Updated Sep. 8th, 2005

MATLAB Antenna Toolbox. A draft

The final report to the NSF DUE grant 0231312 **"MATLAB Antenna Toolbox"**

http://ece.wpi.edu/mom/

Sergey N. Makarov

ECE DEPARTMENT, WORCESTER POLYTECHNIC INSTITUTE

100 INSTITUTE ROAD, WORCESTER, MA 01609-2280

Leo C. Kempel

ELECTRICAL ENGINEERING, COLLEGE OF ENGINEERING

MICHIGAN STATE UNIVERSITY

2120 ENGINEERING BUILDING , EAST LANSING, MI 48824-1226

Developers:

Shashank D. Kulkarni

Andrew G. Marut

Stacey Uy

Anuja Apte

Mark Liffiton

Isaac J. Waldron

July 31st 2005

Table of Contents

Chapter I Introduction

Contents

1.1. MATLAB Antenna Toolbox

This text describes the Method of Moments based software – called the MATLAB Antenna Toolbox (MAT) – for the modeling of basic metal-dielectric antennas and resonators. The MAT uses the Method of Moments, the LAPACK matrix solvers compiled in the MATLAB environment, and the built-in MATLAB 3D mesh generators. The MAT is limited to about 7,000 unknowns (metal plus dielectric). Driven solution with a voltage gap feed, scattering solution, and eigenmode solution are currently included.

The present software is reasonably accurate for the simple patch antennas and cavity resonators. At the same time, it cannot compete with existing commercial FEM/FDTD codes such as Ansoft HFSS or others. The software is applicable to basic antenna configurations when a fast and reasonably accurate solution is required. The MAT is mostly intended for educational purposes and is supported by the NSF CCLI:EMD grant 0231312.

The MAT software and this manual are subject to change and extension. The most recent version can be found at http://ece.wpi.edu/mom/.

1.2. Organization of the text and MAT software

The text includes

i. ten basic antenna application examples (Chapters 1-4)
ii. full software description (Chapters 1-7, 9, Appendix A)
iii. underlying MoM theory (Chapters 5-9) including the convergence tests.

The application examples make up about 70% of the manual. The individual MATLAB codes are documented in the help text at the beginning of every code file. Almost every example is accompanied by a related Ansoft HFSS FEM solution, is compared to the corresponding experimental data, or is compared to another solution. The related Ansoft HFSS projects are also included in the downloadable codes. The list of examples is given below in Table 1. Every example uses the same MATLAB code, but with a different geometry and different frequency/far-field conditions.

Table 1. Application examples.

Example name	Ch.	MATLAB project	Ansoft project/other data
Half-wave antenna			
LP patch antenna (1.0% bandwidth, $\varepsilon_r = 2.33$)	2	example21.zip	example21a.zip
LP patch antenna (2.0% bandwidth, $\varepsilon_r = 2.55$)	2	example22zip	example22a.zip
LP patch antenna (0.6% bandwidth, $\varepsilon_r = 9.29$)	2	example23.zip	example23a.zip
RHCP patch antenna (5% bandwidth, $\varepsilon_r = 3.38$)	2	example24.zip	example24a.zip
Microstrip-fed printed slot antenna (21% bandwidth, $\varepsilon_r = 4.4$)	3	example31.zip	Comparison with experiment
Crossed-slot cavity-backed circularly polarized antenna (4.3% bandwidth, $\varepsilon_r = 2.2$)	3	example32.zip	Comparison with experiment
Quarter-wave antenna			
UHF metal monopole at 400 MHz	4	example41.zip	example41a.zip
Top-hat dielectric-loaded monopole ($\varepsilon_r = 10.0$)	4	example42.zip	example42a.zip
Baseline planar-inverted F-antenna – PIFA (10% bandwidth)	4	example43.zip	example43a.zip
Reduced-size PIFA (3% bandwidth)	4	example44.zip	example44a.zip
Antenna optimization loop			
Top-hat loaded monopole	App.B	Example1b.zip	none
Dielectric resonator			
Sphere DR, disk DR	App.C	example1c.zip	none

Every antenna example includes a frequency sweep for the input impedance/return loss and far/near field calculations.

1.3. How to download the solver and examples

To download the MAT05 codes please refer to http://ece.wpi.edu/mom/ and follow the links. There is only one version of the code which is applied to a number of different examples. The downloaded MATLAB codes are executed following the flowchart given in the next subsection. In the text that follows, we also provide a more detailed step-by-step explanation, based on the example of an isolated cylindrical dielectric resonator.

1.4. Software/hardware requirements

Software: Microsoft Windows XP, MATLAB **R14** or higher, MATLAB PDE Toolbox **R14** or higher. Hardware: minimum 0.5 gigabyte of RAM, **PIV** or higher.
The mesh generator requires that the MATLAB Partial Differential Equations (PDE) Toolbox be included into MATLAB installation. A trial (one month trial) version of the toolbox is available from the MathWorks website.

1.5. Quick execution flowchart

The quick code flowchart is shown in Fig. 1.1 below. A typical project includes three folders: `1_mesh`, `2_basis`, and `3_mom`. First, the scripts `struct2d.m` and `struct3d.m` from the folder `1_mesh` are executed in order to create antenna geometry, identify material parameters and create the antenna feed. Next, the script `wrapper.m` from the folder `2_basis` is executed in order to create the MoM basis functions. Then, one executes the script `impedance.m`, which performs a frequency sweep; finds the input impedance, return loss, and VSWR at every frequency step; and saves the complete MoM solution at every frequency step. This operation is similar to a "Discrete frequency sweep" in Ansoft HFSS. The radiation patterns (co/cross-polarization or right/left-handed CP) are found after executing the script `radpattern.m`. Finally, the near fields within the antenna or on the antenna surface are found using the script `nearfield.m`.

For the eigenmode solution, one uses the scripts `eigenfreq.m` and `mode.m`. For the scattering solution, the script `scatterfield.m` should be used.

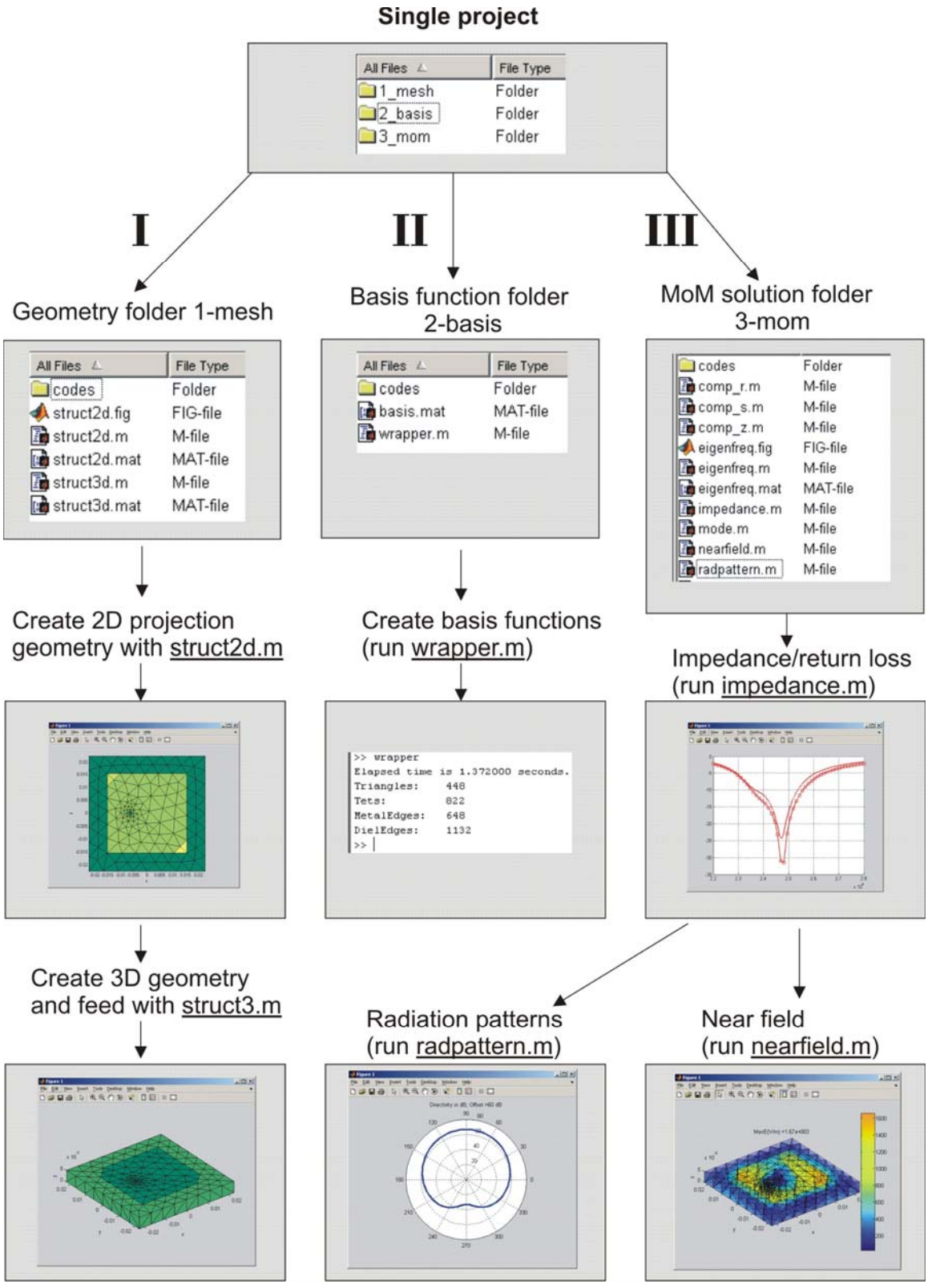

Fig. 1.1. Quick execution flowchart.

