_{1,bot} \\ A^{cross}_{2,top} & A^{bound}_{top} & 0 \\ A^{cross}_{2,bot} & 0 & A^{bound}_{bot} \end{bmatrix} \begin{Bmatrix} E^{\text{int}} \\ E^{bound}_{top} \\ E^{bound}_{bot} \end{Bmatrix} + \begin{bmatrix} 0 & 0 & 0 \\ 0 & Z_{top} & 0 \\ 0 & 0 & Z_{bot} \end{bmatrix} \begin{Bmatrix} E^{\text{int}} \\ E^{bound}_{top} \\ E^{bound}_{bot} \end{Bmatrix} = \begin{Bmatrix} f^{\text{int}} \\ f^{bound}_{top} \\ f^{bound}_{bot} \end{Bmatrix}$$

sparse FE matrices **fully populated BI matrices**

$$[Y]\{E\} = \{f\}$$

Perform $[Y]\{s\}$ many times within iterative solver ({s}: search vector)

Basic Principles of the Adaptive Integral Method (AIM)



- overlay uniform mesh on original mesh
- relate the meshes via moments of basis functions
- utilize convolutional properties of Green's function for uniform mesh
- calculate matrix-vector products (convolutions) in DFT domain and employ FFT algorithm (iterative solver assumed)
- correct for near-coupling terms of original mesh

→ $O(n \log n)$ for matrix-vector product
(planar BI surface assumed)

$O(n)$ memory requirement

Some Details of the AIM Algorithm

Decompose BI matrix in near-zone and far-zone contributions

$$[Z]=[Z]^{near}+[Z]^{far}$$

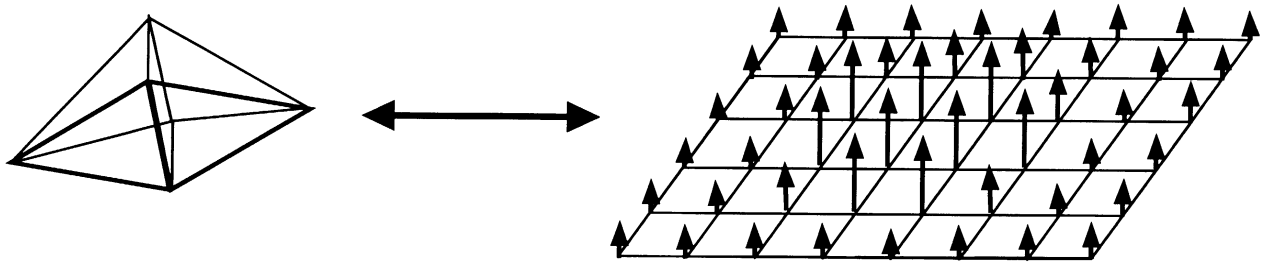
$[Z]^{near}$ is calculated in a conventional way (MPIE formulation in this work).

For the calculation of $[Z]^{far}$ lets consider the scalar coupling integral

$$Z_{mn}=\iint\limits_{S_m} q_m(\vec{r}) \iint\limits_{S_n} G_p(\vec{r}\,|\,\vec{r}^{\,'})\, q_n(\vec{r}^{\,'})\; dS^{'} dS$$

Goal: Calculate the coupling integral in the DFT domain employing the auxiliary uniform mesh.

Therefore, the basis functions q_n must be replaced by equivalent basis functions on the uniform mesh.



$$q_n(\vec{r}) \quad \Leftrightarrow \quad \sum_{i=1}^{I}\sum_{j=1}^{J} \Lambda_{ij}^{n}\; \delta(x-i\Delta x)\,\delta(y-j\Delta y)$$

Some Details of the AIM Algorithm: Continuation 1

Consider $Z_{mn} = \iint_{S_m} q_m(\vec{r}) \underbrace{\iint_{S_n} G_p(\vec{r} \mid \vec{r}\,') \, q_n(\vec{r}\,') \, dS'}_{} \, dS$

$$\boxed{Z_{mn} = \iint_{S_m} q_m(\vec{r}) \; g(\vec{r}) \; dS}$$

Introducing the Taylor series expansion $g(\vec{r}) = \sum_{q=q_1+q_2=0}^{\infty} a_q \, (x-x_1)^{q_1} \, (y-y_1)^{q_2}$

$$\rightarrow \quad Z_{mn} = \sum_{q=q_1+q_2=0}^{\infty} a_q \underbrace{\iint_{S_m} q_m(\vec{r}) \, (x-x_1)^{q_1} \, (y-y_1)^{q_2} \, dS}_{\text{moments of test function}}$$

summation of moments of test function multiplied by Taylor coefficients a_q

Enforcing

$$\boxed{\iint_{S_m} q_m(\vec{r}) \, (x-x_1)^{q_1} \, (y-y_1)^{q_2} \, dS = \sum_{i=1}^{I} \sum_{j=1}^{J} \Lambda_{ij}^{m} \, (i\Delta x - x_1)^{q_1} \, (j\Delta y - y_1)^{q_2} \qquad q = q_1 + q_2 = 0, \ldots, \infty}$$

gives the required relation between the original and uniform meshes and allows the calculation of Λ_{ij}^{m} .

Some Details of the AIM Algorithm: Continuation 2

In numerical implementation:
only finite number of moments possible to relate each test or basis function to a finite number of δ-functions on the uniform grid (accurate for far interactions)

Each test and basis function can now be replaced by the related δ-functions on the uniform grid.

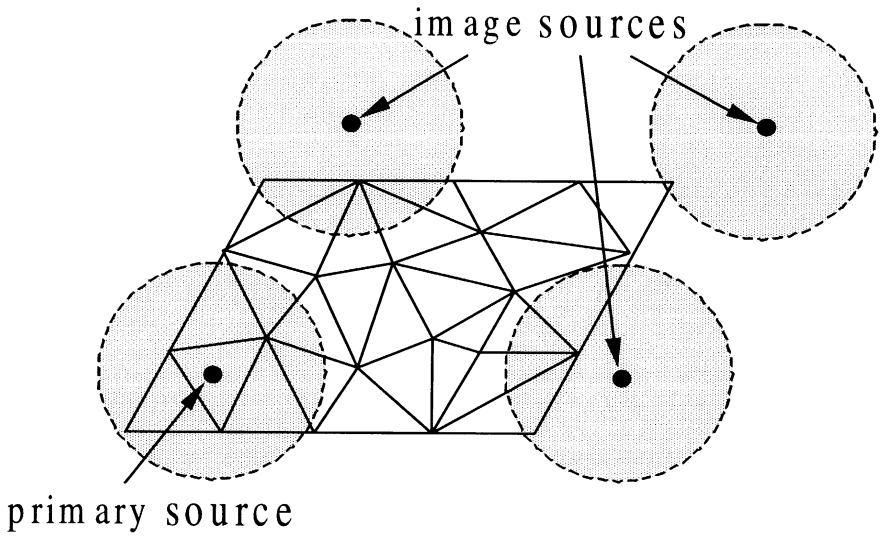
$$Z_{mn} \approx \sum_{i=(i_m-1)}^{(i_m+1)} \sum_{j=(j_m-1)}^{(j_m+1)} \sum_{k=(k_n-1)}^{(k_n+1)} \sum_{l=(l_n-1)}^{(l_n+1)} \Lambda_{kl}^{n}\, G_p((i\Delta x, j\Delta y)\,|\,(k\Delta x, l\Delta y))\;\; \Lambda_{ij}^{m}$$

Important advantage of this concept:

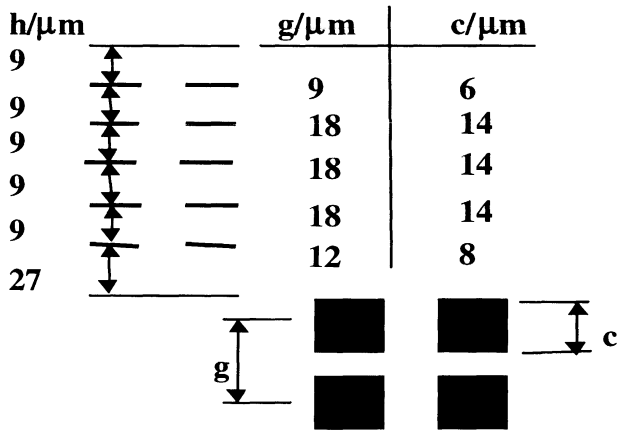
mapping procedure independent of Green's function

→ AIM can directly be applied to periodic problems.

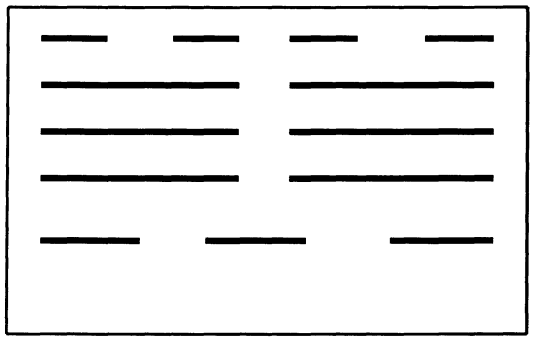
However, "near-zone" must be extended and periodic boundary condition must be taken care of.

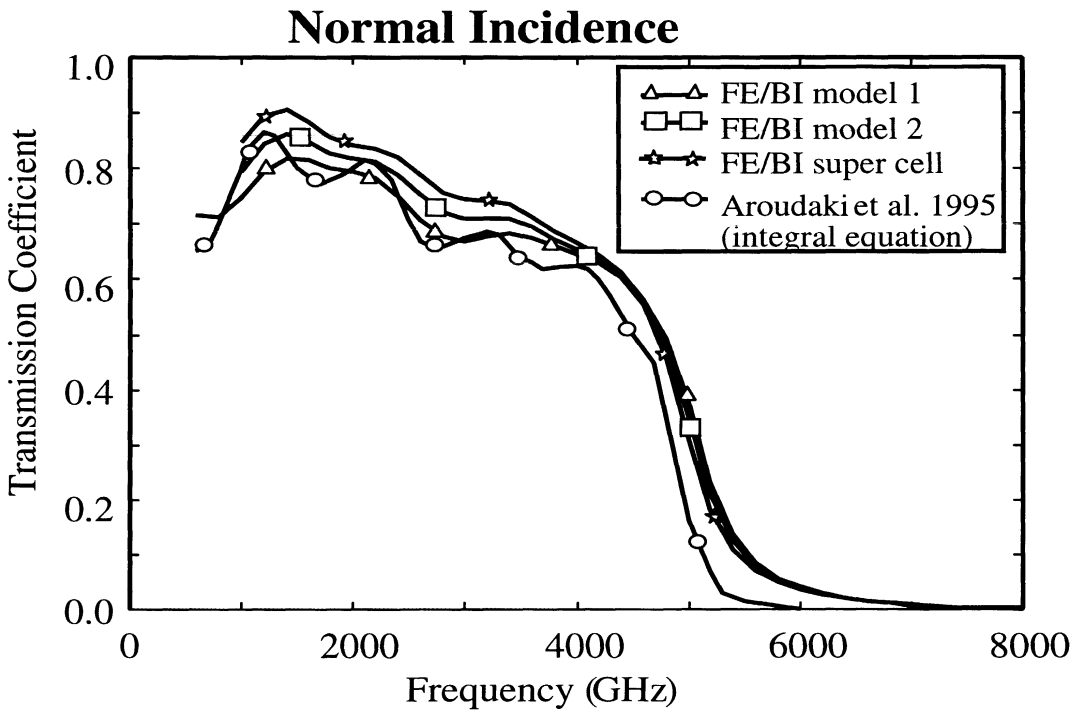


5-Layer FSS with Non-Commensurate Periodicities



FE/BI super cell

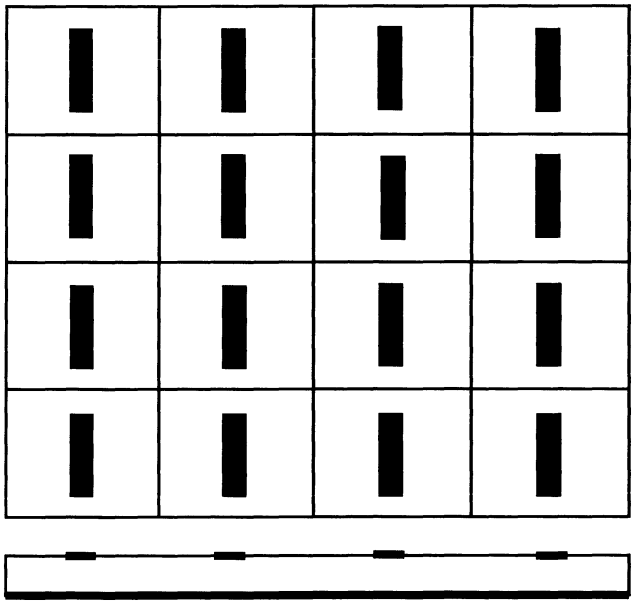




2 x 3960 BI unknowns
200064 Total unknowns
about 3-4 hours per f (on IBM RS6000/590)

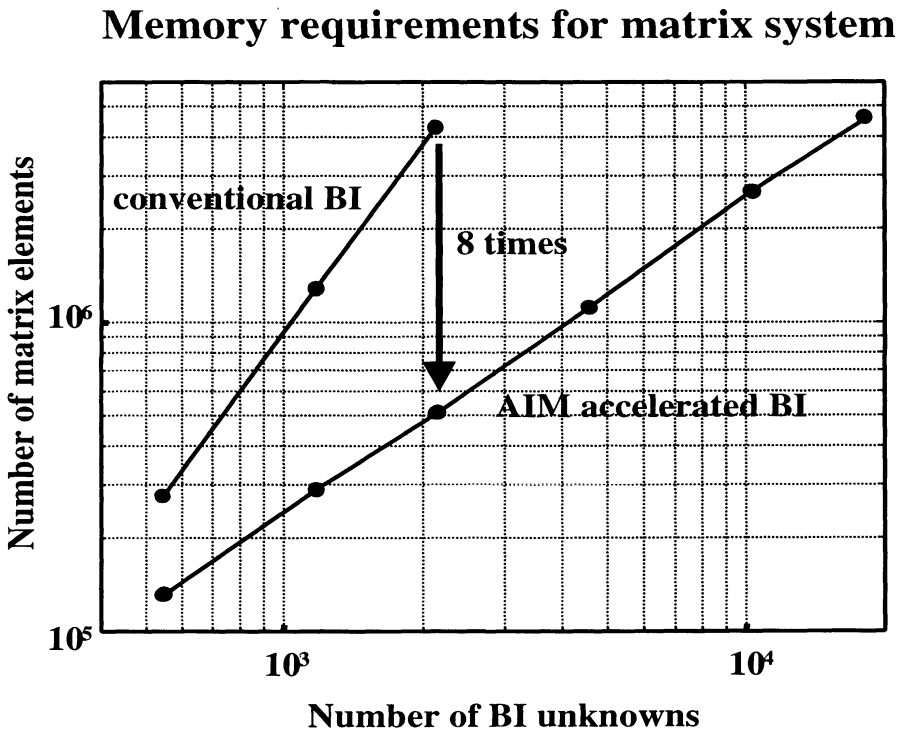
CPU Timings and Memory Comparisons

Test problem:
infinite array of microstrip dipoles

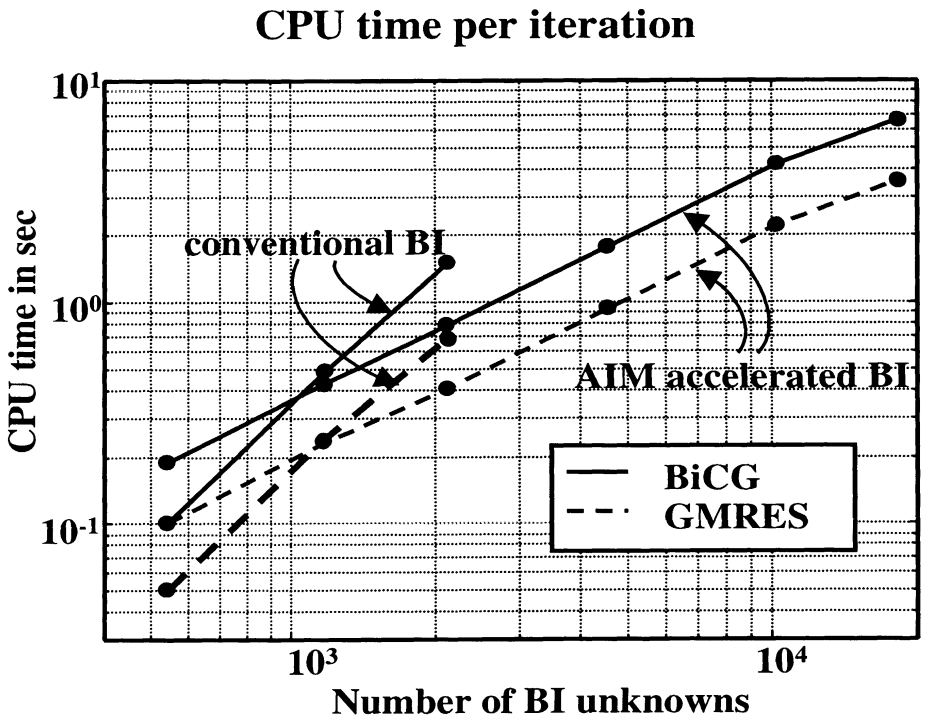


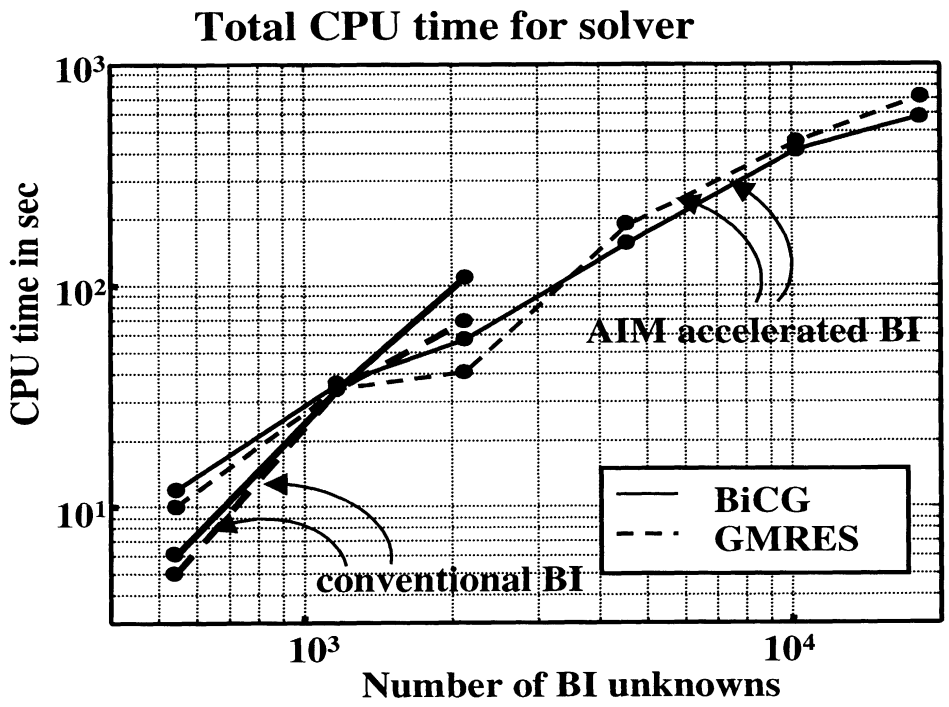
Increasing unknowns count by grouping several dipoles in unit cell

One prism layer



CPU Timings and Memory Comparisons: Continuation 1

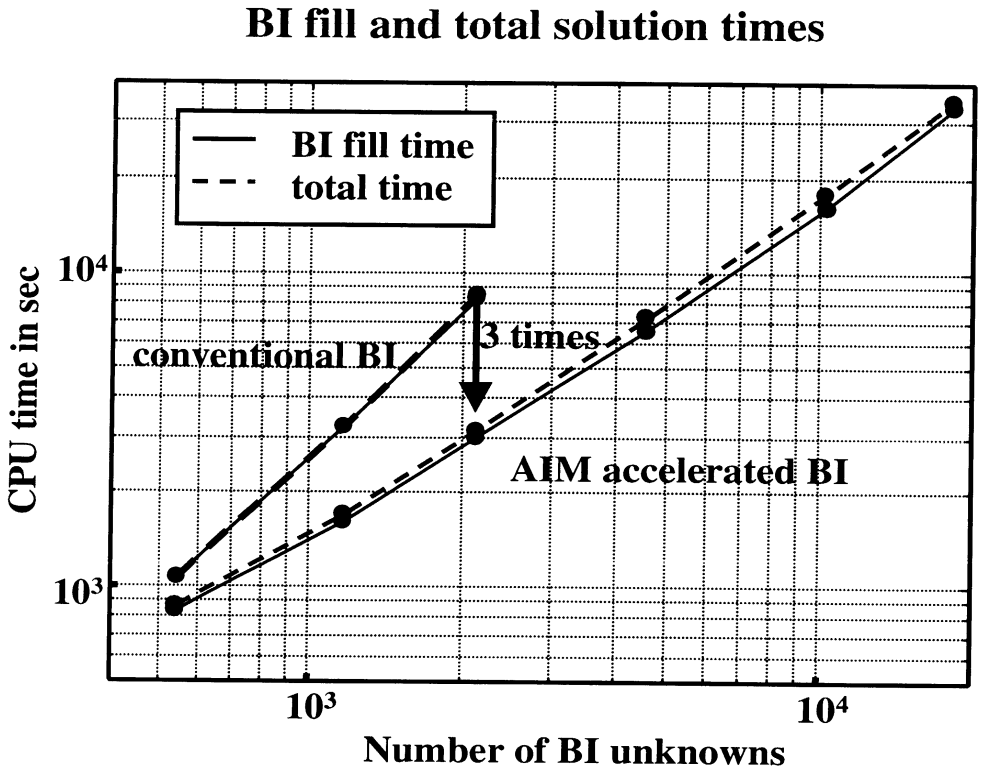




GMRES: restarted after 50 iterations

Computer: Pentium II PC 266 MHz

CPU Timings and Memory Comparisons: Continuation 2



Total solution time dominated by BI fill time.

