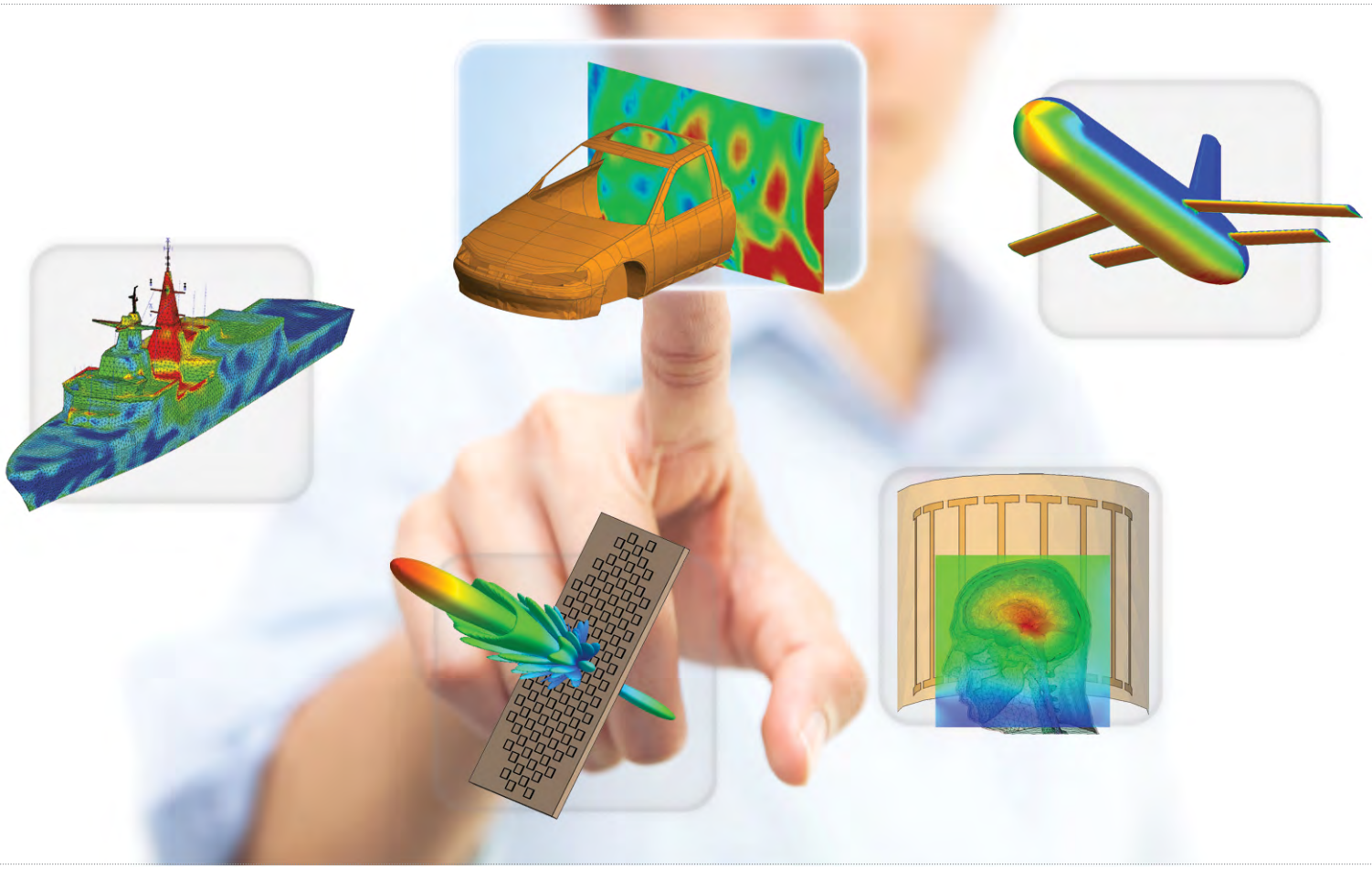
Altair | HyperWorks®

A Platform for Innovation™



Examples Guide for FEKO 14.0

FEKO

Comprehensive Electromagnetic Solutions

Examples Guide

14.0

December 2015

Altair® HyperWorks® Version 14.0
A Platform for Innovation®

Altair | HyperWorks®

Altair Engineering Support Contact Information

Web site
http://www.altairhyperworks.com
http://www.altairhyperworks.com/ClientCenterHWSupportProduct.aspx
http://www.altairhyperworks.com/feko

Third party software licenses

AcuConsole contains material licensed from Intelligent Light (www.ilight.com) and used by permission.

Software Security Measures:

Altair Engineering Inc. and its subsidiaries and affiliates reserve the right to embed software security mechanisms in the Software for the purpose of detecting the installation and/or use of illegal copies of the Software. The Software may collect and transmit non-proprietary data about those illegal copies. Data collected will not include any customer data created by or used in connection with the Software and will not be provided to any third party, except as may be required by law or legal process or to enforce our rights with respect to the use of any illegal copies of the Software. By using the Software, each user consents to such detection and collection of data, as well as its transmission and use if an illegal copy of the Software is detected. No steps may be taken to avoid or detect the purpose of any such security mechanisms.

Contents

Introduction 1

A Antenna synthesis + analysis

A-1 Dipole example A-1-1
A-2 Dipole in front of a cube A-2-1
A-3 Dipole in front of a plate A-3-1
A-4 Monopole antenna on a finite ground plane A-4-1
A-5 Yagi-Uda antenna above a real ground A-5-1
A-6 Pattern optimisation of a Yagi-Uda antenna A-6-1
A-7 Log periodic antenna A-7-1
A-8 Microstrip patch antenna A-8-1
A-9 Proximity coupled patch antenna with microstrip feed A-9-1
A-10 Modelling an aperture coupled patch antenna A-10-1
A-11 Different ways to feed a horn antenna A-11-1
A-12 Dielectric resonator antenna on finite ground A-12-1
A-13 A lens antenna with ray launching geometrical optics (RL-GO) A-13-1
A-14 Windscreen antenna on an automobile A-14-1
A-15 Design of a MIMO elliptical ring antenna (characteristic modes) A-15-1
A-16 Periodic boundary conditions for array analysis A-16-1
A-17 Finite array with non-linear element spacing A-17-1

B Antenna placement

B-1 Antenna coupling on an electrically large object B-1-1
B-2 Antenna coupling using an ideal receiving antenna B-2-1
B-3 Using a point source and ideal receiving antenna B-3-1

C Radar cross section (RCS)

C-1 RCS of a thin dielectric sheet C-1-1
C-2 RCS and near field of a dielectric sphere C-2-1
C-3 Scattering width of an infinite cylinder C-3-1
C-4 Periodic boundary conditions for FSS characterisation C-4-1

D EMC analysis + cable coupling

D-1 Shielding factor of a sphere with finite conductivity D-1-1

D-2 Calculating field coupling into a shielded cable D-2-1
D-3 A magnetic-field probe D-3-1
D-4 Antenna radiation hazard (RADHAZ) safety zones D-4-1

E Waveguide + microwave circuits
E-1 Microstrip filter E-1-1
E-2 S-parameter coupling in a stepped waveguide section E-2-1
E-3 Using a non-radiating network to match a dipole antenna E-3-1
E-4 Subdividing a model using non-radiating networks E-4-1
E-5 Microstrip coupler E-5-1

F Bio electromagnetics
F-1 Exposure of muscle tissue using MoM/FEM hybrid F-1-1
F-2 Magnetic Resonance Imaging (MRI) birdcage head coil example F-2-1

G Time domain examples
G-1 Time analysis of the effect of an incident plane wave on an obstacle G-1-1

H Special solution methods
H-1 Forked dipole antenna (continuous frequency range) H-1-1
H-2 Using the MLFMM for electrically large models H-2-1
H-3 Horn feeding a large reflector H-3-1
H-4 Optimise waveguide pin feed location H-4-1

I User interface tools
I-1 Introduction to application automation I-1-1
I-2 POSTFEKO application automation I-2-1
I-3 Matching circuits generation with Optenni Lab I-3-1
I-4 Using HyperStudy with FEKO to optimise a bandpass filter I-4-1

J Index
Index I-1

Introduction

This *Examples guide* presents a set of simple examples which demonstrate a selection of the features of FEKO. The examples have been selected to illustrate the features without being unnecessarily complex or requiring excessive run times. The input files for the examples can be found in the `examples/ExampleGuide_models` directory under the FEKO installation. No results are provided for these examples and in most cases, the `*.pre`, `*.cfm` and/or `*.opt` files have to be generated by opening and re-saving the provided project files (`*.cfx`) before the computation of the results can be initiated by running the FEKO preprocessor, solver or optimiser.

FEKO can be used in one of three ways. The first and recommended way is to construct the entire model in the CADFEKO user interface. The second way is to use CADFEKO for the model geometry creation and the solution setup and to use scripting for advanced options and adjustment of the model (for example the selection of advanced preconditioner options). The last way is to use the scripting for the entire model geometry and solution setup.

In this document the focus is on the recommended approaches (primarily using the CADFEKO user interface with no scripting).

Examples that employ only scripting are discussed in the *Script Examples* guide. These examples illustrate similar applications and methods to the examples in the *Examples guide* and it is highly recommended that you only consider the *Script Examples* if scripting-only examples are specifically required. It is advisable to work through the *Getting Started Guide* and familiarise yourself with the *Working with EDITFEKO* section in the *User Manual* before attempting the scripting only examples.

What to expect

The examples have been chosen to demonstrate how FEKO can be used in a selection of applications with a selection of the available methods.

Though information regarding the creation and setup of the example models for simulation is discussed, these example descriptions are not intended to be complete step-by-step guides that will allow exact recreation of the models for simulation. This document rather presents a guide that will help the user to discover and understand the concepts involved in various applications and methods that are available in FEKO, while working with the provided models.

In each example, a short description of the problem is given, then the model creation is discussed after which the relevant results are presented.

More examples

This set of examples demonstrates the major features of FEKO. For more step-by-step examples, please consult the *Getting started* guide. Also consult the FEKO website[1] for more examples and models, specific documentation and other FEKO usage FAQ's and tips.

[1] www.feko.info

Chapter A

Antenna synthesis + analysis

A-1 Dipole example

Keywords: dipole, radiation pattern, far field, input impedance

This example demonstrates the calculation of the radiation pattern and input impedance for a simple half-wavelength dipole, shown in Figure A-1-1. The wavelength, λ, is 4 m ($\approx$ 75 MHz), the length of the antenna is 2 m and the wire radius is 2 mm.

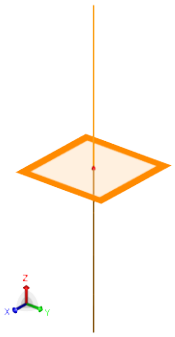


Figure A-1-1: A 3D view of the dipole model with a voltage source, symmetry and the far field pattern to be calculated in CADFEKO are shown.

A-1.1 Dipole

Creating the model

The steps for setting up the model are as follows:

- Define the following variables:
  - `lambda = 4` (Free space wavelength.)
  - `freq = c0/lambda` (Operating frequency.)
  - `h = lambda/2` (Length of the dipole.)
  - `radius = 2E-3` (Radius of the wire.)
- Create a line primitive with the start and end coordinates of (0,0,-h/2) and (0,0,h/2).
- Define a wire vertex port at the centre of the line.
- Add a voltage source to the wire port.
- Set the frequency to the defined variable `freq`.

Requesting calculations

This problem is symmetric around the Z=0 plane. All electric fields will be normal to this plane, and therefore the symmetry is electrical.

The solution requests are:

- Create a vertical far field request. (-180°≤θ≤180°, with ϕ=0° where θ and ϕ denotes the angles theta and phi). Sample the far field at 2° steps.

Meshing information

Use the *standard* auto-mesh setting with wire segment radius equal to `radius`.

CEM validate

After the model has been meshed, run *CEM validate*. Take note of any warnings and errors. Correct any errors before running the FEKO solution kernel.

A-1.2 Results

A polar plot of the gain (in dB) of the requested far field pattern is shown in Figure A-1-2. Click *Axis settings* (*Axes* group) to set the maximum dynamic range of the radial axis to 10 dB.

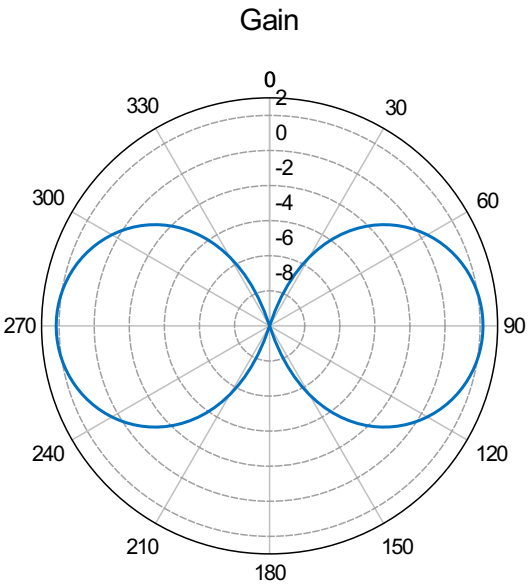


Figure A-1-2: A polar plot of the requested far field gain (dB) viewed in POSTFEKO.

Since the impedance is only calculated at a single frequency it can easily be read from the `*.out` file. The `*.out` file can be viewed in POSTFEKO's *Out file* viewer, or in any other text file viewer. An extract is shown below.

```
                        DATA OF THE VOLTAGE SOURCE NO. 1

                    real part  imag. part   magnitude       phase
Current   in A    1.1241E-02 -4.6364E-03  1.2160E-02      -22.41
Admitt.   in A/V  1.1241E-02 -4.6364E-03  1.2160E-02      -22.41
Impedance in Ohm  7.6025E+01  3.1356E+01  8.2237E+01       22.41
Inductance  in H  6.6585E-08
```

Alternatively, the impedance can be plotted as a function of frequency on a Cartesian graph or Smith chart in POSTFEKO.

A-2 Dipole in front of a cube

Keywords: dipole, PEC, metal, lossy, dielectric

A half wavelength dipole is placed three quarters of a wavelength away from a cube. The radiation pattern is calculated and the effect of the nearby cube on the radiation pattern is demonstrated. Three different cubes are modelled in this example. The first cube is PEC (perfect electrically conducting), the second is a metal cube that has a finite conductivity and the third cube is made as a solid dielectric material.

The second and third models are an extension of the first model. The examples should be set up sequentially.

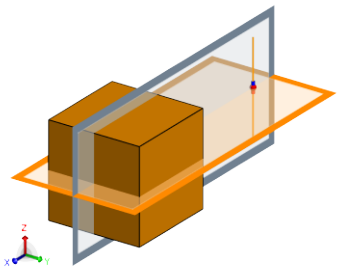


Figure A-2-1: A 3D view of the dipole with a metallic cube model (symmetry planes shown).

A-2.1 Dipole and PEC cube

Creating the model

The steps for setting up the model are as follows:

- Define the following variables:
  - `lambda = 4` (Free space wavelength.)
  - `freq = c0/lambda` (Operating frequency.)
  - `h = lambda/2` (Length of the dipole.)
  - `radius = 2e-3` (Wire radius of dipole.)
- Create a cube. The cuboid is created with the *Base corner, width, depth, height* definition method. The base corner is at (0,-`lambda`/4,-`lambda`/4) and with the *width*, *depth* and *height* set equal to `lambda`/2. By default the cube will be PEC.
- Create a line between the points (0,0,`h`/2) and (0,0,-`h`/2). Place the wire (3/4)* `lambda` away from the cube by translating it by (3/4)*`lambda` in the negative X direction.
- Add a wire port at the centre of the line.
- Add a voltage source to the port.
- Set the frequency to the defined variable `freq`.

Requesting calculations

All electric fields will be tangential to the Y=0 plane, and normal to the Z=0 plane. An electric plane of symmetry is therefore used for the Z=0 plane, and a magnetic plane of symmetry for the Y=0 plane.

The solution requests are:

- A horizontal radiation pattern cut is calculated to show the distortion of the dipole's pattern due to the proximity of the cuboid. ($0° \leq \phi \leq 360°$ with $\theta = 90°$), with $\phi = 0°$ where θ and ϕ denotes the angles theta and phi.

Meshing information

- Use the *standard* auto-mesh setting.
- Wire segment radius: `radius`.

CEM validate

After the model has been meshed, run *CEM validate*. Take note of any warnings and errors. Correct any errors before running the FEKO solution kernel.

A-2.2 Dipole and lossy metal cube

The calculation requests and mesh settings are the same as in the previous model.

Extending the first model

The model is extended with the following steps performed sequentially:

- Create a metallic medium called `lossy_metal`. Set the conductivity of the metal to `1e2`.
- Set the region inside the cuboid to *free space*.
- Set lossy metal properties on the cuboid faces by right-clicking in the details tree. Set the *Face medium* to `lossy_metal` and the thickness to 0.005.

Meshing information

- Use the *standard* auto-mesh setting.
- Wire segment radius: `radius`.

A-2.3 Dipole and dielectric cube

The calculation requests are the same as in the previous model.

Extending the model

The model is extended with the following steps performed sequentially:

- Create a dielectric medium called `diel` and relative permittivity of 2.
- Set the region of the cuboid to `diel`.
- Set the face properties of the cuboid to *default*. This means that CADFEKO will decide the face medium based on the geometry.
- Delete the `lossy_metal` metallic medium.

Meshing information

- Use the *standard* auto-mesh setting.
- Wire segment radius: `radius`.

CEM validate

After the model has been meshed, run *CEM validate*. Take note of any warnings, notes and errors. Please correct error before running the FEKO solution kernel.

A-2.4 Comparison of the results

The gain (in dB) of all three models are shown on a polar plot in Figure A-2-2. We can clearly see the pronounced scattering effect of the PEC and lossy metal cube with almost no difference between their results.

We also see that the dielectric cube has a dramatic different effect. The dielectric cube results in a gain increase in the direction of the cube.

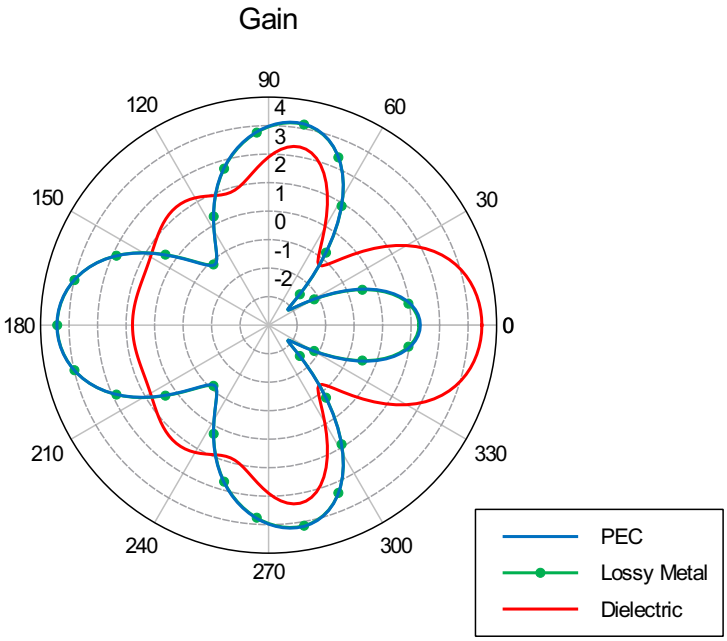


Figure A-2-2: A comparative polar plot of the requested far field gain in dB.

A-3 Dipole in front of a plate modelled using MoM, FDTD, UTD, RL-GO and PO

Keywords: MoM, HOBF, UTD, PO, LE-PO, RL-GO, FDTD, dipole, radiation pattern, far field, electrically large plate

A dipole in front of an electrically large square plate is considered. This simple example illustrates how to solve a model using different techniques. First the dipole and the plate is solved with the traditional MoM. The plate is then modified so that it can be solved with higher order basis function MoM, FDTD, UTD, RL-GO and later PO and LE-PO. The MoM/UTD, MoM/RL-GO, MoM/PO and MoM/LE-PO hybrid solutions demonstrated here are faster and require fewer resources than the traditional MoM solution. When applicable, these approximations can be used to reduce the required solution time and resources.

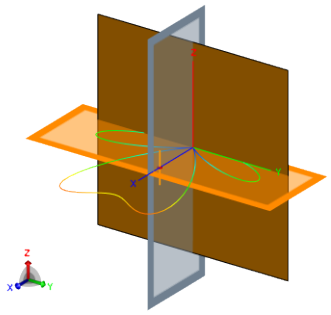


Figure A-3-1: A 3D view of the dipole in front of a metallic plate.

A-3.1 Dipole in front of a large plate

Creating the model

The steps for setting up the model are as follows:

- Define the following variables:
  - `d = 2.25` (Separation distance between dipole and plate. [3*lambda/4])
  - `h = 1.5` (Length of the dipole. [lambda/2])
  - `a = 4.5` (Half-side length of plate.)
  - `rho = 0.03` (Wire radius of the dipole.)
- The wire dipole is a distance `d` from the plate in the *U* axis direction. The dipole is `h` long and should be centred around the *U* axis. Create the dipole (line primitive) by entering the following 2 points: `(d,0,-h/2)`, `(d,0,h/2)`.
- Create the plate by:
  - Creating a rectangle with definition method: *Base centre, width, depth*. Enter the centre as (0,0,0) and the *width* = `2*a` and *depth* = `2*a`.

  - Rotating the workplane 90 degrees around the V axis.

- Add a segment port on the middle of the wire.
- Add a voltage source to the port. (1 V, 0°, 50 Ω)
- Set the total source power (no mismatch) to 1 W.
- Set the frequency to `c0/3`. The model is built relative to a chosen wavelength of $\lambda = 3$ m.

Requesting calculations

The model contains two planes of symmetry and may be added to accelerate the solution. A magnetic plane of symmetry is added on the Y=0 plane, and an electric plane of symmetry on the Z=0 plane.

Request a default horizontal far field cut of the far field ($0° \leq \phi \leq 360°$, $\theta = 90°$, 2° increments).

Meshing information

Use the *standard* auto-mesh setting with the wire radius set to `rho`.

CEM validate

After the model has been meshed, run *CEM validate*. Take note of any warnings and errors. Correct any errors before running the FEKO solution kernel.

A-3.2 Dipole and a plate using HOBF MoM

Creating the model

The model is identical to the traditional MoM model. The only change that is required is to enable HOBF functions for the solution.

HOBF is activated by navigating to the *Solver settings* dialog on the *Solve/Run* ribbon tab and activating the "*Solve with higher order basis functions (HOBF)*" check box. Also ensure the basis function order is set to auto.

Meshing information

Use the *standard* auto-mesh setting with the wire radius set to `rho`. After changing the solution method to use HOBF, the model must be remeshed since HOBF elements are larger than traditional MoM elements.

CEM validate

After the model has been meshed, run *CEM validate*. Take note of any warnings and errors. Correct any errors before running the FEKO solution kernel.

A-3.3 Dipole and a plate using FDTD

Creating the model

The model is identical to the traditional MoM model. The only change that is required is to enable FDTD by navigating to the *Solver settings* dialog on the *Solve/Run* ribbon tab.

Meshing information

Use the *standard* auto-mesh setting with the wire radius set to `rho`. After changing the solution method, the model must be remeshed to obtain a voxel mesh representation of the geometry.

CEM validate

After the model has been meshed, run *CEM validate*. Take note of any warnings and errors. Correct any errors before running the FEKO solution kernel.

A-3.4 Dipole and a UTD plate

Creating the model

The model is identical to the MoM model. The only change that is required is that the solution method to be used on the plate must be changed.

This change is made by going to the face properties of the plate in the detail tree of CADFEKO. On the solution tab use the dropdown box named *Solve with special solution method* and choosing *Uniform theory of diffraction (UTD)*. Now when meshing is done in CADFEKO the plate will not be meshed into triangular elements.

Remove the symmetry definitions for the UTD example - the number of elements is so small that it is faster to simulate without symmetry.

Meshing information

Use the *standard* auto-mesh setting with the wire radius set to `rho`. After changing the solution method on the plate to UTD, the model must be remeshed. UTD plates are not meshed and a single element will be created for the entire plate.

CEM validate

After the model has been meshed, run *CEM validate*. Take note of any warnings and errors. Correct any errors before running the FEKO solution kernel.

A-3.5 Dipole and a RL-GO plate

Creating the model

The model is identical to the MoM model (note that we are changing the MoM model and not the UTD model). The only change that is required is that the solution method to be used on the plate must be changed.

This change is made by going to the face properties of the plate in the detail tree of CADFEKO. On the solution tab use the dropdown box named *Solve with special solution method* and choosing *Ray launching - geometrical optics (RL-GO)*.

Meshing information

Use the *standard* auto-mesh setting with the wire radius set to `rho`. After changing the solution method on the plate to RL-GO, the model must be remeshed. The triangle sizes are determined by the geometrical shape and not the operating wavelength. Unlike the UTD plate, the plate will be meshed into triangular elements for the RL-GO.

CEM validate

After the model has been meshed, run *CEM validate*. Take note of any warnings and errors. Correct any errors before running the FEKO solution kernel.

A-3.6 Dipole and a PO plate

Creating the model

The model is identical to the MoM model. The only change that is required is that the solution method to be used on the plate must be changed.

This change is made by going to the face properties of the plate in the detail tree of CADFEKO. On the solution tab use the dropdown box named *Solve with special solution method* and choosing *Physical optics (PO) - always illuminated*. The 'always illuminated' option may be used in this case, as it is clear that there will be no shadowing effects in the model. With this option, the ray-tracing required for the physical optics solution can be avoided thereby accelerating the solution.

Meshing information

Use the *standard* auto-mesh setting with the wire radius set to `rho`. The auto-mesh feature takes the solution method into account.

CEM validate

After the model has been meshed, run *CEM validate*. Take note of any warnings and errors. Correct any errors before running the FEKO solution kernel.

A-3.7 Dipole and a LE-PO plate

Creating the model

The model is identical to the MoM model. The only change that is required is that the solution method to be used on the plate must be changed.

This change is made by going to the face properties of the plate in the detail tree of CADFEKO. On the solution tab use the dropdown box named *Solve with special solution method* and choosing *Large element (PO) - always illuminated*. The 'always illuminated' option may be used in this case, as it is clear that there will be no shadowing effects in the model. With this option, the ray-tracing required for the physical optics solution can be avoided thereby accelerating the solution.

Meshing information

Set the mesh size to *custom*:

- Triangle edge length: `a/4`
- Wire segment length: `h/10`

CEM validate

After the model has been meshed, run *CEM validate*. Take note of any warnings and errors. Correct any errors before running the FEKO solution kernel.

A-3.8 Comparative results

The total far field gain of the dipole in front of the PEC plate is shown on a dB polar plot in Figure A-3-2. The MoM/UTD, MoM/PO and full MoM reference solution are shown. To obtain the large element PO (LE-PO) models, set the solution method from *Physical optics (PO) - always illuminated* to *Large element physical optics (LE-PO) - always illuminated* and save the PO example files under a new name. Since the dipole is less than a wavelength away from the plate, the standard auto-meshing will not work. Mesh the plate with a triangle edge length of `a/4`.

The comparison between memory requirements and runtimes are shown in Table A-3-1. The method of moments (MoM) is used as reference and all other methods are compared using a memory and runtime factor. Requirements for the MoM solution was 10 s and 51.88 MB.

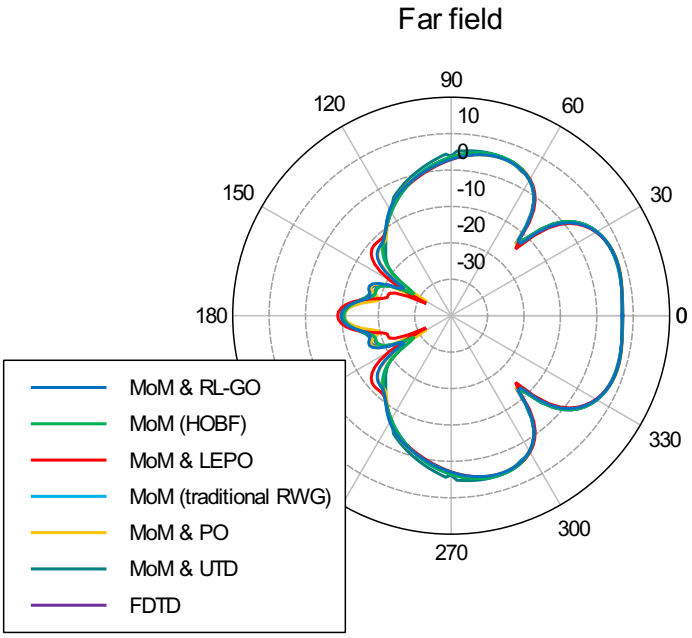


Figure A-3-2: A polar plot of the total far field gain (dB) computed in the horizontal plane using the MoM/RL-GO, MoM/UTD, MoM/PO, MoM/LE-PO and FDTD methods compared to the full MoM (traditional and HOBF) reference solution.

Solution method	*Memory (% of MoM)*	*Runtime (% of MoM)*
MoM (HOBF - auto)	5.6	143.5
Physical optics: MoM/PO	1.9	9.4
Large element PO: MoM/LE-PO	0.3	2.6
Uniform theory of diffraction: MoM/UTD	0.1	1.3
Ray launching geometrical optics: MoM/RL-GO	14.2	26.9
FDTD	104.1	490.3

Table A-3-1: Comparison of resource requirements for the model using different solver techniques.

A-4 Monopole antenna on a finite ground plane

Keywords: monopole, finite ground, radiation pattern, far field, current

A quarter wave monopole antenna on a finite circular ground plane is constructed and simulated. The circular ground has a circumference of three wavelengths, and the wire has a radius of 1 mm. The operating frequency is 75 MHz.

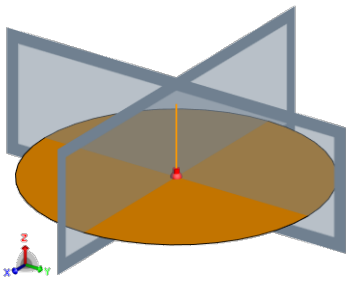


Figure A-4-1: A 3D view of the monopole on a finite circular ground (symmetry planes shown).