1.6. Mesh generator

a. Built-in mesh generator – projection concept

The MAT includes a 3D built-in interactive mesh generator (folder `1_mesh`) integrated into a MATLAB GUI. The mesh generator creates, modifies, and saves a metal/dielectric antenna mesh including the feeding elements. A (inhomogeneous) lossy dielectric material may be defined as well as metal surfaces of arbitrary planar or cylindrical shape. This mesh generator is used for any antenna/resonator type. No multiple custom MATLAB scripts are necessary.

The mesh generator employs Delaunay triangulation in 2D and Delaunay tessellation in 3D, both available in MATLAB using the standard functions `initmesh` or `delaunayn`, respectively. The following mesh generation steps are used:

1. The mesh generator creates an arbitrary unstructured planar mesh – a horizontal projection of all elements of the anticipated volumetric antenna or resonator.
2. Based on that planar mesh, the mesh generator creates one planar layer of tetrahedra. This layer is called the base layer. The base layer may have an arbitrary non-convex shape in the horizontal plane.
3. The generator creates each of the other layers of the structure by shifting the base layer up or down, with some tetrahedra being removed or the dielectric constant being changed if necessary.
4. The user selects the metal faces including ground plane(s), microstrips, vertical via(s), and the feed edges by selecting faces of the base volume mesh. The metal faces/edges are selected for every layer (group of layers), with the polygon tool, or individually, or both.

The mesh generator requires that the MATLAB Partial Differential Equations (PDE) Toolbox be included in the MATLAB installation. A trial version of the toolbox is available from the MathWorks website. For a complete flowchart of the folder `1_mesh` please refer to other supporting documentation.

b. Planar mesh generator

First, a 2D planar projection surface mesh for the base layer is created using 2D Delaunay triangulation (implemented in MATLAB PDE toolbox as `initmesh.m`). The planar mesh generator is called `struct2d.m`, and it is located in folder `1_mesh`. It is used to define the shape of the base layer by defining rectangles, ellipses, and a polygon, from which the structure can be created by taking the union, intersection, or set difference of any combination of these shapes. The GUI of `struct2d.m` is shown in Fig. 1.2 (a patch antenna from example24 in Chapter II will be used here and in what follows).

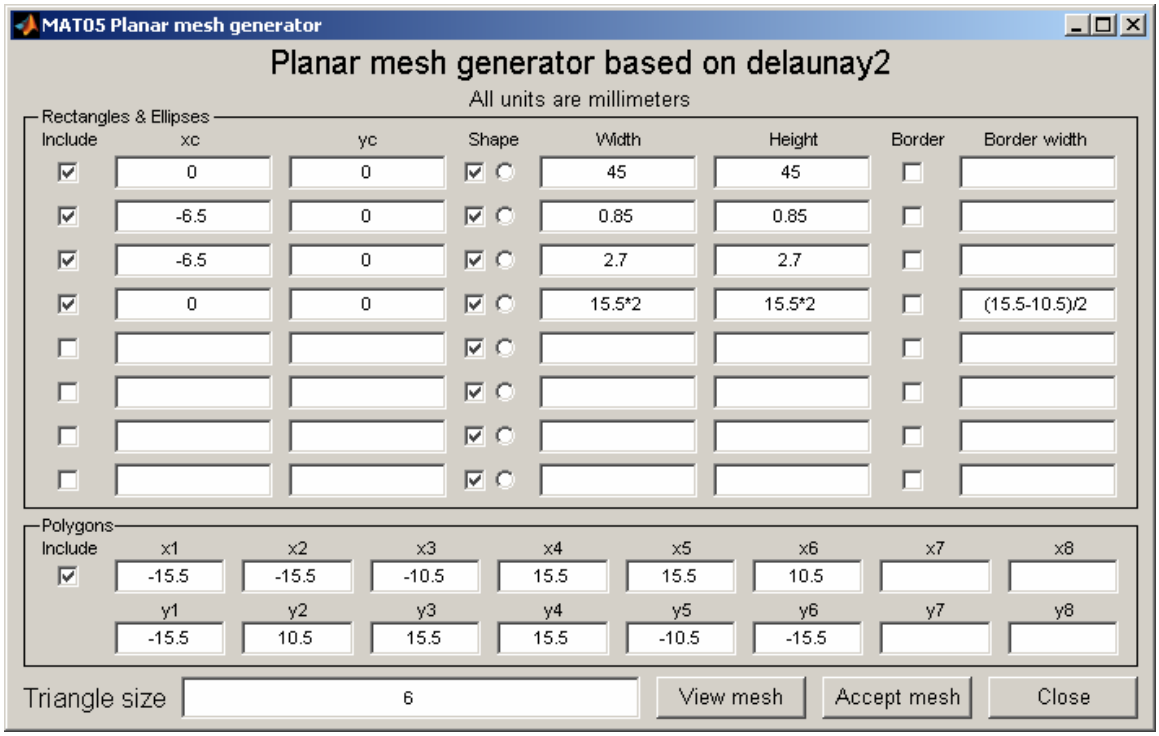

Fig. 1.2. A 2D projection geometry of the base layer. All units are in millimeters.

To define the geometry, one may delimit up to a total of eight unique rectangles and ellipses in addition to one polygon having up to eight vertices. An internal border may be included in a rectangle or ellipse for finer meshing close to the boundary. The first column in Fig. 1.2 is a checkbox labeled `Include` that includes or excludes the geometry object defined in that row from the structure. The next two items, labeled `xc` and `yc`, are the *x*- and *y*-coordinates of the object center. MATLAB vector expressions can be entered

into the text fields in these columns to make copies of the same shape in different locations on the *xy*-plane; an example of this appears in Fig. 1.14. Furthermore, any text field may contain a MATLAB arithmetical expression, as shown in the fourth row of Fig. 1.2, instead of the number to which this expression corresponds.

The next column, `Shape`, selects between rectangle and circle/ellipse. The next two columns, `Width` and `Height`, allow the user to enter the appropriate dimensions of the object along the *x*- and *y*-axes, respectively. The next column is a checkbox `Border` that defines whether a border of the specified width (the next column) should be drawn for the object defined in that row. The lower frame in Fig. 1.2 defines a polygon. The `Include` checkbox has the same meaning as in the upper frame. The remaining columns allow the user to enter up to eight vertices of the polygon.

At the bottom of the GUI, there is an important field labeled `Triangle size`. This field defines the overall mesh grid size.

After delimiting the shapes to be included in the geometry, one presses the `View mesh` button to view a triangular mesh. If the mesh is satisfactory, the button labeled `Accept mesh` needs to be pressed to save the data, followed by the `Close` button to exit. The corresponding output of `struct2d.m` is shown in Fig. 1.3.

When one presses the `Accept mesh` button, `struct2d.m` saves the planar mesh to the file `struct2d.mat`. This file contains two matrices:

`P` – a 3-by-M array of nodes; P(1, *i*) is the *x*-coordinate of the i^{th} node, P(2, *i*) is the *y*-coordinate, and P(3, *i*) is zero and represents the *z*-coordinate of the node.

`t` – 4-by-N array of triangles; t(1:3, *j*) are the three indices in `P` of the vertices of the j^{th} triangle. In addition, t(4, *j*) is the subdomain number of the j^{th} triangle.