However, this is due to time consuming series representation of periodic Green's function.

Computer: Pentium II PC 266 MHz

Conclusions

- Hybrid FE-BI modeling of doubly periodic structures
- Need for speed-up of BI
- Derivation of the Adaptive Integral Method (AIM)
- Validation results
- Memory requirements and CPU timings

Fast Multipole Method Solutions of Finite Element-Boundary Integral Implementations for Antenna Modeling Involving Curved Geometries by K Sertel and J.L. Volakis

Fast Multipole Method (FMM) Solutions of Finite Element-Boundary Integral (FE-BI) Implementations for Antenna Modeling Involving Curved Geometries

Kubilay SERTEL and John L. Volakis

Radiation Laboratory
Dept. of Electrical Engin. and Computer Science
University of Michigan
Ann Arbor, MI 48109-2122

Goals

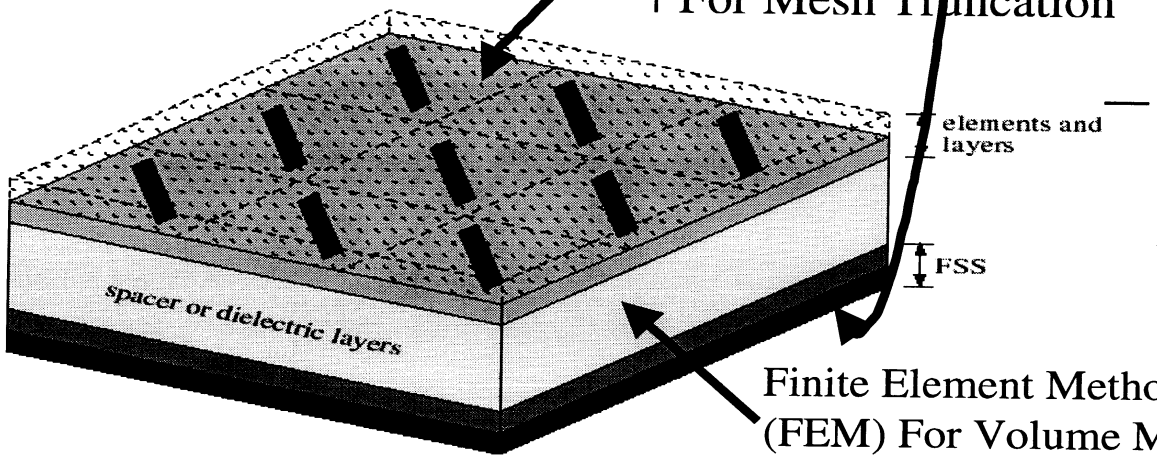
- Fast and Practical Analysis of Finite Arrays on Doubly Curved Surfaces
- Evaluate the Accuracy of FE-BI with Fast Algorithms for Antenna Simulations
- Study the Effects Curvature and the Finite Aperture on Array Performance

Outline

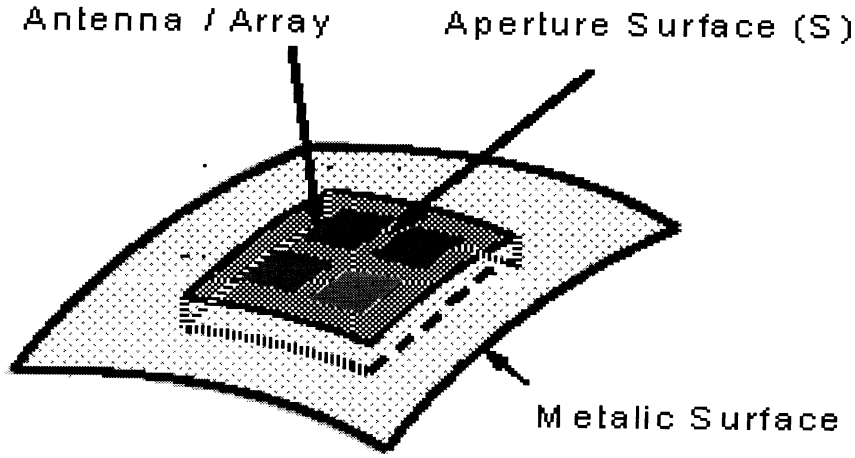
- Problem Geometry: Cavity Backed Patch Antenna Arrays on Curved Platforms
- FE-BI and the Fast Multipole Formulation
- Time and Memory Reduction by Employing FMM
- Examples
- Conclusions

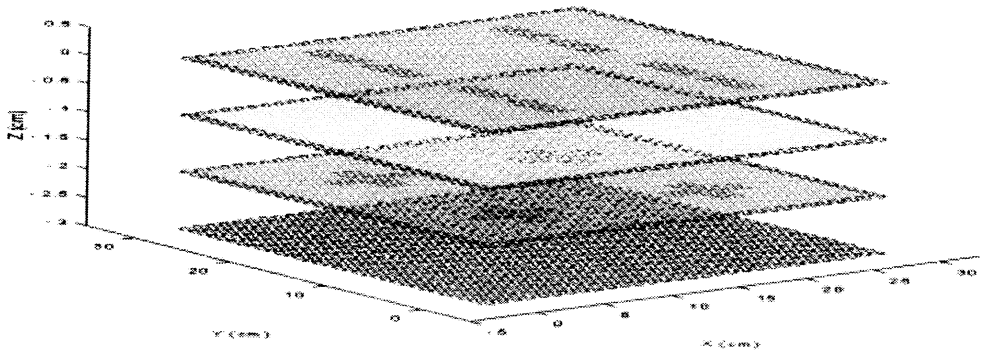
Problem Geometry

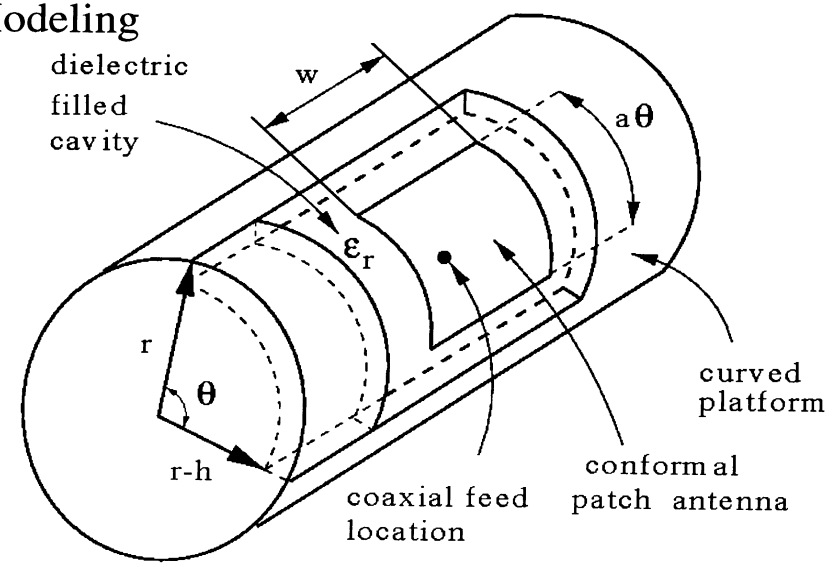
Boundary Integral(BI)
For Mesh Truncation



Finite Element Method
(FEM) For Volume Modeling







FE-BI Formulation

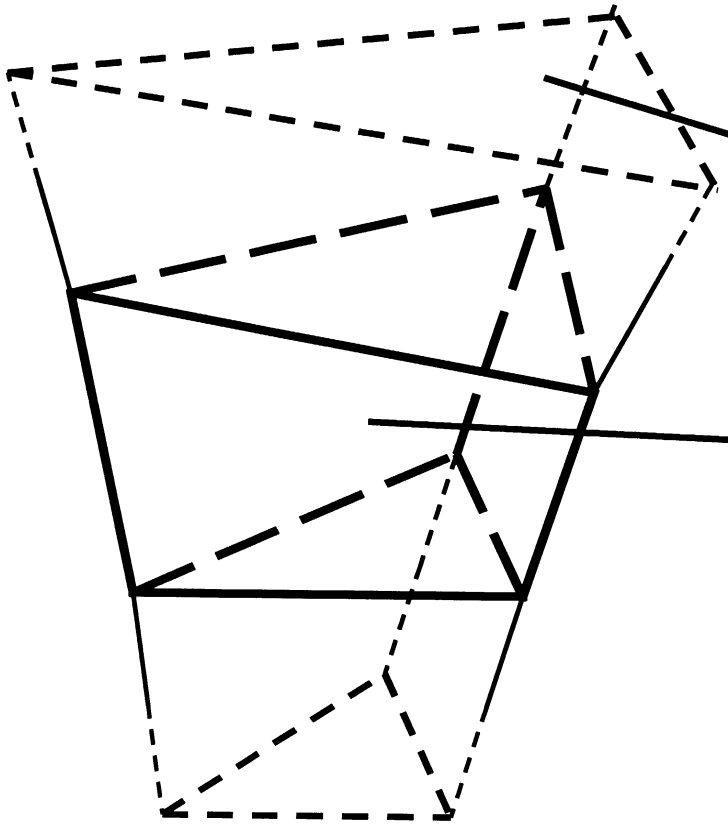
$$\langle \mathbf{E}, \mathbf{T} \rangle = \iiint_{V} \left[\frac{1}{\mu_r} (\nabla \times \mathbf{T}) \cdot (\nabla \times \mathbf{E}) - \varepsilon_r k^2 \mathbf{E} \cdot \mathbf{T} + jkZ_0 \mathbf{T} \cdot \mathbf{J}^{\text{int}} \right] \mathrm{dv} + jkZ_0 \iint_{S} \mathbf{T} \cdot (\mathbf{H} \times \hat{n}) \, \mathrm{ds}$$

$$\mathbf{H} = -2j \frac{k}{Z_0} \left[\iint_{S} g(\mathbf{r}, \mathbf{r}') (\mathbf{E} \times \hat{n}) \mathrm{ds} - \frac{1}{k^2} \nabla \iint_{S} g(\mathbf{r}, \mathbf{r}') \nabla' \cdot (\mathbf{E} \times \hat{n}) \mathrm{ds} \right] + \mathbf{H}^{inc}$$

$$B_{ij} = -2j \frac{k}{Z_0} \iint_{S} \mathrm{ds} \, \mathbf{b_i}(\mathbf{r}) \cdot \iint_{S'} \mathrm{ds}' \left[\mathbf{b_j}(\mathbf{r}') - \frac{1}{k^2} \nabla' \cdot \mathbf{b_j}(\mathbf{r}') \nabla \right] \frac{e^{-jkR}}{R}$$

FE-BI Formulation

Distorted Triangular Prism Elements
[Ozdemir and Volakis, AP-May 97]



BI part generates
Full Matrix

FE part generates
Sparse Matrix

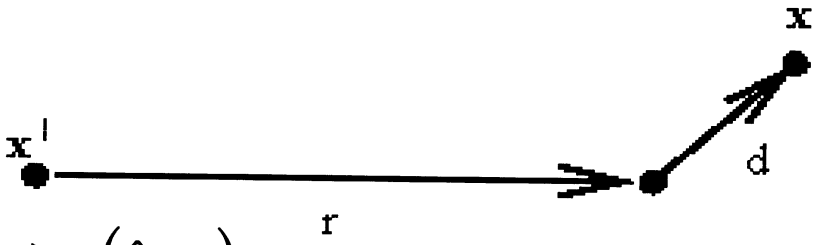
Sparse Matrix

Dense

$$\begin{bmatrix} A^{\text{int}} & A^{cross} \\ A^{Tcross} & A^{bound} \end{bmatrix} \begin{Bmatrix} E^{\text{int}} \\ E^{bound} \end{Bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & G \end{bmatrix} \begin{Bmatrix} E^{\text{int}} \\ E^{bound} \end{Bmatrix} = \begin{Bmatrix} f^{\text{int}} \\ f^{ext} \end{Bmatrix}$$

FE-BI and the Fast Multipole Method [Coifman et al., AP-June 93]

- **Translation Formulas**



$$\frac{e^{-jk|r+d|}}{|r+d|} = -jk\sum_{l=0}^{\infty}(-1)^l(2l+1)j_l(kd)h_l^{(2)}(kr)P_l\left(\hat{d}\cdot\hat{r}\right)$$

$$(-1)^l 4\pi\, j^l\, j_l(kd)P_l\left(\hat{d}\cdot\hat{r}\right) = \int d^2\hat{k}\, e^{-jk\cdot d}\, P_l\left(\hat{k}\cdot\hat{r}\right)$$

$$\frac{e^{-jk|r+d|}}{|r+d|} = \frac{-jk}{4\pi}\int d^2\hat{k}\, e^{-jk\cdot d}\sum_{l=0}^{\infty}(-j)^l(2l+1)h_l^{(2)}(kr)P_l\left(\hat{k}\cdot\hat{r}\right)$$

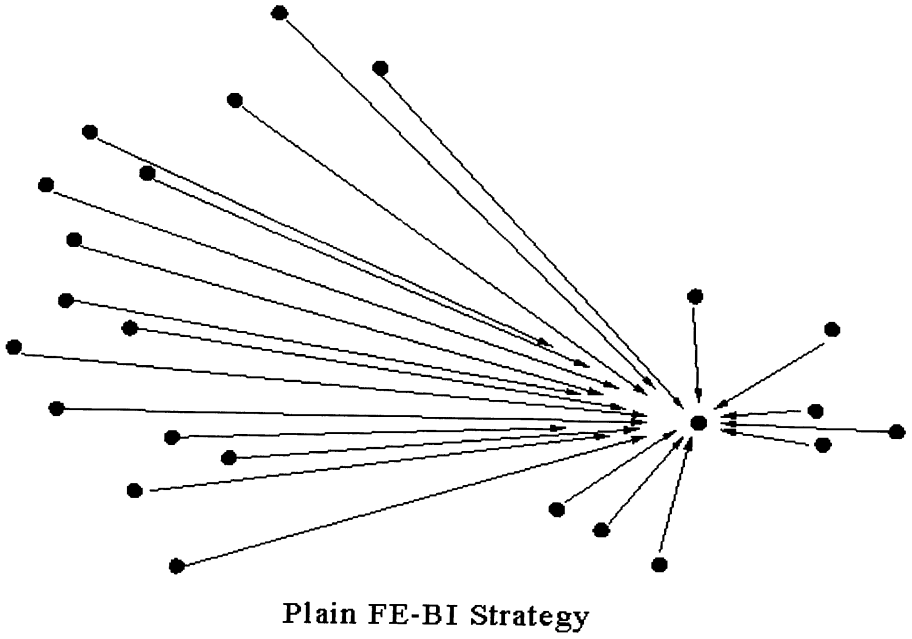
$$T_L\left(kr,\hat{k}\cdot\hat{r}\right) = \sum_{l=0}^{L}(-j)^l(2l+1)h_l^{(2)}(kr)P_l\left(\hat{k}\cdot\hat{r}\right)$$

$$V_{sm'i}\left(\hat{k}\right) = \iint_s ds\; e^{-j\mathbf{k}\cdot\mathbf{r}_{im'}}\left[I-\hat{k}\hat{k}\right]\cdot\mathbf{b}_i(\mathbf{r}_i) \qquad V_{fmj}\left(\hat{k}\right) = \iint_s ds\; e^{-j\mathbf{k}\cdot\mathbf{r}_{jm}}\left[I-\hat{k}\hat{k}\right]\cdot\mathbf{b}_j(\mathbf{r}_j)$$

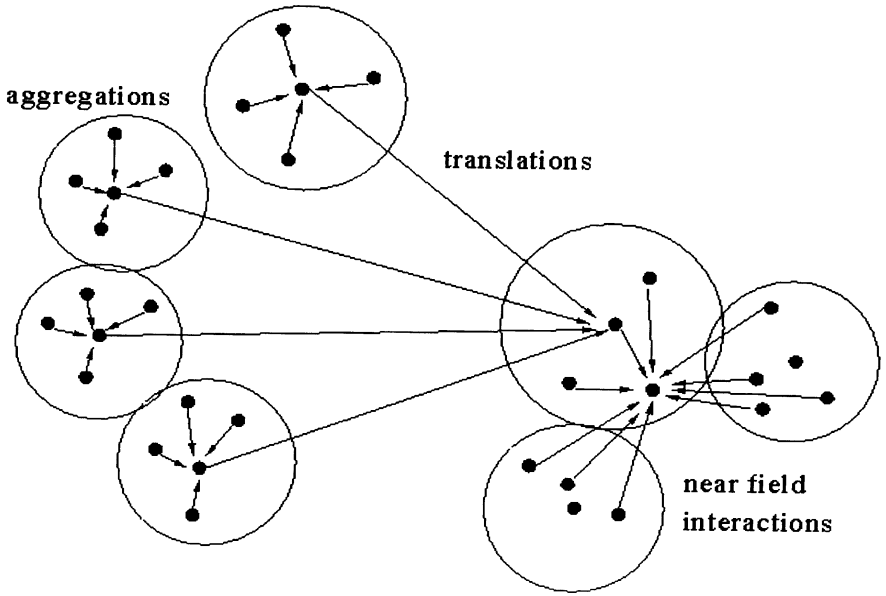
FE-BI and the Fast Multipole Method

$$B_{ij} = -2j\frac{k}{Z_0}\iint_{s} ds\, \mathbf{b_i}(\mathbf{r})\cdot\iint_{s'} ds'\left[\mathbf{b_j}(\mathbf{r}') - \frac{1}{k^2}\nabla'\cdot\mathbf{b_j}(\mathbf{r}')\nabla\right]\frac{e^{-jkR}}{R}$$