A-4.1 Monopole on a finite ground

Creating the model

The steps for setting up the model are as follows:

- Define the following variables:
  - `freq = 75e6` (Operating frequency.)
  - `lambda = c0/freq` (Free space wavelength.)
  - `groundRadius = 2` (Radius of the ground plane.)
  - `wireRadius = 1e-3` (Radius of the monopole wire segments.)
- Create the ground using the ellipse primitive. Set the radii equal to the defined variable `groundRadius` and the label to *ground*.
- Create a line between (0,0,0) and (0,0,`lambda/4`) and rename as `monopole`.
- Union the wire and the ground.
- Add a wire vertex port on the line. The port preview should show the port located at the junction between the wire and the ground plane. If this is not so, change the port position between *Start* and *End*.
- Add a voltage source to the port. (1 V, 0°, 50 Ω)
- Set the frequency equal to `freq`.

Requesting calculations

Two planes of magnetic symmetry are defined at the X = 0 plane and the Y = 0 plane.

The solution requests are:

- A full 3D far field pattern with 2° increments.
- All currents are saved to allow viewing in POSTFEKO.

Meshing information

- Use the *standard* auto-mesh setting.
- Wire segment radius: `wireRadius`.

CEM validate

After the model has been meshed, run *CEM validate*.

A-4.2 Results

A polar plot of the total gain in a vertical cut is shown in Figure A-4-2.

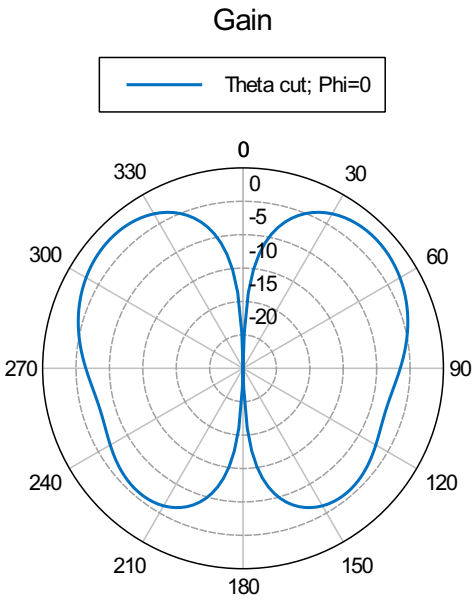


Figure A-4-2: Polar plot of the total gain in a vertical cut.

The full 3D gain pattern is depicted in Figure A-4-3. Since the antenna has an omnidirectional pattern in the ϕ plane, the value of ϕ can be coarser. The far field gain is shown slightly transparent in the figure to allow for visibility of the geometry and the curve of the far field pattern.

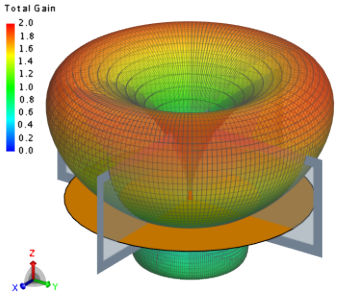


Figure A-4-3: A full 3D plot of the antenna gain.

The currents on all elements (wire segment and surface triangles) are shown in Figure A-4-4. The currents are indicated by the geometry colouring based on the legend colour scale. This allows identification of points where the current is concentrated. The surface currents are displayed in decibels.

The phase evolution of the current display may be animated (as with many other results displays in POSTFEKO) on the *Animate* tab on the ribbon.

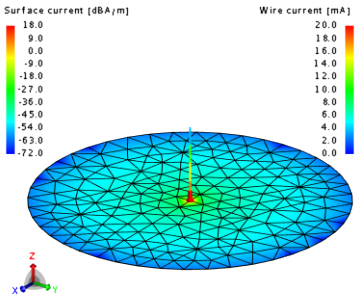


Figure A-4-4: 3D view of the current on the ground plane of the monopole antenna.

A-5 Yagi-Uda antenna above a real ground

Keywords: antenna, Yagi-Uda antenna, real ground, infinite planar Green's function, optimisation

In this example we consider the radiation of a horizontally polarised Yagi-Uda antenna consisting of a dipole, a reflector and three directors. The frequency is 400 MHz. The antenna is located 3 m above a real ground which is modelled with the Green's function formulation.

Note that the model provided with this example includes a basic optimisation. The optimisation is set up such that the optimal dimensions of the antenna may be determined to achieve a specific gain pattern (maximise the forward gain and minimise back lobes).

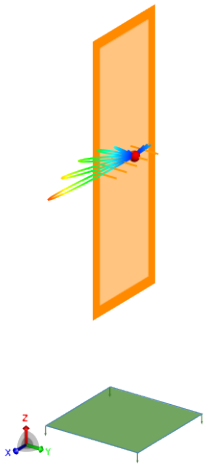


Figure A-5-1: A 3D view of the Yagi-Uda antenna suspended over a real ground.

A-5.1 Antenna and ground plane

Creating the model

The steps for setting up the model are as follows:

- Define the following variables:
  - `freq = 400e6` (Operating frequency.)
  - `lambda = c0/freq` (The wavelength in free space at the operating frequency.)
  - `lr = 0.477*lambda` (Length of the reflector.)
  - `li = 0.451*lambda` (Length of the active element.)
  - `ld = 0.442*lambda` (Length of the directors.)
  - `d = 0.25*lambda` (Spacing between elements.)
  - `h = 3` (Height of the antenna above ground.)
  - `epsr = 10` (Relative permittivity of the ground.)

  - `sigma = 1e-3` (Ground conductivity.)
  - `wireRadius = 1e-3` (Wire radius (1 mm).)

- Create the active element with *start point* as (0, -`li`/2, `h`) and the *end point* as (0, `li`/2, `h`). Set the label as `activeElement`.
- Add a vertex port in the centre of the wire.
- Add a voltage source on the port. (1 V, 0°, 50 Ω)
- Create the wire for the reflector. Set the *Start point* as (-d, -`lr`/2, `h`) and the *End point* as (-d, `lr`/2, `h`). Set the label as `reflector`.
- Create the three wires for the directors.

Director	*Start point*	*End point*
`director1`	(d, -`ld`/2, `h`)	(d, `ld`/2, `h`)
`director2`	(2*d, -`ld`/2, `h`)	(2*d, `ld`/2, `h`)
`director3`	(3*d, -`ld`/2, `h`)	(3*d, `ld`/2, `h`)

- Create a dielectric called `ground` with relative permittivity of `epsr` and conductivity equal to `sigma`.
- Set the lower half space to `ground`. This can be done by setting the infinite plane to use the exact Sommerfeld integrals.
- Set the frequency to `freq`.

Requesting calculations

A single plane of electrical symmetry on the Y = 0 plane is used in the solution of this problem.

The solution requests are:

- Create a vertical far field request above the ground plane. (-90°≤θ≤90°, with ϕ=0 and θ=0.5° increments)
- Set the *Workplane* origin of the far field request to (0, 0, 3).

Meshing information

Use the standard auto-meshing option with the wire segment radius equal to `wireRadius`.

CEM validate

After the model has been meshed, run *CEM validate*. Take note of any warnings and errors. Correct any errors before running the FEKO solution kernel.

Note that a warning may be encountered when running the solution. This is because losses cannot be calculated in an infinitely large medium, as is required for the extraction of antenna directivity information (gain is computed by default). This warning can be avoided by ensuring that the far field gain be calculated instead of the directivity. This is set on the *Advanced* tab of the far field request in the tree.

A-5.2 Results

The radiation pattern is calculated in the H plane of the antenna. A simulation without the ground plane is compared with the results from the model provided for this example in Figure A-5-2. As expected, the ground plane greatly influences the radiation pattern. (Note that the graph is a vertical polar plot of the gain in dB for the two cases.)

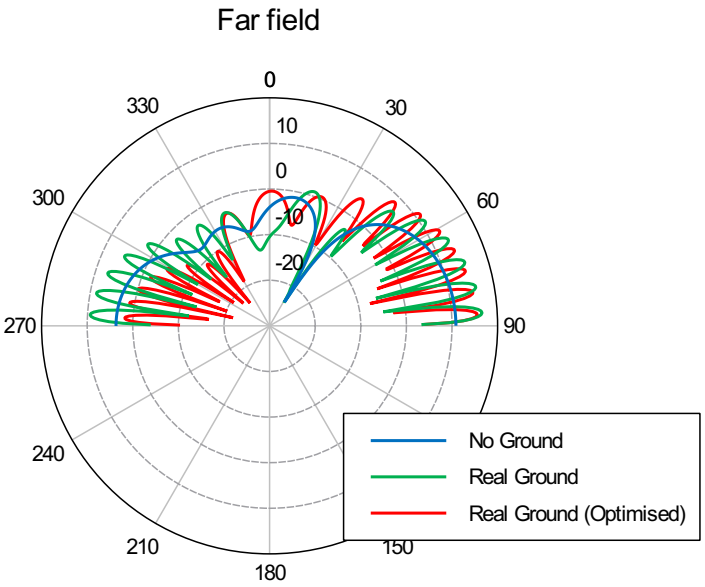


Figure A-5-2: The directivity pattern of the Yagi-Uda antenna over a real ground and without any ground. Note that the optimised pattern is also shown.

A-6 Pattern optimisation of a Yagi-Uda antenna

Keywords: antenna, Yagi-Uda, radiation pattern, optimisation

In this example we consider the optimisation of a Yagi-Uda antenna (consisting of a dipole, a reflector and two directors) to achieve a specific radiation pattern and gain requirement. The frequency is 1 GHz. The antenna has been roughly designed from basic formulae, but we would like to optimise the antenna radiation pattern such that the gain is above 8 dB in the main lobe (-30° $\leq \phi \leq$ 30°) and below -7 dB in the back lobe (90° $\leq \phi \leq$ 270°).

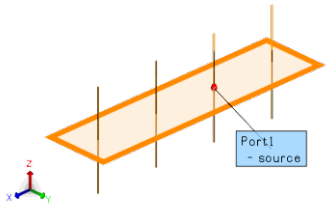


Figure A-6-1: A 3D view of the Yagi-Uda antenna.

A-6.1 The antenna

Creating the model

The steps for setting up the model are as follows:

- Define the following variables (physical dimensions based on initial rough design):
  - `freq = 1e9` (The operating frequency.)
  - `lambda = c0/freq` (The wavelength in free space at the operating frequency.)
  - `L0 = 0.2375` (Length of one arm of the reflector element in wavelengths.)
  - `L1 = 0.2265` (Length of one arm of the driven element in wavelengths.)
  - `L2 = 0.2230` (Length of one arm of the first director in wavelengths.)
  - `L3 = 0.2230` (Length of one arm of the second director in wavelengths.)
  - `S0 = 0.3` (Spacing between the reflector and driven element in wavelengths.)
  - `S1 = 0.3` (Spacing between the driven element and the first director in wavelengths.)
  - `S2 = 0.3` (Spacing between the two directors in wavelengths.)
  - `r = 1e-4` (Radius of the elements.)
- Create the active element of the Yagi-Uda antenna. Set the *start point* as (0, 0, `-L1*lambda`) and the *end point* as (0, 0, `L1*lambda`).
- Add a port on a segment in the centre of the wire.
- Add a voltage source on the port. (1 V, 0°, 50 Ω)

- Set the incident power for a 50 Ω transmission line to 1 W.
- Create the wire for the reflector. Set the *start point* as (-S0*lambda, 0, -L0*lambda) and the *end point* as (-S0*lambda, 0, L0*lambda).
- Create the two directors:
  - Set the *start point* and *end point* for director1 as the following: (S1*lambda, 0, -L2*lambda) and (S1*lambda, 0, L2*lambda), respectively.
  - For director2, set the *start point* as ((S1 + S2)*lambda, 0, -L3*lambda) and the *end point* ((S1 + S2)*lambda, 0, L3*lambda).
- Set the frequency to freq.

Requesting calculations

The Z=0 plane is an electric plane of symmetry.

A magnetic plane of symmetry exists in the Y=0 plane, but since all the wires are in the Y=0 plane, adding the magnetic symmetry setting would not affect the simulation speed and can be neglected.

The solution requests are:

- Create a horizontal far field request labelled 'H_plane'. ($0° \leq \phi \leq 180°$, θ=90 and 2° increments)

Meshing information

Use the *standard* auto-mesh setting with the wire segment radius equal to r.

Setting up optimisation

- An optimisation search is added with the Simplex method and Low accuracy.
- The following parameters are set:
  - L0 (min 0.15; max 0.35; start 0.2375)
  - L1 (min 0.15; max 0.35; start 0.2265)
  - L2 (min 0.15; max 0.35; start 0.22)
  - L3 (min 0.15; max 0.35; start 0.22)
  - S0 (min 0.1; max 0.32; start 0.3)
  - S1 (min 0.1; max 0.32; start 0.3)
  - S2 (min 0.1; max 0.32; start 0.3)
- For this example, it is required that the reflector element be longer than all the director elements. The following constraints are therefore also defined:

  - L2 < L0
  - L3 < L0

- Two optimisation masks are created (see figure A-6-2. The first mask (`Mask_max`) defines the upper limit of the required gain (gain < 15 between 0° and 88°; gain < -7 between 90° and 180°). The purpose of this mask is to define the region that defines the upper boundary. The value of 15 dB in the forward direction was chosen arbitrarily high knowing that this antenna will not be able to achieve 15 dB gain and thus does not have an affect on the optimisation. Upper limit of -7 dB from 90° to 180° will have an effect on the optimisation and determines the size of the back lobes that we are willing to accept.
- The second mask (`Mask_min`) defines the lower limit of the required gain (gain > 8 dB between 0° and 30°; gain > -40 dB between 32° and 180°). This mask is used to determine the desired main lobe gain. The value of -40 dB outside the main lobe was chosen arbitrarily low and thus will not affect the optimisation.
- Two far field optimisation goals are added based on the `H_plane` calculation request. The dB values ($10 * log[]$) of the vertically polarised gain at all angles in the requested range is required to be greater than `Mask_min` and less than `Mask_max`.

A weighting for both goals are equal since neither of the goals are more important than the other. The weighting that should be used depends on the goal of the optimisation.

A-6.2 Results

The radiation pattern (calculated in the E plane of the antenna) is shown in Figure A-6-2 for both the initial design and the antenna resultant after the optimisation process. The gain in the back-lobe region (between 90 and 180 degrees) has been reduced to around -7 dB, while the gain over the main-lobe region (between 0 and 30 degrees) is above 8 dB. (Note that the graph shows the vertically polarised gain plotted in dB with respect to ϕ.)

The extract below from the optimisation log file, indicates the optimum parameter values found during the optimisation search:

```
Optimisation finished (Standard deviation small enough:   9.759663708e-03)

Optimum found for these parameters:
  l0                            =   2.475783864e-01
  l1                            =   1.834495518e-01
  l2                            =   2.258108451e-01
  l3                            =   2.155000036e-01
  s0                            =   2.047418973e-01
  s1                            =   2.620084341e-01
  s2                            =   3.193665629e-01

Optimum aim function value (at no. 140):   7.191160225e-02
No. of the last analysis: 144
```

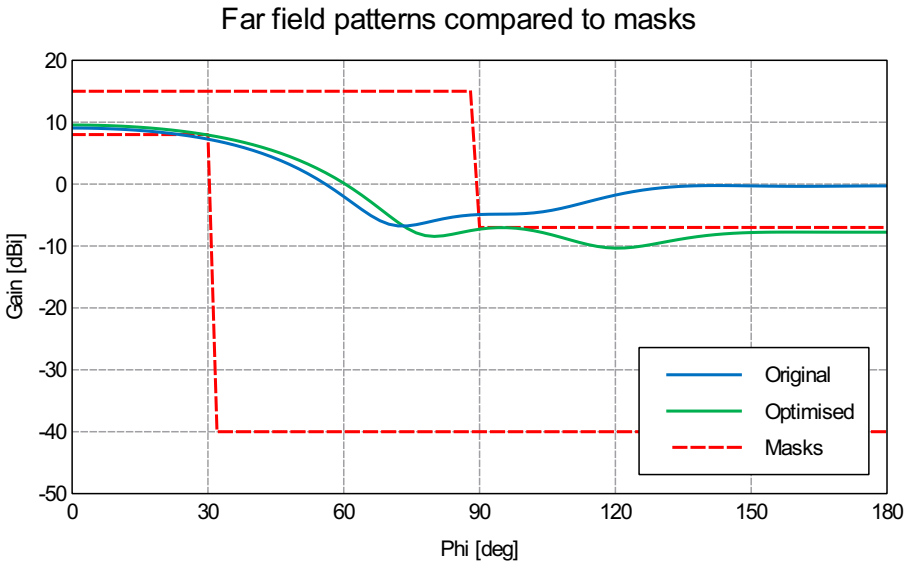


Figure A-6-2: The vertically polarised gain of the Yagi-Uda antenna before and after optimisation.

A-7 Log periodic antenna

Keywords: Transmission line, dipole, array, far field

A log periodic example uses the non-radiating transmission lines to model the boom of a log periodic dipole array antenna. The antenna is designed to operate around 46.29 MHz, with an operational bandwidth over a wide frequency range (35 MHz to 60 MHz).

Figure A-7-1 shows the log periodic dipole array (LPDA) with a transmission line feed network.

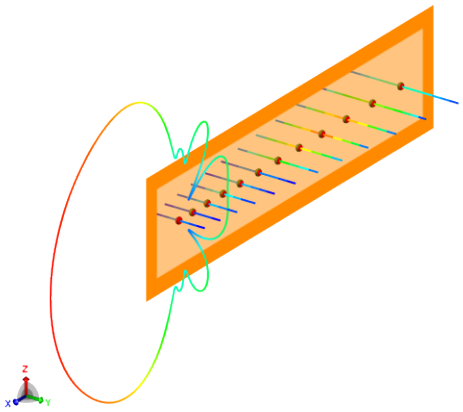


Figure A-7-1: The model of LPDA using transmission lines to model the boom structure.

A-7.1 Log periodic dipole array

Creating the model

The steps for setting up the model are as follows:

- Create the variables required for the model.
  - `freq` = `46.29e6` (The operating frequency.)
  - `tau` = `0.93` (The growth factor.)
  - `sigma0` = `0.7` (Spacing)
  - `len0` = `2` (Length of the first element.)
  - `d0` = `0` (Position of the first element.)
  - `rad0` = `0.00667` (Radius of the first element.)
  - `sigma1...sigma11`: `sigmaN` = `sigma(N-1)/tau`
  - `d1...d11`: `dN` = `d(N-1)` – `sigmaN`
  - `len1...len11`: `lenN` = `len(N-1)/tau`
  - `rad1...rad11`: `radN` = `rad(N-1)/tau`

  - `lambda = c0/freq` (Free space wavelength.)
  - `Zline = 50` (Transmission line impedance.)
  - `Zload = 50` (Shunt load resistance.)

- Create the twelve dipoles using the defined variables. Create line (geometry) number N from `(dN,-lenN/2,0)` to `(dN,lenN/2,0)`. For example, to create dipole 1, create a line from `(d1,-len1/2,0)` to `(d1,len1/2,0)`.
- Add a port in the centre of every dipole.
- Define eleven transmission lines to connect the dipoles. Each transmission line has a characteristic impedance of `Zline` (real part of Z0 (Ohm)) and a transmission line length of `sigmaN`. Check the *Cross input and output ports* to ensure correct orientation of the transmission line connections.
- For each segment set the local wire radius equal to the defined `radN` variable.
- Connect transmission line N between `port(N-1)` and `portN` for all of the transmission lines, see Figure A-7-2.
- Define the shunt load using the admittance definition of a general non-radiating network (Y-parameter). Specify the one-port admittance matrix manually (Y_{11} = `1/Zload`).
- Connect the general network to the final port (i.e. the port of the longest dipole element).
- Set the continuous (interpolated) frequency range from 35 MHz to 60 MHz.
- Add a voltage source to the port at the first dipole at the origin.

Note that it will be required to connect all of the ports, transmission lines and the network together in the schematic view.

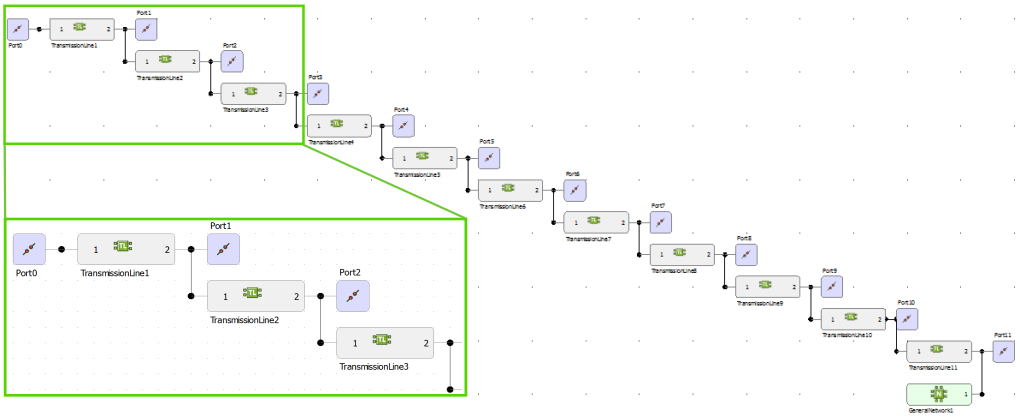


Figure A-7-2: The network schematic view showing the connected transmission lines, general networks and ports.

Requesting calculations

Electric symmetry may be applied to the plane at Y=0.

A far field pattern is requested in the vertical plane (-180°≤θ≤180°, with ϕ=0° and 2° increments).

Meshing information

Use the *standard* auto-mesh setting with wire segment radius equal to `0.01`. Note that all wires have local radii set to `radN`.

CEM validate

After the model has been meshed, run *CEM validate*. Take note of any warnings and errors. Correct any errors before running the FEKO solution kernel.

Save the file and run the solver.

A-7.2 Results

The vertical gain (in dB) at 46.29 MHz and the input impedance over the operating band of the LPDA are shown in Figure A-7-3 and Figure A-7-4 respectively.

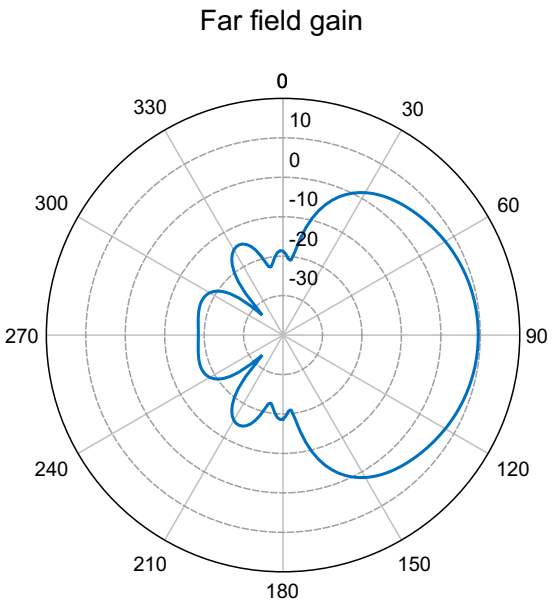


Figure A-7-3: The vertical gain of a LPDA antenna at 46.29 MHz.

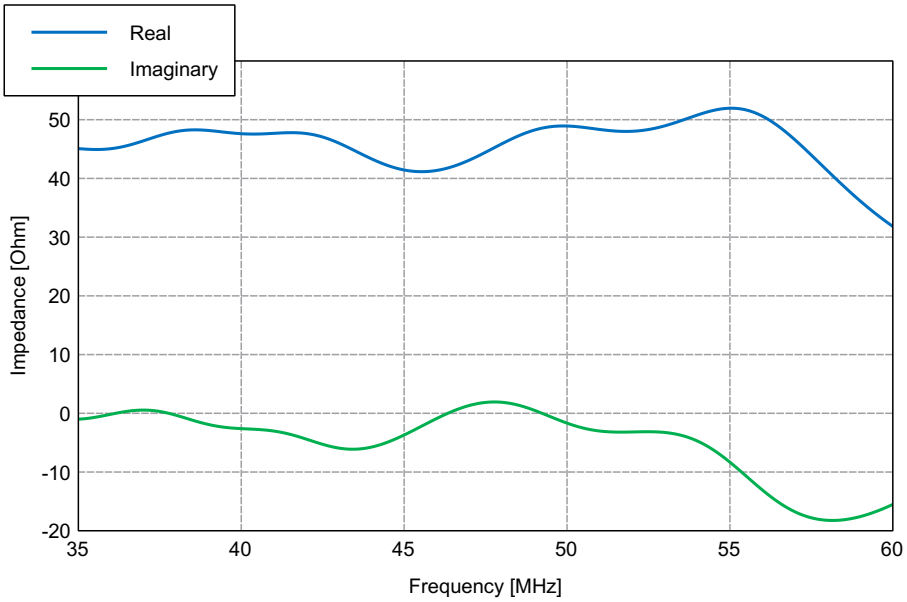


Figure A-7-4: The input impedance (real and imaginary) of the LPDA antenna over the operating band.

A-8 Microstrip patch antenna

Keywords: microstrip, patch antenna, dielectric substrate, pin feed, edge feed, optimisation, SEP, FDTD

A microstrip patch antenna is modelled using different feed methods. The dielectric substrate is modelled as a finite substrate and ground using the surface equivalence principle (SEP) as well as a planar multilayer substrate with a bottom ground layer (using a special Green's function). The simulation time and resource requirements are greatly reduced using an infinite plane, although the model may be less representative of the physical antenna. The two different feeding methods considered are a pin feed and a microstrip edge feed.

In this example, each model builds on the previous one. It is recommended the models be built and considered in the order they are presented. If you would like to build and keep the different models, start each model by saving the model to a new location.

Note that the model provided with this example for the pin-fed patch (SEP) on a finite substrate includes a basic optimisation setup. The optimisation is defined to determine the value for the pin offset which gives the best impedance match to a 50 Ω system.

A-8.1 Pin-fed, SEP model

Creating the model

In the first example a feed pin is used and the substrate is modelled with a dielectric with specified dimensions. The geometry of this model is shown in Figure A-8-1.

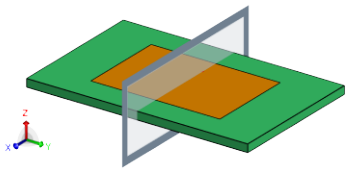


Figure A-8-1: A 3D representation of a pin fed microstrip patch antenna on a finite ground.

The steps for setting up the model are as follows: (Note that length is defined in the direction of the X axis and width in the direction of the Y axis.)

- Set the model unit to millimetres.
- Define the following variables (physical dimensions based on initial rough design):
  - `epsr = 2.2` (The relative permittivity of the substrate.)
  - `freq = 3e9` (The centre frequency.)
  - `lambda = c0/freq*1e3` (The wavelength in free space.)
  - `lengthX = 31.1807` (The length of the patch in the X direction.)

  - `lengthY = 46.7480` (The length of the patch in the Y direction.)
  - `offsetX = 8.9` (The location of the feed.)
  - `substrateLengthX = 50` (The length of the substrate in the X direction.)
  - `substrateLengthY = 80` (The length of the substrate in the Y direction.)
  - `substrateHeight = 2.87` (The height of the substrate.)
  - `fmin = 2.7e9` (The minimum frequency in the simulation range.)
  - `fmax = 3.3e9` (The maximum frequency in the simulation range.)
  - `feedlineWidth = 4.5` (The width of the feedline for the microstrip model feed.)

- Create the patch by creating a rectangle with the *Base centre, width, depth* definition method. Set the *Width* to the defined variable `lengthX` and *Depth* equal to `lengthY`. Rename this label to `patch`.
- Create the substrate by defining a cuboid with the *Base corner, width, depth, height* definition method. Set the *Base corner* to (-`substrateLengthX`/2, -`substrateLengthY`/2, -`substrateHeight`), *Width* = `substrateLengthX`, *Depth* = `substrateLengthY`, *Height* = `substrateHeight`). Rename this label to `substrate`.
- Create the feed pin as a wire between the patch and the bottom of the substrate positioned 8.9 mm (`offsetX`) from the edge of the patch. The pin's offset is in the *X* direction and should have no offset in the *Y* direction.
- Add a segment wire port on the middle of the wire.
- Add a voltage source on the port. (1 V, 0°, 50 Ω)
- Union all the elements and label the union `antenna`.
- Create a new dielectric called `substrate` with relative permittivity equal to `epsr`.
- Set region of the cube to `substrate`.
- Set the faces representing the patch and the ground below the substrate to PEC.
- Set a continuous frequency range from `fmin` to `fmax`.

Requesting calculations

A single plane of magnetic symmetry is used on the Y=0 plane.

The solution requests are:

- Create a vertical (E plane) far field request. (-90°$\leq\theta\leq$90°, with ϕ=0° and 2° increments)
- Create a vertical (H plane) far field request. (-90°$\leq\theta\leq$90°, with ϕ=90° and 2° increments)

Meshing information

Use the *standard* auto-mesh setting with the wire segment radius equal to `0.25`.

CEM validate

After the model has been meshed, run *CEM validate*.

A-8.2 Pin-fed, FDTD model

Creating the model

The model is now solved by means of the FDTD solver. It is still pin-fed as in the previous example.

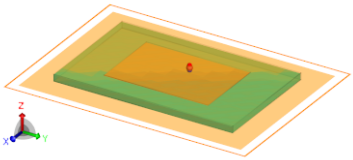


Figure A-8-2: A 3D representation of a pin fed microstrip patch antenna on a finite ground.

The model is extended with the following steps performed sequentially:

- Activate the finite difference time domain (FDTD) solver.
- Change the continuous frequency range to linearly spaced discrete points ranging from `fmin` to `fmax` with the number of frequencies set to `51`.
- Define the boundary condition settings. Set the *Top (+Z)*, *-Y*, *+Y*, *-X* and *+X* boundaries to *Open* and *Automatically add a free space buffer*. Set the *Bottom (-Z)* boundary to *Perfect electric conductor (PEC)* and *Do not add a free space buffer*.

Meshing information

Use the *standard* auto-mesh setting with the wire segment radius equal to `0.25`.

CEM validate

After the model has been meshed, run *CEM validate*.

A-8.3 Pin-fed, planar multilayer substrate

Creating the model

This model is an extensions of the first model. The substrate is now modelled with a planar multilayer substrate (Green's functions). It is still pin-fed as in the previous two examples.

The model is extended with the following steps performed sequentially:

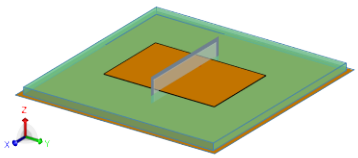


Figure A-8-3: A 3D representation of a pin fed microstrip patch antenna on an infinite ground.