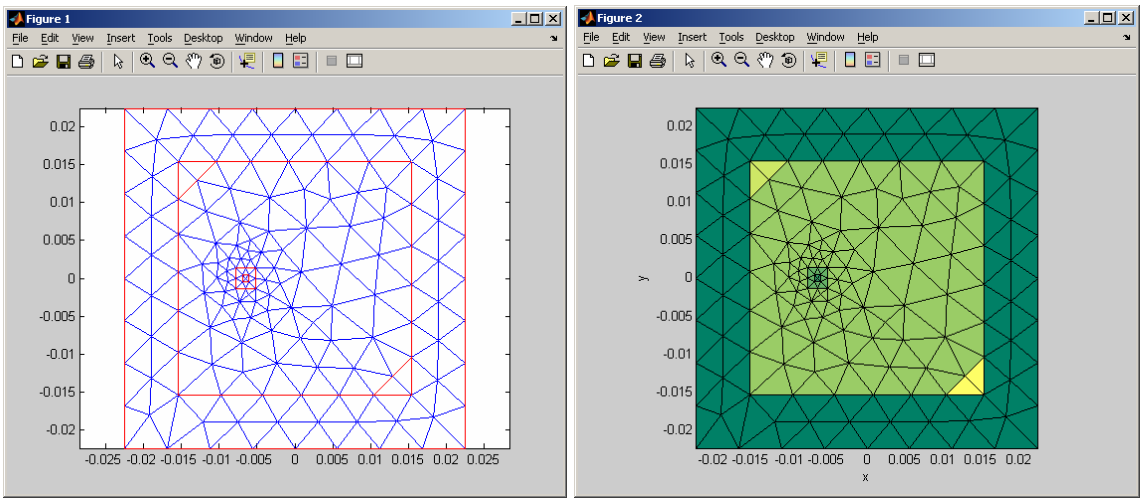

Fig. 1.3. `View` (left) and `Accept` (right) mesh screen. The surface mesh includes a ground plane shape and a polygonal patch antenna shape, with the feed position rectangle, which are both specified in the GUI in Fig. 1.2. Note that the `border` option in Fig. 1.2 is turned off. Different colors in Fig. 1.3 – right indicate different subdomain numbers used in the PDE toolbox.

c. Volume mesh generation

Second, the volume tetrahedral mesh for the structure is created using the script `struct3d.m`. On execution of the file `struct3d.m`, the layer partitioner GUI `layers.m` shown in Fig. 1.4 prompts for data that are used to divide the entire dielectric mesh into layers of tetrahedra. This GUI (located in subfolder `codes`) allows partitioning of the dielectric mesh into multiple planar layers. Each of the layers is initially a duplicate of the base layer. The layers are subdivided into groups of identical layers. This is done in order to simplify operations for multiple identical layers. Every group of layers may then acquire

1. Different physical characteristics (dielectric constant/loss tangent). Please do not introduce layers with $\varepsilon_r = 1$ (air). Instead, cut the entire layers later on.
2. Different dielectric geometric forms by cutting out unwanted tetrahedra.
3. Metal faces of arbitrary shape on the top or bottom of the layer or group of layers.

4. Metal faces embedded vertically (via(s)). Vertically adjacent metal faces will be automatically interconnected to each other or to other horizontal faces for every layer or group of layers.
5. Antenna feed. The feed is initialized by selecting the feeding edges (which are always the bottom edges of the corresponding group of layers). Usually, they correspond to the bottom edges of the metal column representing the coaxial-probe feed.

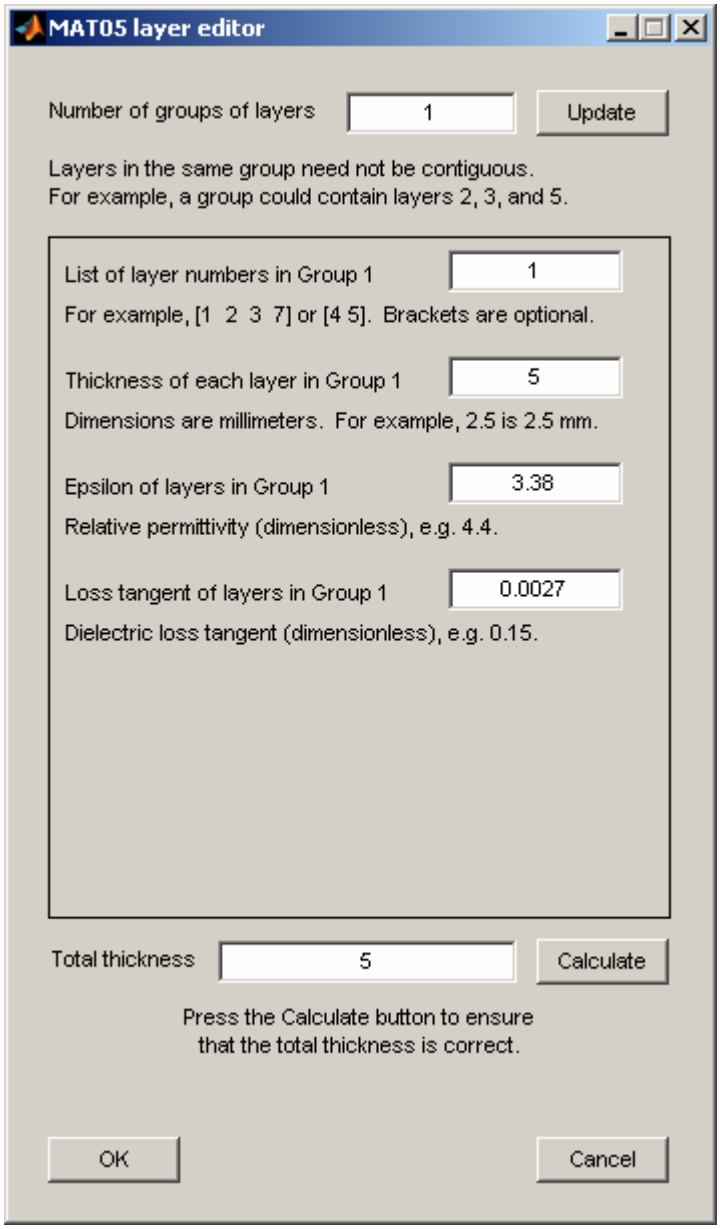

Fig. 1.4. Layer partitioner GUI `layers.m` – specifies parameters for separate layers of tetrahedra. The present GUI includes one layer and one group of layers.

The top text field in Fig. 1.4 specifies the number of groups of layers; the user may enter a positive integer here. After entering the desired number of groups, the user must press the `Update` button to create new layer groups or remove the old groups.

All layers within a group have the same thickness, dielectric constant, dielectric loss tangent, and cross section, but the layers do not need to be contiguous. For each layer group, one enters a list of the layers in the group; each layer is denoted by an integer from 1 to the total number of layers. In addition, the thickness, dielectric constant, and the dielectric loss tangent for all layers in the group should be provided.

After the data for all layer groups are entered, the user should press the `Calculate` button below the frame to ensure that the total thickness of the structure and the layer data are correct. The program ensures that each layer appears in exactly one group. If there is any inconsistency in the layer data or in the total thickness, the program will generate the corresponding warning message.

To continue with the next part of the program, the user next presses the `OK` button. The layer data will be saved in the file `layers.mat`. If the user does not want to save the data and proceed, the `Cancel` button may be pressed to exit the program without saving any changes. Closing the window has the same effect as pressing the `Cancel` button.

After partitioning the dielectric structure into groups of layers, the triangle/edge selector (`triselect.m` in subfolder `codes`) is displayed as shown in Fig. 1.5. The triangle selector is used to remove tetrahedra or groups of tetrahedra from a group of dielectric layers or perform any other operation listed above. For every group, a message in the title bar (see the top of Fig. 1.5) indicates the layers to be handled and the operation to be done.

The triangle selector allows one to select triangles in two ways: individually or as a cluster within a polygon; the latter is the default, and it is done by drawing the corresponding polygon with several mouse clicks on the figure. The polygon is automatically closed either by connecting the first and last vertices when one places a vertex in the triangle containing the first vertex or when the blue button `Close polygon` is pressed. The edges should be selected using only the polygon tool.

Pressing the `Cancel polygon` button erases the polygon drawing from the screen. The `Select all` button selects all the triangles/edges of the surface. The `Zoom`

buttons cause the user's next click to zoom in or out and center the resulting view. The `Done` button closes the polygon, selects all the triangles or edges inside, and closes the window.

Fig. 1.5 below shows the operations necessary to create the metal/dielectric mesh for the patch antenna with two chamfer cuts.