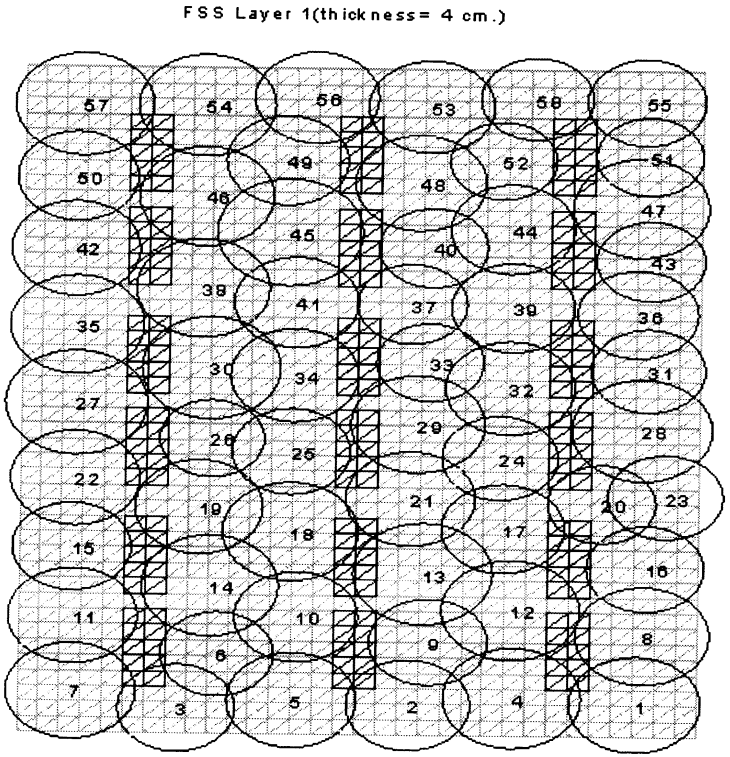
$$B_{ij} \approx \frac{-k^2}{2\pi Z_0}\int d^2\hat{k}\, V_{fmj}\left(\hat{k}\right)\cdot T_L\left(kr, \hat{k}\cdot\hat{r}\right)V^*_{sm'i}\left(\hat{k}\right)$$



Plain FE-BI Strategy



Clustering of the Surface Unknowns

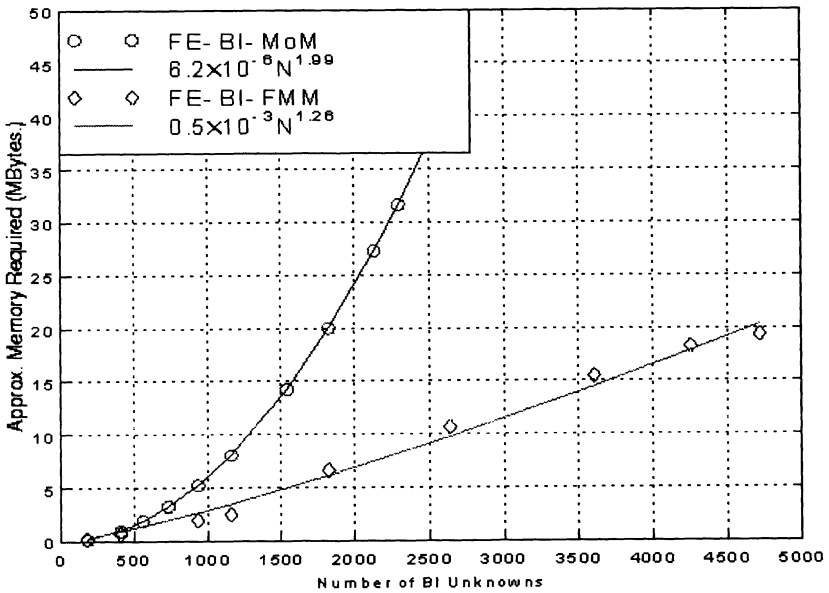


- Shown is 3 by 3 dipole antenna array mesh.
- "k-means" algorithm is used to form the clusters.
- Circles are centered at the cluster centers and pass through the furthest element in the cluster.
- Each element belongs to one cluster only !

- Red : PEC
- Green : Dielectric

CPU Requirements and Complexity

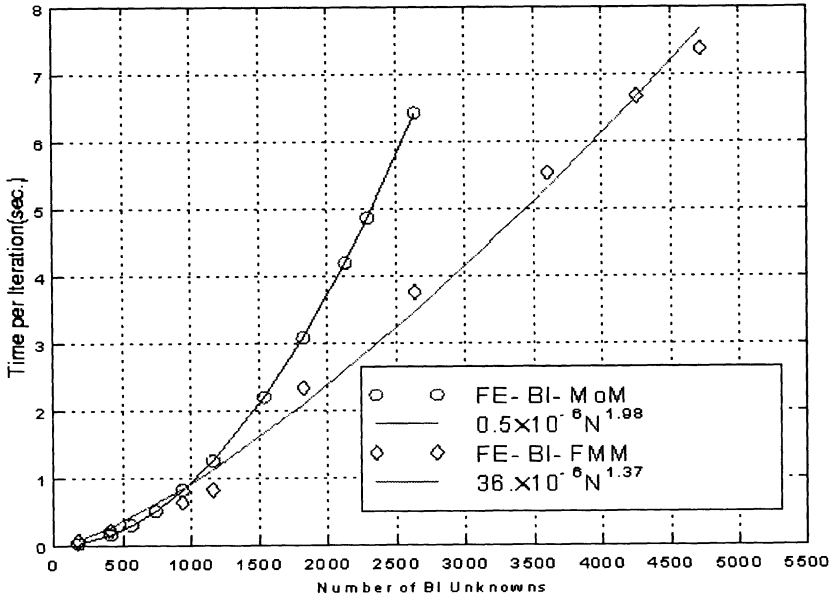
Memory



FE-BI $O(N^{1.99})$

FE-BI-FMM $O(N^{1.26})$

CPU Time per Iteration

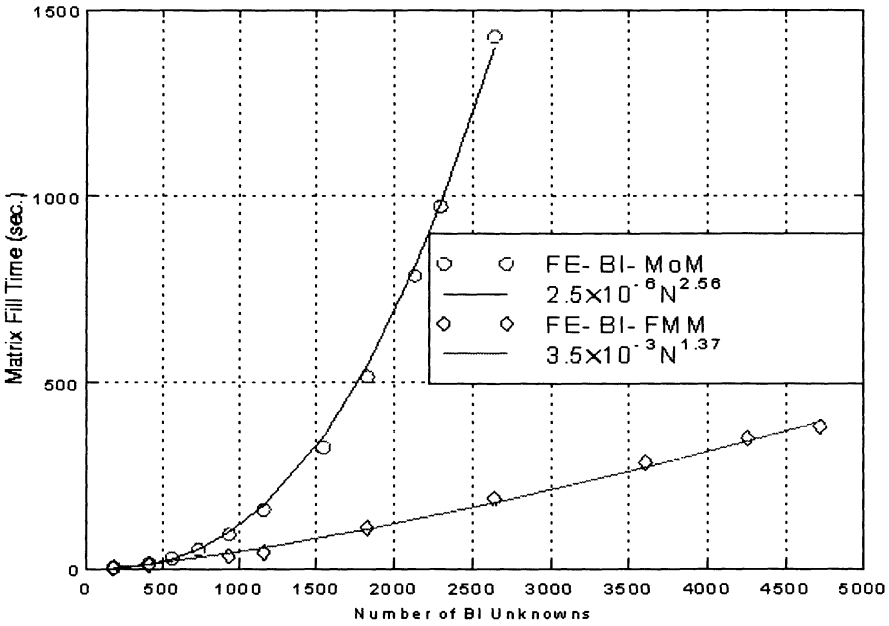


FE-BI $O(N^{1.98})$

FE-BI-FMM $O(N^{1.37})$

CPU Requirements

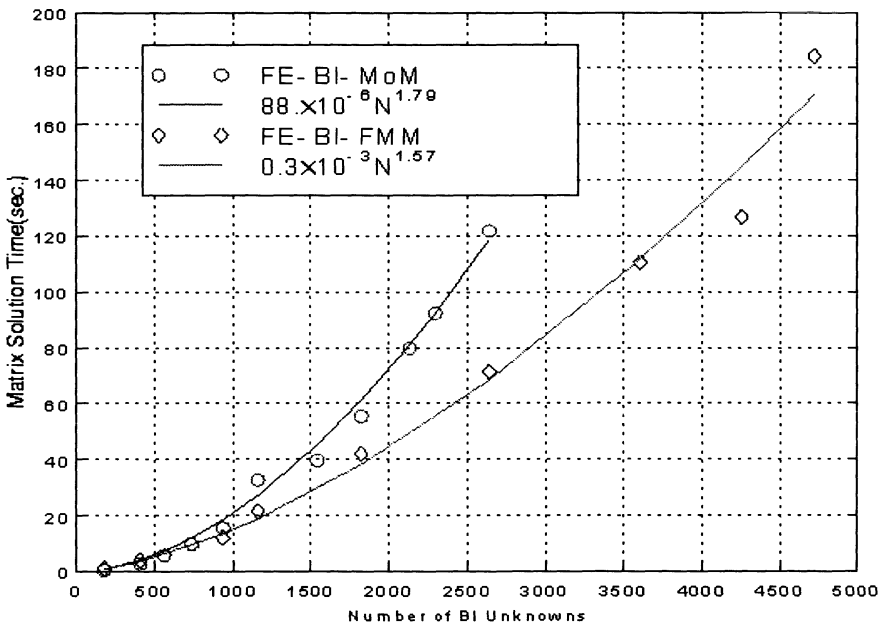
Matrix Fill Time



FE-BI $O(N^{2.56})$

FE-BI-FMM $O(N^{1.37})$

Matrix Solution Time

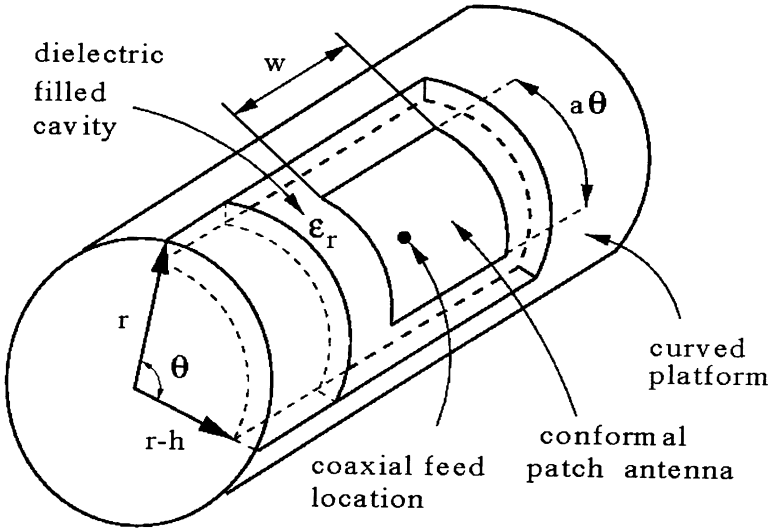


FE-BI $O(N^{1.79})$

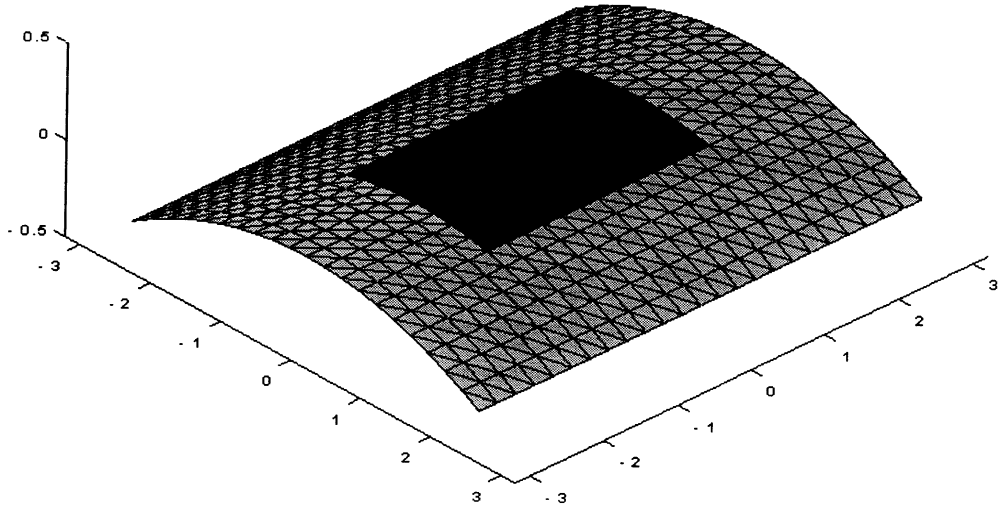
FE-BI-FMM $O(N^{1.57})$

Patch Antenna on a Cylinder

Studied by Sothell[89],
Kempel and
Volakis [AP-Sep. 94].

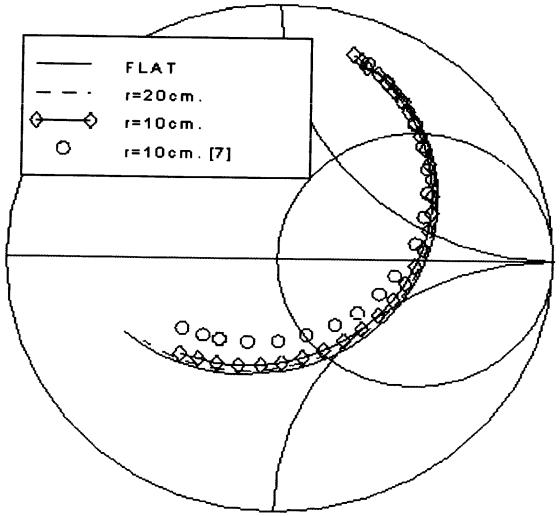


Antenna Simulated for
Different Cylinder Radii

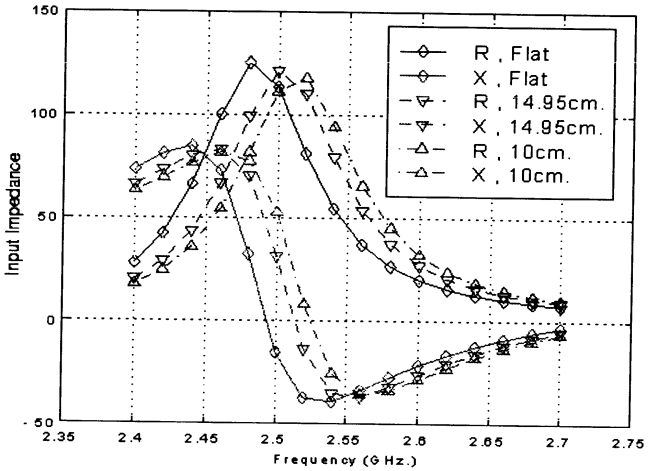


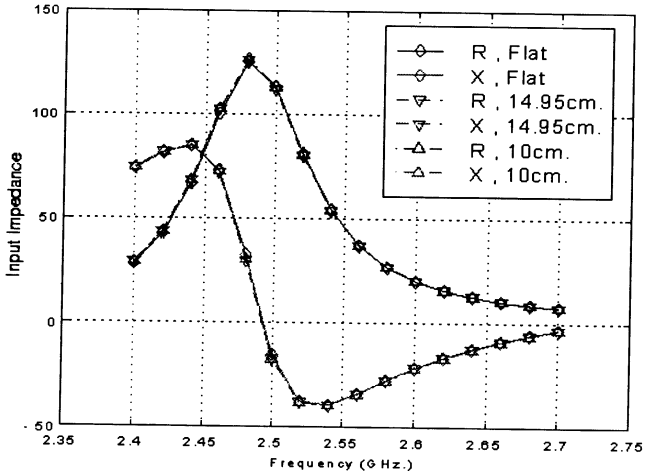
3.5cm by 3.5cm patch antenna on a cylindrical .3175 cm. deep cavity filled with dielectric of relative permittivity 2.32.