- Delete the substrate component from the `antenna` geometry.
- Add a planar multilayer substrate (infinite plane) with a conducting layer at the bottom. *Layer1* must be set to `substrate` with a height of `substrateHeight`.

The meshing values can remain unchanged, as the values used for the previous simulation are sufficient. Run *CEM validate*.

Note that a warning may be encountered when running the solution. This is because losses cannot be calculated in an infinitely large medium, as is required for the extraction of antenna directivity information (gain is computed by default). This warning can be avoided by ensuring that the far field gain be calculated instead of the directivity. This is set on the *Advanced* tab of the far field request in the tree.

A-8.4 Edge-fed, planar multilayer substrate

Creating the model

This fourth model is an extension of the third model. The patch is now edge fed and the microstrip feed is used.

NOTE: This example is only for the purposes of demonstration. Usually, the feed line is inserted to improve the impedance match. Also, for improved accuracy the edge source width (here the width of the line of 4.5 mm) should not be wider than 1/30 of a wavelength. This means that strictly speaking the microstrip port should not be wider than about 3 mm.

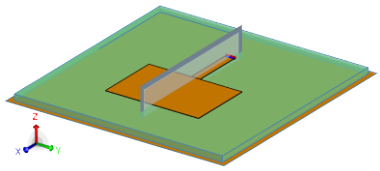


Figure A-8-4: A 3D representation of an edge fed microstrip patch antenna on an infinite ground.

The modification is as follows:

- Copy the patch from the `antenna`.
- Delete the `antenna` geometry so that only the `patch` remains.

- Create a rectangle using the *base corner, width, depth* method, with the base corner at (`-lengthX/2,-feedlineWidth/2,0`). The *width* of the line should be `-lambda/4` and the *depth* should be `feedlineWidth`. Label the part `feedline`.
- Union all the elements.
- Add a microstrip port at the edge of the feed line.
- Add a voltage source on the port. (1 V, 0°, 50 Ω).

All meshing and calculation requests can remain the same as in the previous example. Run the *CEM validate*.

A-8.5 Comparison of the results for the different models

The far field gain patterns for all 4 antenna models at 3 GHz are plotted on the same graph in Figure A-8-5. The model with the finite ground should be the best representation of an antenna that can physically be manufactured, but the simulation time compared to the infinite plane solution is considerably longer. We can also see how the different feeding mechanisms impact the radiation pattern.

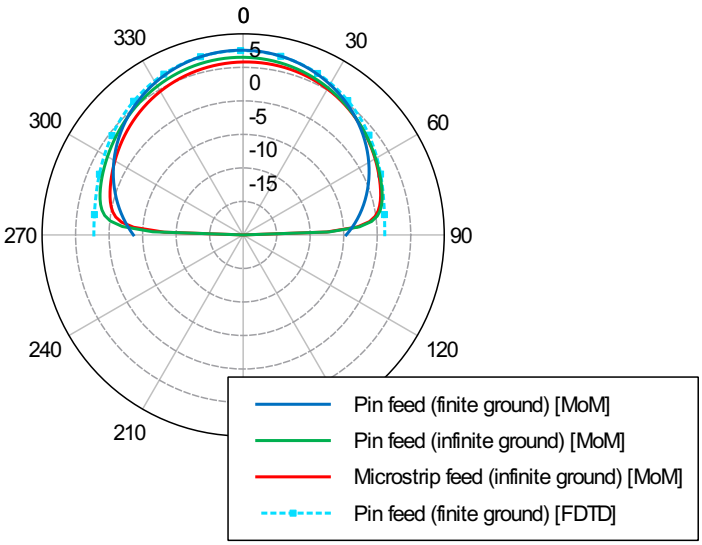


Figure A-8-5: The E plane radiation pattern of the four microstrip patch models.

A-9 Proximity coupled patch antenna with microstrip feed

Keywords: patch antenna, microstrip feed, proximity coupling, voltage on an edge, infinite substrate

This example considers a proximity coupled circular patch antenna from 2.8 GHz to 3.2 GHz. The meshed geometry is shown in Figure A-9-1. The feed line of the patch is between the patch and the ground plane.

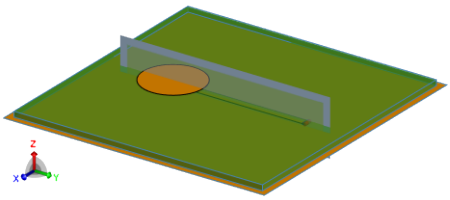


Figure A-9-1: Proximity coupled circular patch antenna. The lighter triangles are on a lower level (closer to the ground plane).

A-9.1 Circular patch

Creating the model

The steps for setting up the model are as follows:

- Set the model unit to millimetres.
- Define the following variables:
  - `epsr = 2.62` (The relative permittivity.)
  - `patch_rad = 17.5` (The patch radius.)
  - `line_len = 79` (The strip line length.)
  - `line_width = 4.373` (The strip line width.)
  - `offset = 0` (Feed line offset from the patch centre.)
  - `substrate_d = 3.18` (The substrate thickness.)
  - `f_min = 2.8e9` (The lowest simulation frequency.)
  - `f_max = 3.2e9` (The highest simulation frequency.)
- Create a new dielectric medium called `substrate` with relative permittivity of `epsr` and dielectric loss tangent of 0.
- Create a circular metallic disk with centre of the disc at the origin with radius = `patch_rad`.
- Create a rectangle with the definition method: *Base corner, width, depth*. Set the *Base corner* as the following: (`-line_width/2`, 0, `-substrate_d/2`). Set the *Width* = `line_width` and *Depth* = `line_len`.

- Add a planar multilayer substrate. The substrate is `substrate_d` thick and is of `substrate` material type with a bottom ground plane. *Layer0* is of type *free space*.
- Create a microstrip port on the edge of the feed line furthest away from the patch element.
- Add a voltage source to the port.
- Request that the continuous frequency range is calculated from `f_min` to `f_max`.

Meshing information

A single plane of magnetic symmetry is used on the X=0 plane.

Use the *standard* auto-mesh setting, but play around with the curvature refinement options on the *Advanced* tab of the mesh dialog. While changing these settings around, create the mesh and investigate the effects of the different settings. Also investigate the difference in the results - this illustrates the importance of performing a mesh conversion test for your model.

No calculation requests are required for this model since the input impedance is available when a voltage source has been defined.

CEM validate

After the model has been meshed, run *CEM validate*. Take note of any warnings and errors. Correct any errors before running the FEKO solution kernel.

A-9.2 Results

Figure A-9-2 shows the reflection coefficient on the Smith chart.

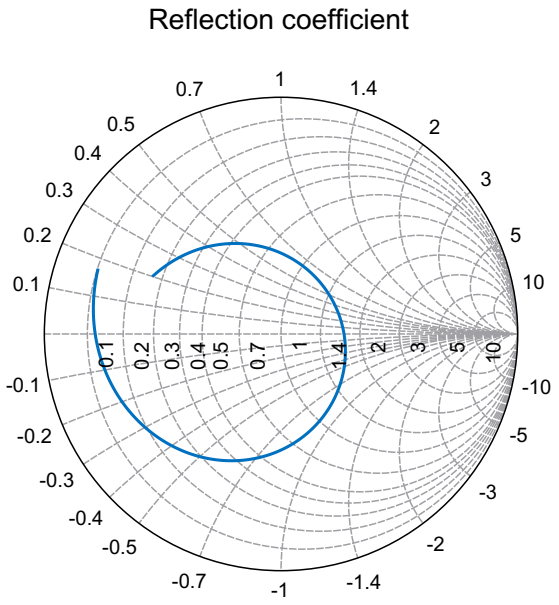


Figure A-9-2: Reflection coefficient of the proximity coupled patch.

A-10 Modelling an aperture coupled patch antenna

Keywords: patch antenna, aperture coupling, aperture triangles, infinite planes, SEP

A patch antenna can be fed using a microstrip feed line, coupling energy through an aperture in the ground plane underneath the patch. This example will demonstrate how to model such a configuration using both a full model where the substrate layers are meshed, as well an equivalent model using an infinite plane approximation. The latter makes use of aperture triangles that allow energy to couple through an infinite PEC ground plane. Figure A-10-1 shows a depiction of the geometry that will be used.

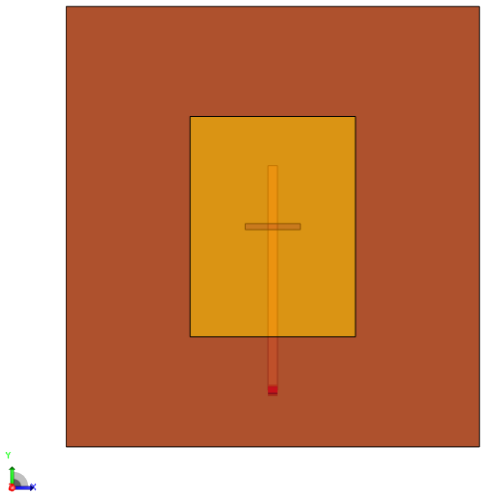


Figure A-10-1: Top view of an aperture coupled patch antenna. Opacity has been set so that all layers can be seen in the image.

A-10.1 Full SEP model

Creating the model

The steps for setting up the model are as follows:

- Define the following variables:
  - `f_min = 2.1e9` (Minimum frequency in operating range.)
  - `f_max = 2.3e9` (Maximum frequency in operating range.)
  - `epsr_a = 10.2` (Relative permittivity of the bottom dielectric layer.)
  - `epsr_b = 2.54` (Relative permittivity of the top dielectric layer.)
  - `lambda_a = c0/f_max/sqrt(epsr_a)*100` (Wavelength in the bottom dielectric layer.)

  - `lambda_b = c0/f_max/sqrt(epsr_b)*100` (Wavelength in top dielectric layer.)
  - `d_a = 0.16` (Height of the bottom dielectric layer.)
  - `d_b = 0.16` (Height of the top dielectric layer.)
  - `patch_l = 4.0` (Length of the patch antenna.)
  - `patch_w = 3.0` (Width of the patch antenna.)
  - `grnd_l = 2*patch_l` (Length of the substrate layers and ground plane.)
  - `grnd_w = 2.5*patch_w` (Width of the substrate layers and ground plane.)
  - `feed_l = lambda_a` (Length of the microstrip feed line.)
  - `feed_w = 0.173` (Width of the microstrip feed line.)
  - `stub_l = 1.108` (Length of the matching stub on the microstrip feed line.)
  - `ap_l = 1.0` (Length of the aperture.)
  - `ap_w = 0.11` (Width of the aperture.)

- Set the model unit to centimetres.
- Create a dielectric medium called `bottom_layer` with relative permittivity of `epsr_a` and a loss tangent of 0.
- Create a dielectric medium called `top_layer` with relative permittivity of `epsr_b` and a loss tangent of 0.
- Create a plate for the ground layer using the rectangle primitive with its centre at (0, 0, 0), a width of `grnd_w` and a depth of `grnd_l`. Label the plate `ground`.
- Create the aperture by using a rectangular plate with its centre at (0, 0, 0), a width of `ap_l` and a depth of `ap_w`. Label the plate `aperture`.
- Subtract `aperture` from `ground`. Note that the ground plane remains, but with a hole in the centre where the aperture plate was defined. Rename this to `slotted_ground`.
- Create the patch antenna using a plate with its centre at (0, 0, `d_b`), a width of `patch_w` and a depth of `patch_l`. Label the plate `patch`.
- Create the microstrip feed line using a plate with base corner at (`-feed_w/2`, `-feed_l/2 + stub_l`, `-d_a`), a width of `feed_w` and a depth of `feed_l`. Label the plate `feed`.
- To excite the model, an edge feed will be used. A plate is created that connects the ground plane to the microstrip line at the farthest edge of the feed. This plate is then split in two parts, one for the positive and negative terminals of the source.
  - Create the feed port by using a plate. The origin of the workplane sits at (`-feed_w/2`, `feed_l/2 + stub_l`, `-d_a`). Rotate the workplane by 90° around the U axis so that the plane where the plate will be created is the vertical XZ plane and is located at the end of the microstrip line. The base corner of the plate is at (0, 0, 0), has a width of `feed_w` and a depth of `d_a`. Label the plate `feedPort`.
  - The feed port must still be split into the positive and negative terminals. Use the *split* command. Split `feedPort` in the UV plane at (0, 0, `-d_a/2`). Rename the two resulting components to `port_bottom` and `port_top`.

- At this point, all of the PEC parts have been created that are required for the model. Union all of the parts and rename the resulting geometry to `conducting_elements`.
- Explicitly set the face properties of all of the faces to PEC. This ensures that the faces will remain PEC after future union operations.
- Create the bottom dielectric layer by using a cuboid whose centre is located at (0, 0, -d_a). The cuboid has a width of `grnd_w`, a depth of `grnd_l` and a height of `d_a`. Label the cuboid `bottom_layer`.
- Create the top dielectric layer by using a cuboid with its base centre at (0, 0, 0). The cuboid has a width of `grnd_w`, a depth of `grnd_l` and a height of `d_b`. Label the cuboid `top_layer`.
- Union all of the geometry components.
- Set the bottom region medium to `bottom_layer` and the top region medium to `top_layer`.
- Ensure that the patch, microstrip line, the feed port and the ground plate are all set to PEC.
- Add an edge port between the two split components of `feedPort`. Let the positive face correspond to the face attached to the ground plane. Add a voltage source to the port with the default source properties.
- In order to obtain accurate results whilst minimising resource requirements, local mesh refinement is necessary on several of the geometry parts:
  - Set local mesh refinement of `lambda_b/40` on the patch edges.
  - Set local mesh refinement of `ap_w*0.7` aperture edges.
  - Set local mesh refinement on the feed face to `feed_w/2`.
- Set the continuous frequency range from `f_min` to `f_max`.

Requesting calculations

Request a full 3D far field. Magnetic symmetry may be applied to the plane at X=0.

Meshing information

Use the *standard* auto-mesh setting. Note that local mesh refinement was used on several of the edges (see description above). To reduce the number of mesh elements, the growth rate was set to 40% between *Slow* and *Fast*. This setting increases the rate at which the size of the mesh elements increase from smaller to larger triangles.

CEM validate

After the model has been meshed, run *CEM validate*. Take note of any warnings and errors. Correct any errors before running the FEKO solution kernel.

A-10.2 Aperture triangles in infinite ground plane

This model uses the planar multilayer substrate to replace the dielectric substrate of the first model. Aperture triangles are used to model the aperture in the PEC ground plane between the layers. This approach provides an equivalent model that requires fewer resources than the full SEP model.

Creating the model

The steps for setting up the model are as follows:

- Define the same variables and media as for the full SEP model and set the model unit to centimetres.
- Follow the same creation steps as for the first model to create some of the components. The following components are required for this model:
  - The `aperture`.
  - The `patch`.
  - The `feed`.
  - The `feedPort`. It is also required to split this plate into a top and bottom half.
- Create a planar multilayer substrate. Add two layers in free space:
  - *Layer 1* should have a PEC ground plane, a height of `d_b` and the medium should be set to `top_layer`.
  - *Layer 2* should not have any ground plane, a height of `d_a` and the medium should be set to `bottom_layer`.
  - The *Z value at the top of layer 1* should be `d_a`.
- Union all of the geometry parts.
- Add and edge port between the two split components of `feedPort`. Let the positive face correspond to the face attached to the ground plane. Add a voltage source to the port with the default source properties.
- Set the solution method of the face representing the aperture to *Planar Green's function aperture*.
- In order to obtain accurate results whilst minimising resource requirements, local mesh refinement is necessary on several of the geometry parts:
  - Set the local mesh refinement for the patch edges to `lambda_b/40`.
  - Set local mesh refinement on the aperture face to `ap_w*0.7`.
  - Set local mesh refinement on the feed face to `feed_w/2`.
- Set the continuous frequency range from `f_min` to `f_max`.

Requesting calculations

Request a full 3D far field. Magnetic symmetry may be applied to the plane at X=0.

Meshing information

Use the *standard* auto-mesh setting. Note that local mesh refinement was used on several of the edges (see description above). The more conservative *slow* growth rate was used.

CEM validate

After the model has been meshed, run *CEM validate*. Take note of any warnings and errors. Correct any errors before running the FEKO solution kernel.

Note that a warning may be encountered when running the solution. This is because losses cannot be calculated in an infinitely large medium, as is required for the extraction of antenna directivity information (gain is computed by default). This warning can be avoided by ensuring that the far field gain be calculated instead of the directivity. This is set on the *Advanced* tab of the far field request in the tree.

A-10.3 Results

Using the correct method to model a problem can dramatically decrease runtime and reduce the memory that is required. In this case, the aperture triangles are used in conjunction with planar multilayer substrates in such a way as to reduce the mesh size of the model. This leads to a reduction in resource requirements. Table A-10-1 shows the resources that are required for the two models.

Table A-10-1: Comparison of resources using different techniques for an aperture coupled patch antenna.

Model	*No. of Triangles*	*RAM [MByte]*	*Time [min:sec]*
Finite ground (Full SEP)	6142	1238	26:14
Infinite ground (with aperture triangles)	992	8.14	1:59

Table A-10-1 has shown the improvement in resource requirements for running the planar multilayer substrate version of the model. The results shown in Figure A-10-2 indicate that the model is a good approximation of the full SEP model. If one increases the size of the finite substrates, the results are expected to converge even more as the infinite plane approximation becomes more appropriate.

Figure A-10-3 shows the far field at broadside over frequency. The far fields are the same shape and the centre frequency deviates by less than 1%.

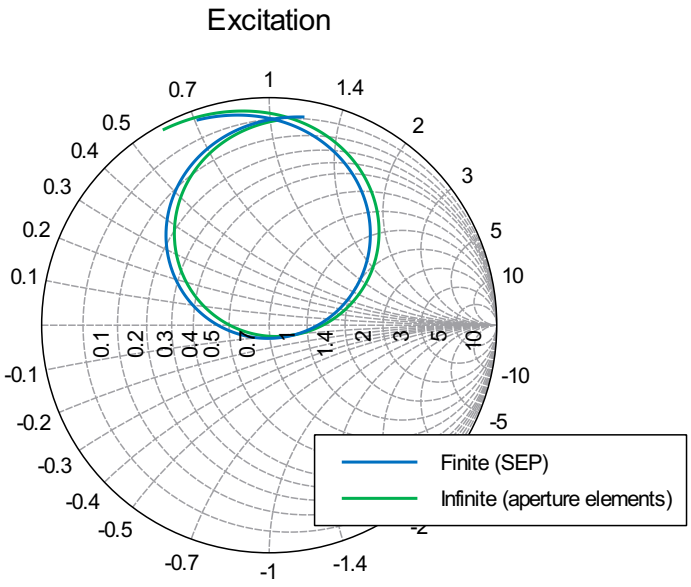


Figure A-10-2: Smith chart showing the reflection coefficient over frequency for the two models.

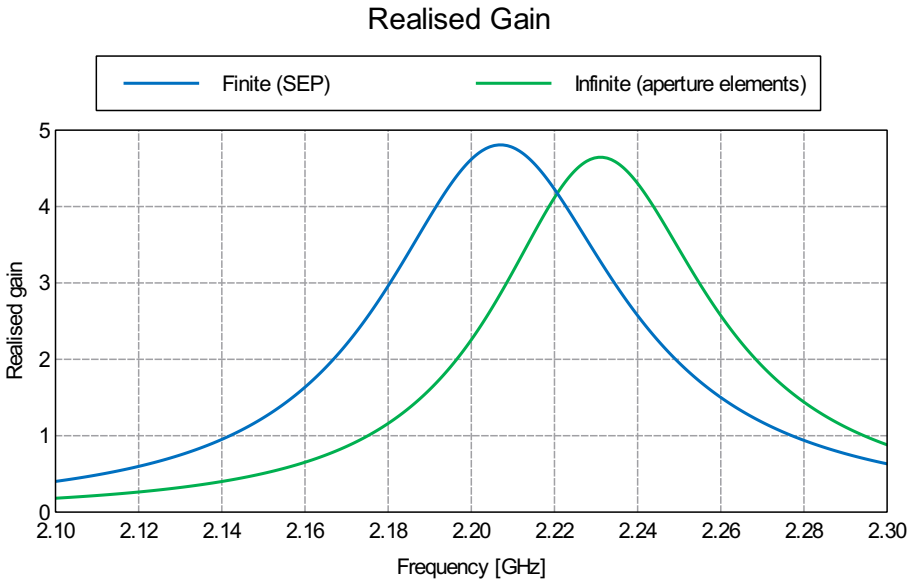


Figure A-10-3: Far field realised gain over frequency (θ=0°; ϕ=0°).

A-11 Different ways to feed a horn antenna

Keywords: horn, waveguide, impressed field, pin feed, radiation pattern, far field

A pyramidal horn antenna with an operating frequency of 1.645 GHz is constructed and simulated in this example. Figure A-11-1 shows an illustration of the horn antenna and far field requests in CADFEKO.

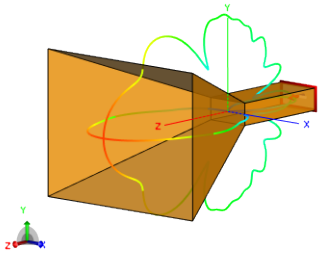


Figure A-11-1: A pyramidal horn antenna for the frequency 1.645 GHz.

In particular, we want to use this example to compare different options available in FEKO to feed this structure. Four methods are discussed in this example:

- The first example constructs the horn antenna with a real feed pin inside the waveguide. The pin is excited with a voltage source.

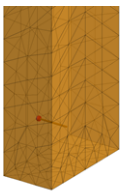


Figure A-11-2: Wire pin feed.

- The second example uses a waveguide port to directly impress the desired mode (in this case a TE_{10} mode) in the rectangular waveguide section.

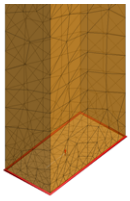


Figure A-11-3: Waveguide feed.

- The third example uses an impressed field distribution on the aperture. This method is more complex to use than the waveguide port, but is worth demonstrating since it can be used for any user defined field distribution or waveguide cross section (which might not be supported directly at the waveguide source). Note that contrary to the waveguide source, the input impedances and S-parameters cannot be obtained using an impressed field distribution.

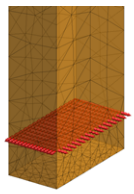


Figure A-11-4: Aperture feed.

- The fourth example uses a FEM modal boundary. The waveguide feed section of the horn is solved by setting it to a FEM region. The waveguide is excited using a FEM modal boundary. Note that for this type of port, any arbitrary shape may be used and the principal mode will be calculated and excited.

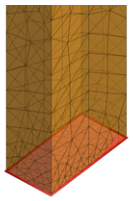


Figure A-11-5: FEM modal port feed.

A-11.1 Wire pin feed

Creating the model

The steps for setting up the model are as follows:

- Set the model unit to centimetres.
- Create the following variables
  - `freq` = 1.645e9 (The operating frequency.)
  - `lambda` = `c0/freq*100` (Free space wavelength.)
  - `wa` = 12.96 (The waveguide width.)
  - `wb` = 6.48 (The waveguide height.)
  - `ha` = 55 (Horn width.)
  - `hb` = 42.80 (Horn height.)

  - `wl` = 30.20 (Length of the horn section.)
  - `fl` = `wl` - `lambda/4` (Position of the feed wire in the waveguide.)
  - `hl` = 46 (Length of the horn section.)
  - `pinlen` = `lambda/4.56` (Length of the pin.)

- Create the waveguide section using a cuboid primitive and the *Base corner, width, depth, height* definition method. The *Base corner* is at (`-wa/2`, `-wb/2`,`-wl`), *width* of `wa`, *depth* of `wb` and *height* of `wl`.
- Delete the face lying on the UV plane.
- Create the horn using the flare primitive with its base centre at the origin using the definition method: *Base centre, width, depth, height, top width, top depth*. The *bottom width* and *bottom depth* are `wa` and `wb`. The *height*, *top width* and *top depth* are `hl`, `ha` and `hb` respectively.
- Delete the face at the origin as well as the face opposite to the face at the origin.
- Create the feed pin as a wire element from (0, `-wb/2`,`-fl`) to (0, `-wb/2` + `pinlen`,`-fl`).
- Add a wire segment port on wire. The port must be placed where the pin and the waveguide meet.
- Add a voltage source to the port. (1 V, 0°, 50 Ω)
- Union the three parts.
- Set the frequency to `freq`.
- Set the total source power (no mismatch) to 5 W.

Requesting calculations

One plane of magnetic symmetry in the X=0 plane may be used.

The solution requests are:

- Define a far field request in the YZ plane with 2° steps for the E plane cut.
- Define a far field request in the XZ plane with 2° steps for the H plane cut.

Meshing information

Use the *coarse* auto-mesh setting with a wire radius of 0.1 cm. We use coarse meshing for this example to keep the simulation times as low as possible.

CEM validate

After the model has been meshed, run *CEM validate*. Take note of any warnings and errors. Correct any errors before running the FEKO solution kernel.

A-11.2 Waveguide feed

Creating the model

The wire feed model is changed to now use the waveguide feed. The line is deleted and the wire port removed. The following additional steps are followed:

- Set a local mesh size of `lambda/20` on the back face of the waveguide.
- A waveguide port is applied to the back face of the guide. CADFEKO automatically determines the shape of the port (rectangular) and the correct orientation and propagation direction. (It is good practice to visually confirm that these have been chosen correctly by observing the port preview in the 3D view.)
- A waveguide mode source is applied to the waveguide port. The option to automatically excite the fundamental propagating mode, and automatically choose the modes to account for in the solution is used.
- Symmetry on the X=0 plane may still be used since the source is symmetric. In addition, electric symmetry may be used in the Y=0 plane.

Meshing information

Remesh the model to account for the setting of the local mesh size on the back face of the waveguide.

CEM validate

After the model has been meshed, run *CEM validate*. Take note of any warnings and errors. Correct any errors before running the FEKO solution kernel.

A-11.3 Near field source feed

Creating the model

Here the modal distribution of the TE_{10} mode in a rectangular waveguide is evaluated directly in FEKO as excitation for the horn by means of an impressed field distribution on an aperture (also see the *FEKO User Manual* for information on the near field source and the AP card). This method may be useful in cases such as when an uncommon propagation mode needs to be impressed.

The application of a near field source is supported in CADFEKO, but the near field distribution must be defined in an external file. This may be done in many ways, but for this example, the setup is done by using another CADFEKO model. A waveguide section is created and a near field request is placed inside the waveguide. Both the electric and magnetic fields are saved in their respective `*.efe` and `*.hfe` files. These files are then used as the input source for the near field feed excited horn model. For more details on how the fields are calculated, see `Create_Mode_Distribution_cf.cfx`.

To add the near field source to the model, first create a near field data definition. Define the near field data file structure with the following properties:

- The electric field file and magnetic field file is stored respectively as:
  - `Create_Mode_Distribution_cf_NearField1.efe`
  - `Create_Mode_Distribution_cf_NearField1.hfe`
- Uncheck the *Also sample along edges* check box.
- The width of the aperture is `wa`.
- The height of the aperture is `wb`.
- The number of points along X/U is `10`.
- The number of points along Y/V is `5`.

Next create a near field equivalent source with the following properties:

- Set the source to use the near field data definition that was just created.
- Set the *Workplane* origin to (`-wa/2`, `-wb/2`, `-wl+lambda/4`).

Note that electric symmetry may be used in the Y=0 plane, as well as magnetic symmetry in the X=0 plane. This is only valid when the near field source also conform to this symmetry.

A-11.4 FEM modal port

Creating the model

The steps for setting up the model are as follows:

- Create a new model.
- Set the model unit to centimetres.
- Create the same variables as for the wire model in section A-11.1.
- Create a dielectric labelled `air` with the default dielectric properties of free space.
- Create the waveguide section using a cuboid primitive and the *Base corner, width, depth, height* definition method. The *Base corner* is at (`-wa/2`, `-wb/2`,`-wl`), *width* of `wa`, *depth* of `wb` and *height* of `wl` (in the Y direction).
- Set the region of the of the cuboid to `air`.
- Select the four faces that represent the waveguide boundary walls (all faces except the one at the origin and the one opposite to it where the modal port will be located) and set their face properties to *Perfect electric conductor*.

- Set the solution method of the region to FEM.
- Create the horn using the flare primitive with its base centre at the origin using the definition method: *Base centre, width, depth, height, top width, top depth*. The *bottom width* and *bottom depth* are `wa` and `wb`. The *height*, *top width* and *top depth* are `hl`, `ha` and `hb` respectively.
- Delete the face at the origin as well as the face opposite to the face at the origin.
- Union the waveguide section and the flare section.
- Set a local mesh size of `lambda/20` on the back face of the waveguide.
- Add a FEM modal port to the back face of the waveguide.
- Add a FEM modal source to the port with the default magnitude and phase.
- Set the frequency to `freq`.
- Set the total source power (no mismatch) to 5 W.

Requesting calculations

A plane of magnetic symmetry at X=0 may be used, as well as a plane of electric symmetry at Y=0.

The solution requests are:

- Define a far field request in the YZ plane with 2° steps for the E plane cut.
- Define a far field request in the XZ plane with 2° steps for the H plane cut.

Meshing information

Use the *coarse* auto-mesh setting.

CEM validate

After the model has been meshed, run *CEM validate*. Take note of any warnings and errors. Correct any errors before running the FEKO solution kernel.

A-11.5 Comparison of the results for the different models

The far field gain (in dB) in the *E_Plane* and *H_Plane* is shown in Figures A-11-6 and A-11-7 respectively.

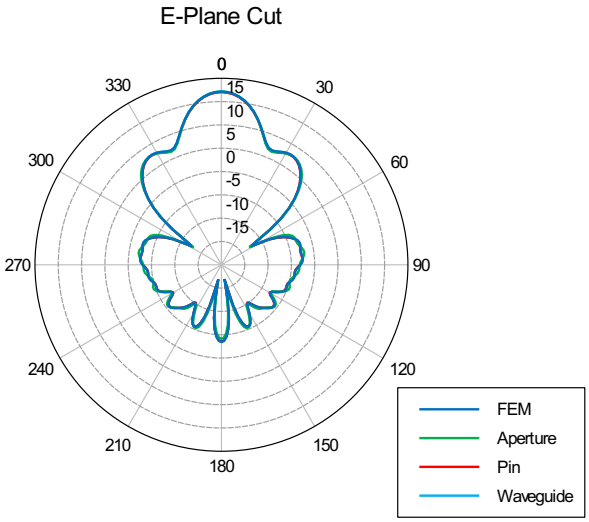


Figure A-11-6: Comparison of the far field gain of the horn antenna with different feeding techniques for the *E_Plane* far field request.