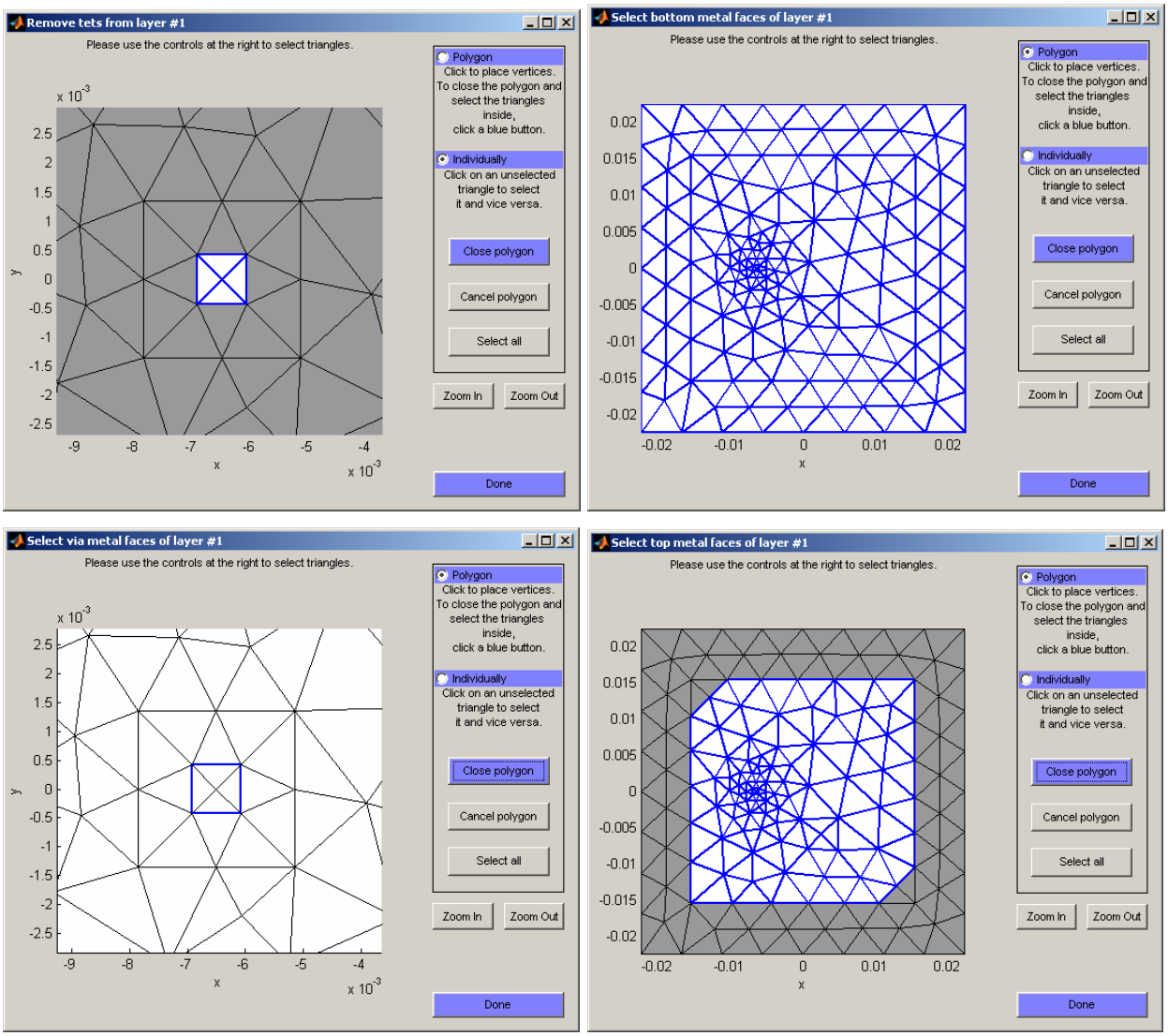

Fig. 1.5. Selection operations used to create the patch antenna mesh. Top left: selection of tetrahedra to be removed from the mesh (feed column only). Top right: selection of the ground plane. Bottom left: selection of the via edges and the feeding edges. Bottom right: selection of the patch.

First, the dielectric tetrahedra are removed from the feed column (Fig. 1.5 top left) using individual selection by mouse click. Then, all metal faces of the ground plane are selected using the `Select all` operation (Fig. 1.5 top right). After that, the four via edges are selected with the polygon tool by carefully drawing a small polygon around every bottom edge of the feed column and then using the `Close polygon` command (Fig. 1.5 bottom left). Next, the feeding edges are selected in exactly the same way as the via edges. For this particular design, the via edges and feed edges coincide. Finally, the top patch is selected by drawing a polygon and closing it (Fig. 1.5 bottom right). The `OK` button is then pressed on the `remove tetrahedra` GUI, which pops up after the mesh operations for a group of layers are complete.

A GUI `view3d.m` (located in subfolder `codes`) pops up after the entire structure is complete, as shown in Fig. 1.6. It displays four radio buttons that allow the user to choose how to view the mesh. One may view the outer dielectric faces, the dielectric tetrahedral grid, the metal faces without the dielectric, or the metal faces with the dielectric. Choosing any view opens a new window containing that view.

When `struct3d.m` finishes, the following variables are saved in the main structure file, `struct3d.mat`:

`P` - array of nodes

`T` - array of tetrahedra

`Faces` - array of dielectric faces

`FacesNontrivial` - number of nontrivial dielectric faces (placed up front: boundary faces, inner faces with nonzero dielectric contrast, inner faces in contact with embedded metal faces)

`Edges` - array of dielectric edges

`AT` - array of adjacent tetrahedra for every dielectric face

`const` - structure array that includes dielectric data for every tetrahedron.

`t` - array of metal faces

`Edgesm` - array of inner metal edges

`FeedIndexes` - array of feed edges

Other arrays related to the RWG basis functions are saved in the same file.

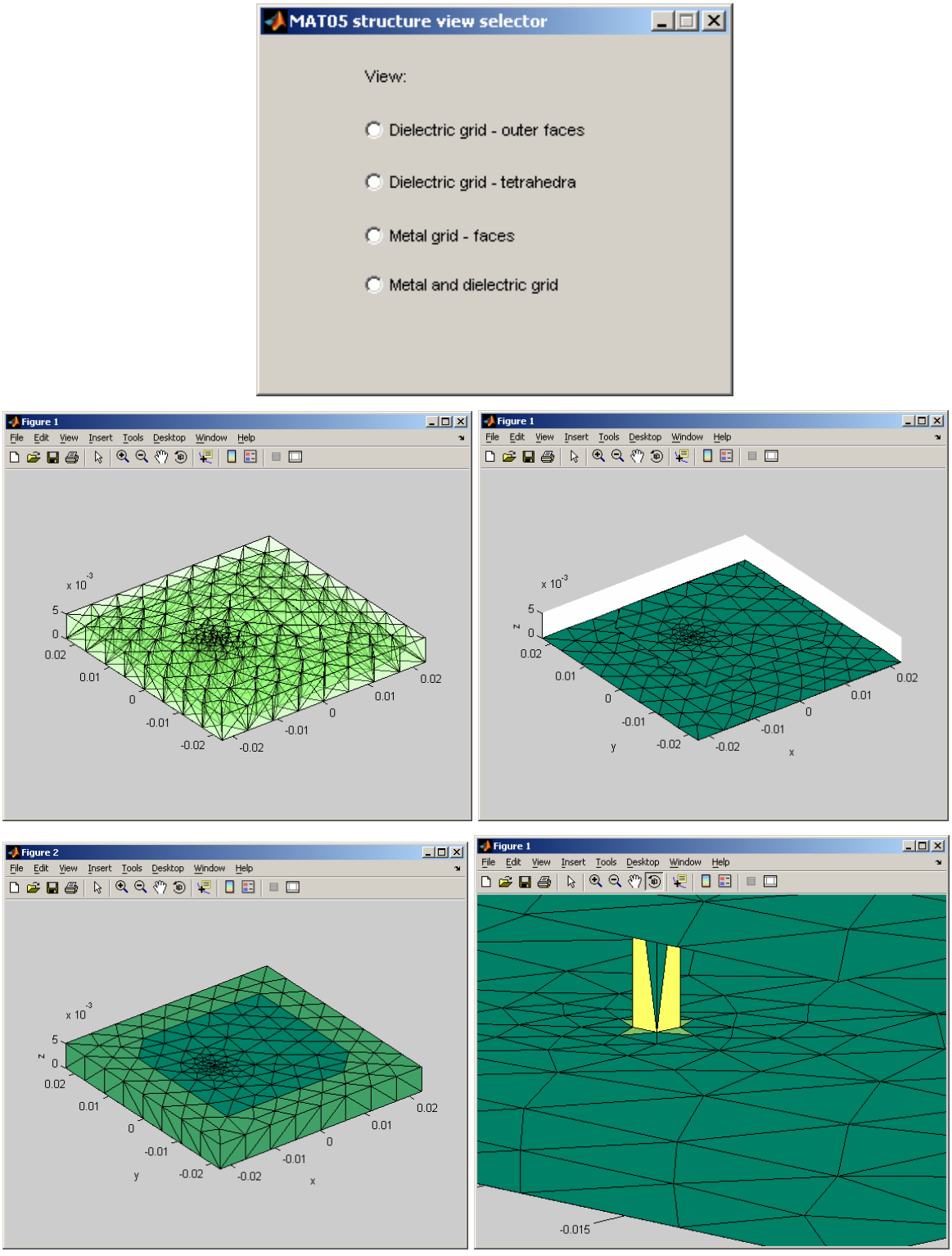

Fig. 1.6. Patch antenna mesh inspection using GUI `view3d.m`. From top left to bottom right: tetrahedral mesh, metal mesh, dielectric/metal mesh, feed column.