Input Impedance Dependence on Curvature

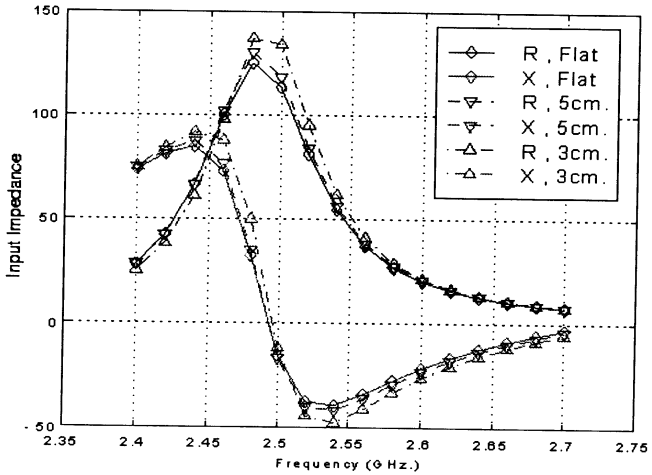


Circumferential Polarization

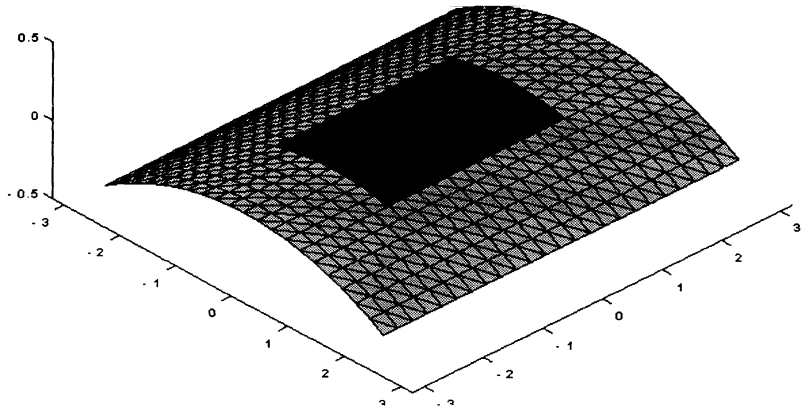




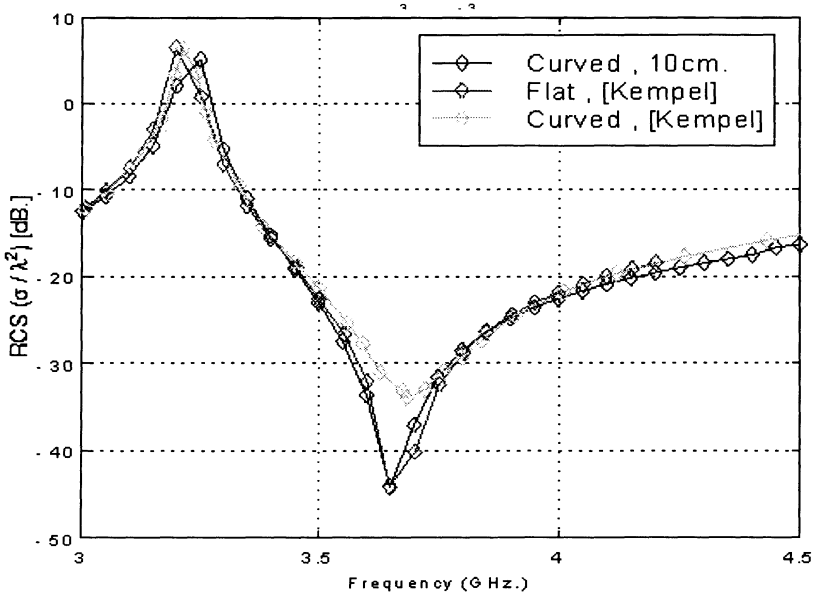
Axial Polarization

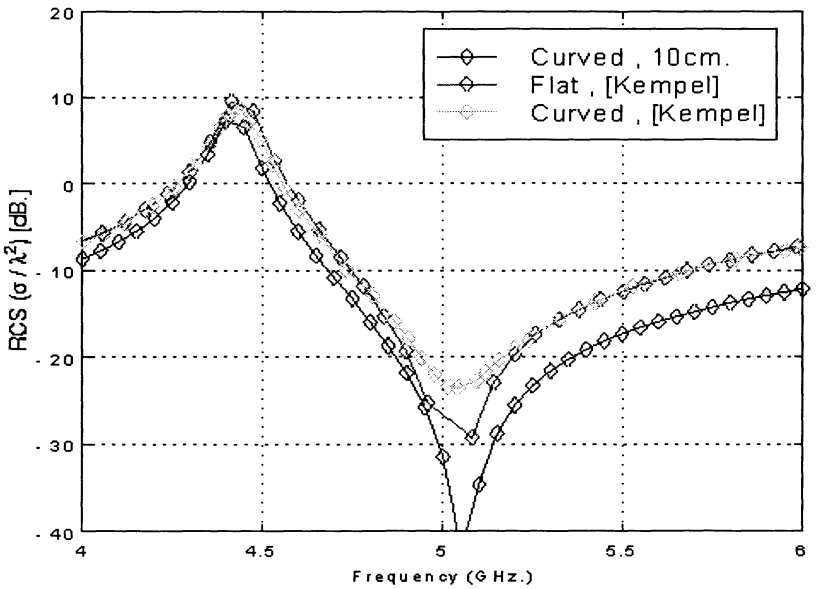


RCS Dependence on Curvature

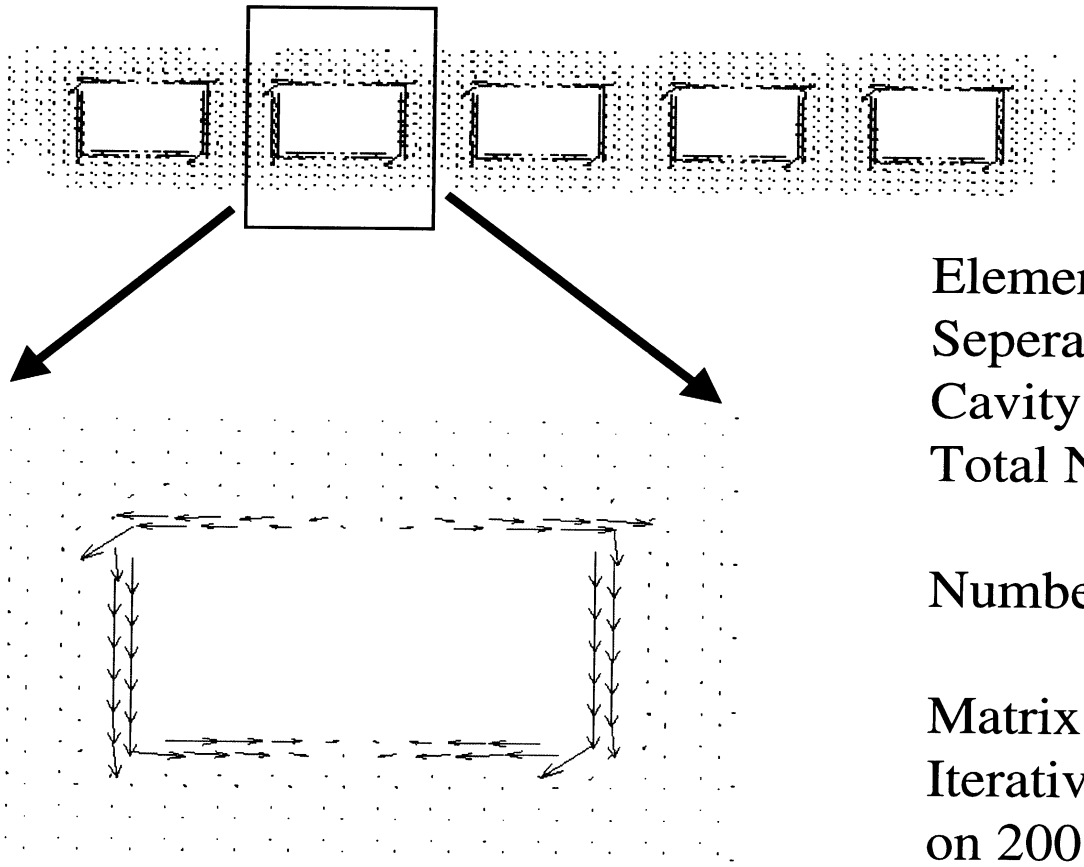


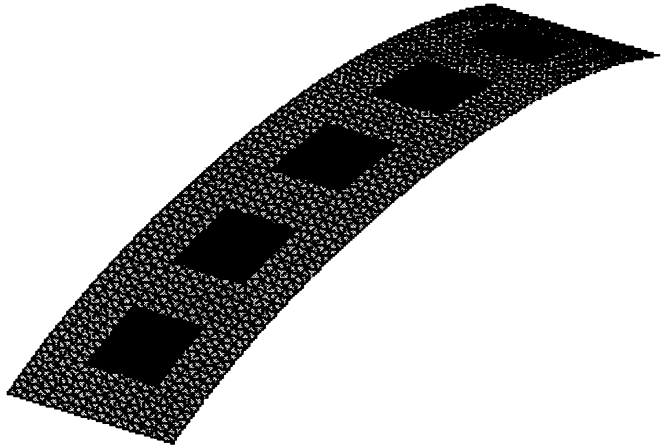
2cm. by 3cm. patch antenna on a 5cm. by 6cm. , 0.078cm. deep cavity filled with dielectric of relative permittivity 2.17





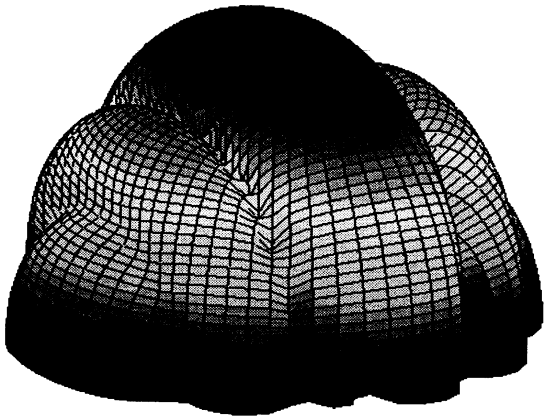
5 Element Antenna Array

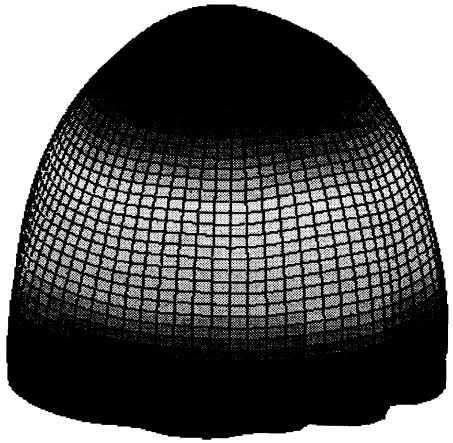


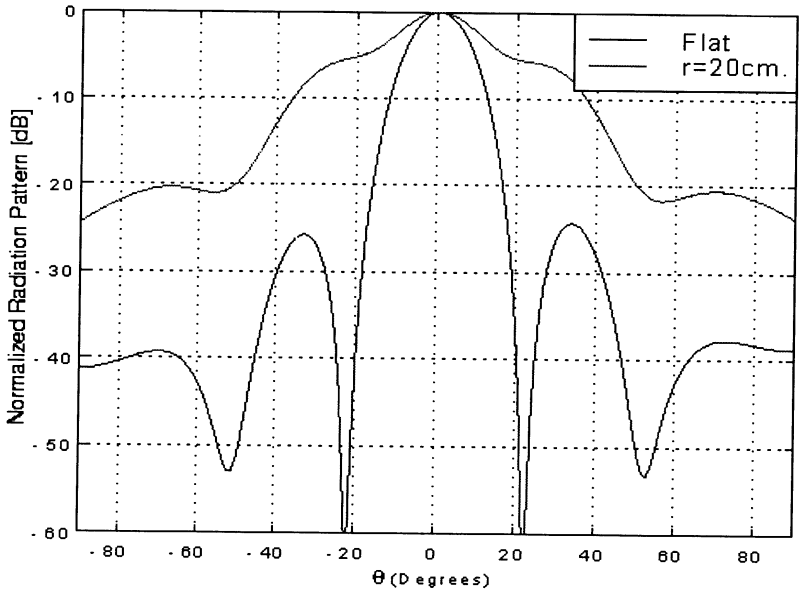


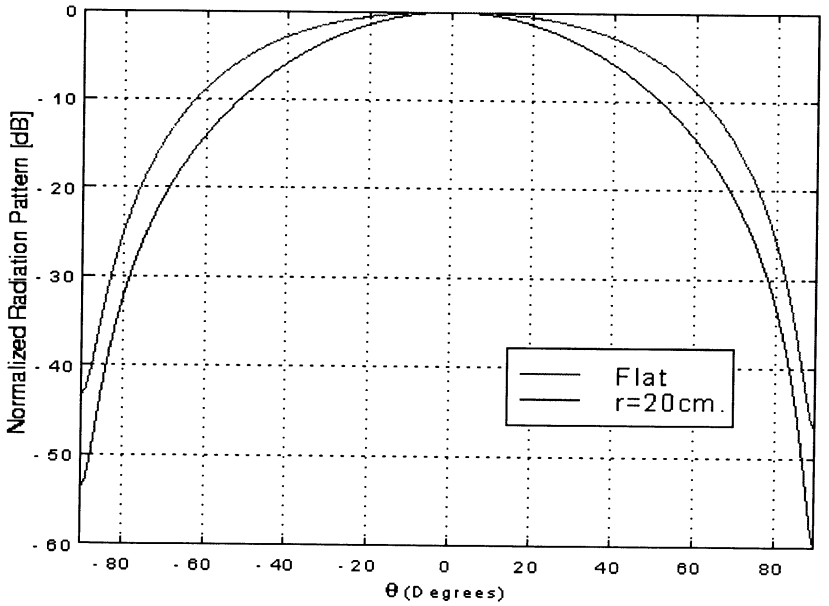
Element size : 3.5cm. by 2.625cm.
Seperation : 6.125cm.
Cavity : 33.25cm. by 6.125cm.
Total Number of Unknowns:
3287
Number of Boundary Unknowns:
2312
Matrix Fill Time : 4 min.
Iterative Solution time : 12 min.
on 200 MHz. IP22 processor.