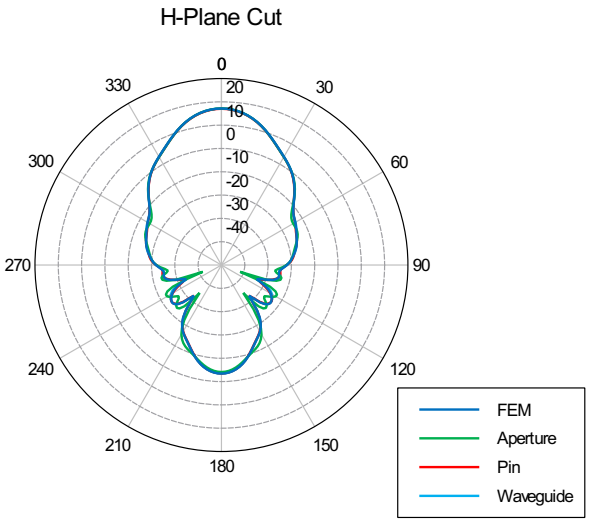


Figure A-11-7: Comparison of the far field gain of the horn antenna with different feeding techniques for the *H_Plane* far field request.

A-12 Dielectric resonator antenna on finite ground

Keywords: dielectric resonator antenna, radiation pattern, far field, input impedance, infinite ground, FEM current source, modal source, waveguide port

The dielectric resonator antenna (DRA) example illustrates how a coaxial pin feed can be modelled. The input impedance and radiation pattern of a DRA on a finite ground plane are considered. Two methods for feeding the model are considered. One method uses a FEM/MoM hybrid, whilst the other uses a pure MoM approach. For the FEM model, a layer of air is added to minimise the number of triangles on the FEM/MoM interface. The antenna geometry (including the finite ground plane and a symmetry plane) is shown in Figure A-12-1.

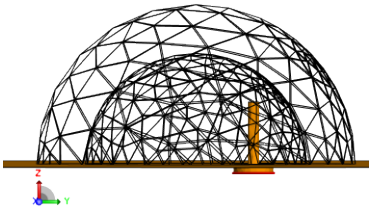


Figure A-12-1: Wire frame display of a dielectric resonator antenna on a finite ground plane showing the dielectric resonator and feed-pin.

A-12.1 DRA fed with a FEM modal port

Creating the model

The steps for setting up the model are as follows:

- Set the model unit to millimetres.
- Define variables:
  - `epsr = 9.5` (Relative permittivity.)
  - `r = 0.63` (Feed element radius.)
  - `hBig = 1` (Feed base height.)
  - `rBig = 2.25` (Feed base radius.)
  - `rDisk = 60` (The ground radius.)
  - `rDome = 12.5` (The inner dome radius.)
  - `rDomeBig = rDome + 5.5` (Outer dome radius.)
  - `h = 7` (Feed element height.)
  - `fmin = 3e9` (Lowest calculation frequency.)
  - `fmax = 6e9` (Highest calculation frequency.)
  - `lambda = c0/fmax*1000` (Free space wavelength in millimetres.)

- Define named points:
  - `excite_b = (0,6.5,-1)`
- Create dielectrics:
  - Create a dielectric named `air` with relative dielectric permittivity of 1 and dielectric loss tangent of 0.
  - Create a dielectric named `dome` with relative dielectric permittivity of `epsr` and dielectric loss tangent of 0.
  - Create a dielectric named `isolator` with relative dielectric permittivity of 2.33 and dielectric loss tangent of 0.
- Create a new workplane and place its origin at `excite_b`. Set this workplane as the default workplane.
- Create a cylinder. Set respectively the *radius* and *height* equal to `rBig` and `hBig`. Modify the label to `FeedBase`.
- Create another cylinder. Set respectively the *radius* and *height* equal to `r` and `h + hBig`. Modify the label to `FeedPin`.
- Union the two cylinders.
- Set the region properties of the cylinder, `FeedPin`, to the dielectric of type `air`.
- Set the region properties of the cylinder, `FeedBase`, to the dielectric of type `isolator`.
- Set the default workplane back to the global `XY` plane.
- Create a disk on the XY plane with the radius set equal to `rDisk`.
- Create a sphere with a radius of `rDomeBig`. Set the label to `OuterDome`.
- Create a sphere with a radius of `rDome`. Set the label to `InnerDome`.
- Union everything and name the unioned part `DRA`.
- Delete the bottom halves of both spheres.
- Set the region of the internal half sphere, to be the dielectric named `dome`.
- Set the region that is left (the space around the internal half sphere) to be the dielectric named `air`.
- For all the regions, set the *solution properties* to *finite element method (FEM)*.
- Set all of the faces in the model to PEC, except:
  - the top and bottom faces of the `FeedBase`;
  - the two faces that remain of the spheres; and
  - the single face inside the `FeedPin`.

  These should be the *default* medium.

- The outer dome can be meshed more coarsely, since the geometry is not of any specific interest. Set a local refinement of `lambda/8`, which is equivalent to a "coarse" mesh size for the MoM boundary between the `air` region and the surrounding free space.
- Use the *Simplify* transform to remove redundant faces and edges in the model.
- Add a FEM modal port to the bottom face of `FeedBase`, at the bottom of the antenna.
- Apply FEM modal source to the modal port.
- Set the frequency to be continuous from `fmin` to `fmax`.

Requesting calculations

A single plane of magnetic symmetry on the X=0 plane may be used for this model.

The solution requests are:

- Create a vertical far field request in the XZ plane. (-180°$\leq\theta\leq$180°, with ϕ=0° and 2° steps)

Meshing information

Use the *Coarse* auto-mesh setting. The curvature of the model will cause further refinement to the complex parts of the model to ensure that the geometry is accurately represented.

CEM validate

After the model has been meshed, run *CEM validate*. Take note of any warnings and errors. Correct any errors before running the FEKO solution kernel.

A-12.2 DRA fed with a waveguide port

Creating the model

The steps for setting up the model are as follows:

- Set the model unit to millimetres.
- Define the same variables as for the FEM/MoM model.
- Define named points:
  - `excite_b = (0,6.5,-1)`
- Create dielectrics:

  - Create a dielectric named `dome` with relative dielectric permittivity of `epsr` and dielectric loss tangent of zero.
  - Create a dielectric named `isolator` with relative dielectric permittivity of 2.33 and dielectric loss tangent of zero.

- Create a new workplane and place its origin at `excite_b`. Set this workplane as the default workplane
- Create a cylinder. Set respectively the *Radius* and *Height* equal to `rBig` and `hBig`. Modify the label to `FeedBase`.
- Create another cylinder. Set respectively the *Radius* and *Height* equal to `r` and `h` + `hBig`. Modify the label to `FeedPin`.
- Union the two cylinders.
- Set the default workplane back to the global `XY` plane.
- Create a disk on the XY plane with the radius set equal to `rDisk`.
- Create a sphere with a radius of `rDome`. Set the label to `InnerDome`.
- Union everything and name the unioned part `DRA`.
- Delete the bottom half of the sphere.
- Set the region of the cylinder, `FeedBase`, to be the dielectric named `isolator`.
- Set the region of the half sphere to be the dielectric named `dome`.
- Set all of the faces in the model to PEC, except:
  - the top and bottom faces of the `FeedBase`; and
  - the remaining face of the half sphere.

  These should be the *default* medium.
- Use the *simplify* transform to remove redundant faces and edges in the model.
- Add a waveguide port to the dielectric face of `FeedBase`, at the bottom of the antenna.
- Apply waveguide source to the waveguide port.
- Set the frequency to be continuous from `fmin` to `fmax`.

Requesting calculations

A single plane of magnetic symmetry on the X=0 plane may be used for this model.

The solution requests are:

- Create a vertical far field request in the XZ plane. (-180°$\leq\theta\leq$180°, with ϕ=0° and 2° steps)

Meshing information

Use the *Coarse* auto-mesh setting. The curvature of the model will cause further refinement to the complex parts of the model to ensure that the geometry is accurately represented.

CEM validate

After the model has been meshed, run *CEM validate*. Take note of any warnings and errors. Correct any errors before running the FEKO solution kernel.

A-12.3 Results

The calculated S_{11} for 3 GHz to 6 GHz is shown in Figure A-12-2. A radiation pattern at 3.6 GHz is shown in Figure A-12-3. Results are shown for both modelling methods.

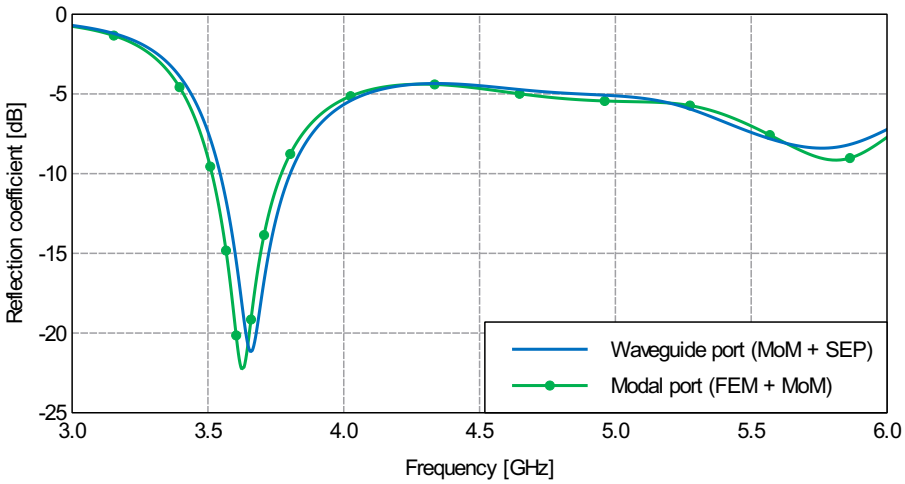


Figure A-12-2: Input reflection coefficient for the DRA antenna.

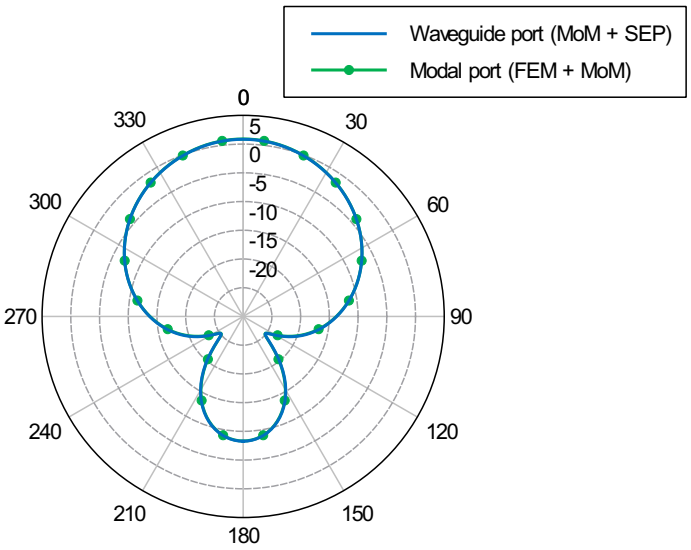


Figure A-12-3: Vertical (XZ plane) gain (in dB) at 3.6 GHz.

A-13 Analysing a lens antenna using the ray launching geometrical optics (RL-GO)

Keywords: Geometrical optics, lens antenna, dielectric, radiation pattern, radiation pattern point source

A dielectric lens with a spherical surface (S_1) and elliptical surface (S_2) is constructed. The lens is illuminated by a radiation pattern point source based on a precomputed (θ) radiation pattern and the far field pattern is computed.

The lens structure is modelled using the ray launching geometrical optics (RL-GO) method. The results are compared to the MoM/FEM result. The model is shown in Figure A-13-1.

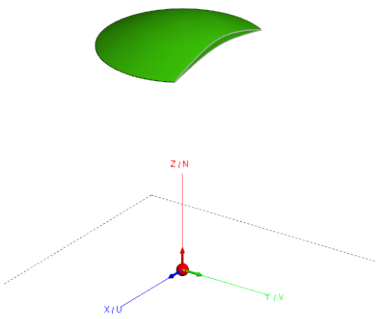


Figure A-13-1: The 3D view of the dielectric RL-GO lens model with a point source.

Creating the model

The model is constructed in millimetres. It is assumed that the focal point of the lens is positioned at the global origin. A lens shape consisting of a spherical surface and an elliptical surface is assumed. The spherical surface is centred at the focal point.

The dielectric lens model has the following user defined parameters:

- `freq = 30e9` (The operating frequency.)
- `lambda_0 = c0/freq` (Free space wavelength.)
- `D = lambda_0*10` (Diameter of lens.)
- `F = 1.5*D` (Focal length.)
- `epsr = 6` (The relative permittivity.)
- `tand = 0.005` (Dielectric loss tangent.)

The following variables that were derived from the above variables are used in the model construction:

- `alpha = arcsin(D/(2*F))` (Included angle to edge of lens.)
- `arclength = alpha*F` (Arc length to edge of lens.)
- `n = sqrt(epsr)` (Refraction index.)
- `T = (2*F - sqrt(4*F^2 - D^2))/(2*(n-1))` (Lens thickness.)
- `v0 = (F + T)/(n + 1)` (Ellipse offset distance.)
- `u0 = sqrt(n^2 - 1)*v0` (Ellipse minor axis length.)
- `w0 = n*v0` (Ellipse major axis length.)

A dielectric medium named *Glass* is defined and the relative permittivity and loss tangent are set to the variables `epsr` and `tand`, respectively.

To construct the lens, a sphere will be subtracted from an elliptical spheroid. The steps are as follow:

- Create a sphere at the origin with a radius `F`.
- Create a spheroid (by choosing the *Centre, radius U, radius V, radius N* method when creating a sphere). Its properties are:
  - *Centre*: (0,0,`v0`)
  - *Radius (Ru)*: `u0`
  - *Radius (Rv)*: `u0`
  - *Radius (Rn)*: `w0`
- Subtract the sphere from the spheroid; the resulting geometry can be labelled *Lens*.
- By default a closed region will be a perfect electric conductor. The region type of the *Lens* part is changed to be dielectric *Glass*.

Requesting calculations

To model the dielectric lens with the geometrical optics approximation we need to specify the solution method for the lens faces. From the *Solution* tab of the *Face properties* dialogue select the *Ray launching - geometrical optics (RL-GO)* solution method. In general, the *View by solution parameters* tool can be used to check which solution methods are being applied.

- Modify the default solver settings by limiting the maximum number of ray interactions to three interactions.
- The analysis is requested at a single frequency of `freq`.
- The dielectric lens is illuminated by a radiation pattern point source.
- The E field pattern is described by $E_x = cos^4(\theta)$, where $0 \leq \theta \leq$ pi/2 is the polar angle measured from the Z axis. Create a far field data definition for this pattern:

  - The pattern data is read from `Ideal_CosineQ4_Xpol.ffe`
  - Number of theta points: 91
  - Number of phi points: 181

- Create a far field source that utilises the far field data definition positioned at the origin. Note that the origin coincides with the focal point of the lens.

Far field pattern cuts ($0 \leq \theta \leq 180$ degree) are calculated in the XY plane ($\phi = 0°$) and YZ plane ($\phi = 90°$). The angular increment is set to 0.25° to capture the fine angular detail.

Meshing requirements

Curved elements are automatically used to represent the geometry for this solution. Note that any of the automatic mesh settings will result in the same mesh. The solution technique only needs an accurate geometric representation, which is independent of the solution frequency.

CEM validate

After the model has been meshed, run *CEM validate*. Take note of any warnings and errors. Correct any errors before running the FEKO solution kernel.

A-13.1 Results

The results (as shown in Figure A-13-2) show the original input gain pattern of the point source. This is used as a reference to see the effect that the glass lens and the solution method has on the gain patterns. Both the FEM and RL-GO solutions show good agreement with one another and show the expected focussing of the electric fields in the main beam direction. The memory and runtime requirements of running the RL-GO model are orders of magnitudes lower than that of the full-wave FEM solution.

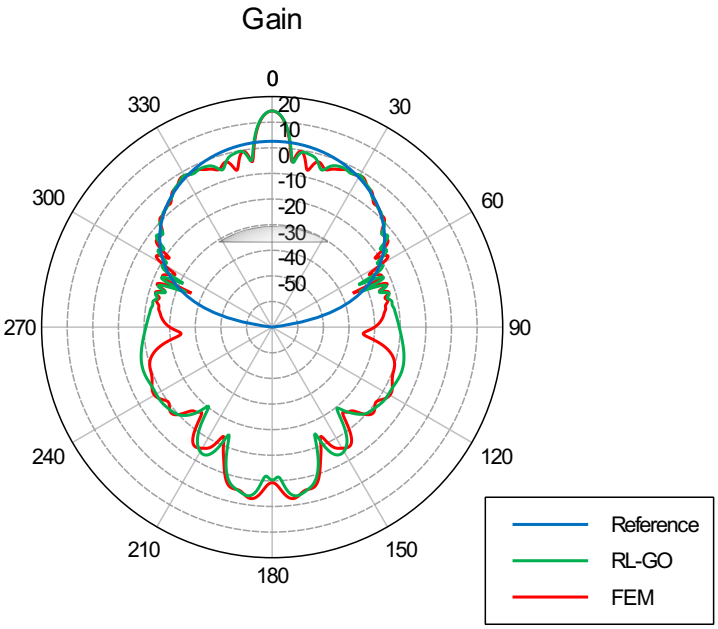


Figure A-13-2: The computed radiation pattern compared to the reference solution and a full FEM result.

A-14 Windscreen antenna on an automobile

Keywords: windscreen, input impedance, antenna

In this example we consider the input impedance of a windscreen antenna. Windscreen antennas are antennas that are located in or on a windscreen. The windscreen can consist of one or more layers and the different layers do not have to be meshed and thus simulation time is greatly reduced when compared to conventional methods. Figure A-14-1 shows a 3D representation of the car and windscreen being simulated in this example.

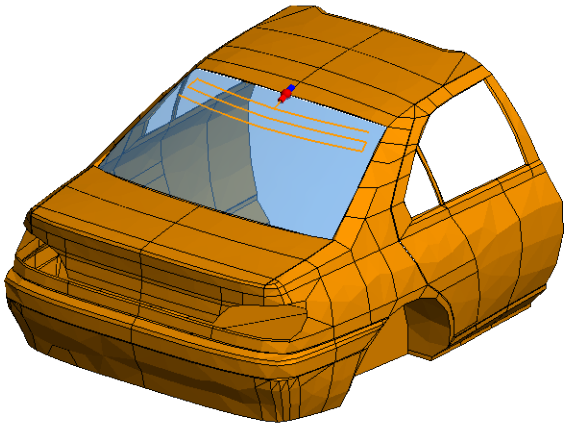


Figure A-14-1: 3D view of an automobile and a windscreen antenna.

A-14.1 Creating the rear windscreen model

The model is created by importing geometry instead of using CADFEKO's geometry tools. The required geometry files for the import are available as part of your FEKO installation and is located in the `ExampleGuide_models` directory.

Creating the model

The steps for setting up the model are as follows:

- Import the Parasolid geometry for the car (`car_geometry.x_b`). Set the *scale* equal to 1.
- Rename the three parts to be more descriptive of the geometry. There should be a part for the `car_body`, one for the `antenna` and one for the `windscreen`.
- Union the car and the antenna. This ensures that these structures will be connected during meshing. Note that the windscreen is not part of the union.
- Add a wire port to the wire that connects the antenna to the car's chassis. See Figure A-14-1 for an indication of where this port is located. Ensure that a vertex wire port is created instead of the default segment wire port.

- Add a voltage source to the port that has been created.
- Define the following dielectric materials:
  - `glass`: ϵ_r=7; tanδ=0.03
  - `pvb_foil`: ϵ_r=3; tanδ=0.05
- Create a layered dielectric called `windscreen_layers` with the following properties:
  - Layer 1: 2.1 mm thick `glass`
  - Layer 2: 0.76 mm thick `pvb_foil`
  - Layer 3: 2.1 mm thick `glass`
- Create a new windscreen definition (similar to how a dielectric is created) that uses the `windscreen_layers` layer definition. Set the *Offset L*=2.1e-3+0.76e-3. This places a reference plane (i.e. the geometry to which this definition will be applied) between the `pvb_foil` and the bottom `glass` layer.
- Select the single face of the `windscreen` and set its properties so that it is solved using the *Windscreen* solution method. This is a windscreen *reference element* and should be displayed in a semi-transparent colour that corresponds to the colour of the windscreen definition. Keep in mind that windscreen reference elements are not solved with the windscreen solution method, but they are used as part of the solution and define the shape and position of the windscreen.
- Set a local mesh refinement of 0.2 on the `windscreen` reference face. The only requirement for the mesh size is that the geometry should be accurately represented. Curvilinear elements are supported for windscreen reference faces and thus only a little refinement is required.
- Select the wires of the antenna, see Figure A-14-2. Set their solution method to be *Windscreen*. These are windscreen *solution elements*. The offset is set to zero, so that the solution elements are defined on the reference plane defined above.
- Set the frequency to be continuous from 90 – 110 MHz.

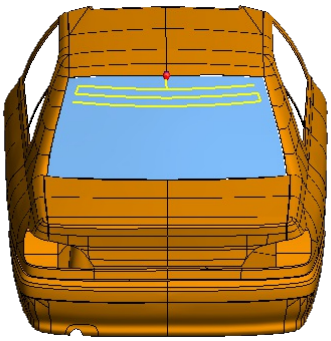


Figure A-14-2: Select the wires of the antenna.

Request calculations

We are interested in the input impedance of the antenna and thus no solution requests are required. The input impedance is automatically calculated for voltage sources.

Meshing information

Use the *Standard* auto-mesh setting with wire segment radius equal to `150e-6`. Due to the fine detail in the geometry of the car, some advanced mesh settings should be utilised to avoid meshing electrically negligible details. On the *Advanced* tab of the meshing dialog, set:

- the *Refinement factor* to *Coarse*
- the *Minimum element size* to *Medium*

Note that a local mesh refinement was set on the windscreen reference face to ensure that the mesh accurately represents the geometry of the face. Roughly 50 curvilinear elements were used in this example, but less elements would also accurately describe the curved windscreen geometry. Since these elements don't cause longer simulation times, more elements are used.

CEM validate

After the model has been meshed, run *CEM validate* and ensure that no errors or warnings are reported. The *Solve/Run → View by solution parameters* dialog can be used to see (and ensure) that the correct elements have been selected as reference and solution elements.

A-14.2 Results

Save and run the FEKO solver. Figure A-14-3 shows the computed input impedance as a function of frequency from 90 – 110 MHz. It is recommended to use MLFMM for simulation at higher frequencies (500 MHz or more).

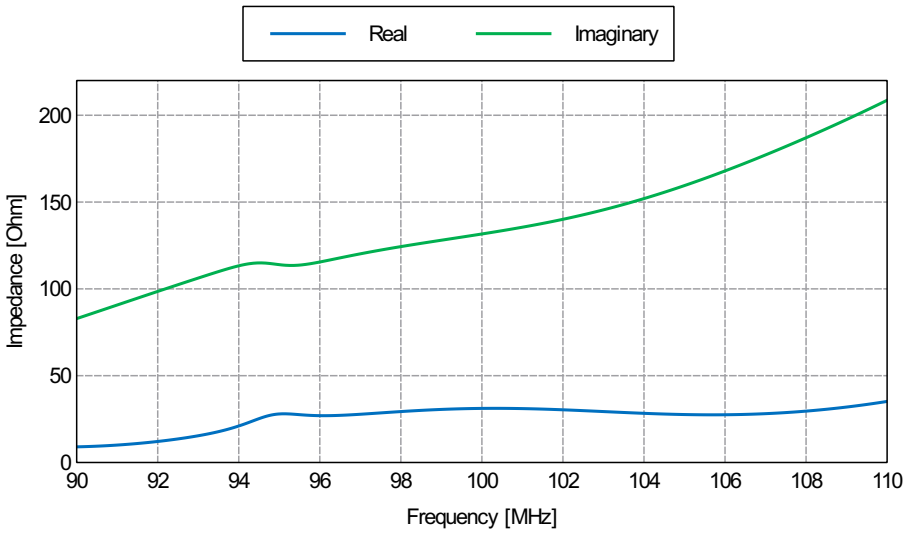


Figure A-14-3: Real and imaginary part of the input impedance of the windscreen antenna.

A-15 Design of a MIMO elliptical ring antenna (characteristic modes)

Keywords: characteristic modes, eigen value, MIMO, multiple-input multiple-output

A characteristic mode analysis of a ring structure is performed. This analysis is independent of any sources and provide insight into how a structure resonates at the calculated frequencies. Using this knowledge, one can strategically excite a structure so that only the desired modes are utilised. Figure A-15-1 shows the first four electric far field modes for the ring structure.

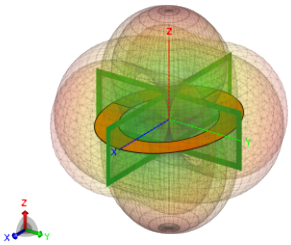


Figure A-15-1: The first four electric far field modes for the MIMO ring.

A-15.1 MIMO ring

Creating the model

The steps for setting up the model are as follows:

- Define the following variables:
  - `rInU = 21` (Inner radius of the ring in the U direction.)
  - `rOutU = 31` (Outer radius of the ring in the U direction.)
  - `rInV = rInU*0.8` (Inner radius of the ring in the V direction.)
  - `rOutV = rOutU*0.8` (Outer radius of the ring in the V direction.)
  - `freq = 2.4e9` (Operating frequency.)
- Set the model unit to millimetres.
- Create an elliptic arc centred at the origin with a radius of `rOutU` and `rOutV` in the u- and v-dimensions respectively. Set the start and end angles from 0° to 90°.
- Create another elliptic arc centred at the origin with a radius of `rInU` and `rInV` in the u- and v-dimensions respectively. Set the start and end angles from 0° to 90°.
- Select the two curves and create a surface by using the *Loft* tool. Name the quarter ring `sector_1`.

  - Copy and mirror `sector_1` around the *UN* plane.
  - Copy and mirror both sectors around the *VN* plane.
  - Union all four sectors to create a single ring structure.

- Set the simulation frequency to `freq`.
- Create four edge ports as indicated:

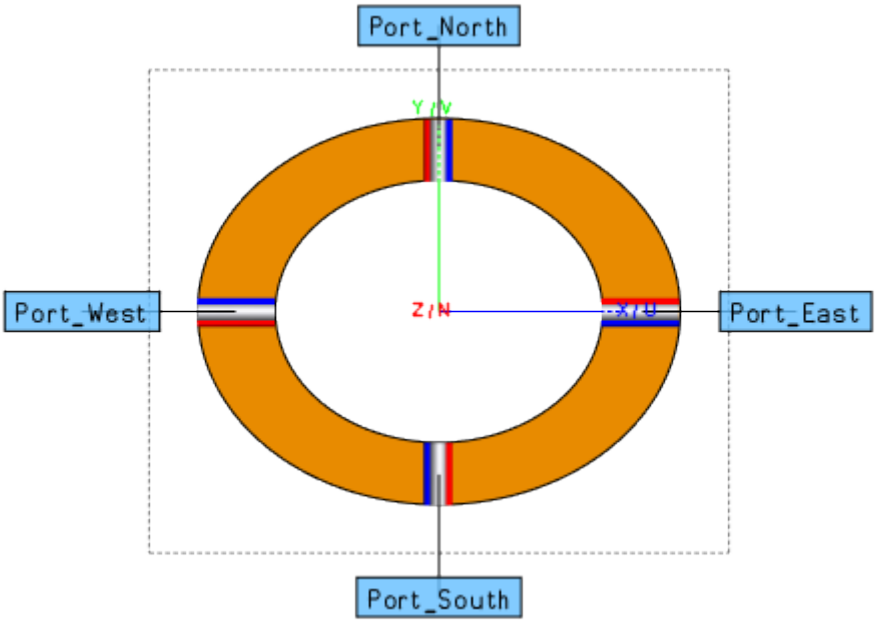


Figure A-15-2: Position of the four ports relative to a top view of the ring.

  - `port_North`: The port edge should fall on the positive *Y* axis.
  - `port_West`: The port edge should fall on the negative *X* axis.
  - `port_South`: The port edge should fall on the negative *Y* axis.
  - `port_East`: The port edge should fall on the positive *X* axis.

  Take note that all of the ports point in the same anticlockwise direction.

Symmetry in general does not apply to characteristic mode analyses, since there is no active sources involved. However, geometric symmetry can be applied to both the X=0 and Y=0 planes. This will result in a symmetrical mesh.

Configurations

Three configurations will be created. The first will be to request the characteristic mode analysis. The other two configurations will illustrate how one can use the insights obtained from a characteristic mode analysis to excite specific modes. Ensure that loads and sources are specified per configuration before continuing.

1. Request a characteristic mode configuration. Only the first five modes should be calculated.

- Request all currents.
- Request a full 3D far field with default sampling.

2. Request a standard configuration. The purpose of this configuration is to excite the first characteristic mode.
   - Add a voltage source to the Eastern port: (1 V, 0°, 50 Ω)
   - Add a voltage source to the Western port: (1 V, 180°, 50 Ω)
   - Request all currents.
   - Request a full 3D far field with default sampling.
3. Request a standard configuration. The purpose of this configuration is to excite the fifth characteristic mode.
   - Add a source to the Eastern port: (1 V, 0°, 50 Ω)
   - Add a source to the Western port: (1 V, 0°, 50 Ω)
   - Add a source to the Northern port: (1 V, 180°, 50 Ω)
   - Add a source to the Southern port: (1 V, 180°, 50 Ω)
   - Request all currents.
   - Request a full 3D far field with default sampling.

Meshing information

Use the *Fine* auto-mesh setting. On the *Advanced* tab of the mesh dialog, set the refinement factor to *Fine*. This is to ensure that the geometry is accurately represented.

CEM validate

After the model has been meshed, run *CEM validate*. Take note of any warnings and errors. Correct any errors before running the FEKO solution kernel.

A-15.2 Results

Characteristic modes are the basic building blocks for many electromagnetic results. When excitations and loads are added to a solution, we unknowingly calculate a weighted sum of the various characteristic modes. The following results will show that if the characteristic modes are properly understood, then one can alter the behaviour of a structure without making any changes to the geometry.