It is important to note that the second (and any subsequent) call to `struct3d.m` will display the GUI `view3d.m` from Fig. 1.6 without updating the existing geometry structure. To create a new geometry file one should either go back and run `struct2d.m` (and press the `Accept mesh` button) or delete the data file `struct3d.mat` manually and then run `struct3d.m` again.

1.7. Basis function generator

The Rao-Wilton-Glisson MoM metal basis functions [1], the dielectric edge basis functions [2-4], and the associated geometry parameters are created in folder `2_basis`. This folder acquires the geometry data saved previously in the structure file `struct3d.mat` (folder `1_mesh`).

To establish the basis functions, one simply runs `wrapper.m`, which calls the `MAT` functions `dielectric.m` and `metal.m`. (subfolder `codes`). The function `dielectric` returns a MATLAB structure `GEOM`, saved in the data file `basis.mat`. The function `metal` returns a MATLAB structure `geom`, saved in the data file `basis.mat`. These structures define all the parameters of the dielectric tetrahedral and the metal triangular mesh. The structures `GEOM` and `geom` are then used to solve the MoM equations.

The structure `GEOM` has a number of fields described in the MATLAB script `dielectric.m`. Likewise, the structure `geom` has a number of fields described in the MATLAB script `metal.m`. These fields give different characteristics of the basis functions described in Chapters V and VI. The most important operation done over the dielectric basis functions is the orthogonalization procedure [3] implemented in the `MAT05` function `basis.dll` (subfolder `codes`).

The `MAT` functions `dielectric.m` and `metal.m`. (subfolder `codes`) define the integration rules over faces/tetrahedra and find the necessary quasi-static potential integrals for neighbor faces/tetrahedra (Chapters V, VI, and Appendix A). A wide range of Gaussian integration formulas are available in the code. The Gaussian formulas are described in Chapters V and VI. The default (low-order) formulas were found to give the best accuracy for the present low-order basis functions.

For a complete flowchart of the folder `2_basis` please refer to the supporting documentation (Chapters V and VI).

1.8. MoM solution

a. Driven lumped port (voltage gap) solution

The antenna solution is performed in the folder `3_mom`. First, the frequency sweep should be specified manually in the script `impedance.m`. Then, one executes the script `impedance.m`, which performs a frequency sweep; finds the input impedance, return loss, and VSWR at every frequency step; and saves the complete MoM solution at every frequency step. This operation is similar to "Discrete frequency sweep" in Ansoft HFSS. Fig. 1.7 shows the script output for the impedance of the patch antenna defined above.

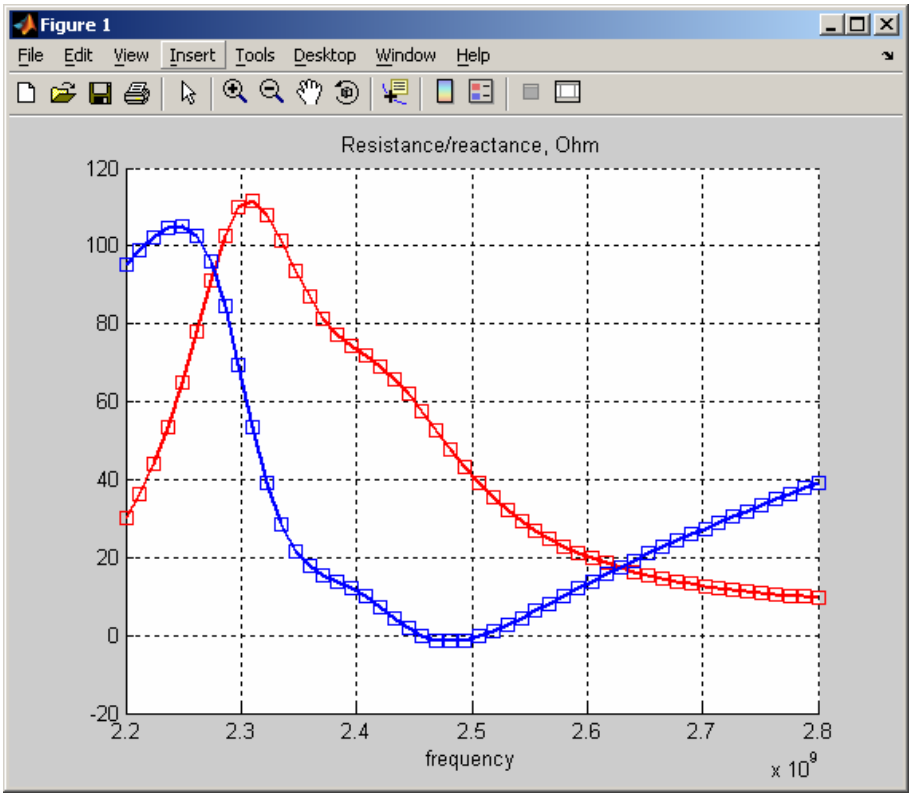

Fig. 1.7. Antenna input impedance. Reactance (blue) crosses zero at approximately 2.45 GHz.

The corresponding return loss (squared curve) in comparison with the FEM Ansoft HFSS solution (solid curve) is shown in Fig. 1.8.

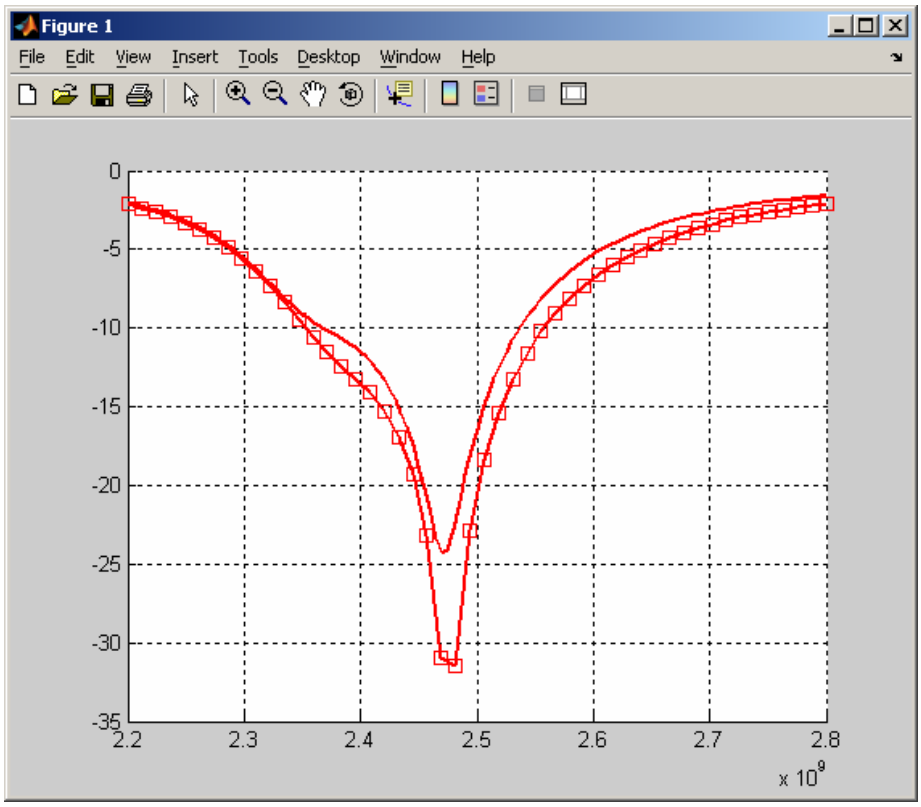

Fig. 1.8. Antenna return loss (squared curve) and the return loss predicted by the Ansoft HFSS solution with approximately 20,000 tetrahedra (solid curve). Note that MATLAB plots the negative value of the return loss.

Next, the radiation patterns (co/cross-polarization or right/left-handed CP) are found by executing the script `radpattern.m`. The choice of the frequency (which should lie within the frequency sweep done previously), polarization component, and the radiation pattern plane is manually done in the script `radpattern.m` (spherical coordinates).

Since the present antenna is circularly polarized, the `ER` (RHCP) and `EL` (LHCP) components should be chosen in the script `radpattern.m`. Fig. 1.9 shows these two components at 2.40 GHz compared to the Ansoft HFSS solution (solid curves). The right-handed circular polarization dominates in the major lobe. The polarization isolation is

approximately 14 dB. Note that the MATLAB script for radiation patterns has the offset of 60 dB in order to be able to plot negative values in dB.