Curved Antenna Array

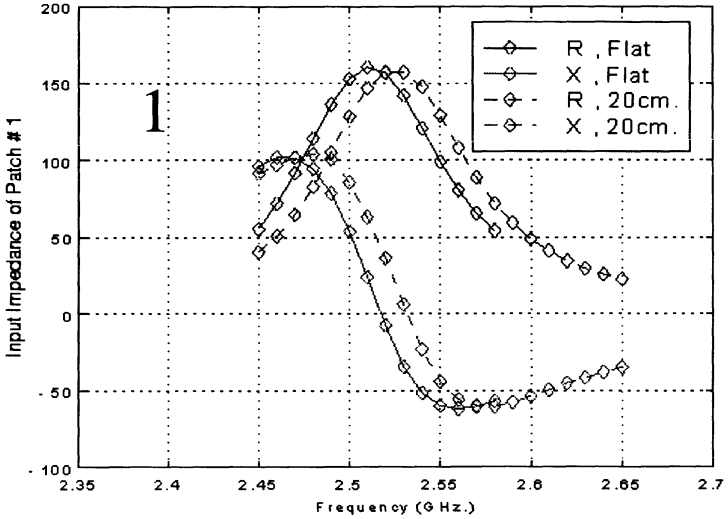


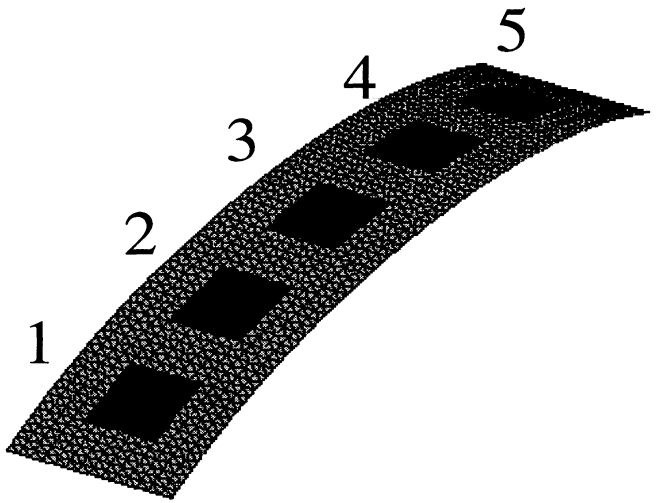


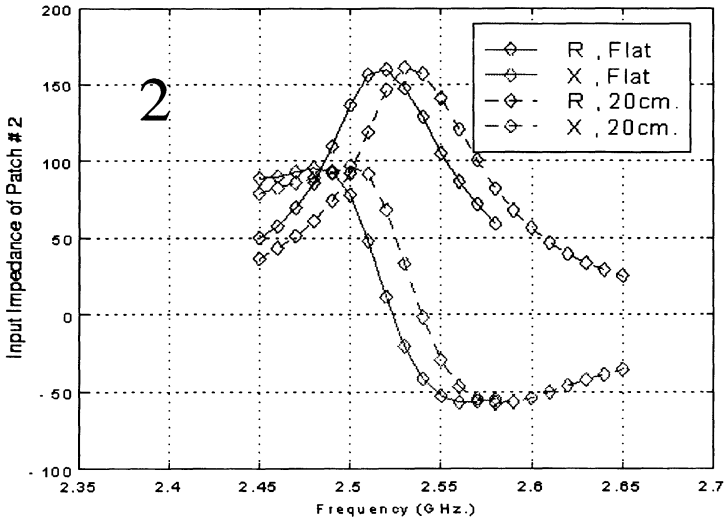


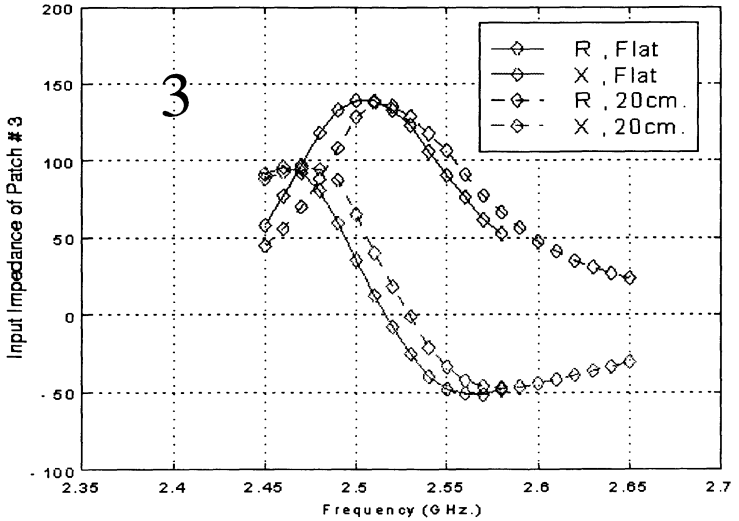


5 Element Antenna Array

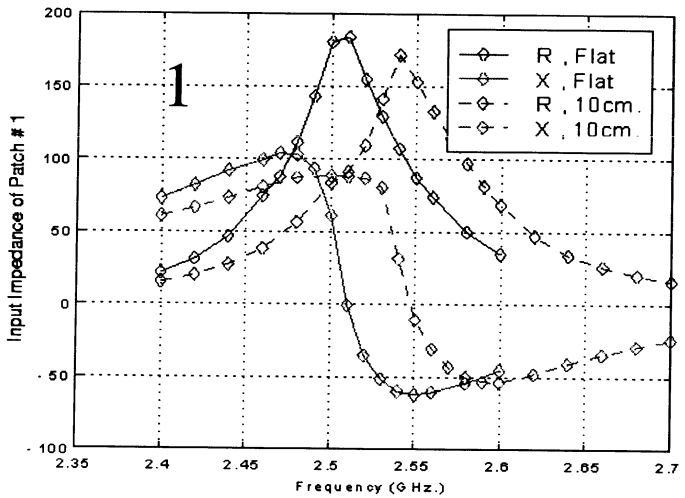




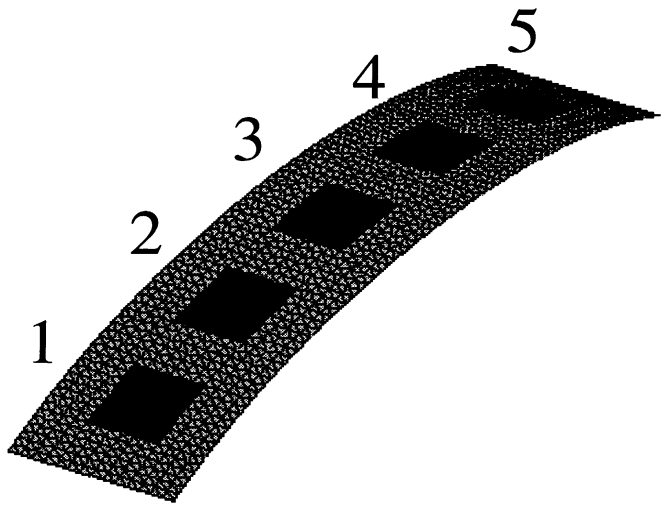


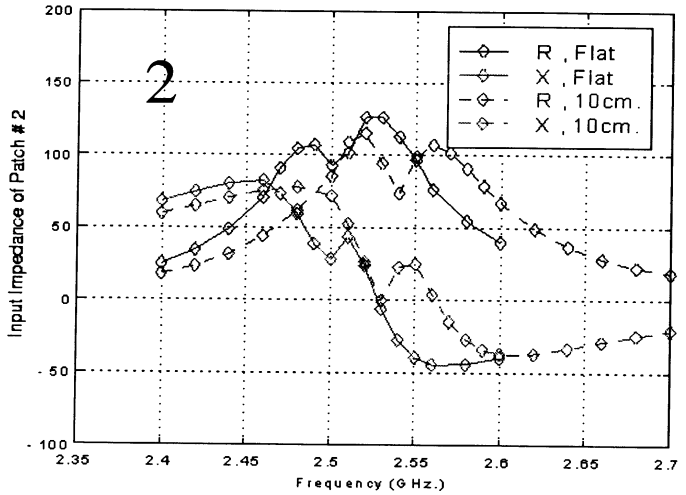


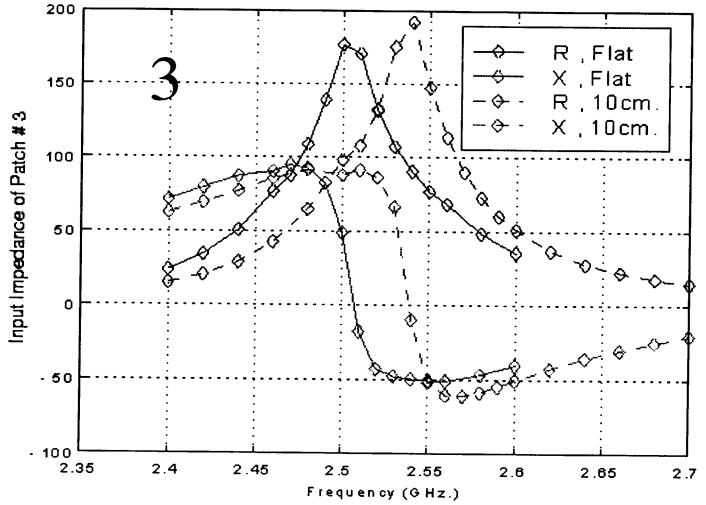
5 Element Antenna Array



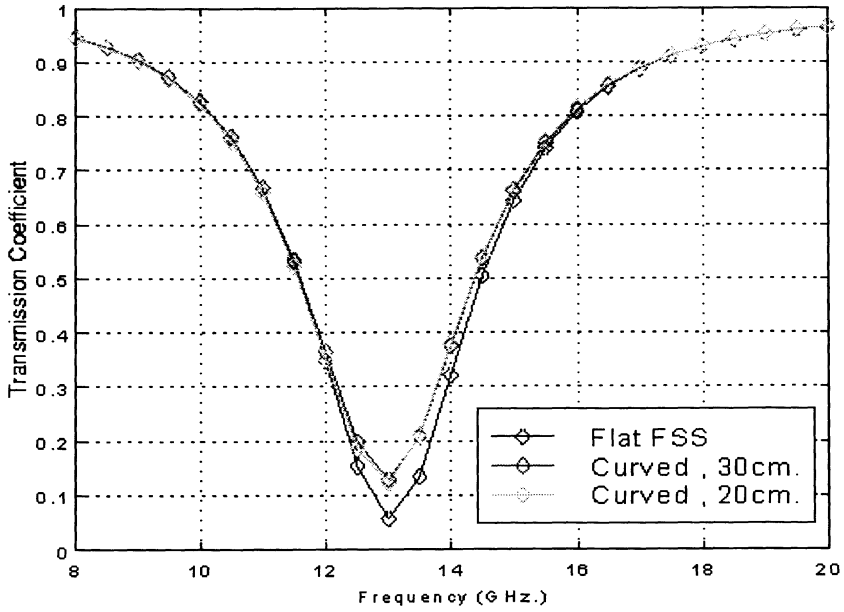
Elements brought closer cause coupling !

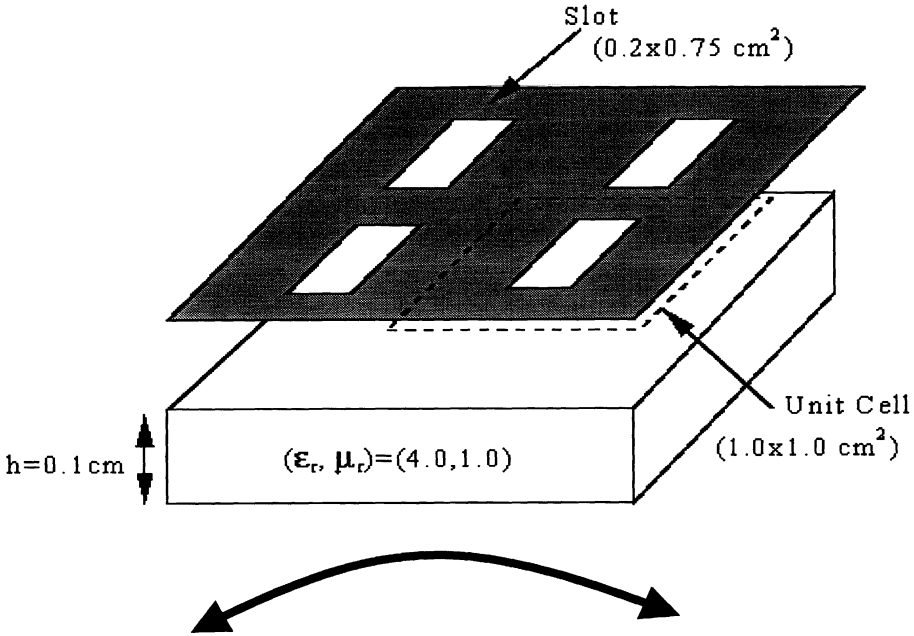






Curved FSS Structure





Curvature direction

Conclusions

- Presented are approximate solutions of antenna problems involving curved arrays.
- FMM is applied to reduce the solution time and memory requirements.
- For platforms with large radius of curvature, approximate solution predicts the effects of curvature.
- Finite arrays are shown to exhibit coupling not present in infinite arrays.
- Method provides a fast but approximate solution of antenna structures in inhomogeneous cavities, possibly curved.

035067-11-T

USERS MANUAL FOR FSS-CURVE

Sanders, A Lockheed Martin Co.
95 Canal Street NCA1-6268
P.O. Box 868
Nashua, NH 030601-0868

July 1998

PROJECT INFORMATION

PROJECT TITLE: Hybrid Finite Element Design Codes for the SERAT Array

REPORT TITLE: Users Manual for FSS-CURVE

U-M REPORT No.: 035067-11-T

CONTRACT
START DATE: October 1996
END DATE: September 1998

DATE: July 28, 1998

SPONSOR: Roland Gilbert
SANDERS, INC, A Lockheed Martin Co.
MER 24-1583
PO Box 868
Nashua, NH 030601-0868
Phone: (603) 885-5861
Email: RGILBERT@mailgw.sanders.lockheed.com

SPONSOR
CONTRACT No.: P.O. QP2047

U-M PRINCIPAL
INVESTIGATOR: John L. Volakis
EECS Dept.
University of Michigan
1301 Beal Ave
Ann Arbor, MI 48109-2122
Phone: (313) 764-0500 FAX: (313) 747-2106
volakis@umich.edu
http://www-personal.engin.umich.edu/~volakis/

CONTRIBUTORS
TO THIS REPORT: K. Sertel, J. Volakis, Y. Erdemli , D. Giszczak

1. General Code Description

FSS_CURVE is a special case of the earlier FSS_PRISM code discussed in the Univ of Michigan Rad Lab report 035067-7-T (July 1998 is the latest revision). FSS_PRISM is used to analyze electromagnetic scattering and radiation characteristics of infinite periodic planar antenna arrays and frequency selective surface (FSS) configurations, as illustrated in Figure 1, with an arbitrary number of FSS layers (metallic patches or slots) and combinations of both. FSS_CURVE considers finite array structures only and in this regard, it is similar to FSS_BRICK discussed in the Univ of Michigan Rad Lab report 035067-5-T. FSS_CURVE, though, employs the same geometry Driver for entering the geometry as FSS_PRISM, with only minor changes. Therefore, this manual repeats much of the same material found in the FSS_PRISM manual (Rad Lab report 035067-7-T). As such FSS_CUREVE is capable of generating the same type of meshes as the FSS_PRISM. The only difference is that FSS_CURVE repeats the meshes of each "periodic" cell of the finite array to generate the entire finite array mesh.

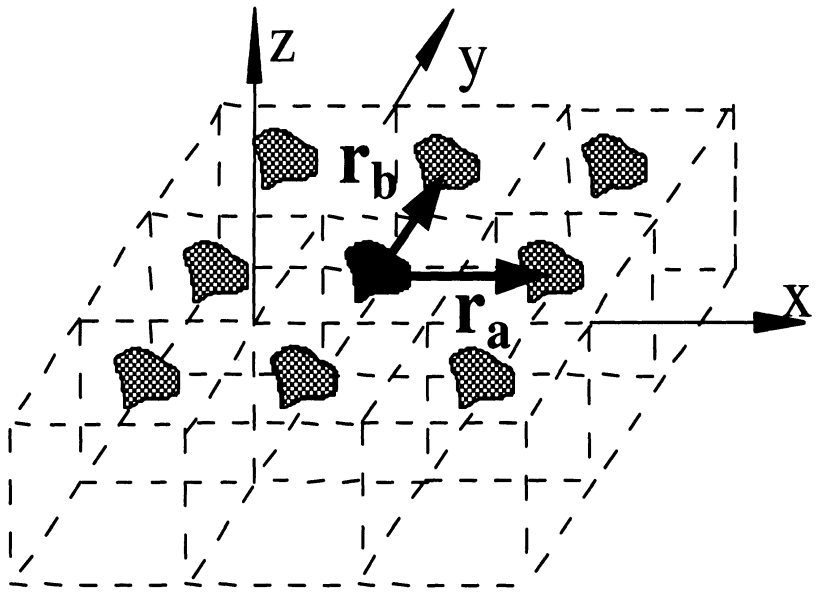


Figure 1: Periodic array configuration.

As in FSS_PRISM, the hybrid finite element-boundary integral (FE-BI) method is again used for field calculation. The finite element formulation is employed within the volumetric part and the boundary integral is used for terminating the mesh. Also, the

code works with prismatic elements in the FE-volume and triangular elements in the BI-surface. The prismatic elements can be right-angled as well as distorted. Distorted prism are essential for describing the geometry of curved finite arrays. First, the code generates triangular surface meshes with all geometrical adaptability for the individual layers while the volumetric FE-mesh is grown along the depth of the volume. The curved configuration mesh is generated by using the curvature information. The new mesh consists of distorted triangular prisms whose level of distortion depends on the curvature of the array. Currently the code has been verified only for cylindrical finite arrays. Like FSS_PRISM, FSS_CURVE has the option to deal with metal backed periodic configurations and with periodic configurations which are open at the top as well as at the bottom surface of the FE-mesh. In the first case, the BI is applied only on the top surface whereas in the latter, the BI-method is used to terminate both surfaces.

For the FSS_CURVE, the number of unknowns in the FE volume and on the BI surfaces are much larger than those in FSS_PRISM which models only a single unit cell. Basically, the degrees of freedom(DOFs) in FSS_CURVE are equal to N_{cells}*N_{FSS_PRISM}, where N_{cells} refers to the number of array elements in the finite array and N_{FSS_PRISM} refers to the DOFs/unknowns used in FSS_PRISM. Consequently, the CPU requirements of FSS_CURVE can be much larger. To alleviate these requirements and allow practical array calculations, the fast multipole method (FMM) is employed to speed-up the iterative solver in FSS_CURVE. Specifically, FMM is used to carry out the matrix-vector products for the BI on the top and bottom surfaces of the array aperture. This reduces the CPU and memory requirements from $O(N^2)$ down to $O(N^{1.5})$ as N increases.

For further details of the analysis and the theory of the formulation the user is referred to [1-10].

FSS_CURVE is written in Fortran 77 and has been verified to run on Unix platforms for HP, Sun, SGI workstations. In this manual, we describe input/output (I/O) operations which are nearly identical to FSS_PRISM. Also, a computational example is presented to demonstrate usage and performance of the code. More of these example will be given later.

2. Features of the FSS_PRISM/FSS_CURVE Driver

This section is primarily borrowed from the FSS_PRISM manual since the geometry Driver used by FSS_CURVE is the same as FSS_PRISM except for minor parameter addition (due to curvature). These are explicitly highlighted in the text.

The geometry Driver is the FORTRAN source file Driver.f. In addition, there is a parameter file "fema.dm1" which is used by both parts of the code and defines the array dimensions for compilation. The driver is an I/O tool that enables users to enter the necessary geometry and excitation information for antenna and/or FSS analysis. Using the geometrical input data, the driver generates the antenna mesh consisting of prismatic elements without a need for an external meshing package. The surface mesh for each FSS layer interface (2-D cross-section), and the whole geometry in 3-D can be viewed using MatLab. Also, the Univ of Michigan viewing tools FSSEfield.exe and FSSCur.exe can be used for this purpose on PCs. Thus, the user can visualize and check the geometry before running the analysis code. The PRISM driver requires the physical dimensions of the geometry in centimeter, angles in degrees, and frequency in GHz. The driver stores the necessary mesh and excitation information in two data files to be used by the analysis code. After executing the driver code, the FSS_CURVE code is run to compute input impedances of radiating elements and/or reflection and transmission coefficients of the corresponding structure. Of course, the input data files for the actual code can also be generated without the delivered driver (using external or specialized gridding packages). This allows the code to be used for more general structures than those supported by the driver.

The driver has the following I/O features:

1) All input data provided by the user is stored in a file with a pre-specified name so that it is convenient to rerun the code using the input file with the same data or modifications without reentering the entire input data set.

2) The user can choose either transmission/reflection coefficient evaluation of FSS structures or radiation computations for antenna configurations on top of multilayered FSS structures. After running the analysis code, for FSS analysis, the transmission and reflection coefficients of the structure for a given scan angle and frequencies are displayed on the screen and stored in corresponding output files

("Reflect", "Trans"). Similarly, plane wave amplitudes in scan direction and input impedances (in Ohm) (file "Imp") are displayed and stored for antenna analysis.

3) The FEM sample size and dimensions for each unit cell must be specified in centimeter. The driver also asks for the thickness or depth of the structure, including the number of layers and their material properties. In addition, beginning, end, and incremental frequencies (in GHz) and angles (in degrees) are specified by the user.

4) FSS_CURVE employs the fast multipole method (FMM) to speed up the BI computation, especially for large problems. Therefore, the driver asks for FMM parameters

5) Strip dipoles, slots, crossed-strip, and crossed-slot antenna or FSS elements are available in the driver as primitives. These elements can be placed at any location within the unit cell. The physical dimensions and the lower-left corner location of elements must be specified for each layer interface. It is assumed that the lower-left corner of each layer interface is the origin (0.,0.). Moreover, the user can generate other types of elements by specifying the size and location of rectangular metallic patches that would form the specific FSS element.

6) The driver allows horizontal (x- or y-directed) as well as vertical (z-directed) probe feeds at any node location within the structure where the user can specify locations, amplitudes, and phases of the probe currents.

7) Vertical short-circuit pins can be placed in all layers by specifying their (x,y) coordinates.

8) Vertical (z-directed) lumped impedance loads can be situated in each layer; their admittance values (in 1/Ohm) can be specified.

9) Horizontally placed lumped impedance loads are also available in the driver. The user can place x- or y- directed loads on each FSS layer as well as at the aperture where antenna elements may reside. The admittance values and locations of the loads are entered by the user.

10) As another option, resistive cards can be selected. The cards can be in all FSS layers and in the aperture. Surface conductance (1/Ohm), location, and size of the cards can be specified.

3. Running the Code

Before compiling the driver source file "Driver.f", the dimension parameters "NdmTri" (maximum number of triangles), "NdmSEd" (maximum number of edges), "NdmLay" (number of prism layers), and "NdmSNo" (maximum number of nodes expected in the final mesh) in the file called "fema.dm1" should be adjusted properly. Next the driver code is compiled by typing

f77 Driver.f -o *executable_filename1*

on UNIX platforms. To run the code, just type *executable_filename1* and follow the instructions for standard I/O operations. If an input file for the driver is already available, the user only needs to enter the name of that file; otherwise, the user is asked to enter the necessary input information one by one on the screen. All information is also stored in a user-specified input file allowing simplified reruns. After executing the driver, two data files "Data1" and "Data2" are generated to be used by the analysis code.

The main code can be compiled using the delivered "Makefile" for UNIX systems. Typing "make" at the prompt starts the compiler for the individual source files and finally links them together to an executable called FSS_C (see Makefile_sgi and Makefile_sun for Makefiles referring to Sun or SGI machines). For optimal use of the "Makefile" it might be necessary to adopt compiler and linker options, for instance with respect to code optimization, to the used system. On PC based systems, typically a project is defined including all the necessary source files. Before the code is compiled the array dimensions defined in the parameter file "fema.dm1" must be set sufficiently large. The meaning of the different parameters is explained in the file "fema.dm1" and also later in this manual. To make sure that no array dimension is exceeded, FORTRAN compilers usually have a compiler option enabling array range checking. Also, most of the array dimensions are checked during run time and the code will terminate with an error message.

After executing the prism code, the output data is displayed on the screen and also stored in the output files "Imp", "Trans", and "Reflect", whose formats will be described later in this manual. Sometimes it might be useful to redirect the output from the screen into an ASCII file by typing "> out.txt" at the end of the command (on Unix), viz.

FSS_C... > out.txt

This causes the screen output to be written in the file "out.txt".

4. I/O Operations

After starting the driver code, the user must specify the input data format. If an input file is available, the user is only asked to enter the name of that file. Otherwise, the user must enter all input data one by one on the screen. However, in this case all input data is stored in a file specified by the user. The driver finally generates two data files, namely "Data1" and "Data2" to be used as input files for the analysis code. The data format of these files is described later in this section.

In case of entering the input data on the screen, the user will be asked specifically for all necessary information where the questions are self-explanatory. However, if the user wants to generate an input file by himself rather than entering each input on the screen, it is necessary to know the input file format of the geometry driver. For this purpose, all input variables appearing in the input file are described in the next subsection.

4.1 Input File Format

Each line in the input file includes at least one entry and the entries must be separated by either a blank or a comma. It is important to pay attention to the type of the individual variables, whether it is a real, integer or complex variable, and specify it accordingly; otherwise, the driver may give error message and terminate. All entries specified in an input file are described below and illustrated by an example. Note that the variables are designated by its representative names.

1) Problem type and configuration: **`ITYP,IBOT,ICOM`**

`ITYP` (integer): Problem = 1 → Radiation *or* = 2 → Scattering

`IBOT` (integer): Bottom aperture = 1 → Open *or* = 2 → Closed (metal-backed)

`ICOM` (integer): = 1 → Commensurate *or* = 2 → Non-commensurate or = 3 → Curved

Example: `1,2,2` [Radiation problem; Non-commensurate configuration with metal-backed bottom surface]

2) Polarization type of incident plane wave and scan direction: **IPOL,PHI,THETA**

IPOL (integer): Polarization = 0 → No plane wave excitation, Radiation (ITYP=1)
Polarization = 1→ TE-pol *or* = 2 → TM-pol (Scattering: ITYP=2)
PHI, THETA (real): Array scan angles (degree)

Example: 1,0.,90. [TE-polarization; ϕ =0°, θ =90°]

3) Frequency of operation: **FRBEG,FREND,FRSTP**

FRBEG, FREND, FRSTP (real) : Beginning, end, and incremental frequencies (GHz)

Example: 1.,10.,0.25 [f_1=1 GHz, f_2=10 GHz, δf =0.25 GHz]

4) AIM computation parameters: **NSB1,NFBX,NFBY**

This is Not used in FSS-CURVE.