Note that when comparing the characteristic modes to the reconstructed modes, all values need to be normalised.

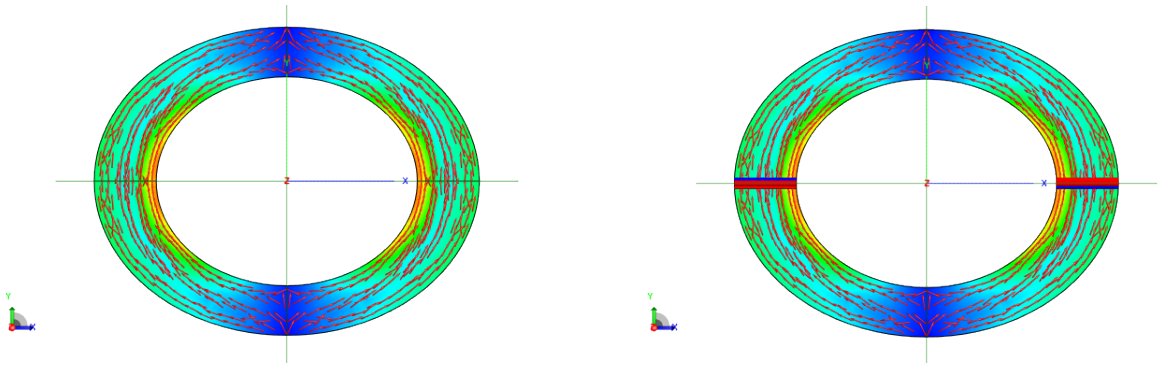


Figure A-15-3: MIMO ring's first characteristic current mode (left) vs. the reconstructed mode (right).

Mode 1 comparison

Figure A-15-3 shows how the first characteristic mode (left) can be recreated when sources are placed in the appropriate locations.

Mode 5 comparison

Figure A-15-4 shows how the fifth characteristic mode (left) can be recreated when sources are placed in the appropriate locations.

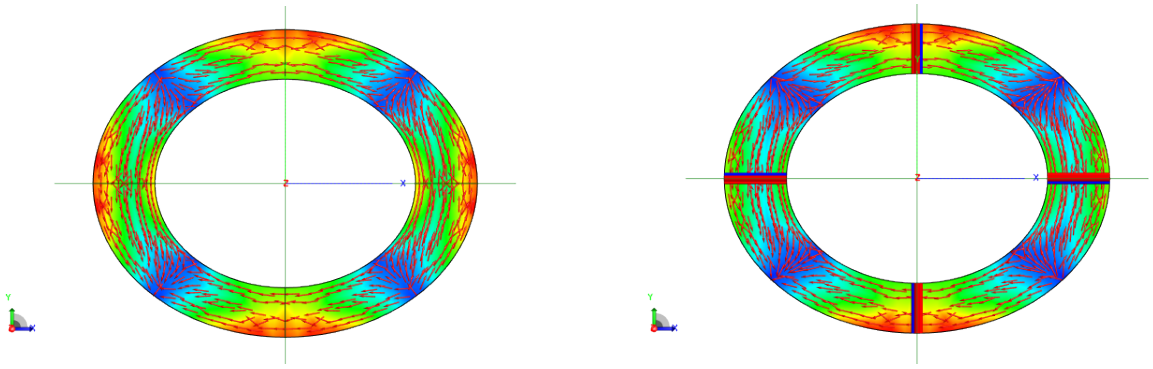


Figure A-15-4: MIMO ring's fifth characteristic current mode (left) vs. the reconstructed mode (right).

Electric far fields

Figure A-15-5 shows how the manually excited electric fields compare to the characteristic field modes. Note that due to the lack of sources when using characteristic modes, the results are normalised in this comparison.

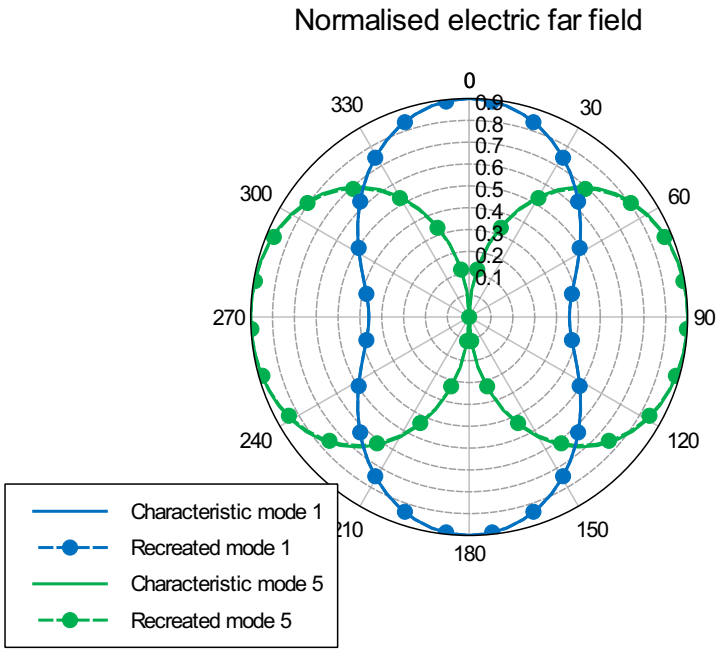


Figure A-15-5: A polar plot comparing the MIMO ring's characteristic electric field modes with the recreated fields.

A-16 Periodic boundary conditions for array analysis

Keywords: periodic boundary condition, voltage source, far field

The periodic boundary condition solution method is used to calculate the far field pattern for a single element in an infinite 2D array of pin fed patch elements as well as the approximated far field pattern for a 10×10 element array. The mutual coupling between elements is taken into account as though the elements were in an infinite array. This means that for large arrays, the results will be very accurate if edge effects can be neglected.

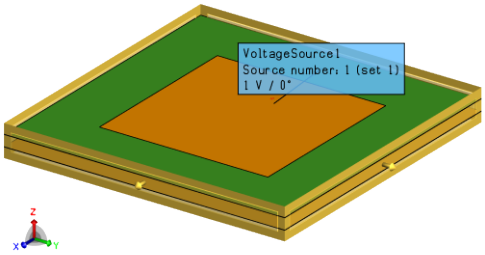


Figure A-16-1: A 3D view of a single element of the infinite patch array.

A-16.1 Pin fed patch: Broadside pattern by phase shift definition

Creating the model

The steps for setting up the model are as follows:

- Define the following variables:
  - `lambda = 0.1` (The spacing for periodic boundary condition.)
  - `freq = c0/lambda` (Operating frequency of the patch)
  - `er = 2.55` (Relative dielectric constant of patch substrate)
  - `base_width = 0.5*lambda` (Width of the patch substrate)
  - `base_length = 0.5*lambda` (Length of the patch substrate)
  - `base_height = 0.02*lambda` (Height of the patch substrate)
  - `patch_width = 0.3*lambda` (Width of the patch antenna)
  - `patch_length = 0.3*lambda` (Length of the patch antenna)
  - `pin_pos = patch_length/4` (Distance of feed pin from patch centre)
- Create a dielectric medium named `substrate` with relative permittivity of `er` and zero dielectric loss tangent.
- Create the substrate using the cuboid primitive with the *base centre, width, depth, height* definition method. The side lengths are `base_width` and `base_length` and has a thickness of `base_height`. Note that the cuboid's base is located at the origin.

- Create the patch by creating a rectangle with the *base centre, width, depth, height* definition method. The *base centre* should be located at (0,0,`base_height`). Set the *width* and *depth* respectively to the defined variable `patch_length` and `patch_width`.
- Create the feed pin as a wire between the patch and the bottom of the substrate. Set the *Start point* to (-`pin_pos`,0,0) and the *End point* to (-`pin_pos`,0,`base_height`).
- Union all the elements and label the union `antenna`.
- Set the region of the cuboid to *substrate*.
- Set the faces representing the patch and the ground below the substrate to PEC.
- Add a segment wire port on the middle of the wire.
- Add a voltage source on the port. (1 V, 0°, 50 Ω)
- Set the frequency to `freq`.
- Set the periodic boundary condition of the model to the end exactly on the edge of the substrate to expand in both the x- and y-dimensions.
- Manually specify the phase shift in both directions to be *u1*=0° and *u2*=0°.

Requesting calculations

The solution requests are:

- Create a vertical (E plane) far field request. (-180°$\leq\theta\leq$180°, with θ=1° and ϕ=0° increments).
- Create another vertical (E plane) far field request. (-180°$\leq\theta\leq$180°, with θ=1° and ϕ=0° increments). This time, request the calculation of a 10×10 array of elements (on the *Advanced* tab).

Meshing information

Use the *standard* auto-mesh setting with wire segment radius equal to `0.0001`.

CEM validate

After the model has been meshed, run *CEM validate*. Take note of any warnings and errors. Correct any errors before running the FEKO solution kernel.

A-16.2 Pin fed patch: Broadside pattern by squint angle definition

Creating the model

Use the same geometry as for the first model. Meshing instructions are also the same. Change the periodic boundary condition settings as follow:

- Determine the phase shift by setting the beam angle for *Theta* and *Phi* to 0°.

Requesting calculations

Use the same far field calculations as for the first model.

CEM validate

After the model has been meshed, run *CEM validate*. Take note of any warnings and errors. Correct any errors before running the FEKO solution kernel.

A-16.3 Pin fed patch: Squint pattern by phase shift definition

Creating the model

Use the same geometry as for the first model. Meshing instructions are also the same. Change the periodic boundary condition settings as follow:

- Manually specify the phase shift in both directions to be $u1 = -61.56°$ and $u2 = 0°$.

Requesting calculations

Use the same far field calculations as for the first model.

CEM validate

After the model has been meshed, run *CEM validate*. Take note of any warnings and errors. Correct any errors before running the FEKO solution kernel.

A-16.4 Pin fed patch: Squint pattern by squint angle definition

Creating the model

Use the same geometry as for the first model. Meshing instructions are also the same. Change the periodic boundary condition settings as follow:

- Determine the phase shift by setting the beam angle for *Theta*=20° and *Phi*=0°.

Requesting calculations

Use the same far field calculations as for the first model.

CEM validate

After the model has been meshed, run *CEM validate*. Take note of any warnings and errors. Correct any errors before running the FEKO solution kernel.

A-16.5 Results

Figure A-16-2 shows the far field gain for the broadside models of a single patch element (with the effect of all the other patch elements taken into account) and also for the 10×10 element array. We can see that the gain for the element array is about 20 dB higher than the single element.

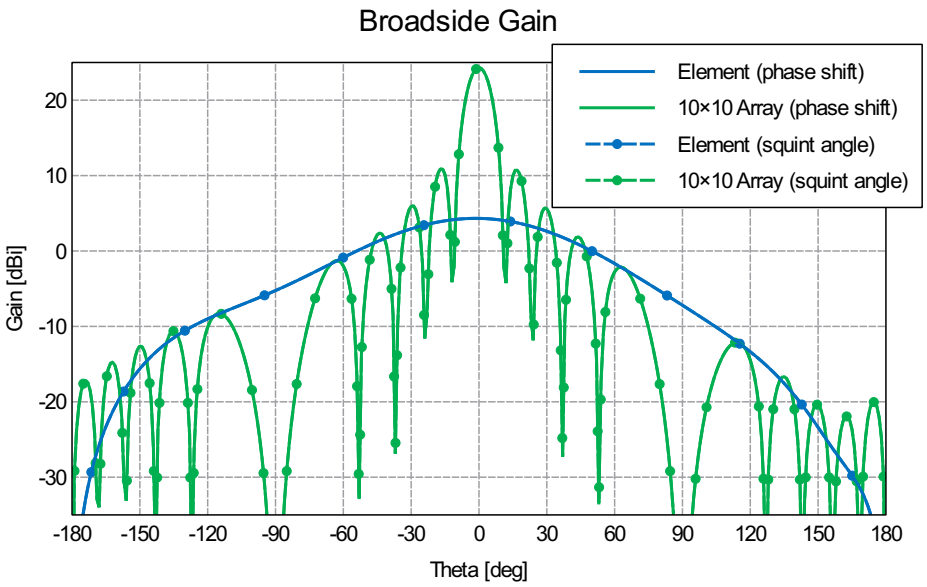


Figure A-16-2: The far field gain for a single element and for a 10 × 10 element 2D array of pin fed patch elements in the broadside direction.

Figure A-16-3 shows the far field gain for the 20° squint angle models of a single patch element (with the effect of all the other patch elements taken into account) and also for the 10 × 10 element array.

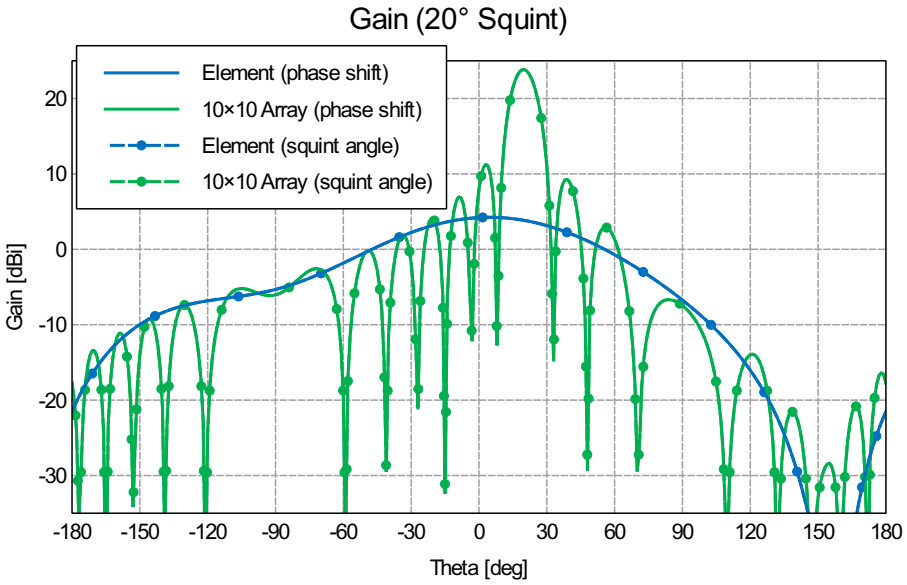


Figure A-16-3: The far field gain for a single element and for a 10×10 element 2D array of pin fed patch elements in the 20 degree squint direction.

A-17 Finite array with non-linear element spacing

Keywords: finite array, DGFM

The finite array that is presented here consists of identical array elements that are arbitrarily spaced. A pin-fed patch antenna serves as the base element for the array. The array is created by arbitrarily placing the base element (see figure A-17-2 for the array's layout).

Arrays of this type are difficult to model and can result in high resource requirements. It will be shown how the array can be constructed using the available finite array tools. When arrays are constructed in this way, the DGFM (domain Green's function method) acceleration can be applied to lower the resources that are needed.

Figure A-17-1 shows the full 3D far field pattern for the array.

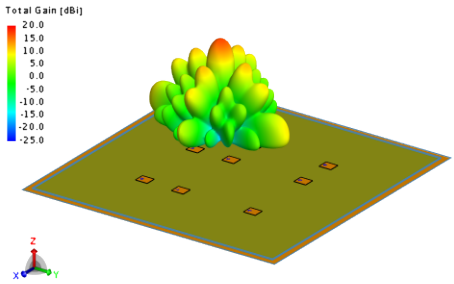


Figure A-17-1: The 3D far field pattern for the array.

A-17.1 Pin-fed patch array

Creating the model

The steps for setting up the model are as follows:

- Define the following variables:
  - `freq = 2.4e9` (Operating frequency of the patch)
  - `lam0 = c0/freq*1000` (Free space wavelength in millimetres)
  - `epsr = 2.08` (Relative dielectric constant of patch substrate)
  - `patchLength = 41` (Length of the patch antenna)
  - `patchWidth = 35` (Width of the patch antenna)
  - `h = 3.5` (The height of the substrate)
  - `pinOffset = -11` (Distance of feed pin from patch centre)
  - `wireRadius = 0.1` (Radius of the feed pin wire)
- Set the model unit to millimetres.

- Create a rectangle centred at the origin with a width of `patchWidth` and a depth of `patchLength`.
- Create a line segment between (0, `pinOffset`, `-h`) and (0, `pinOffset`, 0).
- Union the parts together.
- Create a dielectric medium with a permittivity of `epsr` and label it `substrate`.
- Create a planar multilayer substrate with a height of `h`. The medium should be `substrate`. Ensure that a ground plane is defined at the bottom of the dielectric layer.
- Add a port to the wire segment.
- Add a voltage source to the port with the default values.
- Set the frequency to `freq`.

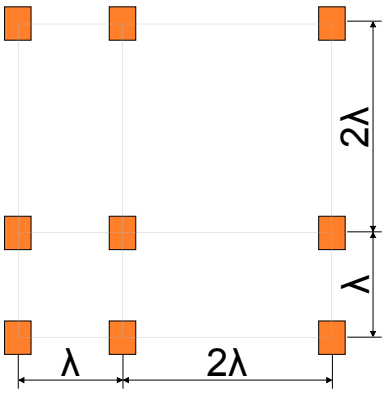


Figure A-17-2: The array layout that will be analysed.

The geometry above represents the base element, which is not included in the array calculations by default. Create the array depicted in figure A-17-2 by performing the following steps:

- Create a planar array:
  - Request four elements in both dimensions
  - Space the elements `lam0` apart
- Convert the planar array into a custom array. This makes it possible to delete, reposition or rotate individual elements of the array.
- Delete all elements corresponding to the third row and column. There should now be nine elements left.

  Note that each element can also have a different orientation. Elements are rotated by modifying the local workplane of the custom antenna array elements. No elements need to be rotated for this example, but the user in encouraged to rotate some of the elements after the simulation has completed to investigate the effect on the array pattern.

Meshing

Use *Standard* meshing to mesh the geometry. Set the segment radius to `wireRadius`.

Requests

Request a 3D far field that covers the top half space. A sampling increment of $\theta = 1.5°$ and $\phi = 1.5°$ is needed to obtain a reasonable resolution. Set the origin of the far field (*Workplane tab*) to `1.5*lam0` for both the *X* and the *Y* components. This does not change the far field pattern (this will affect the phase), but places the display of the far field on the 3D view in the middle of the patch array.

Note that a warning may be encountered when running the solution. This is because losses cannot be calculated in an infinitely large medium, as is required for the extraction of antenna directivity information (gain is computed by default). This warning can be avoided by ensuring that the far field gain be calculated instead of the directivity. This is set on the *advanced* tab of the far field request in the tree.

A-17.2 Results

Figure A-17-3 shows the comparison of two theta cuts of the array approximation with the results obtained using an equivalent full MoM model. The results compare favourably.

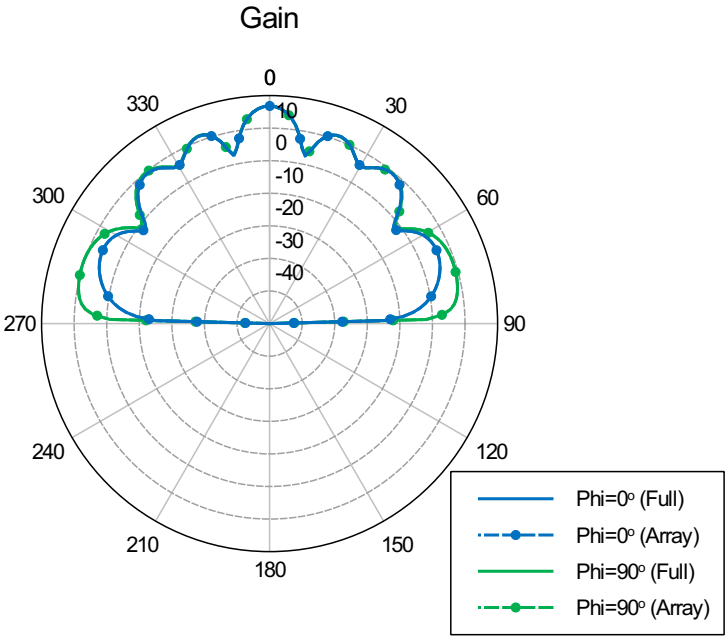


Figure A-17-3: The far field gain pattern for the array.

Apart from the relative ease of construction when using arrays, there is also a performance improvement when using the array techniques. This will become more evident for arrays that contain a larger number of elements.

Chapter B

Antenna placement

B-1 Antenna coupling on an electrically large object

Keywords: electrically large, MLFMM, CFIE, coupling, antenna placement, S-parameters

This example consists of a Rooivalk helicopter mock-up with three monopole antennas located near the front, middle and rear of the model respectively. S-parameters (coupling) are computed between the three antennas over a frequency range.

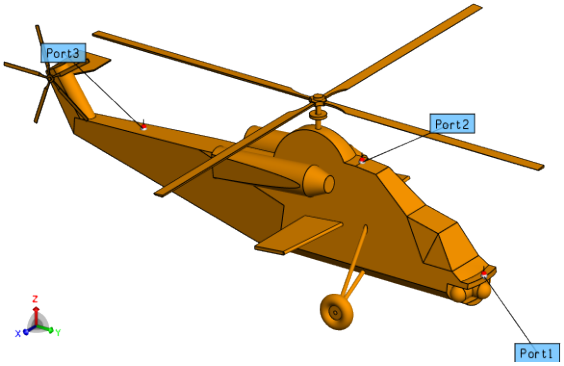


Figure B-1-1: 3D view of the helicopter.

NOTE: Due to calculations over a frequency range as well as the electrical size of the problem, several hours of computation time is required.

B-1.1 Helicopter

This example consists of complicated geometry and the model for this geometry is provided with the FEKO installation. The important features of this model are briefly presented.

- The model is solved with the MLFMM - this is set under *Solver settings*.
- The Combined Field Integral Equation (CFIE) is used. This is set on the face properties. All geometry face normals must point outward.

Requesting calculations

The solution requests are:

- The S-parameters for this model must be calculated. All three ports should be included and all must be active with a reference impedance of 50 Ω.

CEM validate

After the model has been meshed, run *CEM validate*. Correct any warnings and errors before running the FEKO solution kernel.

Note that during the FEKO solver run, the following warning may be displayed: *Inhomogeneous segmentation for triangles.* This warning is due to the occurrence of both very large and small triangles in the rotor of the helicopter. This warning may be ignored for this example.

B-1.2 Results

This example requires an appreciable time to solve (as shown in the extract below from the text `*.out` file). These resource requirements (both time and memory) for the MLFMM solution are notably smaller than for the full MoM solution. The resource requirements are further reduced in this example by the application of the CFIE formulation to the closed PEC structure of the helicopter. In this case, the use of the CFIE requires 30% less memory resources, and halves the simulation time required when compared with the default EFIE solution.

```
                  SUMMARY OF REQUIRED TIMES IN SECONDS

                                          CPU-time      runtime
Reading and constructing the geometry        0.610        0.609
Checking the geometry                        0.231        0.231
Initialisation of the Green's function       0.000        0.000
Calcul. of coupling for PO/Fock              0.000        0.000
Ray launching/tracing phase of RL-GO         0.000        0.000
Calcul. of the MLFMM transfer function       4.162        4.164
Fourier transform of MLFMM basis funct.     42.960       42.958
Calcul. of matrix elements                2283.468     2283.469
Calcul. of right-hand side vector            0.012        0.012
Preconditioning system of linear eqns.      11.746       11.746
Solution of the system of linear eqns.     258.806      258.806
Eigensolution for characteristic modes       0.000        0.000
Determination of surface currents            0.000        0.000
Calcul. of impedances/powers/losses          1.052        1.052
Calcul. of averaged SAR values               0.000        0.000
Calcul. of power ideal receiving ant.        0.000        0.000
Calcul. of cable coupling                    0.000        0.000
Calcul. of error estimates                   0.000        0.000
Calcul. of electric near field               0.000        0.000
Calcul. of magnetic near field               0.000        0.000
Calcul. of far field                         0.000        0.000
                          other              3.350        3.351
                                          ----------   ----------
                      total times:        2606.397     2606.398
             (total times in hours:           0.724        0.724)

Peak memory usage during the whole solution: 314.899 MByte
```

The S-parameters representing the coupling between the antennas mounted on the helicopter and the reflection coefficients of the antennas are shown in Figure B-1-2 as a function of frequency.

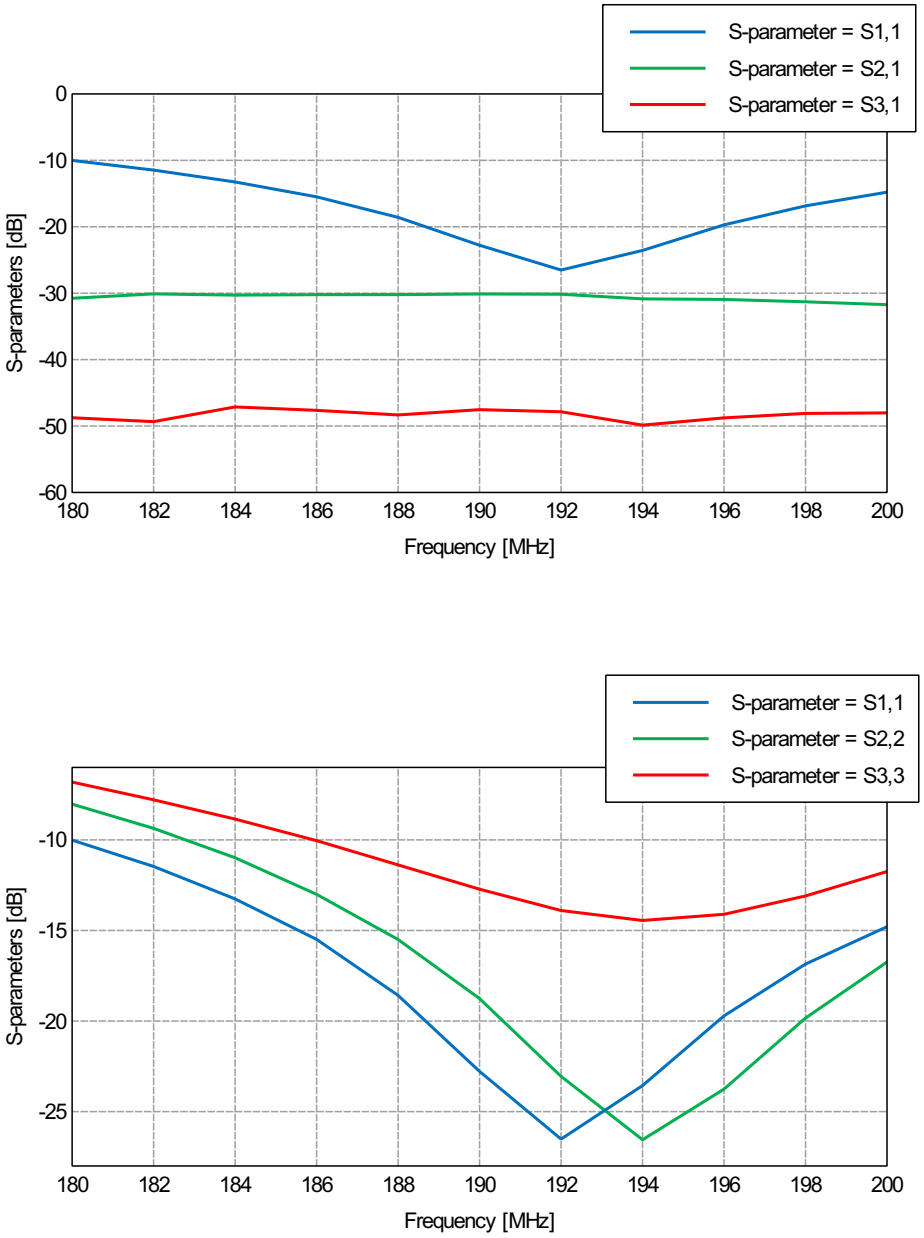


Figure B-1-2: The input reflection coefficients and coupling between the antennas when mounted on the electrically large helicopter.

B-2 Antenna coupling using an ideal receiving antenna

Keywords: coupling, ideal receiving antenna, far field data file, ffe file, helix antenna Yagi-Uda antenna, electrically large

This example involves the calculation of the coupling between a helix antenna and a Yagi-Uda antenna located in front of an electrically large metal plate, as shown in Figure B-2-1.

This is an electrically large problem, and two approaches are used to accelerate the solution significantly. The plate is efficiently modelled as a UTD plate and the helix antenna is modelled as an ideal receiving antenna. The receiving antenna can be modelled using a far field radiation pattern, far field spherical modes or a near field aperture. The ideal receiving antenna formulation can be used only when the required far field, spherical modes or near field aperture data is available (or can be calculated in a separate simulation) for an antenna. In addition, the antenna must be located an acceptable distance from all other physical structures that may influence the currents on the antenna.

This example will illustrate the use of all three receiving antenna techniques in a single model. The simulation time, received power and coupling between the antennas modelled using the three receiving antenna formulations will be compared to the full solution. Note that usually only one of these receiving antennas would be required in a model.

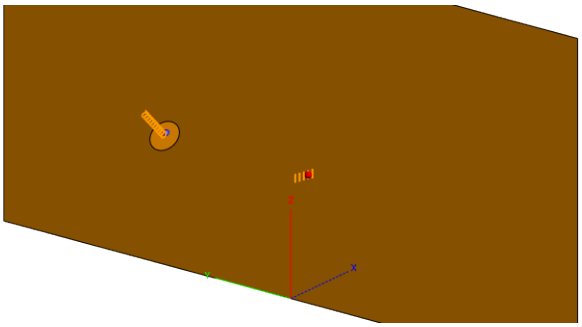


Figure B-2-1: Geometry of *Antenna_Coupling_Full* showing full helix model.

Three models are provided for this example:

Antenna_Coupling_Helix_Antenna.cfx Model of the helix antenna used to pre-calculate the far field pattern, spherical modes and near field aperture that will be used in the ideal receiving antennas.

Antenna_Coupling_Receiving_Antenna.cfx The model used to calculate the coupling between the Yagi-Uda and helix antennas using the ideal receiving antenna techniques.

Antenna_Coupling_Full.cfx The model used to calculate reference result using the full models for both antennas.