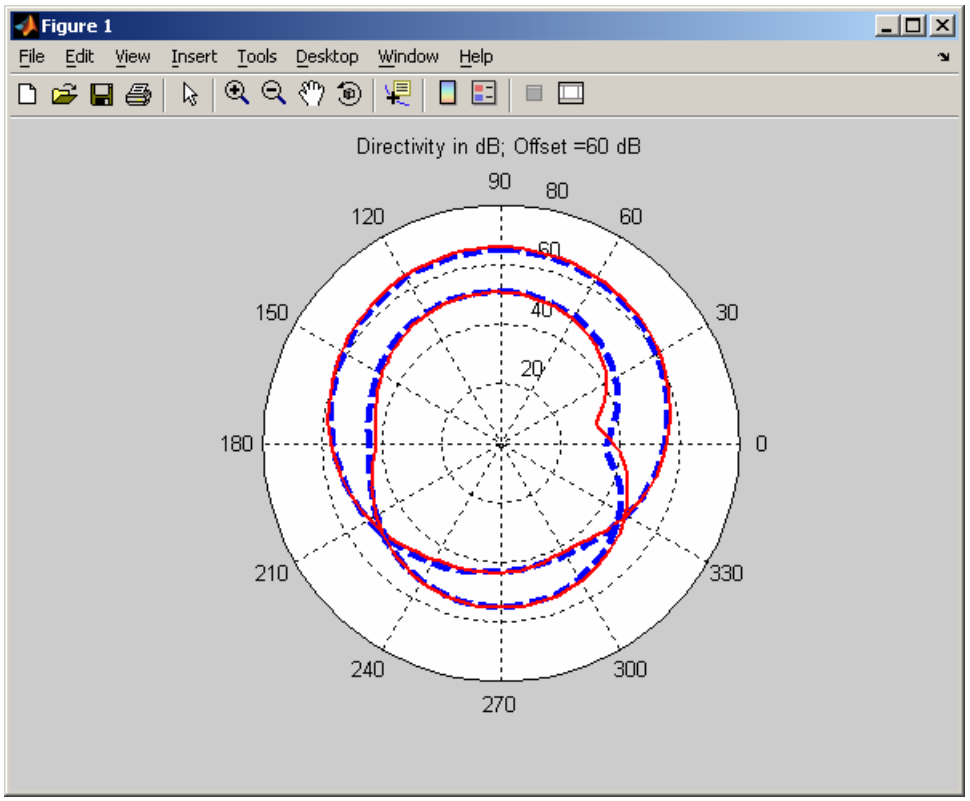

Fig. 1.9. Absolute directivity of RHCP (higher at zenith) and LHCP (higher in the back lobe) in the xz-plane of the patch antenna at 2.40 GHz. The corresponding Ansoft HFSS solution is shown by dashed curves. Note the offset marked in the figure title.

Finally, the near fields within the antenna or on the antenna surface are found using the script `nearfield.m`. This script acquires the frequency of interest and uses the MATLAB GUI `viewfields.m` (subfolder `codes`) to inspect the field distribution. The field distribution includes electric and magnetic field in the dielectric, bound and free surface charges, and the metal current density. Fig. 1.10 shows the GUI output for the electric field in the dielectric (the magnitude of the dominant vertical component) within the patch antenna at 2.40 GHz. Since, for the circular polarization, two nearly degenerate

TM modes are simultaneously excited in the patch cavity, the field does not have a standard TM mode shape (Chapter II).

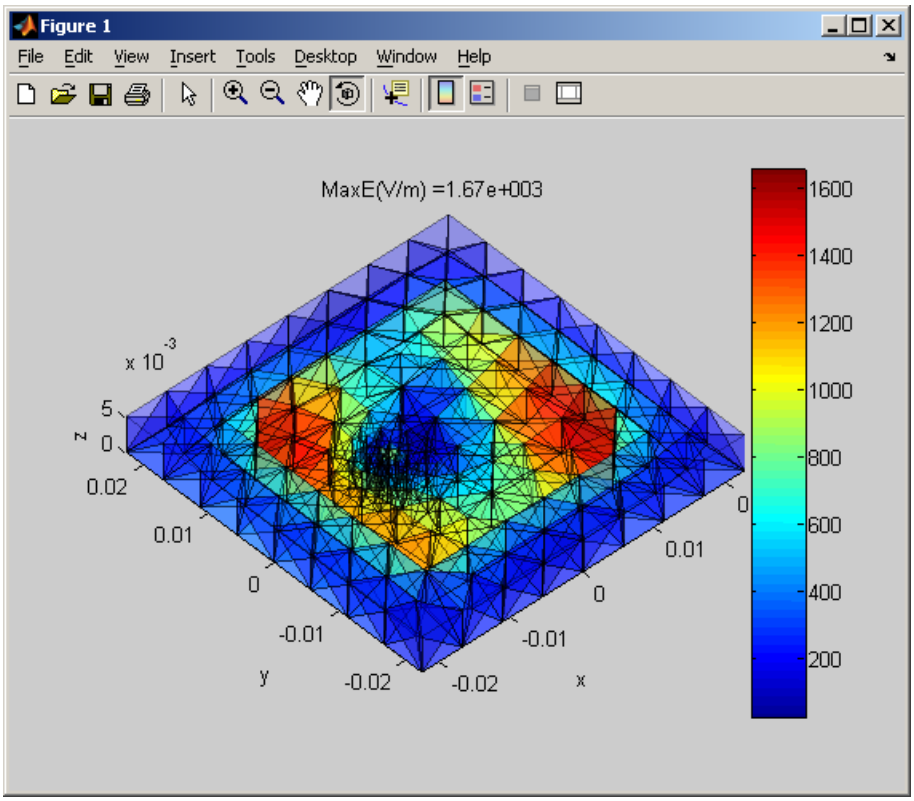

Fig. 1.10. The GUI output for the electric field in the dielectric (the magnitude of the dominant vertical component) within the patch antenna at 2.40 GHz.

b. Eigenmode solution

Along with the driven solution, one can obtain the eigenmode solution for the present antenna. The algorithm for the search for the resonant frequency and the *Q*-factor is described in Chapter VI, Section 6.6. The input parameters include the search range for the real part of the frequency (resonant frequency) and the search range for the imaginary part of the frequency (which determines the *Q*-factor). When the approximate starting resonant frequency is not known, it is recommended that the user choose a wider range of frequency variation.

To find the resonant frequency and the Q-factor, one needs to run the MATLAB GUI `eigenfreq.m` in folder `3_mom`. The GUI script `eigenfreq.m` (direct eigenmode search) in the folder 3_mom is intended for the eigenfrequency search. It will not run for

the present antenna configuration. To find the eigenfrequencies of the corresponding TM resonator one must go back to the folder `1_mesh` and create the same structure, but without the antenna feed (do not remove tetrahedra from the feed, do not select metal via patches for the feed, and do not select any feed edges). Then, create the basis functions and run `eigenfreq.m` in order to find the resonant frequency and the *Q*-factor of the resonator. The script output for this patch antenna is shown in Fig. 1.11. One can see two resonant modes, at approximately 2.32 and 2.44 GHz. The *Q*-factors are about 16 and 10, respectively. The eigenmode search can have a finer resolution. The eigenmode fields might be found using the script `mode.m` (which works for the pure dielectric only).