NSB1 (integer): Number of samples per dimension in the regular grid (Commensurate: ICOM=1) *or* in the bottom regular grid (Non-commensurate: ICOM=2)
NFBX, NFBY (integer): Nearfield threshold in the regular *or* in the bottom regular grid for the *x*- and *y*-directions

Only for non-commensurate case, additionally: **NST1,NFTX,NFTY**

NST1 (integer): Number of samples per dimension in the top regular grid

NFTX, NFTY (integer): Nearfield threshold in the top regular grid for *x*- and *y*-directions

Example: 50,18,18 [Commensurate case: NSB1=50, NFBX=NFBY=18]

The regular AIM grid overlays the possibly irregular triangular surface mesh. In the commensurate case, the surface mesh is equal throughout the whole structure and therefore the BI matrices on the top and bottom surfaces must be calculated only once. Similarly, only one regular AIM grid is needed. In the non-commensurate case, the surface meshes on the top and bottom BI terminations can have different extent and the employed Green function grid can be different as well. Therefore, different regular AIM grids can be defined on the top and bottom surfaces. Good AIM performance can typically be achieved by three to four regular grid points per triangle side length in the original mesh. In this case, the nearfield thresholds in *x*- and *y*-directions should not be smaller than 15 (typically 18). For instance, if the extent of the original mesh in *x*- and *y*-directions is 10 cm and the average side length of the triangles is 0.5 cm, a total number of 60 regular grid points per dimension would lead to three regular grid points per

triangle side length. In the current implementation of the code, a 2-D FFT algorithm is used which requires the same number of elements in both dimensions. Therefore, the number of samples per dimension equally applies to *x*- and *y*-directions.

5) Fss unit cell size and FEM sample size: **XL,YL,DX,DY**

XL, YL (real): FSS unit cell size (ICOM=1) *or* Largest unit cell size (ICOM=2) (cm)

DX, DY (real): FEM sample size (cm)

Example: 2.,2.,0.05,0.05 [FSS unit cell size: 2x2 cm^2, FEM sample size: 0.05x0.05 cm^2]

6) Depth of structure and number of layers: **ZL,NZ**

ZL (real): Depth of structure (cm)

NZ (integer): Number of layers along the depth

Example: 1.,3 [3-layered structure with 1 cm in depth]

7) FSS unit cell size: **FXL,FYL**

Only if ICOM=2 (Non-commensurate case), for each layer from the bottom to the top:

FXL, FYL (real): *x-y* dimensions of FSS unit cell in cm (each layer's top surface)

Example: 2.,2. [FSS unit cell size for Layer # 1 (bottom layer): 2x2 cm^2]
1.5,1.25 [FSS unit cell size for Layer # 2: 1.5x1.25 cm^2]
1.25,1. [FSS unit cell size for Layer # 3 (top layer): 1.25x1 cm^2]

8) Layer parameters: **DT,EPS,MU**

For each layer from the bottom to the top:

DT (real): Layer thickness (cm)

EPS, MU (complex): Relative permittivity and permeability

Example: 0.5,(1.,0.),(1.,0.) [t_1=0.50 cm, $\varepsilon_{r1}=\mu_{r1}$=1.+j0.]
0.25,(2.,0.),(1.,0.) [t_2=0.25 cm, ε_{r2}=2.+j0., μ_{r2}=1.+j0.]
0.25,(1.,0.),(1.,-0.05) [t_3=0.25 cm, ε_{r3}=1.+j0., μ_{r3}=1.-j0.05]

9) FSS elements:

For each layer from the bottom to the top:

Flag for FSS elements: **IYNFS**

IYNFS (integer): = 1 → FSS elements on the top of layer *or* = 2 → No FSS elements

If *only* IYNFS=1:

Type of FSS elements and number of sections: **NFSEL,NMET**

NFSEL (integer): FSS element type = 1 → Strip *or* = 2 → Slot *or* = 3 → Crossed-strip
or = 4 → Crossed-slot *or* = 5 → User-specified element

NMET (integer): Number of sections = 1 (NFSEL=1,2) *or* = 2 (NFSEL=3,4)
or = user-specified (NFSEL=5)

Size of each section: **DPX,DPY**

DPX, DPY (real): *x*-*y* dimensions of each section for each FSS element (cm)

Location of each section: **X1,Y1**

X1, Y1 (real): Lower-left corner coordinate of each section in cm (for each layer, the origin is assumed to be the lower-left corner of the corresponding unit cell)

Example:	1	[FSS elements exist on Layer # 1 (bottom layer)]
	1,1	[FSS element type: Strip with one section only]
	0.5,0.75	[Size of Strip element: 0.5x0.75 cm^2]
	0.25,0.35	[(*x*,*y*) coordinate of Strip element in cm]
	2	[There is no FSS element on Layer # 2]
	1	[FSS elements exist on Layer # 3 (top layer)]
	3,2	[FSS element type: Crossed-strip with two sections]
	0.5,0.5	[Size of 1^{st} section: 0.5x0.5 cm^2]
	0.25,0.35	[(*x*,*y*) coordinate of 1^{st} section in cm]
	0.75,0.75	[Size of 2^{nd} section: 0.75x0.75 cm^2]
	0.35,0.25	[(*x*,*y*) coordinate of 2^{nd} section in cm]

10) Resistive Cards (R-cards):

For each layer from the bottom to the top:

Number of R-cards: **NRES**

NRES (integer): Number of R-cards placed on each layer's top surface

Size of each R-card: **RCX,RCY**

RCX, RCY (real): *x*-*y* dimensions of each R-card in cm

Location of each R-card: **RX1,RY1**

RX1, RY1 (real): Lower-left corner coordinate of each R-card in cm (for each layer, the origin is assumed to be the lower-left corner of the corresponding unit cell)

Surface Conductance of each R-card: **SCN**

SCN (complex): Surface conductance of the R-card (1/Ohm)

Example:

```
0                 [There is no R-card on Layer # 1 (bottom layer)]

0                 [There is no R-card on Layer # 2]

2                 [There are two R-cards on Layer # 3 (top layer)]
0.5,0.5           [Size of R-card # 1: 0.5x0.5 cm^2]
0.25,0.25         [(x,y) coordinate of R-card # 1 in cm]
(100.,0.)         [Conductance value of R-card # 1: 100+j0 (1/Ohm)]
0.25,0.25         [Size of R-card # 1: 0.25x0.25 cm^2]
0.25,0.25         [(x,y) coordinate of R-card # 2 in cm]
(10.,-0.5)        [Conductance value of R-card # 2: 10-j0.5 (1/Ohm)]
```

11) Lumped Impedance Loads:

x-directed loads (*x*-loads):

For each layer from the bottom to the top:

Number of *x*-loads: **NXD**

NXD (integer) : Number of *x*-loads located on each layer's top surface

Location of each *x*-load: **X1,Y1,X2,Y2**

X1, Y1 (real) : Beginning coordinate of each *x*-load in cm

X2, Y2 (real) : End coordinate of each *x*-load in cm

Note that an *x*-load is located between its beginning and end coordinates, and also Y1 should be equal to Y2.

Admittance of each *x*-load: **AXLD**

AXLD (complex): Admittance value of each *x*-load (1/Ohm)

Example:

```
0                 [There is no x-load on Layer # 1]

1                 [There is only one x-load on Layer # 2]
0.1,0.1,0.3,0.1   [(x1,y1) = (0.1,0.1) and (x2,y2) = (0.3,0.1) cm]
(150.,0.)         [Admittance of the x-load: 150+j0 (1/Ohm)]

0                 [There is no x-load on Layer # 3]
```

y-directed loads (*y*-loads) :

For each layer from the bottom to the top:

Number of *y*-loads: **NYD**

NYD (integer) : Number of *y*-loads located on each layer's top surface

Location of each *y*-load: **X1,Y1,X2,Y2**

X1, Y1 (real) : Beginning coordinate of each *y*-load in cm

X2, Y2 (real) : End coordinate of each *y*-load in cm

Note that a *y*-load is located between its beginning and end coordinates, and also X1 should be equal to X2 .

Admittance of each *y*-load: **AYLD**

AYLD (complex) : Admittance value of each *y*-load (1/Ohm)

Example:

```
0                       [There is no y-load on Layer # 1]

1                       [There is only one y-load on Layer # 2]
0.1,0.3,0.1,0.3         [(x1,y1) = (0.1,0.1) and (x2,y2) = (0.3,0.1) cm]
(10.,10.)               [Admittance of the y-load: 10+j10 (1/Ohm)]

0                       [There is no y-load on Layer # 3]
```

z-directed loads (*z*-loads) :

For each layer from the bottom to the top:

Number of *z*-loads: **NZD**

NZD (integer) : Number of *z*-loads located in each layer

Location of each *z*-load: **X1, Y1**

X1, Y1 (real) : (*x*,*y*) coordinate of each *z*-load within the unit cell (cm)

Admittance of each *z*-load: **AZLD**

AZLD (complex) : Admittance value of each *z*-load (1/Ohm)

Example:

```
1                       [There is only one z-load in Layer # 1]
0.15,0.3                [Location of the load: (x,y) = (0.15,0.3) cm]
(10.,1.5)               [Admittance of the z-load: 10+j1.5 (1/Ohm)]

1                       [There is only one z-load in Layer # 2]
0.15,0.3                [Location of the load: (x,y) = (0.15,0.3) cm]
(10.,1.5)               [Admittance of the z-load: 10+j1.5 (1/Ohm)]

1                       [There is only one z-load in Layer # 3]
0.15,0.3                [Location of the load: (x,y) = (0.15,0.3) cm]
(10.,1.5)               [Admittance of the z-load: 10+j1.5 (1/Ohm)]
```

Note that a cascade of three vertical loads located at (*x*,*y*) = (0.15,0.3) are formed along the depth of the structure in the above example.

12) Short-Circuit Pins:

For each layer from the bottom to the top:

Number of pins: **NPIN**

NPIN (integer) : Number of short-circuit pins inserted in each layer

Location of each pin: **X1,Y1**

X1 , Y1 (real) : (x,y) coordinate of each pin within the unit cell (cm)

Example:

```
1                [There is only one pin in Layer # 1 (bottom layer)]
0.3,0.3          [Location of the load: (x,y) = (0.3,0.3) cm]

1                [There is only one pin in Layer # 2]
0.3,0.3          [Location of the pin: (x,y) = (0.3,0.3) cm]

1                [There is only one pin in Layer # 3 (top layer)]
0.3,0.3          [Location of the pin: (x,y) = (0.3,0.3) cm]
```

The top and bottom surfaces of the structure can be short-circuited by placing pins at the same (x,y) coordinate in each layer as done in the above example.

13) Probe-Current Feeds:

x-directed probe-current feeds (x-feeds) :

For each layer from the bottom to the top:

Number of x-feeds: **NXFED**

NXFED (integer) : Number of x-feeds located on each layer's top surface

Location of each x-feed: **X1,Y1,X2,Y2**

X1 , Y1 (real) : Beginning coordinate of each x-feed in cm

X2 , Y2 (real) : End coordinate of each x-feed in cm

Note that an x-feed is located between its beginning and end coordinates, and also Y1 should be equal to Y2 .

Amplitude and phase of each x-feed: **XAMP,XPHS**

XAMP (real) : Amplitude of each x-feed (Amp)

XPHS (real) : Phase of each x-feed (degree)

Example:

```
0                [There is no x-feed on Layer # 1]

0                [There is no x-feed on Layer # 2]

1                [There is only one x-feed on Layer # 3]
0.2,0.1,0.5,0.1  [(x1,y1) = (0.2,0.1) and (x2,y2) = (0.5,0.1) cm]
```

`1.,45.` [Probe-current: $I_{x\text{-feed}}=1.\ e^{j\pi/4}$ Amp]

y-directed probe-current feeds (y-feeds) :

For each layer from the bottom to the top:

Number of y-feeds: **NYFED**

`NYFED` (integer) : Number of *y*-feeds located on each layer's top surface

Location of each y-feed: **X1,Y1,X2,Y2**

`X1,Y1` (real) : Beginning coordinate of each *y*-feed in cm

`X2,Y2` (real) : End coordinate of each *y*-feed in cm

Note that an *y*-feed is located between its beginning and end coordinates, and also `X1` should be equal to `X2`.

Amplitude and phase of each y-feed: **YAMP,YPHS**

`YAMP` (real) : Amplitude of each *y*-feed (Amp)

`YPHS` (real) : Phase of each *y*-feed (degree)

Example:

`0`	[There is no *y*-feed on Layer # 1]
`0`	[There is no *y*-feed on Layer # 2]
`1`	[There is only one *y*-feed on Layer # 3]
`0.3,0.1,0.3,0.4`	[(x_1,y_1) = (0.3,0.1) and (x_2,y_2) = (0.3,0.4) cm]
`1.,0.`	[Probe-current: $I_{y\text{-feed}}=1.\ e^{j0.}$ Amp]

z-directed probe-current feeds (z-feeds) :

For each layer from the bottom to the top:

Number of z-feeds: **NZFED**

`NZFED` (integer) : Number of *z*-feeds inserted in each layer

Location of each z-feed: **X1,Y1**

`X1,Y1` (real) : (x,y) coordinate of each *z*-load within the unit cell (cm)

Amplitude and phase of each z-feed: **ZAMP,ZPHS**

`ZAMP` (real) : Amplitude of each *y*-feed (Amp)

`ZPHS` (real) : Phase of each *y*-feed (degree)

Example:

`1`	[There is only one *z*-feed in Layer # 1 (bottom)]
`0.2,0.2`	[Location of the *z*-feed: (x,y) = (0.2,0.2) cm]
`1.,90.`	[Probe-current: $I_{z\text{-feed}}=1.\ e^{j\pi/2}$ Amp]
`1`	[There is only one *z*-feed in Layer # 3]

`0.2,0.2`	[Location of the z-feed: (x,y) = (0.2,0.2) cm]
`1.,90.`	[Probe-current: $I_{z\text{-feed}}=1.\ e^{j\pi/2}$ Amp]
`1`	[There is only one z-feed in Layer # 3 (top)]
`0.2,0.2`	[Location of the z-feed: (x,y) = (0.2,0.2) cm]
`1.,90.`	[Probe-current: $I_{z\text{-feed}}=1.\ e^{j\pi/2}$ Amp]

As shown in the above example, one can construct a continuous current feed between the bottom and top surfaces of the structure by placing probe-currents with same amplitude and phase value at the same (x,y) coordinate in each layer.

14) Curvature:

THI IS SPECIFIC TO FSS_CURVE

Radius of curvature in x-direction : **CURADX**

CURADX (real) : Radius of curvature in x-direction, i.e. the mesh is wrapped on a cylindrical surface such that the y=constant edges have cylindrical curvature.

4.2 Format of Data Files

In this subsection we describe the formats of the input data files "Data1" and "Data2", the output files "Imp", "Trans", "Reflect", "EqvCur", "EqvCurB", and "EdegUnk", as well as the parameter file "fema.dm1". "Data1" and "Data2" are generated by the driver, but can also be provided from another source. These files are the input files for the analysis code and contain all necessary data for the code to be run. "Data1" contains excitation and global geometry information and "Data2" contains surface mesh data (node and triangular element numbering), definitions for periodic boundary conditions, and locations of discrete elements (metallic sections, R-cards, loads, pins, feeds) in terms of node and triangle numbers. "Imp", "Trans", and "Reflect" are generated by the prism code and contain computed input impedances related to probe feeds, transmission and reflection coefficients, respectively, with respect to frequency for the specified scan direction. The files "EqvCur", "EqvCurB", and "EdgeUnk" are generated for the frequency specified by the parameter NFREQ in the file "Data1" and

contain equivalent surface current densities and the values of the edge unknowns, respectively.

A. DATA1

1) Problem type and configuration: **ITYP,IBOT,ICOM**

ITYP (integer): Problem = 1 → Radiation *or* = 2 → Scattering

IBOT (integer): Bottom aperture = 1 → Open *or* = 2 → Closed (metal-backed)

ICOM (integer): = 1 → Commensurate *or* = 2 → Non-commensurate *or* =3 Curved

2) Polarization type of incident plane wave and scan direction: **IPOL,PHI,THETA**

IPOL (integer): Polarization = 0 → No plane wave excitation, Radiation (ITYP=1)
Polarization = 1→ TE-pol *or* = 2 → TM-pol (Scattering: ITYP=2)

PHI, THETA (real): Array scan angles (degree)

3) Frequency of operation: **FRBEG,FREND,FRSTP**

FRBEG, FREND, FRSTP (real): Beginning, end and incremental frequencies (GHz)

4) Miscellaneous parameters:

i) **NFREQ,RACC,MCON,IDIS**

NFREQ (integer): Number of frequency for which the files "EqvCur", "EqvCurB", and "EdgeUnk" are saved ; NFREQ=1 (default)

RACC (real): Relative accuracy for the solver; RACC=0.01 (default)

In most cases, a relative accuracy RACC=0.01 gives good results, sometimes smaller values might be necessary.

MCON (integer): Monitor convergence; MCON=0 (Default: Monitoring → OFF),
MCON=1 (Monitoring → ON)

IDIS (integer): Type of prisms used for the mesh; IDIS=1 → Distorted (default for FSS-CURVE)
IDIS=2 → Right-angled (default for FSS-PRISM)

ii) **NPRE,DEF,IDF1,IDF2**

NPRE (integer): Flag for preconditioning; NPRE=0 → No preconditioning
NPRE=1 → Diagonal precond. (default)
NPRE=2 → AIPC (not recommended)

For non-perpendicular scan direction no preconditioning usually gives better results than diagonal preconditioning. AIPC is not recommended, especially not in conjunction with AIM. **Preconditioning is not used in FSS-CURVE.**

DEF (real): control parameter for AIPC

IDF1 (integer): flag for solver selection; IDEF1=**0** → BiCG solver (default)

IDEF1=1 → GMRES solver

IDF2 (integer): Number of search vectors per GMRES restart (for BiCG, value is ignored but must be present)

The default solver for FSS-CURVE is CGS (Conjugate Gradient Squared).

AIM computation parameters:

NOTE that AIM parameters are not used in FSS-CURVE.

Commensurate: **IAM1,NSB1,NSB2,NFBX,NFBY**

IAM1 (integer) : Flag for AIM computation; IAM1=1 → AIM is active,
IAM1=0 → AIM is not active

NSB1, NSB2 (integer) : NSB1→ Number of samples in the regular grid; NSB2=2*NSB1

NFBX, NFBY (integer) : Nearfield threshold in the regular grid for the *x*- and *y*-directions

Non-commensurate: **IAM1,NSB1,NSB2,NFBX,NFBY,NST1,NST2,NFTX,NFTY**

IAM1 (integer) : Flag for AIM computation; IAM1=1 → AIM is active,
IAM1=0 → AIM is not active

NSB1, NSB2 (integer) : Number of samples in the bottom surface grid; NSB2=2*NSB1

NFBX, NFBY (integer) : Nearfield threshold in the top surface grid for both directions

NST1, NST2 (integer) : Number of samples in the top surface grid; NST2=2*NST1

NFTX, NFTY (integer) : Nearfield threshold in the top surface grid for both directions

6) FSS unit cell size and FEM sample size: **XL,YL,DX,DY**

XL, YL (real): FSS unit cell size (ICOM=1) *or* Largest unit cell size (ICOM=2) (cm)

DX, DY (real): FEM sample size (cm)

7) Depth of structure and number of layers: **ZL,NZ**

ZL (real): Depth or thickness of structure (cm)

NZ (integer): Number of layers along the depth

8) FSS unit cell size: **FXL,FYL,GAMMA**

Only if ICOM=2 (Non-commensurate case), for each layer from the bottom to the top:

FXL, FYL (real) : *x-/y*-dimensions of FSS unit cell in cm (each layer's top surface)

GAMMA: Unit cell angle between the two lattice vectors. One of the vectors is always aligned with the *x*-axis. If GAMMA is not 90^0 the height YL in *y*-direction is shorter than the length of the second lattice vector.