B-2.1 The helix antenna in free space

Creating the model

The steps for setting up the model are as follows:

- Define variables:
  - `freq = 1.654e9` (The design frequency of the helix.)
  - `lambda = c0/freq` (The wavelength in free space.)
  - `n = 10` (The number of turns for the helix.)
  - `helix_alpha = 13` (The pitch angle of the helix.)
  - `helix_radius = lambda*cos(helix_alpha*pi/180)/pi/2` (The radius of the helix.)
  - `plate_radius = 0.75*lambda` (The radius of the ground plate.)
  - `wire_radius = 0.65e-3` (The radius of the helix wire segments.)
- An ellipse primitive is used to create a circular plate, centred around (0, 0, 0) with a radius of `plate_radius`. This is used to model the finite ground plane of the helix.
- The helix antenna is created using a helix primitive with:
  - Definition method of *Base centre, radius, pitch angle, turns*.
  - Origin at (0, 0, 0).
  - Radius of `helix_radius`.
  - Pitch angle of `helix_alpha`.
  - Number of turns is `n`.
- The helix and ellipse are unioned to indicate connectivity.
- A wire port is added on the segment at the start of the helix.
- A voltage source is applied to the port.
- Set the frequency to `freq`.

Meshing information

Use the *standard* auto-mesh setting with wire segment radius equal to `wire_radius`.

Requesting calculations

A full 3D far field is requested. The far field pattern receiving antenna requires a far field pattern file to be exported. On the *Advanced* tab, choose to export the field to an ASCII file. The far field pattern will be stored in a file with the `*.ffe` extension.

The spherical modes required for the spherical modes receiving antenna can be calculated as part of the same far field request. On the *Advanced* tab, choose to calculate spherical mode coefficients and also to export the coefficients to an ASCII file. The spherical modes expansion coefficients will be stored in a file with the `*.sph` extension.

The near field aperture receiving antenna requires near field points around the antenna to be calculated and stored. The near field aperture can be any shape and the aperture can also be

constructed from multiple near field request surfaces. It is important that the near field surface spans the entire radiating area of the antenna. For the helix antenna the surface needs to be all around the antenna and thus using a spherical near field surface would work well. Create a near field request with the following settings:

- Use the spherical coordinate system.
- On the *Advanced* tab, choose to export the fields to ASCII file.
- Request the following ranges:

Axis	*Start*	*End*	*Increment*
r	0.45	0.45	0
θ	0	180	5
ϕ	0	360	5

The files generated by this simulation will be used in the following model of this example.

B-2.2 Using receiving antenna techniques to model the helix antenna

In the `Antenna_Coupling_Receiving_Antenna.cfx` model, the Yagi-Uda antenna and the large conducting sheet are added. The receiving antenna is correctly positioned and rotated relative to these by specifying a local coordinate system.

Creating the model

The steps for setting up the model are as follows:

- Define variables.
  - `freq = 1.654e9` (Design frequency of the helix.)
  - `lambda = c0/freq` (The wavelength in free space.)
  - `yagi_ld = lambda*0.442` (The length of director element.)
  - `yagi_li = lambda*0.451` (The length of active element.)
  - `yagi_lr = lambda*0.477` (The length of reflector element.)
  - `yagi_d = 0.25*lambda` (Spacing between Yagi elements.)
  - `yagi_rho = lambda*0.0025` (The radius of Yagi elements.)
- Define named points.
  - `helix_centre = (-1.5/2, 3/4, 1.5)` as the helix antenna location
  - `yagi_centre = (-1.5/2, -3/4, 1.5)` as the Yagi-Uda antenna location

- The Yagi-Uda antenna is created using line primitives. Create a line with *start point* (0, 0, -yagi_li/2) and *end point* as (0, 0, yagi_li/2). Set the label to yagi_active.
- Create a line with *start point* (0, -yagi_d, -yagi_ld/2) and *end point* (0, -yagi_d, yagi_ld/2). Set the label to yagi_director.
- Create a line with the *start point* (0, yagi_d, -yagi_lr/2) and the *end point* (0, yagi_d, yagi_lr/2). Set the label to yagi_reflector.
- Create a copy of yagi_director. Translate form (0, 0, 0) to (0, -yagi_d, 0).
- Create a copy of yagi_director. Translate form (0, 0, 0) to (0, -2*yagi_d, 0).
- Union the wires and modify the label to yagi_antenna.
- A wire port is added on the vertex at the centre of the active element of the Yagi antenna.
- A voltage source is applied to the port.
- Set the simulation frequency to freq.
- Rotate yagi_antenna with -(90+15)°. Note that an expression may be added to the field, not just a value.
- Translate yagi_antenna from (0, 0, 0) to (yagi_centre, yagi_centre, yagi_centre).
- The rectangle primitive is used to create the plate:
  - Set the local workplane to the global YZ plane (Ctrl+Shift and click on the *Global YZ* workplane.
  - Use the *Base corner, width, depth* definition method with the corner at (-3, 0, 0).
  - Set *width* = 6 and *depth* = 3.

  Set the label to metal_plate.
- The properties of the rectangle face in the details tree are set so that the UTD method will be applied to the face.

Meshing information

Use the *standard* auto-mesh setting with wire segment radius equal to yagi_rho.

Requesting calculations

- A far field data definition is created
  - The number of theta and phi points is 37 and 73 respectively. (If changes have been made to the provided models, care should be taken to ensure that the number of field points specified for the receiving antenna is consistent with the values stored in the *.ffe file.)
  - The pattern file is chosen as the *.ffe file generated using the free-space helix model.

- A spherical modes data definition is created
  - The spherical modes expansion file is chosen as the `*.sph` file generated using the free-space helix model.
- A near field data definition is created
  - The near field files are chosen as the `*.efe` and `*.hfe` files for electric and magnetic near fields respectively. These files were generated using the free-space helix model.
  - Select the spherical coordinate system.
  - The *radius* should be set to `0.45`.
  - The number of theta and phi points is 37 and 73 respectively. (If changes have been made to the provided models, care should be taken to ensure that the number of field points specified for the receiving antenna is consistent with the values stored in the `*.efe` and `*.hfe` files.)
  - The origin of the workplane is set to the `helix_centre` named point. The *U* axis direction is (1, 0, 1) to define the orientation of the helix that the pattern represents.
- Create the receiving antenna requests:
  - Create a far field receiving antenna that uses the far field data definition. The origin of the workplane is set to the `helix_centre` named point. The *U* axis direction is (1, 0, 1) to define the orientation of the helix that the pattern represents.
  - Create a spherical modes receiving antenna that uses the spherical modes data definition. The origin of the workplane is set to the `helix_centre` named point. The *U* axis direction is (1, 0, 1) to define the orientation of the helix that the pattern represents. Be sure to use the far field approximation for the internal approximation of the spherical modes receiving antenna. In cases where the request is very close to geometry, using spherical modes for the internal approximation may be inaccurate.
  - Create a near field receiving antenna using the near field data definition.

The Yagi-Uda antenna is oriented so that its first side-lobe is aimed directly at the helix. The radiated power is configured as 100 W by selecting the *Total source power (no mismatch)* option on the power settings dialog.

CEM validate

After the model has been meshed, run *CEM validate*. Take note of any warnings and errors. Correct any errors before running the FEKO solution kernel.

B-2.3 The full model

The ideal receiving antenna calculates the coupling assuming a matched load. To make the results directly comparable, the helix antenna in the full model is loaded with the complex conjugate of its input impedance. The power loss in the applied load represents the total power received by the antenna. This can be found in the `*.out` file.

B-2.4 Results

As coupling parameters cannot be computed directly when using the ideal receiving antenna, the coupling information must be derived from received power information. In POSTFEKO the received power may be plotted on a graph or found in the text-based output of the `*.out` file. On the same graph, the power lost in the matching load from the full model may be plotted.

As we have chosen the radiated power to be exactly 100 W for all cases, the coupling can be calculated from:

$$\text{Coupling}_{(dB)} = 10 * log10 \left[\frac{\text{Received power}}{100} \right]$$

The comparative results are shown in Table B-2-1.

	Received power (mW)	*Coupling (dB)*	*Runtime (s)*
Full model	3.0274	-45.2	77
Receiving antenna (far field pattern)	2.9534	-45.3	<10
Receiving antenna (spherical modes)	2.9739	-45.3	<10
Receiving antenna (near field)	2.9103	-45.4	<10

Table B-2-1: Coupling results with 100 W transmitted power.

The power-loss computed in the full model is shown in this extract from the output file named `Antenna_Coupling_Full.out`

```
                    POWER LOSS METAL (in Watt)

         Losses due to impedances at vertices/connection points
               basis fu. no.                power loss
                1147                        3.0274E-03 W

         Total loss in the vertices:        3.0274E-03 W

         Loss metal (total):                3.0274E-03 W
```

Both the power lost in the conjugate load (from the full model) and the power received by the ideal receiving antennas can be plotted in POSTFEKO.

```
Receiving antenna (far field pattern) with name: FarfieldReceivingAntenna1

              RECEIVED POWER FOR IDEAL RECEIVING ANTENNA (FAR FIELD PATTERN)

         Received power (ideal match assumed):    2.9534E-03 W

         Relative phase of received signal:      -9.1181E+01 deg.
```

```
Receiving antenna (spherical modes) with name: SphericalModesReceivingAntenna1

              RECEIVED POWER FOR IDEAL RECEIVING ANTENNA (FAR FIELD PATTERN)

         Received power (ideal match assumed):    2.9739E-03 W

         Relative phase of received signal:      -9.2620E+01 deg.
```

```
Receiving antenna (near field aperture) with name: NearfieldReceivingAntenna1

               RECEIVED POWER FOR RECEIVING ANTENNA (NEAR FIELD APERTURE)

          Received power:                           2.9103E-03 W

          Relative phase of received signal:      -8.0782E+01 deg.
```

The phase is relative to the global FEKO phase reference.

The ideal receiving antenna solutions require considerably less resources than the full model. If the receiving antenna is moved further away from the transmitting antenna and geometry, then the difference in the results will be smaller.

B-3 Antenna coupling using a point source and ideal receiving antenna

Keywords: coupling, S-parameters, far field point source, ideal receiving antenna

This example demonstrates the computation of coupling between two horn antennas using an equivalent far field source approximation and an ideal far field receiving antenna. The far field source can also be replaced by a spherical modes source, while a spherical modes or a near field data definition could have been used for the ideal receiving antenna. The best options for the source and the receiving antenna depends on the problem at hand. It is left to the reader to experiment with the different techniques that can be used to model the source and receiving antenna.

The model geometry consists of two horn antennas pointing towards one another, separated by a distance of 60 wavelengths. Exactly halfway between the antennas is a metallic plate that is effectively blocking the line–of–sight coupling between the antennas.

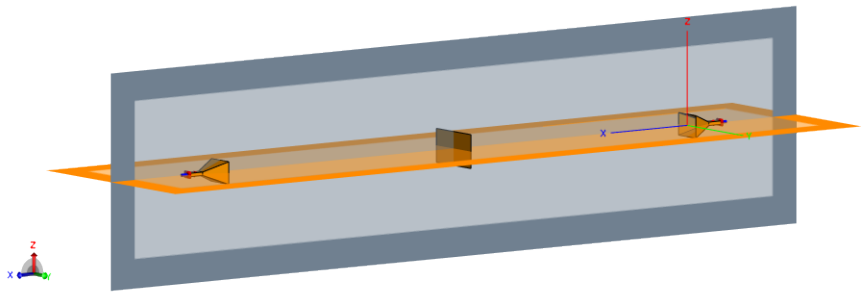


Figure B-3-1: The full 3D model representation of the problem considered in this example.

This example consists of 3 models.

Pyramidal_Horn.cfx A model of a horn antenna in free space used to precompute the far field pattern to be used in the far field source and the far field receiving antenna parts of the ensuing models.

Point_Source_Coupling.cfx A model that uses the far field pattern of the horn antenna to efficiently extract the coupling between two horns as shown in Figure B-3-1.

Full_Model.cfx The reference model used to compute the coupling between two horn antennas as shown in Figure B-3-1, without using a precomputed far field pattern.

B-3.1 The horn antenna in free space

In the `Pyramidal_Horn.cfx` model shown in Figure B-3-2 the 3D radiation pattern of a horn at 1.645 GHz is computed and saved to an `*.ffe` file. The horn is excited using a waveguide port.

The horn is placed with its source on the YZ plane. To account for the phase centre offset, the far field is calculated with the offset axis origin at X= -21.6 cm. Two planes of symmetry are used in the model in the $Y = 0$ and $Z = 0$ planes.

The calculation of the phase centre that is required for accurate placement of the radiation pattern of the horn is beyond the scope of this example, but is discussed in Example 35 of the *ScriptExamples.pdf* guide. A phase centre calculation script is also available on the FEKO website that allows the phase centre to be calculated in POSTFEKO. Technically, the phase offset needs to be calculated for each frequency. The far field pattern should also be calculated for each frequency. Due to the narrow bandwidth of the calculation, this step is omitted in this example.

B-3.2 Using the computed horn far field pattern in a coupling calculation

In `Point_Source_Coupling.cfx`, the two horn antennas are substituted with a correctly oriented and positioned ideal receiving antenna and a equivalent far field source.

The coupling can be computed based on the power that is received (in W) by the receiving antenna and comparing it to the power that was transmitted. By setting the transmitting antenna power to 1 W (on the power settings tab), only the received power needs to be recorded. The coupling is then related to the received power by:

$$\text{Coupling}_{(dB)} = 10 * log10 \left[\frac{\text{Received power}}{\text{Transmitted power}} \right]$$

B-3.3 The reference model

In `Full_Model.cfx`, both horns and the plate are included. Symmetry is used in the $Y = 0$ and $Z = 0$ planes.

The coupling between the antennas is computed directly by requesting S-parameters.

CEM validate

After the model has been meshed, run *CEM validate*. Take note of any warnings and errors. Correct any errors before running the FEKO solution kernel.

B-3.4 Results

The received power at the ideal receiving antenna is extracted from POSTFEKO. Figure B-3-3 shows the fully calculated model compared to the point source approximation. It can be seen that the results match reasonably well, where the differences can be attributed to the point source approximation.

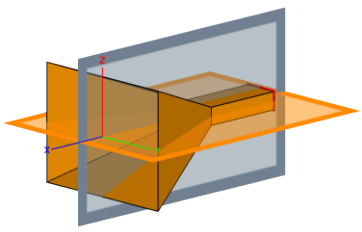


Figure B-3-2: The 3D model of the horn that is used to generate the `*.ffe` file.

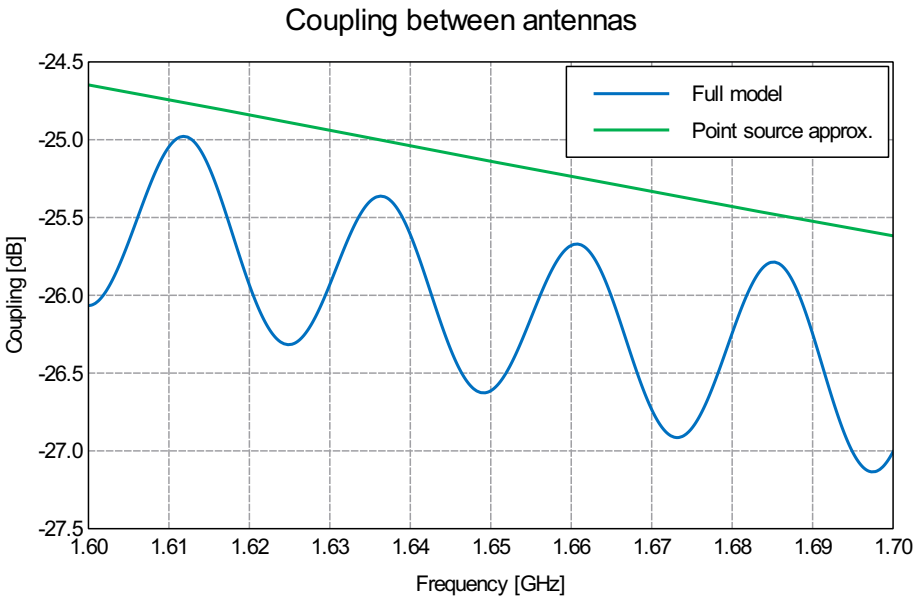


Figure B-3-3: The comparative results for the full model simulation and the simulation using pre-calculated radiation pattern representations of the horn antennas in the model.

Chapter C

Radar cross section (RCS)

C-1 RCS of a thin dielectric sheet

Keywords: RCS, thin dielectric sheet (TDS), plane wave

An electrically thin dielectric plate, modelled with the thin dielectric sheet approximation, is illuminated by an incident plane wave such that the bistatic radar cross section may be calculated at 100 MHz.

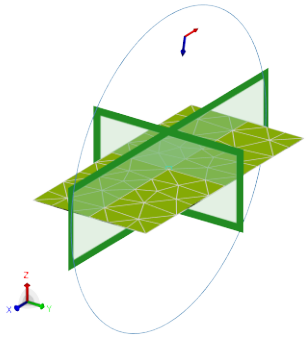


Figure C-1-1: A 3D representation of a thin dielectric sheet with a plane wave source (source, far field request and symmetry planes shown).

C-1.1 Dielectric sheet

Creating the model

The steps for setting up the model are as follows:

- Define the following variables:
  - `freq = 100e6` (Operating frequency.)
  - `d = 0.004` (Plate thickness.)
  - `a = 2` (Width of plate.)
  - `b = 1` (Depth of plate.)
  - `epsr = 7` (Relative permittivity.)
  - `tand = 0.03` (Loss tangent.)
  - `thetai = 20` (Zenith angle of incidence.)
  - `phii = 50` (Azimuth angle of incidence.)
  - `etai = 60` (Polarisation angle of incident wave.)
- Create a dielectric called `substrate` with relative permittivity equal to `epsr` and dielectric loss tangent set to the value of `tand`.
- Create a layered dielectric with a single layer named `thin_dielsheet`. Select `substrate` as the dielectric material for the layer and set the thickness equal to variable `d`.

- Create a rectangular plate in the XY plane centred around the origin. The *width* (X axis) is the value of `a` and *depth* is `b`.
- Set the face medium property of the plate to `thin_dielsheet`.
- Add a single incident plane wave source from the direction θ=`thetai` and ϕ=`phii`. Set the polarisation angle to `etai`.
- Set the frequency to `freq`.

Requesting calculations

The geometry of the problem is symmetric around the X=0 and Y=0 planes, but the source has no symmetry. Two planes of geometric symmetry are therefore specified in the model settings.

The solution requests are:

- Create a vertical far field request. (-180°$\leq\theta\leq$180°, with ϕ=0°)

Meshing information

- Use the *standard* auto-mesh setting.

CEM validate

After the model has been meshed, run *CEM validate*. Take note of any warnings and errors. Correct any errors before running the FEKO solution kernel.

C-1.2 Results

The bistatic RCS of the dielectric sheet at 100 MHz as a function of the angle θ, in the plane $\phi = 0°$ is shown in Figure C-1-2.

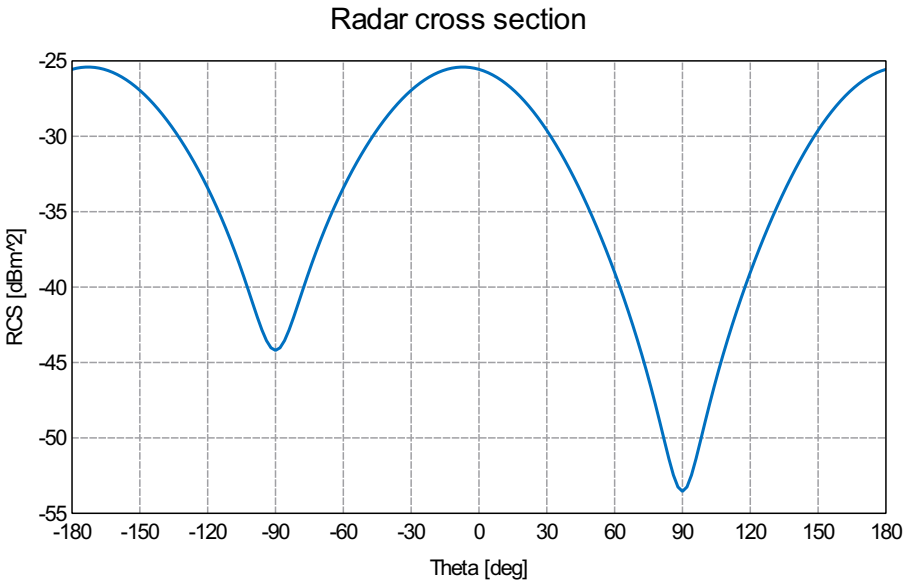


Figure C-1-2: Bistatic RCS of a thin dielectric sheet.

C-2 RCS and near field of a dielectric sphere

Keywords: dielectric, plane wave, sphere, bistatic RCS, monostatic RCS

A lossless dielectric sphere with radius of 1 m and relative permittivity equal to 36 is excited by means of an incident plane wave. The wavelength of the incident field is 20 m in free space (3.33 m in the dielectric). The near field inside and outside the sphere as well as the RCS of the sphere are calculated and compared to theoretical results.

The calculation is done using the surface equivalence principle.

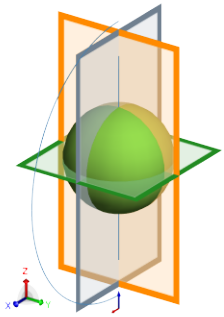


Figure C-2-1: A 3D view of the dielectric sphere and plane wave source. The CADFEKO preview of the far field request and the symmetry planes are also shown on the image.

C-2.1 Dielectric sphere

Creating the model

The steps for setting up the model are as follows:

- Define the following variables:
  - `lambda = 20` (Free space wavelength.)
  - `freq = c0/lambda` (Operating frequency.)
  - `radius = 1` (Sphere radius.)
  - `epsilon = 36` (Relative permittivity.)
- Create a new dielectric called `diel` and set its relative permittivity equal to `epsilon`.
- Create a sphere with at the origin and set its radius to the value of `radius`.
- Set the region properties of the sphere to be of medium `diel`.
- Add a plane wave source with θ=180° and ϕ=0°.
- Set the frequency equal to variable `freq` ($\approx$ 15 MHz).

Requesting calculations

The geometry in this problem is symmetric around all 3 principal planes, but the source is not. As the electric fields of the incident plane wave are purely X directed for the chosen incident angle, electric symmetry may be used in the X=0 plane, magnetic symmetry may be used in the Y=0 plane, but only geometric symmetry may be used in the Z=0 plane.

The solution requests are:

- Create a vertical far field request. ($0° \leq \theta \leq 180°$ and $\phi = 0°$)
- Create a near field request along the *Z* axis. Set the *Start* position for the near field to (0,0,-2*`radius`) and the *End* position to (0,0,2*`radius`). Request 80 field points in the *Z* direction.

Meshing information

Use the *custom* mesh option with the following settings:

- Triangle edge length: `0.2`.
- Wire segment length: Not applicable.
- Tetrahedron edge length: Not applicable.
- Wire segment radius: Not applicable.

Since the wavelength at the simulation frequency is large compared to the size of the model, we need to mesh the model such that it accurately represents a sphere. A triangle edge length of `0.2` is fine enough to accurately represent the sphere.

CEM validate

After the model has been meshed, run *CEM validate*. Take note of any warnings and errors. Correct any errors before running the FEKO solution kernel.

C-2.2 Results

Figures C-2-2 and C-2-3 compare the near field along the *Z* axis and the radar cross section as a function of the angle to exact mathematical results.

RCS calculations are displayed on a far field graph. The *Y* axis of the RCS graph has been changed to a logarithmic scale for improved visualisation.

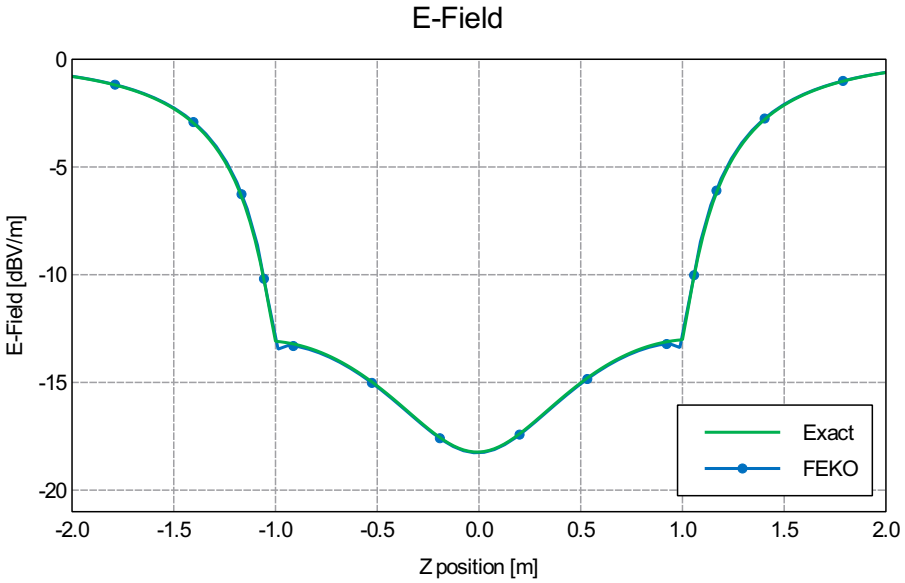


Figure C-2-2: Near field along the Z axis.

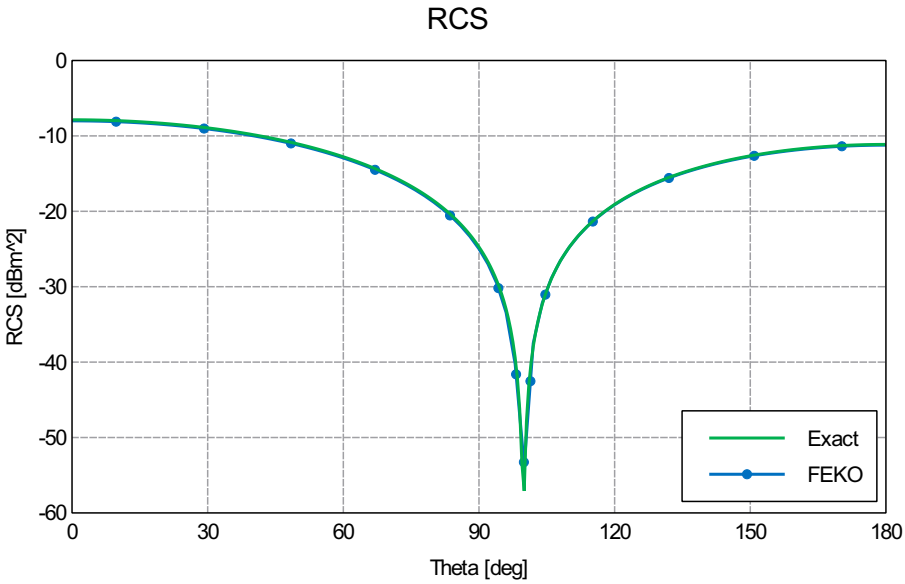


Figure C-2-3: Bistatic radar cross section of the dielectric sphere.

C-3 Scattering width of an infinite cylinder

Keywords: periodic boundary condition, plane wave, MoM

Using a 1-dimensional periodic boundary condition, the scattering of an infinite cylinder is efficiently computed (as defined below). The results are compared with a literature reference (C. A. Balanis, Advanced Engineering Electromagnetics, Wiley, 1989, pp. 607.)

$$SW = \frac{1}{\lambda} \lim_{\rho \to \infty} [2\pi\rho \frac{|E_z^s|^2}{|E_z^i|^2}]$$

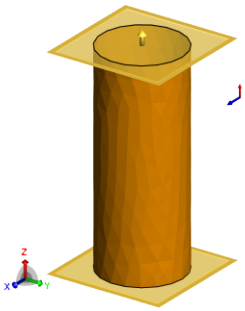


Figure C-3-1: A 3D view of the unit-cell of the infinite cylinder with the 1D periodic boundary condition shown.

C-3.1 Infinite cylinder

Creating the model

The model consists of a cylindrical section of variable radius and height of half a wavelength at the excitation frequency. The cylinder is realised by creating a cylinder primitive and deleting the upper and lower faces of the cylinder.

- Define the following variables:
  - `lambda = 1` (Wavelength in free space.)
  - `freq = c0/lambda` (Frequency for free space wavelength.)
  - `h = lambda/2` (Height of the cylinder.)
  - `r = 0.1` (Radius of the infinite cylinder.)
- Create a cylinder with a base centre at (0, 0,`-h/2`). Use a radius `r` and a height `h`.
- Delete the top and bottom faces of the cylinder.
- Set a single frequency of `freq`.
- Enable the use of higher order basis functions under the solver settings.

Requesting calculations

For this example, the scattering width of the cylinder for an incident plane wave normal to the cylinder will be considered. A plane wave source for θ=90° and ϕ=180° is used.

The 1D periodic boundary condition is defined along the axis of the cylinder so that the unit-cell touches the edges of the periodic region.

A near field request is used to determine the direction-dependent scattered field, from which the scattering width is derived:

- Request a cylindrical coordinate system near field
- Set the radius as: ρ=`500*lambda`
- Specify ϕ=`0.5`° angular resolution
- It is important to only calculate the scattered part of the near field. This removes the effect of the plane wave on the calculated field and ensures that only the scattered fields are considered.

Meshing information

Use the *fine* auto-mesh setting. Also try modifying the parameters on the *Advanced* tab to obtain a mesh that better represents the geometry as a cylinder.

CEM validate

After the model has been meshed, run *CEM validate*. Take note of any warnings and errors. Correct error before running the FEKO solution kernel.

C-3.2 Results

Figure C-3-2 shows the computed scattering width as a function of the bistatic observation angle (ϕ) for a cylinder radius of `r=0.1` and `r=0.6`. The results agree well with the literature reference.