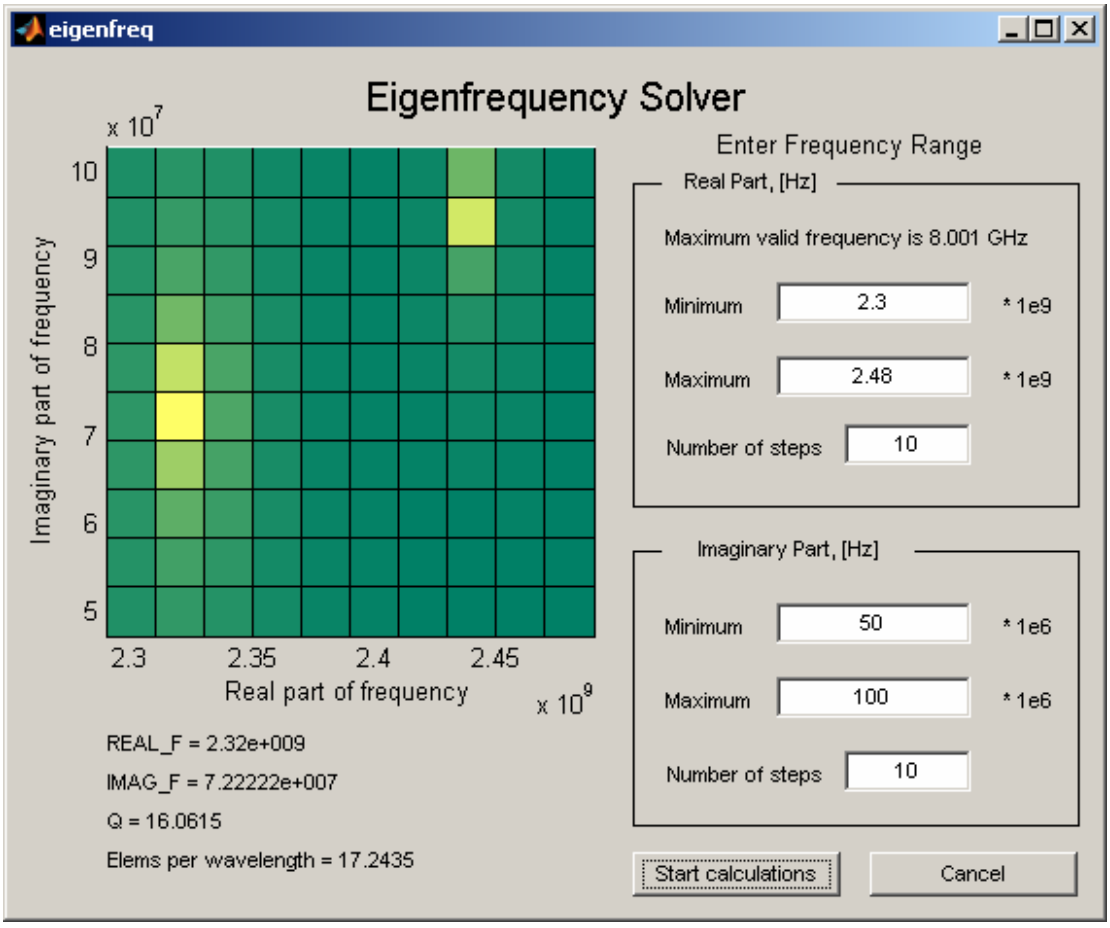

Fig. 1.11. The GUI output for the direct eigenmode solution for the patch antenna cavity (with the feed removed). The light dots in the plane of complex frequency indicate two close resonances.

c. Driven scattering solution

One can solve the scattering problem at a given frequency, using the `scatterfield.m` script. The GUI `nearfield.m` is then executed to inspect the near field distribution.

1.9. Antenna mesh refinement

For thin dielectric substrates (Chapters II and III), an accurate solution is obtained with one layer of tetrahedra (see convergence tests in Chapter VII). Mesh refinement in the vertical direction is therefore not necessary.

For thick substrates, such as that used for the present patch antenna, the mesh refinement in both the vertical direction and the lateral direction might be useful. As an example, Fig. 1.12 shows the input impedance behavior for two different meshes. One (coarse) mesh is used in Fig. 1.6. Another (finer) mesh is specially created in order to estimate the method convergence. The Ansoft HFSS impedance solution is provided for comparison. Clearly, the finer mesh somewhat better describes the impedance behavior, although the differences still remain. It is believed that these differences are related to the antenna feed, differently implemented in the Ansoft HFSS solution (see the next Section). At the same time, the return loss and the radiation patters remain nearly the same for both the meshes. Note that the execution time very significantly increases for finer meshes.

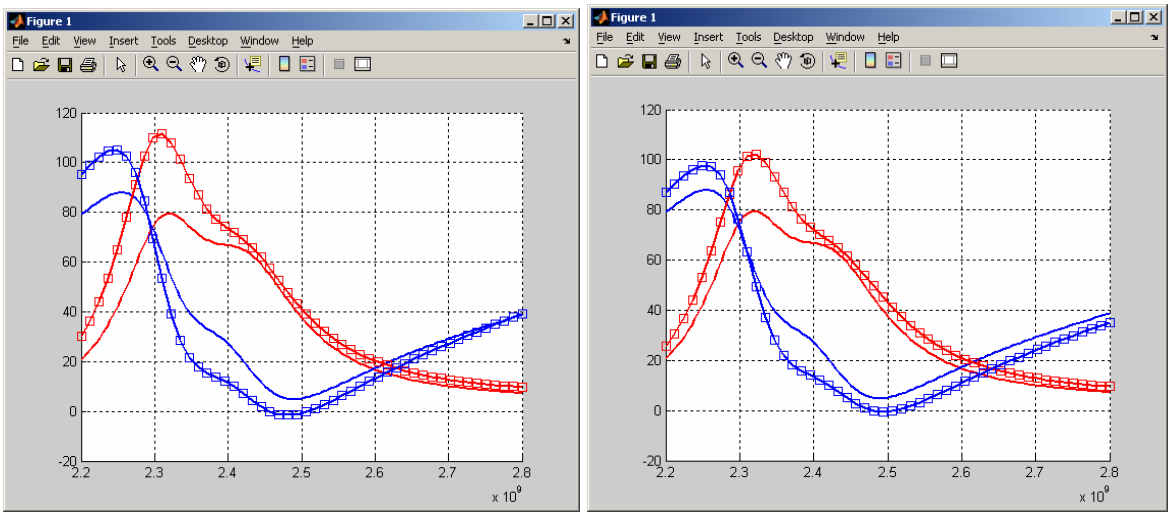

Fig. 1.12. Input impedance for the mesh with 1780 unknowns (left) and 4898 unknowns (right). The Ansoft HFSS solution is given by solid curves.

1.10. Probe feed model

The default MATLAB feed model (script `impedance.m`) is the voltage gap for every feeding edge of a metal column (rectangular or cylindrical) or a metal strip (cf. [5]). The related Ansoft HFSS solutions use two different lumped port models. A more realistic feed model uses the perfect magnetic boundary – a disk gap around the probe in the ground plane shown in Fig. 1.13 left. The voltage (electric field) is given across this gap. Another feed model determines the voltage gap directly between the probe and the ground plane as shown in Fig. 1.13 center. The first feed model better accounts for a finite opening of the coaxial probe feed.

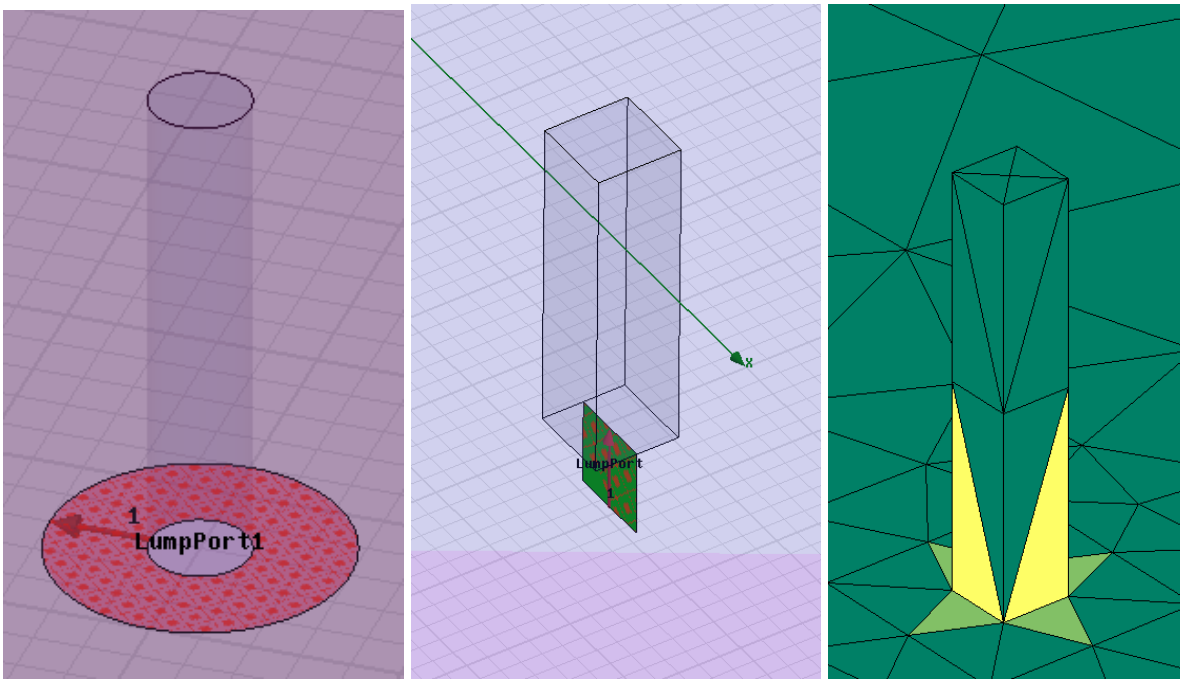

Fig. 1.13 Two feed models used in Ansoft HFSS solutions. Left – disk gap; center – voltage gap between the feed and the ground plane. The MoM voltage gap feed for the bottom edges of the metal column (two basis functions per edge) is shown on the right.

The MATLAB MoM delta-gap feed model (Fig. 1.13 right) perhaps holds an intermediate position between these two models. It does not take into account the finite opening of the coaxial cable. The accurate feed model is of somewhat lesser importance

for the single-frequency probe-feed patch antennas on thin substrate (Chapter II) but becomes important for antennas on thick substrates.