9) Layer parameters: **DT,EPS,MU**

For each layer from the bottom to the top:

DT (real): Layer thickness (cm)

EPS, MU (complex): Relative permittivity and permeability

B. DATA2

1) Surface mesh parameters:

i) Number of nodes and triangles: **NNOD,NTRI**

NNOD (integer): Number of nodes in the surface mesh

NTRI (integer): Number of triangular elements in the surface mesh

Note that the size of the surface mesh is one FEM sample larger (in both *x*- and *y*-directions) . That is, there is at least one additional row of triangular elements around the periphery of the aperture ;

ii) Triangle and node numbering: **ITR,NODE1,NODE2,NODE3**

ITR (integer): Triangle number (1 → NTRI)

NODE1, NODE2, NODE3 (integer): Node numbers for triangular element ITR
Note, all triangles must be numbered counterclockwise.

iii) Node number and (*x,y,z*) coordinate: **IN,XSNOD,YSNOD,ZSNOD**

IN (integer): Node number (1 → NNOD)

XSNOD, YSNOD, ZSNOD (real) : (*x,y,z*) coordinates of node IN; ZSNOD=0. (default)

2) Interface information:

Number of layer interfaces: **NZI**

NZI (integer): Number of layer interfaces along the depth of the structure;
NZI= NZ (Number of layers) + 1

Note that in the driver, surface elements (*x*- and *y*-directed impedance loads and feeds, resistive cards and metallic sections) can be placed on each layer's top surface, corresponding to interface # 2 → # NZI. However, in the "Data2" file also surface elements in the bottom surface of the lowest layer can be defined (interface # 1).

For each interface from the bottom to the top: **ZCO**

ZCO (real): *z*-coordinate of each interface in cm;
Top surface (interface number: NZI) → ZCO= 0.
Bottom surface (interface number: 1) → ZCO= -ZL (the depth)

3) Absorber triangles and periodic boundary condition (PBC) : **(NOT USED IN FSS_CURVE)**

For *commensurate* case:

i) Number of absorber triangles: **NABTRI**

NABTRI (integer): Number of absorber triangles (as defined above) around the unit cell

ii) Absorber triangle numbering: **IAT**

IAT (integer): triangle number

iii) Number of PBC nodes: **NXPER**

NXPER (integer): Number of PBC nodes (outer nodes surrounding the unit cell) in x-direction

iv) Bottom and top PBC node list: **IPBOT, IPTOP**

For all PBC nodes along the *x*-direction:

IPBOT (integer): Node number along the lower side of the unit cell

IPTOP (integer): Corresponding node number along the upper side of the unit cell

v) Number of PBC nodes: **NYPER**

NYPER (integer): Number of PBC nodes (outer nodes surrounding the unit cell) in y-direction

vi) Left and right PBC node list: **IPLFT, IPRGT**

For each PBC node along the *y*-direction:

IPLFT (integer): Node number along the left side of the unit cell (along *y*-axis)

IPRGT (integer): Corresponding node number along the right side of the unit cell

For *non-commensurate* case: Repeat input options above (i-ii) for all layer interfaces. Follow with (iii-iv) for all layers and (v-vi) for all layers.

(1 → NZI). In addition, for each interface, specify the following:

vii) Number of periodic edges: **ICP**

ICP (integer) : Number of edges in the outer region of the FSS unit cell

viii) Periodic edge list: **N1,N2,PN1,PN2**

N1, N2 (integer) : Nodes located in the outer region of the cell and representing an edge

PN1, PN2 (integer) : Nodes located in the inner region of the cell and representing the corresponding periodic edge

4) Metallic Triangles:

For each layer interface, repeat these steps as a pair:

Number of metallic triangles: **NOMT**

NOMT (integer): Number of metallic triangles in each interface's surface mesh

Triangle number: **INMT**

INMT (integer): metallic triangle number

5) Resistive Triangles:

For each layer interface, repeat these steps as a pair:

Number of resistive triangles: **NORT**

NORT (integer): Number of resistive triangles in each interface's surface mesh

Triangle number and Conductance: **INRES,SCN**

INRES (integer): resistive triangle number

SCN (complex): Surface conductance of resistive triangle in 1/Ohm

6) Lumped Impedance Loads:

x-directed loads (*x*-loads):

For each layer interface, repeat these steps as a triplet:

Number of *x*-loads: **NXD**

NXD (integer): Number of *x*-loads placed on each layer interface

Admittance: **AXLD**

AXLD (complex): Admittance value of each x-load in 1/Ohm

Node location: **LXB, LXE**

LXB, LXE (integer): Corresponding beginning and end node numbers in the surface mesh for the x-load

y-directed loads (y-loads):

For each layer interface, repeat these steps as a triplet:

Number of y-loads: **NYD**

NYD (integer): Number of y-loads placed on the layer interface

Admittance: **AYLD**

AYLD (complex): Admittance value of y-load in 1/Ohm

Node location: **LYB, LYE**

LYB, LYE (integer): Corresponding beginning and end node numbers in the surface mesh for the y-load

z-directed loads (z-loads):

For each layer, repeat these steps as a triplet:

Number of z-loads: **NZD**

NZD (integer): Number of z-loads embedded in the layer

Admittance: **AZLD**

AZLD (complex): Admittance value of the z-load in 1/Ohm

Node location: **LZC**

LZC (integer): Corresponding node number in the surface mesh for the z-load

7) Short-Circuit Pins:

For each layer, repeat these steps as a pair:

Number of pins: **NPIN**

NPIN (integer): Number of short-circuit pins inserted in the layer

Node location: **LPC**

LPC (integer): Corresponding node number in the surface mesh for the pin

8) Current-Probe Feeds:

Only for radiation problems:

x-directed feeds (*x*-feeds):

For each layer interface, repeat these steps as a triplet:

Number of *x*-feeds: **NXFED**

NXFED (integer): Number of *x*-feeds placed on the layer interface

Current: **XCUR**

XCUR (complex): Current of the probe feed in Amp

Node location: **FXB, FXE**

FXB, FXE (integer) : Corresponding beginning and end node numbers in the surface mesh for the *x*-feed

y-directed feeds (*y*-feeds):

For each layer interface, repeat these steps as a triplet:

Number of *y*-feeds: **NYFED**

NYFED (integer): Number of *y*-feeds placed on each layer interface

Current: **YCUR**

YCUR (complex): Current value of each *y*-feed in Amp

Node location: **FYB, FYE**

FYB, FYE (integer) : Corresponding beginning and end node numbers in the surface mesh for the *y*-feed

z-directed feeds (*z*-feeds):

For each layer, repeat these steps as a triplet:

Number of *z*-feeds: **NZFED**

NZFED (integer): Number of *z*-feeds embedded in each layer

Current: **ZCUR**

ZCUR (complex): Current value of each *z*-feed in Amp

Node location: **FZC**

FZC (integer): Corresponding node number in the surface mesh for each *z*-feed

C. Imp

The output file "Imp" contains the input impedances for the individual probe feeds. For all feeds it contains a line with the data:

```
f     FEEDNR     Z_REAL     Z_IMAG     ITER
```

f: Frequency

FEEDNR: Number of feed

Z_REAL: Real part of input impedance

Z_IMAG: Imaginary part of input impedance

ITER: Number of iterations

D. Reflect

The output file "Reflect" contains the reflection coefficients at the top BI surface for plane wave excitation and the radiated field amplitude in scan direction for radiation problems. For all frequencies the files contains a line with the data:

```
f     TM_AMP     TM_PHASE   TE_AMP     TE_PHASE
```

f: Frequency

TM_AMP: Amplitude of TM component

TM_PHASE: Phase (degree) of TM component

TE_AMP: Amplitude of TE component

TE_PHASE: Phase (degree) of TE component

E. Trans

The output file "Trans" contains the transmission coefficients at the bottom BI surface (if open) for plane wave excitation and the radiated field amplitude in scan direction for radiation problems. For all frequencies the files contains a line with the data:

```
f     TM_AMP     TM_PHASE   TE_AMP     TE_PHASE
```

f: Frequency

TM_AMP: Amplitude of TM component

TM_PHASE: Phase (degree) of TM component

TE_AMP: Amplitude of TE component

TE_PHASE: Phase (degree) of TE component

F. EqvCur/EqvCurB

The file "EqvCur" contains the tangential magnetic current components in the top surface of the mesh and the file "EqvCurB" the corresponding data in the bottom surface. Both files are only generated for one frequency specified by the parameter `NFREQ` in the file "Data1". The first line of the files contains two integer numbers:

```
Ntri        Wnum
```

`NTri:` Number of triangles in the surface

`Wnum:` Wavenumber

Then for each triangle a line follows with the data:

```
IT    Area    Xc    Yc    Zc    Mx    My    Mz
```

`IT`: Triangle number

`Area`: Area of triangle

`Xc`: x-coordinate of triangle center

`Yc`: y-coordinate of triangle center

`Zc`: z-coordinate of triangle center

`Mx`: x-component of magnetic current

`My`: y-component of magnetic current

`Mz`: z-component of magnetic current

G. EdgeUnk

The file "EdgeUnk" contains the values for the FE unknowns in the mesh. The file is only generated for one frequency specified by the parameter `NFREQ` in the file "Data1". For each edge in the volume mesh, a line is present with the data:

```
I    X1    Y1    Z1    X2    Y2    Z2    Amp    Real    Imag
```

`I`: Number of edge

X1: x-coordinate of first grid point

Y1: y-coordinate of first grid point

Z1: z-coordinate of first grid point

X2: x-coordinate of second grid point

Y2: y-coordinate of second grid point

Z2: z-coordinate of second grid point

Amp: Amplitude of the edge unknown

Real: Real part of the unknown

Imag: Imaginary part of the unknown

H. fema.dm1

"fema.dm1" is the parameter file for the FORTRAN codes. It defines parameters which determine the size of the data arrays in the codes. The file contains the same parameters for the driver as well as the analysis codes. However, in the driver not all parameters are used. The parameters must be larger than the largest possible size of the quantity for the given model. The list of these parameters with their descriptions are given in the following:

NdmSNo: Number of nodes in the surface mesh (used by the Driver)

NdmVNo: Number of nodes in the volume mesh (number of layers times surface nodes)

NdmPNo: Number of nodes on periodic boundary in a layer (per side) (not used by FSS-CURVE)

NdmTri: Number of triangles in the surface mesh (used by the Driver)

NdmPri: Number of prisms in the volume mesh (number of layers times triangles)

NdmSEd: Number of edges in the surface mesh

NdmNZS: Number of non-zero edges in the surface mesh

NdmVEd: Number of edges in the volume mesh

NdmNZE: Number of non-zero edges in the volume mesh

NdmPEd: Number of edges on periodic boundary in a layer (per side) (not used by FSS-CURVE)

NdmLay: Number of prism layers in the volume mesh (used by the Driver)

NdmREd: Number of x- and y-directed resistive edges in the volume mesh

NdmRTri: Number of resistive triangles in the volume mesh (not used by FSS-CURVE)

NdmZPin: Number of vertical short circuit pins (edges) in the volume mesh

NdmZEd: Number of vertical resistive edges in the volume mesh

NdmRow: Number of matrix elements in the sparse matrix (FE + BI) (FSS-CURVE stores three matrices for top, bottom BI and FE systems. It assumes NdmRow as the number of matrix elements in the sparse top BI matrix.

NdmRowP: Number of matrix elements in the AIPC preconditioner matrix (must be larger than NNZE if GMRES is applied in conjunction with diagonal preconditioner) (not used by FSS-CURVE)

NdmFFTPad: Size of 2-D-FFTPad in one dimension (at least two times number of regular grid points)
(not used by FSS-CURVE)

NdmLoIt: Number of local GMRES iterations (not used by FSS-CURVE)

NdmFd: Number of probe feeds

NdmRowb: Number of matrix elements in the sparse bottom BI matrix

NdmRowFE: Number of matrix elements in the sparse FE matrix

NdmClus: Number of clusters (Choose to be > int(sqrt(NdmNZS)))

NdmTh: Number of series terms in FMM expansion (a good value is 10)

4.3 A Sample Run for Driver.f

We now present a sample run for the driver code by entering input data on the screen. If there is an input file generated for the driver as described in section 4.1, then the user only needs to specify the name of that file.

Example 1:

```
************* FSS_PRISM DRIVER *************

CHOOSE: 1) READ  INPUT DATA FROM THE FILE
        2) ENTER INPUT DATA ON SCREEN
1
```

```
ENTER THE INPUT FILE NAME
Fss_Input
```

Then, the driver gets the data from the specified file and processes them to generate data files "Data1" and "Data2" to be used by the analysis code. On the other hand, if the user wants to enter the input information on the screen, he needs to choose the first input option as "2" referring to the above example. The necessary geometry and excitation information are then asked to enter one by one as shown in the following example.

Example 2:

```
************* FSS_PRISM DRIVER *************
CHOOSE: 1) READ  INPUT DATA FROM THE FILE
        2) ENTER INPUT DATA ON SCREEN
1

ENTER THE INPUT FILE NAME
Fss_Input

CHOOSE: 1) RADIATION PROBLEM
        2) SCATTERING PROBLEM
2

CHOOSE: 1) COMMENSURATE FSS
        2) NONCOMMENSURATE FSS
        3) CURVED FSS
2

THE BOTTOM SURFACE IS OPEN OR CLOSED ?
 1) OPEN    2) CLOSED (METAL BACKED)
1

CHOOSE POLARIZATION TYPE
 1) TE-POL    2)TM-POL
1

ENTER INCIDENCE ANGLE: PHI,TETA (DEG)
0.,90.

ENTER BEGINNING,END AND STEP FREQUENCIES:
      FRBEG,FREND,FRSTP (GHZ)
1.,10.,1.

ENTER NO. OF SAMPLES IN THE BOTTOM SURFACE GRID
50

ENTER NEARFIELD THRESHOLD IN THE BOTTOM SURFACE
GRID FOR THE X- AND Y-DIRECTIONS: NFBX,NFBY
25,25

ENTER NO. OF SAMPLES IN THE TOP SURFACE GRID
30

ENTER NEARFIELD THRESHOLD IN THE TOP SURFACE
GRID FOR THE X- AND Y-DIRECTIONS: NFTX,NFTY
```

```
20,20

ENTER THE LARGEST UNIT CELL SIZE: XL,YL (CM)
2.,2.

ENTER THE FEM SAMPLE SIZE: DX,DY (CM)
0.05,0.05

ENTER THE CAVITY DEPTH: ZL (CM)
1.25

ENTER NUMBER OF LAYERS ALONG THE DEPTH
3

EACH LAYER IS NUMBERED STARTING FROM THE BOTTOM.
NOTE : THE CAVITY APERTURE IS AT Z=0 PLANE.
FOR EACH LAYER, ENTER INFORMATION REQUESTED.
FSS ELEMENTS, X-,Y- DIRECTED LUMPED LOADS,
X-,Y-DIRECTED FEEDS, AND RESISTIVE CARDS MAY
BE PLACED ON TOP SURFACE OF EACH LAYER. ALSO,
Z-DIRECTED LUMPED LOADS, FEEDS, AND SHORT-CKT.
PINS MAY BE EMBEDDED IN EACH LAYER.

****************** LAYER # 1 ******************

ENTER THICKNESS, RELATIVE PERMITTIVITY
AND PERMEABILITY: DT(CM),EPS,MU
0.5,(1.,0.),(1.,0.)

ENTER FSS UNIT CELL SIZE: FXL,FYL (CM)
2.,2.

ARE THERE FSS ELEMENTS ON LAYER # 1 ?
        1) YES          2) NO
2

ENTER NUMBER OF RESISTIVE CARDS ON LAYER # 1
0

ENTER NUMBER OF X-DIRECTED LOADS ON LAYER # 1
0

ENTER NUMBER OF Y-DIRECTED LOADS ON LAYER # 1
0

ENTER NUMBER OF Z-DIRECTED LOADS IN LAYER # 1
0

ENTER NUMBER OF SHORT CIRCUIT PINS IN LAYER # 1
1

FOR THE PIN # 1, ENTER:

(X,Y) COORDINATE OF THE PIN: PIX,PIY (CM)
NOTE: THE LOWER-LEFT CORNER OF THE UNIT
CELL IS THE ORIGIN (0.,0.)
1.,1.

****************** LAYER # 2 ******************

ENTER THICKNESS, RELATIVE PERMITTIVITY
AND PERMEABILITY: DT(CM),EPS,MU
0.25,(2.,0.),(1.,0.)
```

```
ENTER FSS UNIT CELL SIZE: FXL,FYL (CM)
1.5,1.5
ARE THERE FSS ELEMENTS ON LAYER # 2 ?
        1) YES          2) NO
1


CHOOSE AN FSS ELEMENT TYPE:
1) STRIP      3) CROSSED-STRIP
2) SLOT       4) CROSSED-SLOT
   5) USER-SPECIFIED
3

ENTER SIZE OF 1ST SECTION OF THE CROSS: DPX,DPY (CM)
0.7,0.3

ENTER SIZE OF 2ND SECTION OF THE CROSS: DPX,DPY (CM)
0.3,0.7

ENTER LOCATION OF THE LOWER-LEFT
CORNER OF THE 1ST SECTION: XF,YF (CM)
NOTE: THE LOWER-LEFT CORNER OF
THE UNIT CELL IS THE ORIGIN (0.,0.)
0.4,0.6

ENTER LOCATION OF THE LOWER-LEFT
CORNER OF THE 2ND SECTION: XF,YF (CM)
NOTE: THE LOWER-LEFT CORNER OF
THE UNIT CELL IS THE ORIGIN (0.,0.)
0.6,0.4

ENTER NUMBER OF RESISTIVE CARDS ON LAYER # 2
0

ENTER NUMBER OF X-DIRECTED LOADS ON LAYER # 2
1

FOR THE X-DIRECTED LOAD # 1, ENTER:

BEGINNING AND END COORDINATES OF
THE LOAD: CX1,CY1,CX2,CY2 (CM)
NOTE: THE LOWER-LEFT CORNER OF THE UNIT
CELL IS THE ORIGIN (0.,0.)
CY1 SHOULD BE EQUAL TO CY2.
0.4,0.4,0.6,0.4

ADMITTANCE OF THE LOAD: AXLD (1/OHM)
(20.,0)

ENTER NUMBER OF Y-DIRECTED LOADS ON LAYER # 2
1

FOR THE Y-DIRECTED LOAD # 1, ENTER:

BEGINNING AND END COORDINATES OF
THE LOAD: CX1,CY1,CX2,CY2 (CM)
NOTE: THE LOWER-LEFT CORNER OF THE UNIT
CELL IS THE ORIGIN (0.,0.)
CX1 SHOULD BE EQUAL TO CX2.
0.4,0.4,0.4,0.6

ADMITTANCE OF THE LOAD: AYLD (1/OHM)
(10.,0)
```

```
ENTER NUMBER OF Z-DIRECTED LOADS IN LAYER # 2
0

ENTER NUMBER OF SHORT CIRCUIT PINS IN LAYER # 2
1

FOR THE PIN # 1, ENTER:
(X,Y) COORDINATE OF THE PIN: PIX,PIY (CM)
NOTE: THE LOWER-LEFT CORNER OF THE UNIT
CELL IS THE ORIGIN (0.,0.)
0.75,0.75

****************** LAYER # 3 ******************

ENTER THICKNESS, RELATIVE PERMITTIVITY
AND PERMEABILITY: DT(CM),EPS,MU
0.5,(1.,0.),(1.,0.)

ENTER FSS UNIT CELL SIZE: FXL,FYL (CM)
1.,1.

ARE THERE FSS ELEMENTS ON LAYER # 3 ?
        1) YES          2) NO
2

ENTER NUMBER OF RESISTIVE CARDS ON LAYER # 3
1

FOR THE R-CARD # 1, ENTER:

SIZE OF THE R-CARD: RX,RY (CM)
1.,1.

THE LOWER-LEFT CORNER OF THE R-CARD: CRX,CRY (CM)
NOTE: THE LOWER-LEFT CORNER OF THE UNIT
CELL IS THE ORIGIN (0.,0.)
(0.,0.)

SURFACE CONDUCTANCE OF THE R-CARD: SCN (1/OHM)
(1000.,0)

ENTER NUMBER OF X-DIRECTED LOADS ON LAYER # 3
0

ENTER NUMBER OF Y-DIRECTED LOADS ON LAYER # 3
0

ENTER NUMBER OF Z-DIRECTED LOADS IN LAYER # 3
1

FOR THE Z-DIRECTED LOAD # 1, ENTER:

(X,Y) COORDINATE OF THE LOAD: ZLX,ZLY (CM)
NOTE: THE LOWER-LEFT CORNER OF THE UNIT
CELL IS THE ORIGIN (0.,0.)
0.5,0.5

ADMITTANCE OF THE LOAD: AZLD (I/OHM)
(100.,0)

ENTER NUMBER OF SHORT CIRCUIT PINS IN LAYER # 3
0

ENTER THE CURVATURE IN X-DIRECTION
```

```
5000.0
**************** END OF INPUT DATA ****************
```

In addition to the data files, "Data1" and "Data2", a MatLab file "Setup.m" with its data files "MeshDs" and "Attr" is generated by the driver. When this setup file is used along with the available MatLab files "Mesh2.m" and "Mesh3.m", the user can display a 2-D mesh for each layer interface in (x,y) and a 3-D view of the whole geometry in the (x,y,z) coordinate system, respectively. Each planar and volumetric discrete element in the structure is represented by different colors in these figures. This way, the user can view and check the location and size of all specified elements. The colors shown in the mesh view and what they stand for are described as follows:

- Black triangles: "Absorber" triangles. Also, for metal-backed structures, the bottom surface will be in black.
- Black lines: x- or y-directed feeds.
- Black stars (*): z-directed feeds.
- Red triangles: Metallic sections.
- Yellow triangles: Resistive cards.
- Blue lines: x- or y-directed impedance loads.
- Blue stars (*): z-directed impedance loads.
- Black circles(o): short-circuit pins.