The scattering width was obtained by using the equation at the top of the example with the values provided to simplify to:

$$SW = 2\pi 500 \mid E \mid^2.$$

To use the equation, ensure the magnitude of the electric field is displayed and activate the *Enable maths*. Enter the following equation in POSTFEKO:

```
2*pi*500*ABS(self)^2
```

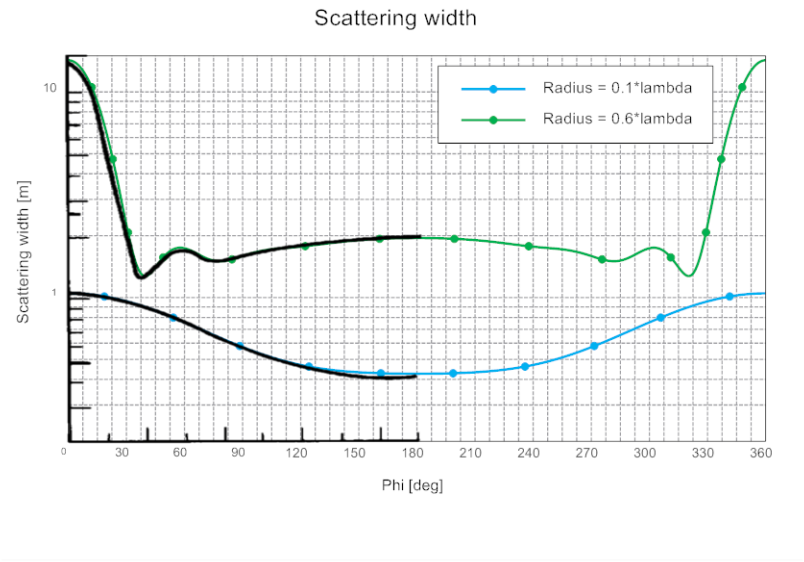


Figure C-3-2: The scattering width of an infinite cylinder with two different radii modelled.

C-4 Periodic boundary conditions for FSS characterisation

Keywords: periodic boundary condition, plane wave, frequency selective surface, near field, optimisation

A Jerusalem cross FSS (frequency selective surface) structure modelled using infinite periodic boundary conditions is excited with an incident plane wave (as shown in Figure C-4-1). The frequency dependent transmission and reflection coefficients of the surface are computed and considered. These results may be compared to those reported in the literature (Ivica Stevanovic, Pedro Crespo-Valero, Katarina Blagovic, Frederic Bongard and Juan R. Mosig, Integral-Equation Analysis of 3-D Metallic Objects Arranged in 2-D Lattices Using the Ewald Transformation, IEEE Trans. Microwave Theory and Techniques, vol. 54, no. 10, October 2006, pp. 3688–3697).

Note that the model supplied with this example includes an optimisation setup to determine the best set of geometrical parameters to maximise reflection and minimise transmission at 8 GHz. To perform the optimisation the frequency request should be set to a single frequency equal to 8 GHz.

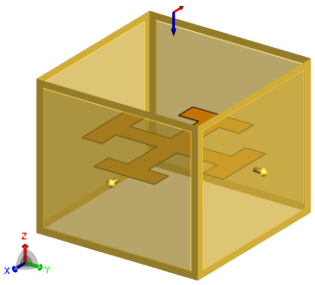


Figure C-4-1: A 3D view of the FSS structure. The Jerusalem cross unit-cell structure is shown with the plane wave source and periodic boundary condition.

C-4.1 Frequency selective surface

Creating the model

The steps for setting up the model are as follows:

- Define the following variables:
  - `d = 15.2` (The spacing for periodic boundary condition.)
  - `armLength = 13.3` (The length of the arm of the cross.)
  - `armWidth = 1.9` (The width of the arm of the cross.)
  - `stubLength = 5.7` (Length of stub at the end of the cross.)
  - `stubWidth = armWidth` (Width of stub at the end of the cross.)
  - `fmin = 2e9` (The minimum frequency.)

  - `fmax = 12e9` (The maximum frequency.)

- Set the model unit to millimetres (mm).
- Create a rectangle centred at the origin with a width of `armWidth` and a depth of `armLength`. This rectangle is the main arm of the cross.
- Create a rectangle at the origin with a width of `stubLength` and a depth of `stubWidth`.
- Translate the stub rectangle to be flush with the end of the arm.
- Copy and mirror the stub so that there is a stub at each end of the arm.
- Union the three rectangles.
- Copy and rotate the structure by 90°.
- Union and simplify the parts.
- Set a single plane wave to excite the model with θ=0° and ϕ=0°.
- Set a continuous frequency range from `fmin` to `fmax`.

Requesting calculations

Create a single transmission / reflection coefficient request; leave the phase origin at (0, 0, 0).

Set the periodic boundary condition in two dimensions. The following points should be used:

- *Start point*: (`-d/2`, `-d/2`, 0)
- *End point of first vector*: (`d/2`, `-d/2`, 0)
- *End point of second vector*: (`-d/2`, `d/2`, 0)

Determine the phase shift based on the incident plane wave direction.

Meshing information

Use the *standard* auto-mesh setting.

CEM validate

After the model has been meshed, run *CEM validate*. Take note of any warnings and errors. Correct any errors before running the FEKO solution kernel. Save the file and run the solver.

C-4.2 Results

Figure C-4-2 shows the computed total transmission and reflection coefficients.

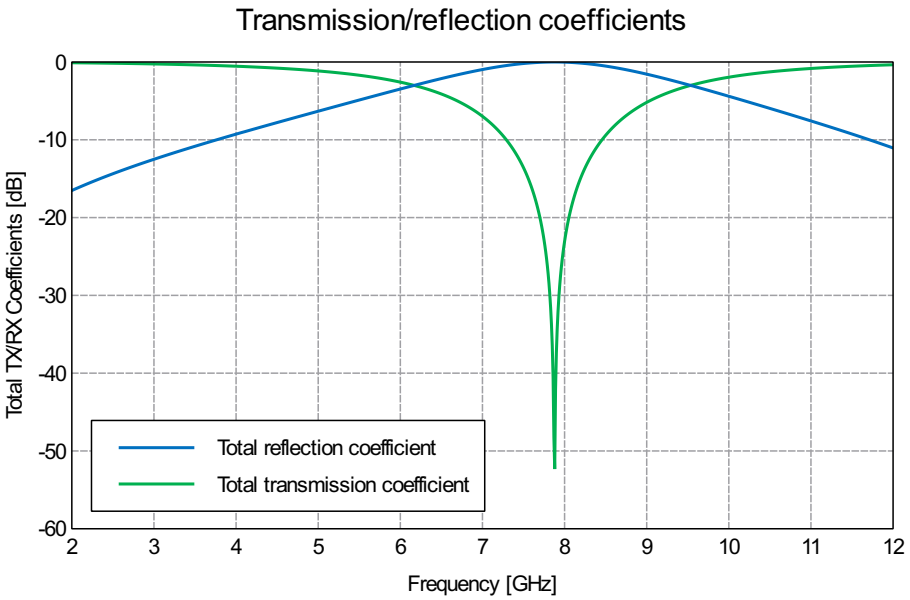


Figure C-4-2: The transmission and reflection coefficients for the specified incident plane wave.

Chapter D

EMC analysis + cable coupling

D-1 Shielding factor of a sphere with finite conductivity

Keywords: shielding, EMC, plane wave, near field, finite conductivity, FEM

A hollow sphere is constructed from a lossy metal with a given thickness. An incident plane wave is defined between the frequencies of 1–100 MHz. Near fields calculated at the centre of the sphere are used to compute the shielding factor of the sphere. The results are compared to values from the literature for the case of a silver sphere with a thickness of 2.5 nm.

Figure D-1-1 shows a 3D view of the sphere and the plane wave source in the CADFEKO model.

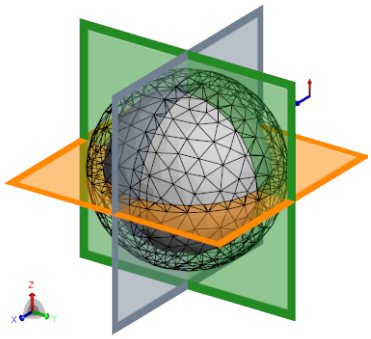


Figure D-1-1: A 3D view of the sphere with a plane wave source.

D-1.1 Finite conductivity sphere (MoM)

Creating the model

The steps for setting up the model are as follows:

- Define the following variables:
  - `r0 = 1` (Radius of sphere.)
  - `f_min= 1e6` (Lower operating frequency.)
  - `f_max= 100e6` (Upper operating frequency.)
  - `d = 2.5e-9` (Thickness of the shell.)
- "Silver" is a predefined metallic medium in the media library. Add the medium to the model.
- Create a sphere at the origin with radius set equal the defined variable `r0`.
- Set the region of the sphere to free space.
- Set the medium type of the sphere's face to `Silver` and set the thickness equal to the variable `d`.
- Create a single incident plane wave with direction set to θ=90° and ϕ=180°.
- Set the frequency to calculate a continuous range between `f_min` and `f_max`.

Requesting calculations

In the X=0 plane, use geometric symmetry. In the Y=0 plane, use magnetic symmetry and in the Z=0 plane, use electric symmetry.

The solution requests are:

- Create a single point near field request in the centre of the sphere. (Use the Cartesian coordinate system.)

Meshing information

Use the *standard* auto-mesh setting.

CEM validate

After the model has been meshed, run *CEM validate*. Take note of any warnings and errors. Correct any errors before running the FEKO solution kernel.

D-1.2 Finite conductivity sphere (FEM)

Creating the model

The steps for setting up the model are as follows:

- Define the following variables:
  - `r0 = 1` (Radius of sphere.)
  - `r1 = 1.2` (Radius of FEM vacuum sphere.)
  - `f_min = 1e6` (Lower operating frequency.)
  - `f_max = 100e6` (Upper operating frequency.)
  - `d = 2.5e-9` (Thickness of the shell.)
- "Silver" is a predefined metallic medium in the media library. Add the medium to the model.
- Create a new dielectric medium with the default properties of free space. Label the medium `air`.
- Create a sphere at the origin with radius equal to the defined variable `r0`.
- Create another sphere at the origin with radius equal to the defined variable `r1`.
- Set the region of both spheres to `air`. A dielectric material is used with the properties of free space instead of using the free space medium directly to ensure that the region is meshed as a tetrahedral volume for the FEM.

- Set the medium type of the inner sphere's face to `Silver` and set the thickness equal to the variable `d`.
- Union the two spheres.
- Set the solution method for the regions to *FEM (finite element method)*.
- Create a single incident plane wave with direction set to θ=90° and ϕ=180°.
- Set the frequency to calculate a continuous range between `f_min` and `f_max`.

Requesting calculations

In the X=0 plane, use geometric symmetry. In the Y=0 plane, use magnetic symmetry and in the Z=0 plane, use electric symmetry.

The solution requests are:

- Create a single point near field request in the centre of the sphere. (Use the Cartesian coordinate system.)

Meshing information

Use the *standard* auto-mesh setting.

CEM validate

After the model has been meshed, run *CEM validate*. Take note of any warnings and errors. Correct any errors before running the FEKO solution kernel.

D-1.3 Results

The subject of interest is the shielding capability of the sphere with respect to the incident electric and magnetic fields. In other words, we calculate the ratio between the field measured inside the sphere and the field incident on the sphere.

The incident field strength was set as $E_i = 1\ V/m$. From the wave impedance for a plane wave in free space, the incident magnetic field can be calculated.

$$H_i = \frac{E_i}{\eta_0} = \frac{1}{376.7} = 2.6544 \times 10^{-3}\ A/m$$

The shielding factor is therefore

$$S_e = -20 * log \frac{E}{E_i}\ [dB]$$

$$S_h = -20 * log \frac{H}{H_i}\ [dB]$$

Figures D-1-2 and D-1-3 show respectively the shielding of the electric and magnetic fields as a result of a sphere with the finite conductivity properties provided.

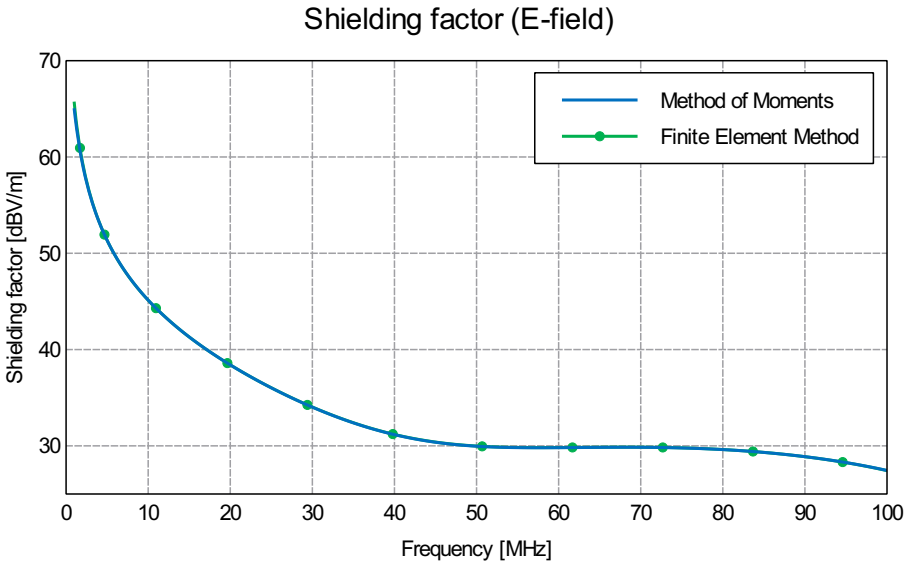


Figure D-1-2: Shielding of the electric field.

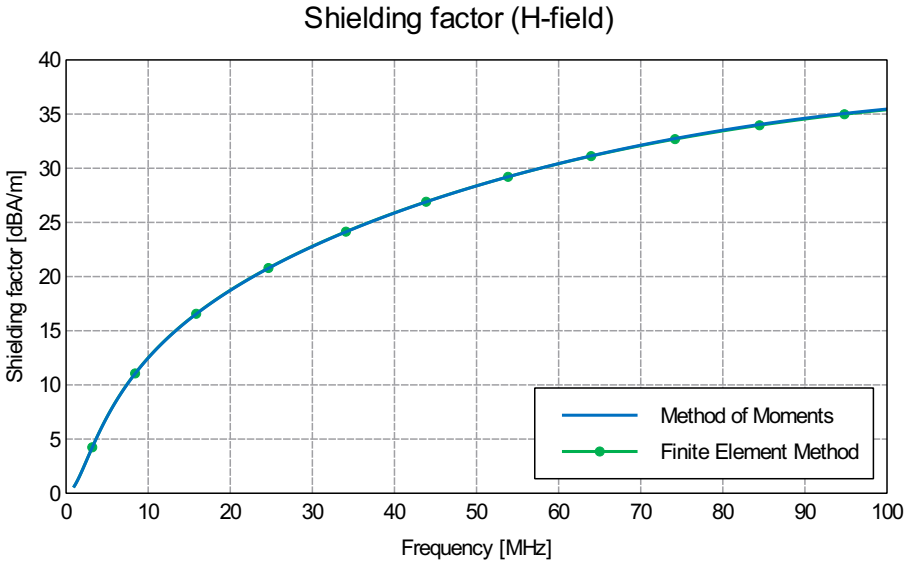


Figure D-1-3: Shielding of the magnetic field.

D-2 Calculating field coupling into a shielded cable

Keywords: cable modelling, cable analysis, shielded cable, coupling, EMC

The coupling from a monopole antenna into a nearby shielded cable that follows an arbitrary path near a ground plane is calculated from 1 MHz to 35 MHz in this example. The *cable analysis* option in FEKO is used for the analysis. This method solves the model without the cable first, and then calculates the coupling into the cable using the transfer impedance of the cable. The same problem could be modelled by by building a full MoM model of the cable, but that would be much more resource intensive for complex cables. The cable analysis solution also allows the use of a database of measured cable properties (integrated in FEKO).

The geometry shown in Figure D-2-1 consists of a driven monopole antenna and a section of RG58 shielded cable over an infinite ground plane. The RG58 cable is terminated with 50 Ω loads at both ends. Figure D-2-1 shows the geometry of this model.

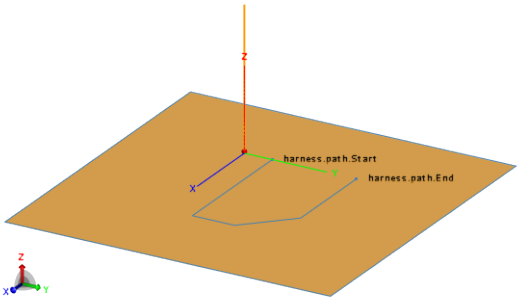


Figure D-2-1: RG58 shielded cable illuminated by a monopole above an infinite ground plane.

D-2.1 Monopole and ground

Creating the model

The steps for setting up the model are as follows:

- Define the following variables:
  - `fmin = 1e6` (The minimum operating frequency.)
  - `fmax = 35e6` (The maximum operating frequency.)
  - `wireRadius = 1e-3` (The wire radius of the monopole.)
- Create a monopole 10 m high with beginning and end point coordinates of (0,0,0) and (0,0,10).
- Add a wire segment port to the base of the monopole.
- Add a voltage source to the port. (1 V, 0°, 50 Ω)

- Set the total source power (no mismatch) equal to 10 W.
- Define a PEC ground plane.
- Set the frequency to be continuous from `fmin` to `fmax`.
- Create a cable path definition. The cable path for this example consists of the following list of (x, y, z) coordinates.
  - (0 , 2, 0.01)
  - (10 , 2, 0.01)
  - (10 , 5, 0.01)
  - (7 , 8, 0.01)
  - (0 , 8, 0.01)
- Create a cable harness. A harness can contain multiple cables and make up the different sections of a cable path.
- Create two cable connectors. One at the start and one at the end of the cable path. Label them `startConnector` and `endConnector` respectively. Both connectors will have two pins; one that is live and one for ground (i.e. the cable shield).
- Create the RG58 cable cross section. The RG58 cable is one of the predefined coaxial cable cross sections that can be selected from the list. The cross section is used to define what type of cable is being routed.
- Create a cable instance that runs from the `startConnector` and `endConnector`. Connect the two live pins together and the two ground pins together. Ensure that the live pins connect to the centre conducting wire and the ground pins connect to the outer shielding of the cable. This can be verified by looking at the labels in the preview.
- Open a schematic view.
- Add a 50 Ω complex load to each connector. The load terminates the cable and must be connected between the live and ground pins at each end of the cable.
- Ensure that the outer shields of the cables (i.e. the ground pins) are connected to the global ground in the schematic view.

Requesting calculations

Add a voltage probe over the load terminating the `startConnector`. If the port impedance or power is also of interest, then a current probe must also be requested in series to the terminating load. The values can then be derived using Ohm's law.

Meshing information

Use the *standard* auto-mesh setting. Set the wire radius of the monopole to `wireRadius`.

CEM validate

After the model has been meshed, run *CEM validate*. Take note of any warnings and errors. Correct any errors before running the FEKO solution kernel.

D-2.2 Results

Results are shown in Figure D-2-2.

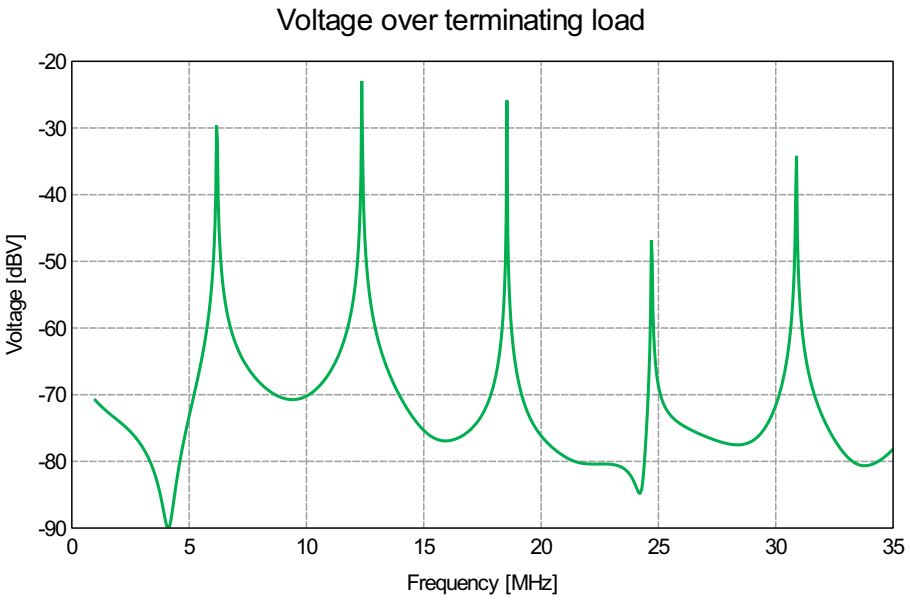


Figure D-2-2: Voltage induced in a terminated shielded cable by an external source.

D-3 A magnetic-field probe

Keywords: shielding, EMC, HOBF, probe, current, plane wave, magnetic field

A magnetic field probe in the form of a frame antenna with shielding against electric fields is constructed and simulated. The wavelength at the operating frequency (30 MHz) is approximately 10 m.

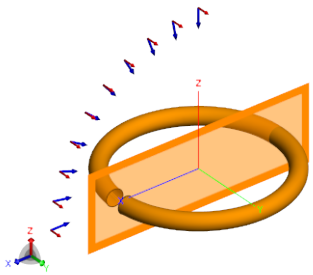


Figure D-3-1: A 3D view of the H-probe and the plane wave incidence excitation (symmetry plane shown).

D-3.1 Magnetic-field probe

Creating the model

The steps for setting up the model are as follows:

- Define the following variables:
  - `freq = 30e6` (The operating frequency.)
  - `lambda = c0/freq` (The free space wavelength.)
  - `rBig = 1` (Radius of revolution.)
  - `rSmall = 0.1` (The pipe radius.)
  - `wireRad = 5e-3` (The radius of the inner wire segments of the probe.)
- Create an elliptic arc with radius = `rSmall`. Set the workplane of the elliptic arc at an origin of (`-rBig`,0,0) and set the *U* and *V* vectors respectively to [0,0,1] and [1,0,0].
- Rotate the arc over an angle of 185° around the *Z* axis.
- Spin the ellipse over an angle of 350° around the *Z* axis.
- Draw an elliptic arc through the centre of the toroidal section. (radius = `rBig`, start angle = 0°, end angle = 360°)
- Add a plane wave source that loops over multiple incident angles. Let $0° \leq \theta \leq 90°$ and $\phi = 0°$ with θ in 10° steps. Set the polarisation angle equal to 90°.
- Set the frequency equal to `freq`.
- Activate HOBF for the model. Click on the *Solver settings* button (*Solve/Run* tab). On the *General* tab, check the *Solve with higher order basis functions (HOBF)* check box. Ensure the basis function order is set to auto.

Requesting calculations

Since all E-fields will be normal to the Y=0 plane, electric symmetry can be defined.

The solution requests are:

- Add a request to store all wire currents.

Meshing information

Use the fine or standard auto-meshing setting with the wire segment radius equal to `wireRad`. The user is encouraged to play around with the mesh settings on the *Advanced* tab of the mesh dialog.

CEM validate

After the model has been meshed, run *CEM validate*. Take note of any warnings and errors. Correct any errors before running the FEKO solution kernel.

D-3.2 Results

The current induced in a specific segment on the probe is shown versus solution number in Figure D-3-2. This is plotted on a currents and charges graph. Each solution represents a different plane wave source direction, starting at θ=0° in steps of 10° to θ=90° (for ϕ=0°).

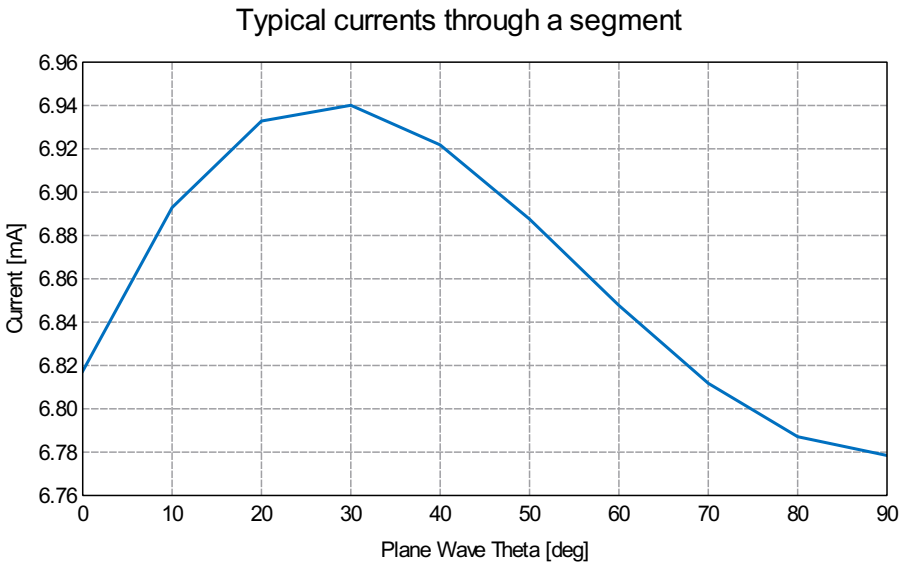


Figure D-3-2: The current in an arbitrary segment is plotted as a function of the plane wave source incidence angle. Note that each segment will result in a slightly different current as a function of the plane wave source.

D-4 Antenna radiation hazard (RADHAZ) safety zones

Keywords: yagi, radiation hazard, scripting

Safety standards differ from country to country, industry to industry and may change over time. This example illustrates how POSTFEKO scripts can be used to fully customise the calculation of such results. A yagi antenna is simulated with a full 3D near field cube of the antenna's immediate surroundings. Using math scripts in POSTFEKO, the radiation standards are used to identify the safety zones for the antenna.

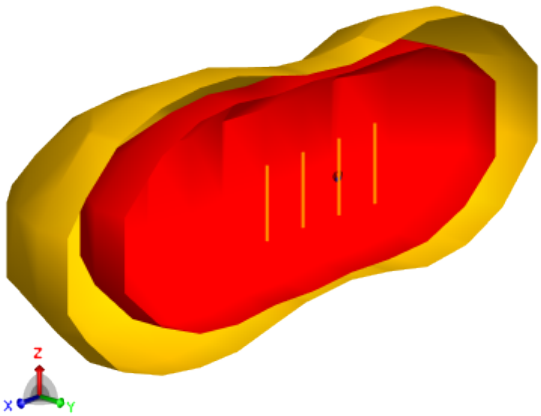


Figure D-4-1: The 3D safety zones for 80% and maximum (i.e. 100%) exposure levels according to the INIRC 88 standard.

D-4.1 CADFEKO

The antenna in a given environment gives rise to electric and magnetic fields. These fields will differ in strength and shape depending on the input power, antenna design and surrounding environment.

Creating the model

The steps for setting up the model are as follows:

- Define the following variables (physical dimensions based on initial rough design):
  - `freq = 1e9` (The operating frequency.)
  - `fmin = 0.4e9` (The minimum simulation frequency.)
  - `fmax = 1.5e9` (The maximum simulation frequency.)
  - `lambda = c0/freq` (The wavelength in free space at the operating frequency.)
  - `L0 = 0.2375` (Length of one arm of the reflector element in wavelengths.)

  - `L1 = 0.2265` (Length of one arm of the driven element in wavelengths.)
  - `L2 = 0.2230` (Length of one arm of the first director in wavelengths.)
  - `L3 = 0.2230` (Length of one arm of the second director in wavelengths.)
  - `S0 = 0.3` (Spacing between the reflector and driven element in wavelengths.)
  - `S1 = 0.3` (Spacing between the driven element and the first director in wavelengths.)
  - `S2 = 0.3` (Spacing between the two directors in wavelengths.)
  - `r = 0.1e-3` (Radius of the elements.)
- Create the active element of the antenna. Set the *Start point* as (0, 0, `-L1*lambda`) and the *End point* as (0, 0, `L1*lambda`).
- Add a port in the centre of the wire.
- Add a voltage source on the port. (1 V, 0°, 50 Ω)
- Set the incident power for a 50 Ω transmission line to 25 W.
- Create the wire for the reflector. Set the *Start point* as (`-S0*lambda`, 0, `-L0*lambda`) and the *End point* as (`-S0*lambda`, 0, `L0*lambda`).
- Create the two directors.
  - Set the *Start point* and *End point* for *Director1* as the following: (`S1*lambda`, 0, `-L2*lambda`) and (`S1*lambda`, 0, `L2*lambda`), respectively.
  - For *Director2*, set the *Start point* and *End point* as ((`S1` + `S2`)`*lambda`, 0, `-L3*lambda`) and ((`S1` + `S2`)`*lambda`, 0, `L3*lambda`), respectively.
- Use a continuous frequency range from `fmin` to `fmax`.

Requesting calculations

The Z=0 plane is an electric plane of symmetry. A magnetic plane of symmetry exists in the Y=0 plane.

The solution requests are:

- Create a 3D Cartesian near field block:
  - Start: (-0.6, -0.6, -0.6)
  - End: (1.2, 0.6, 0.6)
  - Number of U field points: 20
  - Number of V field points: 10
  - Number of N field points: 10

Meshing information

Use the *standard* auto-mesh setting with the wire segment radius equal to `r`.

D-4.2 POSTFEKO

A session containing the following instructions is provided (`radiation_zones.pfs`). The session contains the results from the antenna simulation and three script results:

INIRC88 This script generates a near field result that incorporates the calculated near fields and the INIRC 88 safety standards for occupational limits.

NRPB89 This script generates a near field result that incorporates the calculated near fields and the NRPB 89 safety standards.

standards This is a custom dataset that contains both of the standards used here. The result can be plotted on a 2D graph and shows the maximum field limits for both magnetic and electric fields over the calculated frequency band. Figure D-4-2 shows these results for both the magnetic and electric field limits.

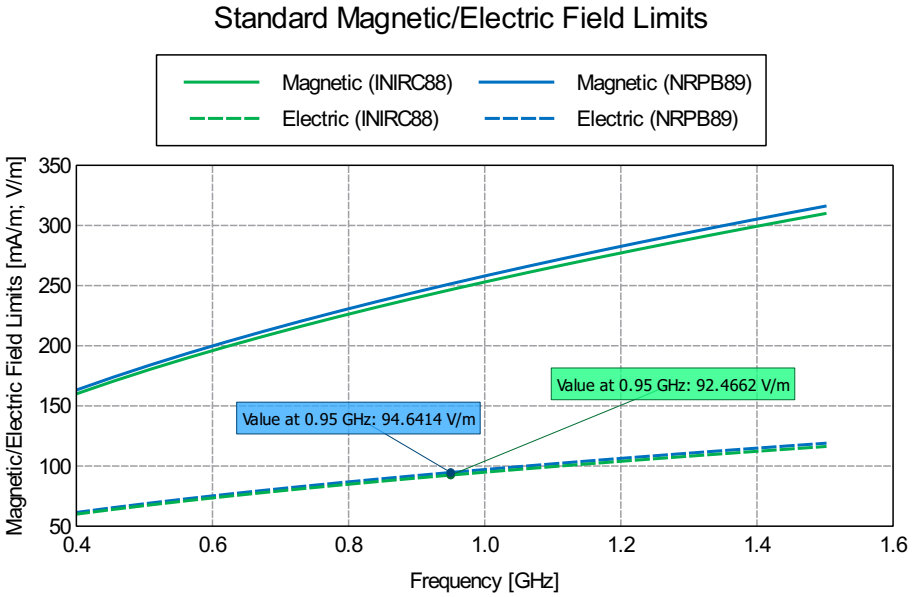


Figure D-4-2: The definition of the standards used in the calculated band.