1.11. Mesh generator capability

The mesh generator discussed in Section 1.6 can be used to create periodic finite structures like a small antenna array or a textured substrate. As an example, we present in Fig. 1.14 the GUI `struct2d.m` that creates a textured dielectric substrate used in Ref. [6]. The substrate combines two ceramic materials with the base $\varepsilon_r = 30$ and the embedded ceramic disks with $\varepsilon_r = 100$ and has total 36 embedded disks. The structure is created by using MATLAB vector expressions that clone a single element at different *x* and *y*-locations and are directly inserted into the GUI input fields. The output of the volume mesh generator is shown in Fig. 1.15. The structure has 3564 tetrahedra (4828 dielectric unknowns) and still can be computed in a reasonable amount of time.

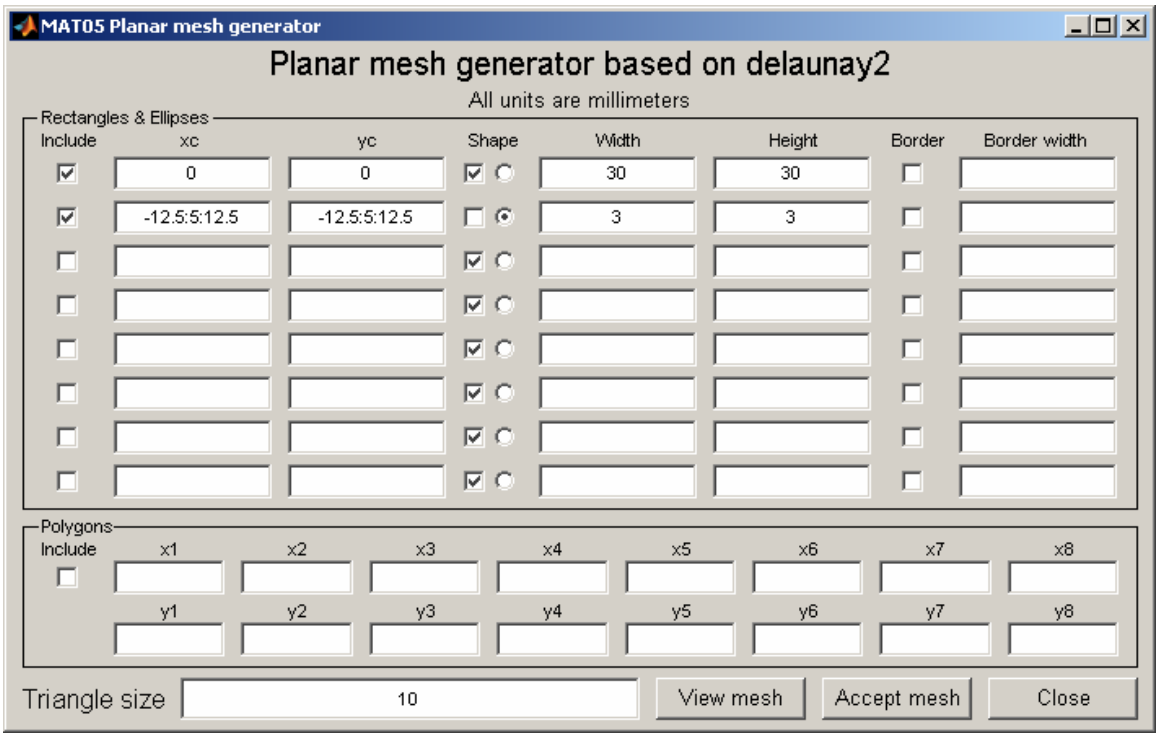

Fig. 1.14. Vector operation to create a finite periodic structure.

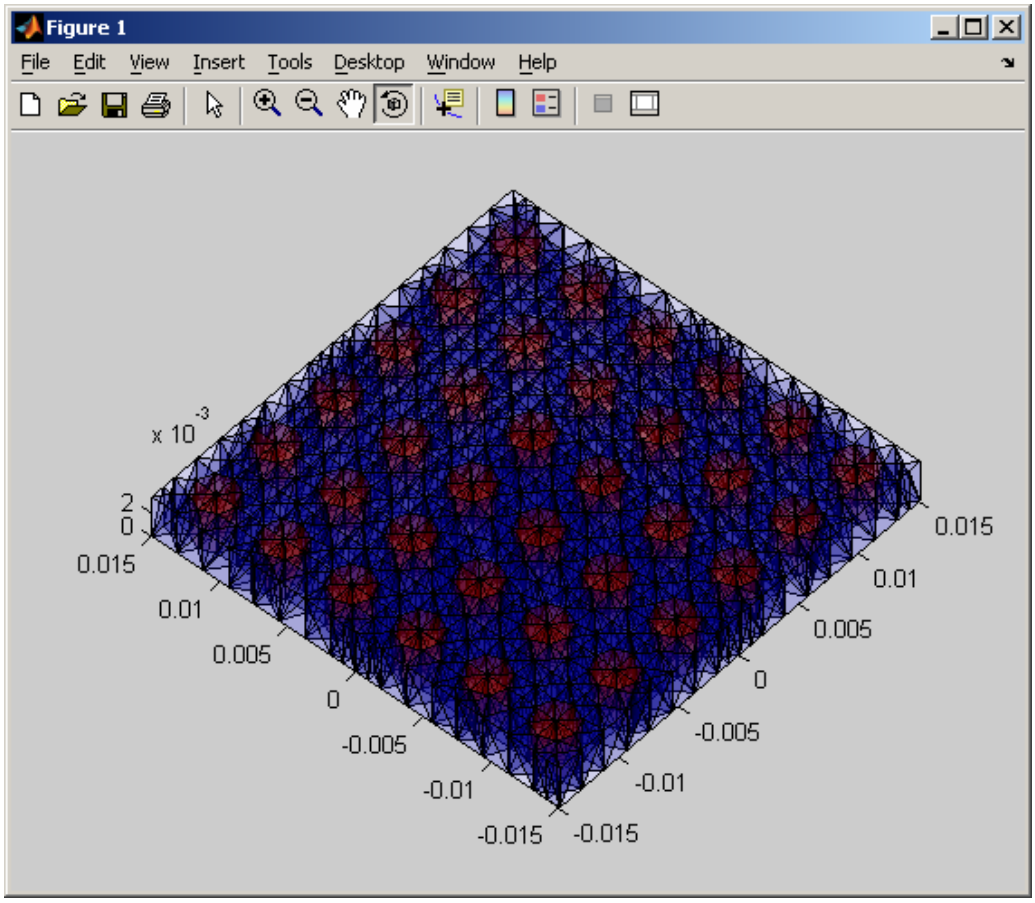

Fig. 1.15. The textured dielectric substrate.

1.12. Antenna optimization

The following optimization operations are straightforward (see Appendix B):

1. Optimization loop with regard to the value of the dielectric constant/loss tangent.
2. Loop over the overall geometry size.
3. Loop over the antenna size in one dimension.

Other operations may be performed by creating multiple meshes manually first and then saving them as, say, `struct3d1.mat`, …, `struct3d20.mat`, in folder `1_mesh`. Then a loop similar to the loop given in Appendix B may be organized.

References

1. S. M. Rao, D. R. Wilton, and A. W. Glisson, "Electromagnetic scattering by surfaces of arbitrary shape," *IEEE Trans. Antennas and Propagation*, vol. AP-30, no. 3, pp. 409-418, May 1982.
2. S. A. de Carvalho and L. de Souza Mendes, "Scattering of EM waves by inhomogeneous dielectrics with the use of the method of moments and the 3-D solenoidal basis functions", *Microwave and Optical Technology Letters*, vol. 23, No. 1, pp. 42-46, Oct. 1999.
3. S. Kulkarni, R. Lemdiasov, R. Ludwig, and S. Makarov, "Comparison of two sets of low-order basis functions for tetrahedral VIE modeling," *IEEE Trans. Antennas and Propagation*, vol. AP-52, no. 10, pp. 2789-2795, Oct. 2004.
4. S. Kulkarni, S. Uy, R. Lemdiasov, R. Ludwig, and S. Makarov, "MoM VIE solution for an isolated metal-dielectric resonator with the zeroth-order edge-based basis functions," *IEEE Trans. Antennas and Propagation*, vol. AP-53, no. 4, pp. 1566-1571, April 2005.
5. S. Makarov, *Antenna and EM modeling in MATLAB*, Wiley, 2002.
6. D. Psychoudakis, Y.-H. Koh, J.L. Volakis, and J. H. Halloran, "Design method for aperture-coupled microstrip patch antennas on textured dielectric substrates," *IEEE Trans. Antennas and Propagation*, vol. AP-52, no. 10, pp. 2763-2766, Oct. 2004.