The numbering of the nodes and triangles can be visualized using the available MatLab files "GloNod.m" and "TriNum.m", respectively.

5. Examples For FSS-CURVE

This section presents example runs using FSS-CURVE. We give sample input files and the results from the code.

A. 2x2 Curved Slot Array

The input file for the FSS slot array as suggested in [4] is given by:

```
      2           1           3
     1  0.0000000E+00  0.0000000E+00
  8.000000       20.000000       1.000000
```

```
       60          15          15


2.000000       2.000000       0.100000       0.0500000

0.100000               2

0.050000     (4.000000,0.0000000E+00) (1.000000,0.0000000E+00)
0.050000     (4.000000,0.0000000E+00) (1.000000,0.0000000E+00)

        2

        1
        5    6
        0.4  2.0
        0.0  0.0
        0.8  2.0
        0.6  0.0
        0.4  2.0
        1.6  0.0
        2.0  0.1
        0.0  0.0
        2.0  0.25
        0.0  0.85
        2.0  0.15
        0.0  1.85

        0

        0

        0

        0

        0

        0

        0

        0

        0

        0

        5000.0
```

This data defines the 2x2 slot array on a dielectric substrate of height 0.1 cm with relative permittivity ε_r=4 (see Figure 4). The FSS has bandpass characteristics with a resonance frequency of about 13 GHz.

The contents of file "Data1" are:

```
2  1  3

1  0.  0.

  8.00000    20.0000    1.00000

1    1.00000E-02  1  1

1   0.750000  0  39

1  60  120  15  15

  2.00000    2.00000    1.00000E-01    5.00000E-02    90.0000

  1.00000E-01  2

  5.00000E-02  (    4.00000,  0.)  (    1.00000,  0.)
  5.00000E-02  (    4.00000,  0.)  (    1.00000,  0.)
```

The contents of the file "Reflect" are:

```
 8.0000      0.9831    166.0866      0.0010   -100.9983
 9.0000      0.9598    161.3455      0.0013   -107.1797
10.0000      0.9116    154.0573      0.0017   -115.1561
11.0000      0.8007    141.1884      0.0024   -127.9058
12.0000      0.4749    113.1864      0.0033   -150.6444
13.0000      0.2446    -83.3590      0.0036    165.9516
14.0000      0.7342   -131.8348      0.0024    128.5226
15.0000      0.8794   -150.7698      0.0014    119.7041
16.0000      0.9297   -159.0788      0.0009    115.1112
17.0000      0.9543   -163.8897      0.0005    117.0717
18.0000      0.9635   -167.0473      0.0004    119.5416
19.0000      0.9668   -169.3119      0.0002     77.7105
20.0000      0.9677   -171.0102      0.0001     41.7021
```

The contents of the file "Trans" are:

```
 8.0000      0.2505     62.7871      0.0007   -112.5256
 9.0000      0.3384     54.4083      0.0010   -118.9334
10.0000      0.4688     43.7879      0.0014   -133.6464
11.0000      0.6802     28.7348      0.0021   -156.7587
12.0000      1.0145      1.5960      0.0031   -127.3688
13.0000      1.1814    -41.7512      0.0031    152.7710
14.0000      0.8554    -76.9957      0.0034     73.3046
15.0000      0.5768    -92.8665      0.0010     77.7187
16.0000      0.4432   -101.5774      0.0009    113.6983
```

17.0000	0.3639	-107.1637	0.0010	162.5027
18.0000	0.3153	-111.7577	0.0009	163.9082
19.0000	0.2819	-115.7032	0.0021	-1.4368
20.0000	0.2545	-119.1319	0.0018	-20.1334

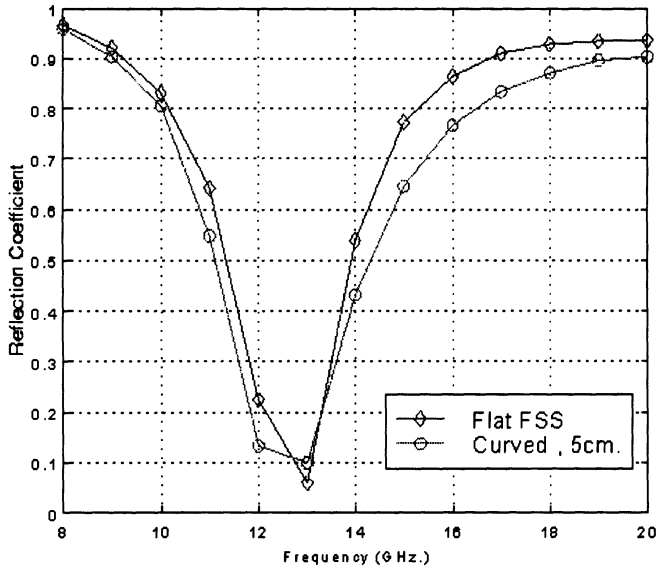
When the same array is wrapped on a 5 cm. radius cylinder the following data is output:

In "Trans":

8.0000	0.2494	61.7632	0.0007	-113.8381
9.0000	0.3397	52.4518	0.0010	-122.1603
10.0000	0.4769	41.2179	0.0014	-137.8322
11.0000	0.6827	21.6030	0.0019	-148.5131
12.0000	0.9197	-9.4047	0.0025	174.9585
13.0000	0.9271	-48.1427	0.0024	141.4440
14.0000	0.7243	-75.8568	0.0014	111.5877
15.0000	0.5542	-92.0735	0.0008	98.0854
16.0000	0.4387	-103.5694	0.0009	146.2644
17.0000	0.3623	-112.2991	0.0012	68.7593
18.0000	0.3067	-119.1433	0.0013	143.5213
19.0000	0.2654	-125.3090	0.0001	-91.4070
20.0000	0.2328	-129.5307	0.0014	-37.0264

In "Reflect":

8.0000	0.9793	166.1468	0.0010	-102.0748
9.0000	0.9507	161.3576	0.0012	-108.4458
10.0000	0.8973	153.9700	0.0017	-116.8433
11.0000	0.7402	142.2992	0.0022	-132.7885
12.0000	0.3656	131.4384	0.0028	-160.5980
13.0000	0.3164	-138.4491	0.0026	163.3590
14.0000	0.6573	-144.2372	0.0018	140.0772
15.0000	0.8034	-153.8060	0.0013	129.0685
16.0000	0.8755	-159.9580	0.0009	123.1191
17.0000	0.9135	-163.9261	0.0006	111.4032
18.0000	0.9332	-166.7416	0.0004	119.7375
19.0000	0.9466	-168.8011	0.0002	116.2907
20.0000	0.9505	-170.4048	0.0000	169.0984


Reflection Coefficient
Frequency (GHz.)
Flat FSS
Curved , 5cm.
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
8
10
12
14
16
18
20

6. Electric field viewer for FSS-Prism by Dan Giszczak

This program was written for Windows 95 or Windows NT using OpenGL.

This program will view EdgeUnk files when a Data2 file is available. Large files may take a while to load. The status bar will show progress. The DOS window will prompt for the number of samples you wish per average edgelength. This fits a chosen number of samples across a distance equal to an average triangle edge length. For parts with few elements, say a hundred, a value like 10 will produce good results. For 10000 elements, 1.0 is good. In general, take 100/sqrt(number of elements) to get a good looking plot. Fraction values are permitted. Values under 0.5 produce no display.

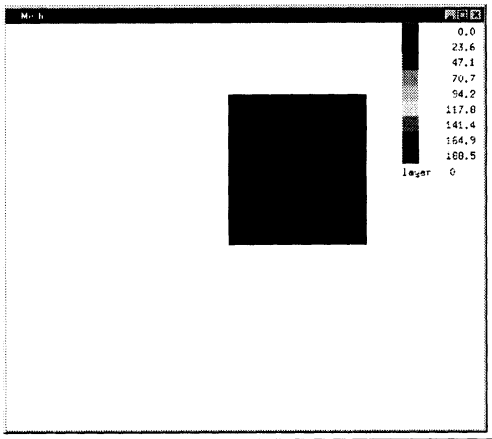
Once the view is loaded, press the first mouse button and drag the mouse on the display window to rotate to model. Press the right mouse button and drag to move the part around. To zoom in and out, either press both left and right buttons together or press the middle button and drag up or down.

T	T alternates between triangle and interpolated modes.
Q	Press q to display the quiver plot. This plot is a set of arrows showing the magnitude and direction of the real part of electric field.

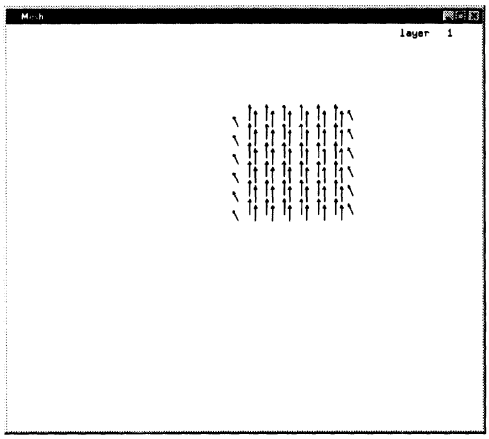
	While showing triangles, the following options are available
G	G will display the geometry. (T or Q returns to data display.)
X	Press x to display the x component of the electric field. Either the real part or the imaginary part may be shown depending upon which mode is current.
Y	Y displays the y component of the electric field. Both real and imaginary parts can be shown separately.
M	This displays the magnitude of the electric field, real or imaginary part.
R	Press r to display the real component x direction, y direction, or magnitude.
I	I will show the imaginary component x direction, y direction, or magnitude.
A	Animate the fields when overall magnitude is displayed.
	While showing interpolated data
R	R will bring up the real part magnitude.
I	I shows the imaginary magnitude.
	While in either mode
O	This displays the overall magnitude of electric field, combining both the real and imaginary magnitudes.
+	This key moves up a sample layer in the FSS.
-	- moves down a sample layer.
L	L displays all layers.
[	To decrease the distance between layers, press [.
]	To spread the layers apart more, press].

Use the Data1 and Data2 files from the Prism example with the generated EdgeUnk. Put them in a directory with FSSEfield.exe and run the executable. When questioned, use 8.8 samples.

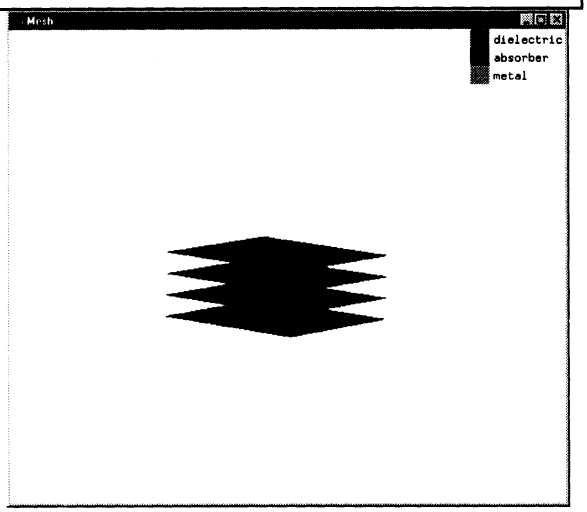
When it loads up, it should look like this.



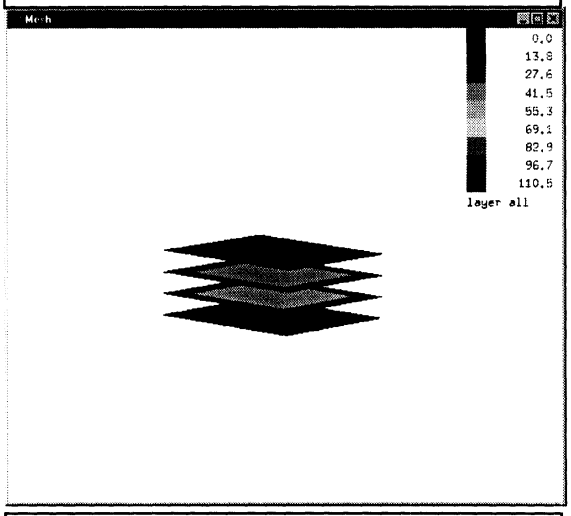
Hit 'Q' for the quiver plot.



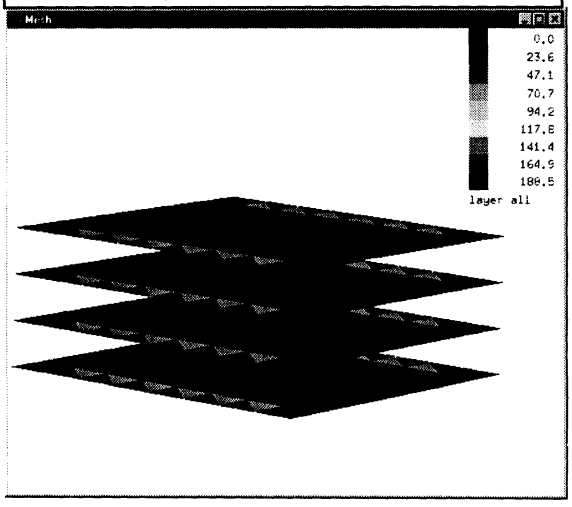
Hit 'G' then press the left mouse button on the window and rotate by dragging to see all the groups.



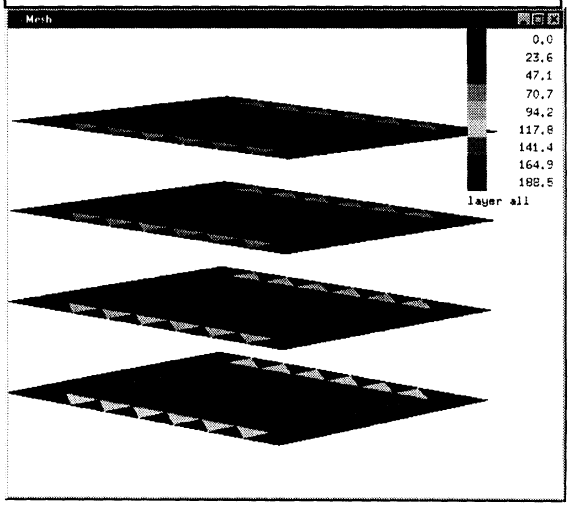
Hit 'T', 'I' to see the imaginary part of the fields.



Press 'T','O' to see the overall magnitude mesh data. Press the middle or left and right buttons and drag up to zoom in.



Press ']' several times to increase the distance between the layers so that you can see them all. Then press the right mouse button and drag up until you can see the whole drawing.



Magnetic current viewer for FSS-Prism by Dan Giszczak

This program will view EqvCur and EqvCurB files when a Data2 file is available.

X	Show magnitude of the current in the x direction.
Y	Magnitude of the currents in the y direction.
Z	Z direction current magnitudes.
O	Show the overall <x,y,z> magnitude.
T	View magnitudes for the top layer.
B	Bottom layer magnitudes.

Bibliography

[1] T. F. Eibert, J. L. Volakis, D. R. Jackson, D. R. Wilton, "Hybrid FE/BI Modeling of 3-D Doubly Periodic Structures Utilizing Triangular Prismatic Elements and a MPIE Formulation Accelerated by the Ewald Transformation," *to be published.*

[2] T. F. Eibert, J. L. Volakis, "Adaptive Integral Method for Hybrid FE/BI Modeling of 3-D Doubly Periodic Structures," *to be published.*

[3] T. F. Eibert, J. L. Volakis, "FE/BI Modelling of 3-D Doubly Periodic Structures with Non-Commensurate Layer Periodicities," *to be published.*

[4] R. Mittra, C. H. Chan, T. Cwik, "Techniques for Analyzing Frequency Selective Surfaces - A Review," *Proc. IEEE*, Vol. 76, No. 12, pp. 1593-1615, Dec. 1988.

[5] H. Aroudaki, V. Hansen, H.-P. Gemünd, E. Kreysa, "Analysis of Low-Pass Filters Consisting of Multiple Stacked FSS's of Different Periodicities with Applications in the Submillimeter Radioastronomy," *IEEE Trans. AP*, Vol. 43, No. 12, pp. 1486-1491, Dec.1995.

[6] J. Gong, D. Jackson, J. Volakis, D. Wilton, "Hybrid Finite Element and Moment Method Software for the SERAT Array," *1st Quarterly Report 035067-1-T.*

[7] S. Bindiganavale, Y. Erdemli, J. Gong, D. Jackson, J. Volakis, D. Wilton, "Hybrid Finite Element and Moment Method Software for the SERAT Array," *2nd Quarterly Report 035067-2-T.*

[8] Y. Erdemli, T. Eibert, D. Jackson, J. Volakis, D. Wilton, "Hybrid Finite Element and Moment Method Software for the SERAT Array," *3rd Quarterly Report 035067-3-T.*

[9] T. Eibert, Y. Erdemli, K. Sertel, D. Jackson, J. Volakis, D. Wilton, "Hybrid Finite Element and Moment Method Software for the SERAT Array," *4th Quarterly Report 035067-4-T.*

[10] T. Eibert, Y. Erdemli, K. Sertel, D. Jackson, J. Volakis, D. Wilton, "Hybrid Finite Element and Moment Method Software for the SERAT Array," *5th Quarterly Report 035067-6-T.*