The 3D representation for the safety zones can be depicted in a variety of formats. In the description image (Figure D-4-1), the safety zones for 0.95 GHz are displayed. Similar zones can be drawn for any value in the calculated frequency band as shown in Figure D-4-2. Figure D-4-3 shows how the field values at a specific position can be monitored to test for compliance to the radiation standards. It can be seen that the electric field exceeds the maximum limit between 1.024 – 1.173 GHz.

The scripts

The scripts that are used for the calculations are provided. Note that they adhere to the standards only for the frequency band over which the model was simulated. The resulting values for the

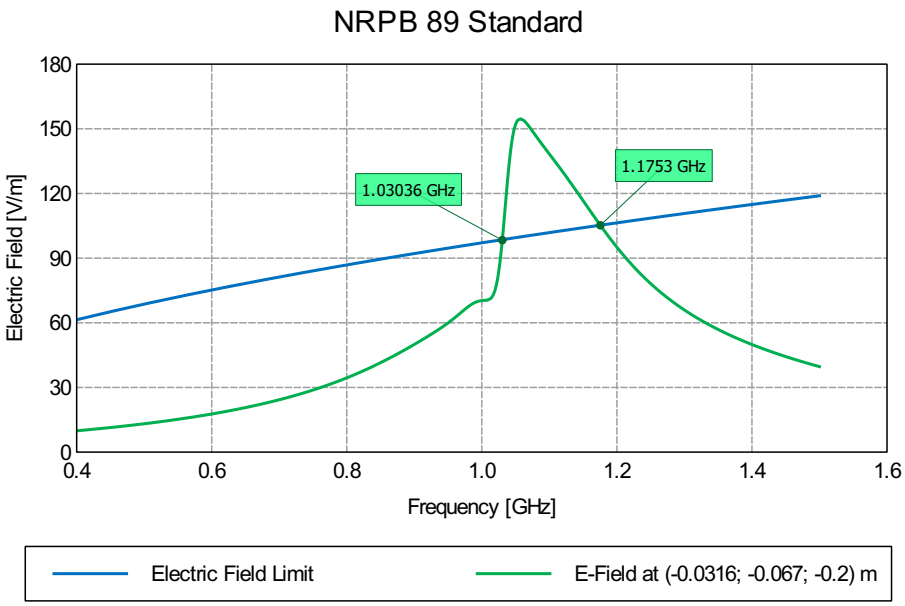


Figure D-4-3: The electric field values at a given location over frequency.

INIRC88 and NRPB89 near fields are technically no longer near fields. The calculated near field is normalised to the maximum field value of the standard, so that a value of "1" corresponds to the maximum threshold, a value of "0.8" corresponds to 80% of the maximum threshold and so on.

This means that the safety zones can easily be determined and visualised.

standards

The definitions for the standards are given in Tables D-4-1 and D-4-2.

Table D-4-1: Definition for electric and magnetic field limits according to INIRC 88 between 0.4 – 2.0 GHz.

Field Type	*Defintion* (f in MHz)	*Unit*
Electric field	$3\sqrt{f}$	$\frac{\mathrm{V}}{\mathrm{m}}$
Magnetic field	$0.008\sqrt{f}$	$\frac{\mathrm{A}}{\mathrm{m}}$

Table D-4-2: Definition for electric and magnetic field limits according to NRPB 89 between 0.4 – 2.0 GHz.

Field Type	*Defintion* (F in GHz)	*Unit*
Electric field	$97.1\sqrt{F}$	$\frac{\mathrm{V}}{\mathrm{m}}$
Magnetic field	$0.258\sqrt{F}$	$\frac{\mathrm{A}}{\mathrm{m}}$

```
-- Create a dataset containing the standards formulae for reference
standards = pf.DataSet.New()
standards.Axes:Add( pf.Enums.DataSetAxisEnum.Frequency, pf.Enums.FrequencyUnitEnum.Hz,
                    400e6, 1.5e9 ,21 )
standards.Quantities:Add( "E_inirc88", pf.Enums.DataSetQuantityTypeEnum.Scalar, "V/m")
standards.Quantities:Add( "E_nrpb89", pf.Enums.DataSetQuantityTypeEnum.Scalar, "V/m")
standards.Quantities:Add( "H_inirc88", pf.Enums.DataSetQuantityTypeEnum.Scalar, "A/m")
standards.Quantities:Add( "H_nrpb89", pf.Enums.DataSetQuantityTypeEnum.Scalar, "A/m")

for freqIndex = 1,standards.Axes[pf.Enums.DataSetAxisEnum.Frequency].Count do
    local freqHz  = standards[freqIndex]:AxisValue(pf.Enums.DataSetAxisEnum.Frequency)
    local freqMHz = freqHz/1e6 -- frequency in MHz
    local freqGHz = freqHz/1e9 -- frequency in GHz

    local standardsPt = standards[freqIndex]

    -- Electric field limits
    standardsPt.E_inirc88 = 3*math.sqrt(freqMHz)
    standardsPt.E_nrpb89  = 97.1*math.sqrt(freqGHz)
    -- Magnetic field limits
    standardsPt.H_inirc88 = 0.008*math.sqrt(freqMHz)
    standardsPt.H_nrpb89  = 0.258*math.sqrt(freqGHz)
end

return standards
```

INIRC88

```
-- This example illustrates how advanced calculations
-- can be performed to display radiation hazard zones.

-- The INIRC 88 standards are used.

nf = pf.NearField.GetDataSet("yagi.StandardConfiguration1.nf3D")

function calculateRADHAZThresholds(index, nf)
    -- Get a handle on the indexed near field point
    local nfPt = nf[index]

    -- Set up the threshold according to the standards
    -- Frequency in MHz
    local freq = nfPt:AxisValue(pf.Enums.DataSetAxisEnum.Frequency)/1e6
    local EfieldLimit = 3*math.sqrt(freq)
    local HfieldLimit = 0.008*math.sqrt(freq)

    -- SCALE THE ELECTRIC FIELD VALUES
    -- Scale the values to indicate percentages.  The percentage represents
    -- the field value relative to the limit of the standard.
    nfPt.EFieldComp1 = nfPt.EFieldComp1/(EfieldLimit)
    nfPt.EFieldComp2 = nfPt.EFieldComp2/(EfieldLimit)
    nfPt.EFieldComp3 = nfPt.EFieldComp3/(EfieldLimit)

    -- SCALE THE MAGNETIC FIELD VALUES
    -- Scale the values to indicate percentages.  The percentage represents
    -- the field value relative to the limit of the standard.
    nfPt.HFieldComp1 = nfPt.HFieldComp1/(HfieldLimit)
    nfPt.HFieldComp2 = nfPt.HFieldComp2/(HfieldLimit)
    nfPt.HFieldComp3 = nfPt.HFieldComp3/(HfieldLimit)
end
pf.DataSet.ForAllValues(calculateRADHAZThresholds, nf)

-- Note that in essence, the values being returned are
-- no longer near fields.  As such, interpret them
-- carefully in POSTFEKO.
return nf
```

NRPB89

```
-- This example illustrates how advanced calculations
-- can be performed to display radiation hazard zones.

-- The NRPB 89 standards are used.

nf = pf.NearField.GetDataSet("yagi.StandardConfiguration1.nf3D")

function calculateRADHAZThresholds(index, nf)
    -- Get a handle on the indexed near field point
    local nfPt = nf[index]

    -- Set up the threshold according to the standards
    -- Frequency in GHz
    local freq = nfPt:AxisValue(pf.Enums.DataSetAxisEnum.Frequency)/1e9
    local EfieldLimit = 97.1*math.sqrt(freq)
    local HfieldLimit = 0.258*math.sqrt(freq)

    -- SCALE THE ELECTRIC FIELD VALUES
    -- Scale the values to indicate percentages.  The percentage represents
    -- the field value relative to the limit of the standard.
    nfPt.EFieldComp1 = nfPt.EFieldComp1/(EfieldLimit)
    nfPt.EFieldComp2 = nfPt.EFieldComp2/(EfieldLimit)
    nfPt.EFieldComp3 = nfPt.EFieldComp3/(EfieldLimit)

    -- SCALE THE MAGNETIC FIELD VALUES
    -- Scale the values to indicate percentages.  The percentage represents
    -- the field value relative to the limit of the standard.
    nfPt.HFieldComp1 = nfPt.HFieldComp1/(HfieldLimit)
    nfPt.HFieldComp2 = nfPt.HFieldComp2/(HfieldLimit)
    nfPt.HFieldComp3 = nfPt.HFieldComp3/(HfieldLimit)
end
pf.DataSet.ForAllValues(calculateRADHAZThresholds, nf)

-- Note that in essence, the values being returned are
-- no longer near fields.  As such, interpret them
-- carefully in POSTFEKO.
return nf
```

Chapter E

Waveguide + microwave circuits

E-1 Microstrip filter

Keywords: microstrip filter, FEM, SEP, input impedance, microstrip source, FEM current source, edge source, reflection coefficient, S-parameters, planar multilayer substrate

A simple microstrip notch filter is modelled. The filter is solved using several different techniques: the surface equivalence principle (SEP), the finite element method (FEM) and on an infinite substrate using a planar multilayer substrate modelled with Green's functions. The reference for this example may be found in: G. V. Eleftheriades and J. R. Mosig, "On the Network Characterization of Planar Passive Circuits Using the Method of Moments", IEEE Trans. MTT, vol. 44, no. 3, March 1996, pp. 438-445, Figs 7 and 9.

The geometry of the finite substrate model is shown in Figure E-1-1:

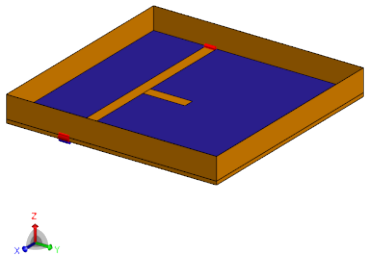


Figure E-1-1: A 3D view of the simple microstrip filter model in CADFEKO. (A cutplane is included so that the microstrip lines of the filter inside the shielding box are visible.)

E-1.1 Microstrip filter on a finite substrate (FEM)

Creating the model

The substrate and shielding box are made using cuboid primitives. The microstrip line is built using a cuboid primitive and removing the undesired faces. The stub is added by sweeping a line that forms a leading edge of the stub.

The steps for setting up the model are as follows:

- Set the model unit to millimetres.
- Create the following variables:
  - `fmax = 4e9` (Maximum frequency.)
  - `fmin = 1.5e9` (Minimum frequency.)
  - `epsr = 2.33` (Substrate relative permittivity.)
  - `shielding_height = 11.4` (Height of the shielding box.)
  - `substrate_height = 1.57` (Substrate height.)

  - `gnd_length = 92` (Length and width of substrate.)
  - `port_offset = 0.5` (Inset of the feed point.)
  - `strip_width = 4.6` (Width of the microstrip sections.)
  - `strip_offset = 23` (Offset of the microstrip from the ground edge.)
  - `stub_length = 18.4` (Length of the stub.)
  - `stub_offset = 41.4` (Inset length from the ground edge to the stub.)

- Create a dielectric medium named `air` with the default properties of a vacuum.
- Create a dielectric medium named `substrate` with relative permittivity of `epsr` and zero dielectric loss tangent.
- Create the substrate using the cuboid primitive with the *Base corner* at (0, 0, 0). The side lengths are `gnd_length` and has a height of `substrate_height`. Label the cuboid `substrate`.
- Create the shielding box using the cuboid primitive with the *Base corner* at (0, 0, 0). The side lengths are `gnd_length` and it is `shielding_height` high and label the cuboid `shielding_box`.
- Create a cuboid for the microstrip at *Base corner* (`port_offset`, `strip_offset`, 0). The cuboid width is set to `gnd_length-port_offset*2`, a depth of `strip_width` and with a height of `substrate_height`. Label the cuboid `mircostrip`.
- Delete all four vertical faces of `mircostrip` (cube created above).
- To illustrate the sweep tool, the stub will be created by sweeping a line segment:
  - Create a line segment that spans from (`stub_offset`, `strip_offset+strip_width`, `substrate_height`) to (`stub_offset+strip_width`, `strip_offset+strip_width`, `substrate_height`).
  - Select the line and sweep it from (0,0,0) to (0, `stub_length`, 0) to generate a rectangular patch.
- Create the following line segments with labels `Feed1` and `Feed2`.
  - `Feed1` spans from (0, `strip_offset+strip_width/2`, `substrate_height`) to (`port_offset`, `strip_offset+strip_width/2`, `substrate_height`).
  - `Feed2` spans from (`gnd_length-port_offset`, `strip_offset+strip_width/2`, `substrate_height`) to (`gnd_length`,`strip_offset+strip_width/2`, `substrate_height`).
- Union all the geometry and label the union `shielded_filter`.
- Set the region properties of the substrate region to `substrate` and the remaining of the region inside the shielding box to `air`.
- Set the solution method for the regions to FEM.

- Ensure that the face properties of the microstrip line, the face defining the ground below the substrate as well as all of the outside faces of the shielding box are set to PEC.
- Select a continuous frequency from `fmin` to `fmax`.

The FEM line port is used to define the excitation points for this model. Add FEM line ports to `Feed1` and `Feed2`. One of the line ports is shown in Figure E-1-2. The ports are labeled *Port1* and *Port2*.

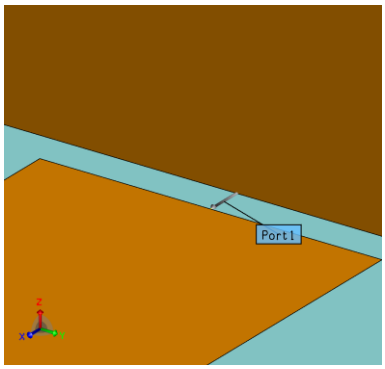


Figure E-1-2: A zoomed in 3D view of one of the FEM current sources applied to a line port.

Requesting calculations

Create an S-parameter configuration where `Port1` is active with a 50 Ω reference impedance. Port2 should be added, but not be active. This should be the only configuration in the model.

Meshing information

Use the *standard* auto-mesh setting.

CEM validate

After the model has been meshed, run *CEM validate*. Take note of any warnings and errors. Correct any errors before running the FEKO solution kernel.

E-1.2 Microstrip filter on a finite substrate (SEP)

Creating the model

The model is based on the FEM model described above. In order to use the SEP, the excitation method must be adjusted.

The steps for modifying the FEM model are as follows:

- Delete both of the line-ports (including the line geometry used to define the port locations). Note that the configuration now reverts to a standard configuration.

- Set the region properties of the two regions back to *MoM/MLFMM with surface equivalence principle (SEP) - default*. Also set the `air` region back to *Free space*.
- As described in the note below, we need to create an excitation point inside the dielectric region. In order to do this, create a face extending from the microstrip edge down to the ground plane, just inside the dielectric region (as shown in Figure E-1-3). (The simplest way to do this is to select and copy the edges at the microstrip line feed points, and sweep them down to the ground to create a plate.)
- Split the two feed plates (added above) in the middle to create a feed edge inside the substrate.
- Union all the geometry. Ensure that all faces are still represented by the correct materials and that no entities have gone suspect.
- Create the edge port connections as illustrated in Figure E-1-3.
- Ensure that the face properties of the microstrip line, the ground below the substrate and sides of the substrate are PEC.
- Set a local mesh size on the microstrip lines (faces) of `strip_width*0.7`.

NOTE: When the edge source is used together with a finite sized dielectric, the edge for the port must be surrounded on all sides by the same medium. In this case, we needed to embed the ports fully inside the substrate dielectric.

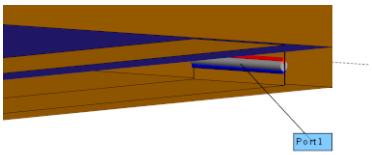


Figure E-1-3: A zoomed in 3D view of one of the edge feeds.

Requesting calculations

Create an S-parameter configuration where `Port1` is active with a 50 Ω reference impedance. Port2 should be added, but not be active. This should be the only configuration in the model.

Meshing information

Use the *standard* auto-mesh setting.

CEM validate

After the model has been meshed, run the *CEM validate*.

E-1.3 Microstrip filter on an infinite substrate (Planar multilayer Green's function)

Creating the model

Only the shielding box and the microstrip lines are required. The lower face of the shielding box and the substrate are removed and modelled using a planar multilayer substrate. The changes that must be made to the FEM model are given below.

- Delete both of the line-ports (including the line segment geometry used to define the port locations). Note that the configuration now reverts to a standard configuration.
- Set the region properties of the two regions back to *MoM/MLFMM with surface equivalence principle (SEP) - default*. Also set both the `air` and the `substrate` regions back to *Free space*.
- Delete the bottom face of the shielding box, as well as the bottom part of the microstrip line. The box should now be open from below and all faces should be PEC.
- Delete the face surrounding the microstrip line and stub. The only horizontal faces remaining are then the top of the microstrip line and the top of the shielding box.
- Create a planar multilayer substrate. Add a layer of type `substrate` that has a thickness of `substrate_height`. The top of the substrate is at `z=substrate_height)`. Add a PEC ground plane to the bottom of the substrate layer.
- As there is an infinite ground plane in this model, the microstrip port may be used to define the excitation. Microstrip ports are attached to each of the port edges. These ports are then referenced in the S-parameter solution request. The polarisation of the ports should be chosen such that the positive terminals (indicated by a red cylinder in the 3D view) are on the microstrip.
- Set a local mesh size on the microstrip lines (faces) of `strip_width*0.7`.

Requesting calculations

Create an S-parameter configuration where `Port1` is active with a 50 Ω reference impedance. Port2 should be added, but not be active. This should be the only configuration in the model.

Meshing information

Use the *standard* auto-mesh setting.

CEM validate

After the model has been meshed, run *CEM validate*. Take note of any warnings and errors. Correct any errors before running the FEKO solution kernel.

E-1.4 Results

The S-parameters for all 3 cases are computed over the frequency range 1.5 GHz to 4 GHz. The results for the S-parameters are shown in Figure E-1-4 and E-1-5. From the scattering parameters at the input and output ports, it can be seen that almost all energy incident on the filter at 2.75 GHz is reflected back to the input port.

The effect of the different solution methods and feed techniques can be seen in the results, but all results agree very well with the reference measurements and with each other.

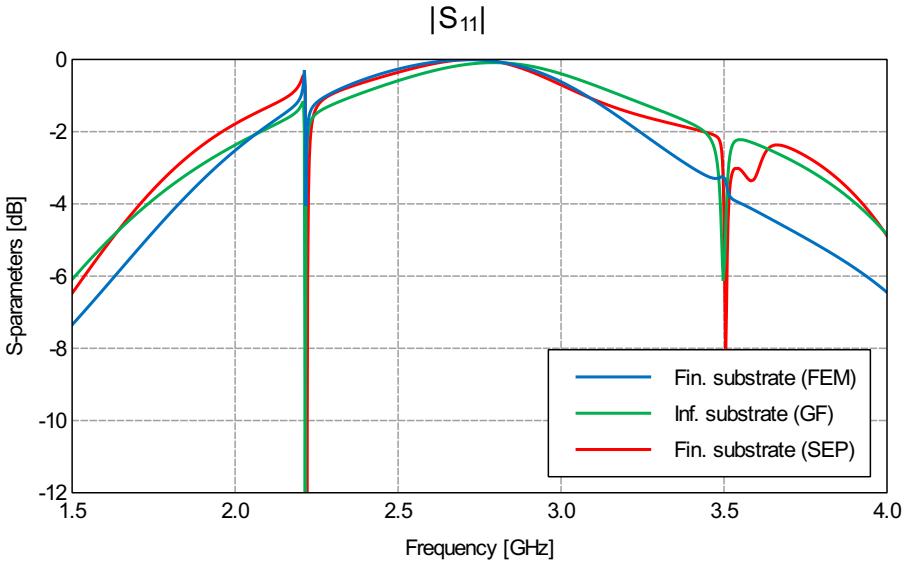


Figure E-1-4: S_{11} in dB of the microstrip filter on an infinite and finite substrate from 1.5 GHz to 4 GHz.

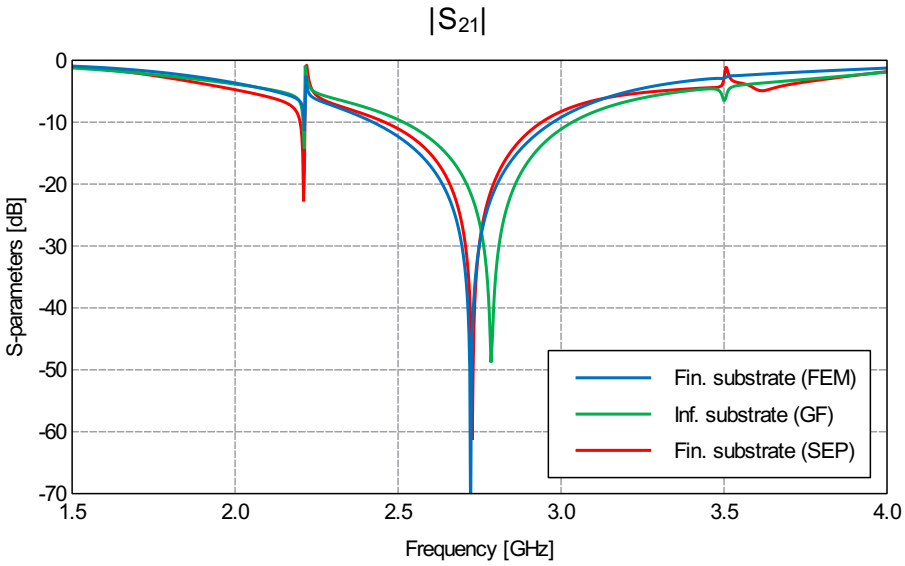


Figure E-1-5: S_{21} in dB of the microstrip filter on an infinite and finite substrate from 1.5 GHz to 4 GHz.

E-2 S-parameter coupling in a stepped waveguide section

Keywords: waveguide, S-parameter, coupling

In this example we consider a waveguide transition from Ku- to X-band by a simple step discontinuity (as shown in Figure E-2-1) using two solution methods available in FEKO. The model is first simulated using the MoM using waveguide ports and then using the FEM and modal ports. The rectangular waveguide dimensions are a=15.8 mm and b=7.9 mm for the Ku-band waveguide, and a=22.9 mm and b=10.2 mm for the X-band waveguide, respectively. Only the H_{10} mode is considered.

The critical frequency for the chosen H_{10} mode in the smaller Ku-band waveguide is $f_c=\frac{c_0}{2a}$=9.4871 GHz. We want to compute S-parameters from this cut–off frequency up to 15 GHz using adaptive frequency sampling.

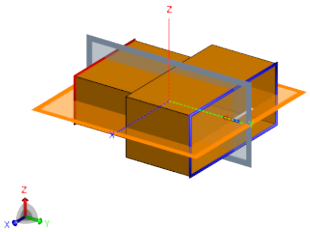


Figure E-2-1: 3D view of a waveguide step from Ku to X band.

E-2.1 Waveguide step model (MoM)

Creating the model

The steps for setting up the model are as follows:

- Set the model unit to millimetres.
- Define the following variables:
  - `a1 = 15.8` (Width of the Ku section.)
  - `b1 = 7.9` (Height of the Ku section.)
  - `l1 = 12` (Length of the Ku section.)
  - `a2 = 22.9` (Width of the X section.)
  - `b2 = 10.2` (Height of the X section.)
  - `l2 = 12` (Length of the X section.)
  - `fmin = 9.4872e9` (The minimum calculation frequency.)
  - `fmax = 15e9` (The maximum calculation frequency.)

Note that the minimum calculation frequency `fmin` is just above the Ku-band cutoff frequency f_c as explained previously.

- Create the Ku-band waveguide section using a cuboid with its base corner at (`-a1/2`, `-l1`, `-b1/2`) with a *width* of `a1`, a *depth* of `l1` and a height of `b1`.
- Create the X-band waveguide section on the positive Y axis. Use antoher cuboid with its base corner at (`-a2/2`, `0`, `-b2/2`) with a *width* of `a2`, a *depth* of `l2` and a *height* of `b2`.
- Union the two cubes and then simplify the model.
- Set the region inside the waveguide step to free space.
- Rename the faces that form the waveguide to `Port1` and `Port2`, where `Port1` sits on the outer face of the Ku-band waveguide section and `Port2` sits on the outer face of the X-band waveguide section.
- Apply waveguide ports to both faces `Port1` and `Port2`. Ensure that the reference vector (indicated by a white line) for both ports are in the *XY* plane.
- Confirm that the propagation direction of the waveguide source is into the waveguide in both cases.
- Set the frequency to be continuous from `fmin` to `fmax`.

Requesting calculations

Magnetic symmetry in the X=0 plane and electric symmetry in the Z=0, are used.

The solution requests are:

- An S-parameter calculation is requested (Fundamental mode for both ports). Only `Port1` needs to be active.

Meshing information

Use the *fine* auto-mesh setting.

CEM validate

After the model has been meshed, run *CEM validate*. Take note of any warnings and errors. Correct any errors before running the FEKO solution kernel.

E-2.2 Waveguide step model (FEM)

The model is almost the same as for the MoM model, so it will be used as a base for the FEM model.

Creating the model

Make a copy of the MoM model and make the following changes:

- Remove the two waveguide ports.
- Apply FEM modal ports to both faces `Port1` and `Port2`.
- Create a new dielectric medium with the name `air` and use all the default values of free space.
- Set the region property of the waveguide to be a dielectric and select `air` as the dielectric.
- Ensure that the faces that form the walls of the waveguide are set to PEC. Note that the port faces must be dielectrics for FEM ports.
- Set the solution method of the region to *finite element method (FEM)*.
- Decouple the FEM and MoM (the setting is available on the FEM tab of the Solver settings dialog).

Requesting calculations

The solution requests are:

- An S-parameter calculation is requested (Fundamental mode for both ports). Only `Port1` needs to be active.

Meshing information

Use the *fine* auto-mesh setting.

CEM validate

After the model has been meshed, run *CEM validate*. Take note of any warnings and errors. Correct any errors before running the FEKO solution kernel.

E-2.3 Results

Save and run the FEKO solver. Figure E-2-2 shows the computed S-parameters with FEKO for both the MoM and the FEM solutions. It is clear that the cut-off frequency is at about 9.4871 GHz. These results agree well with available references (both measurements and computations) and with one another.

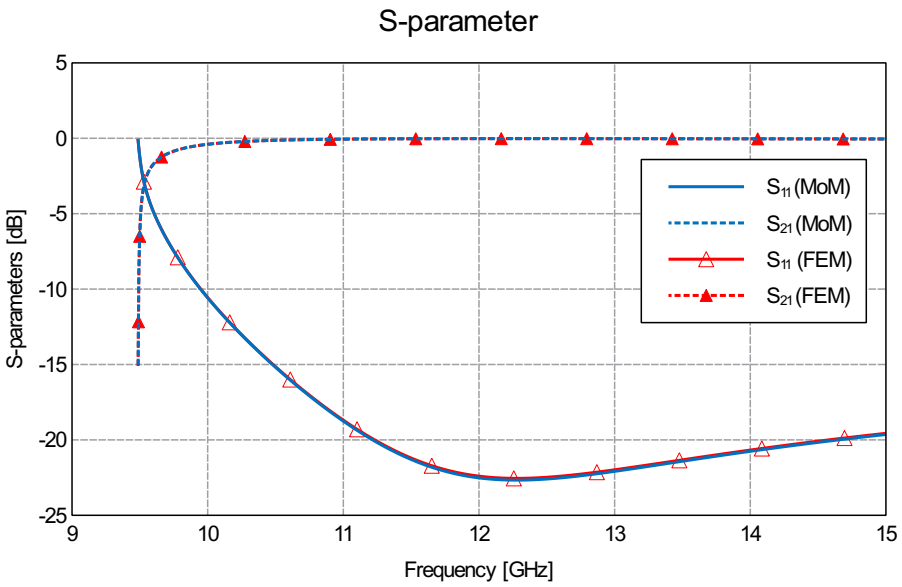


Figure E-2-2: S-parameters for the waveguide step discontinuity.

E-3 Using a non-radiating network to match a dipole antenna

Keywords: network, S-parameters, Z-parameters, Y-parameters, Touchstone, ideal matching, dipole

A short dipole (approximately $\frac{1}{3}\lambda$) is matched to be resonant at 1.4 GHz. This is done using two methods; one using an LC matching section and another using the matching network's S-parameters as an input network. The S-parameters are provided in the `Matching.s2p` Touchstone file.

The matching network is simply a 2.43 pF shunt capacitor and a 41.2 nH series inductor connected between the source and the dipole.

Figure E-3-1 is an illustration of the short dipole with a network feed as well as the matching network schematic.

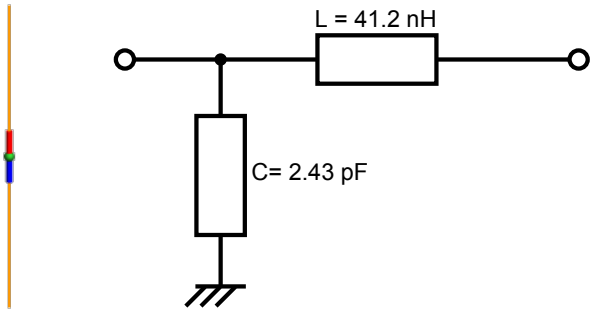


Figure E-3-1: A model of a dipole. The schematic represents the matching network used at the port.

E-3.1 Dipole matching using a SPICE network

Creating the model

The steps for setting up the model are as follow:

- Set the model unit to millimetres.
- Define the following variables:
  - `fmin = 1.3e9` (The minimum operating frequency.)
  - `fmax = 1.5e9` (The maximum operating frequency.)
  - `h = 70` (The height of the dipole.)
  - `wireRadius = 0.1` (Radius of dipole wire segments.)
- Create a 70 mm (`h`) line along the Z axis with its centre at the origin. Label the wire `Dipole`.
- Add a wire port to the centre of the wire. Label the port `Port1`.

- Set the frequency to be continuous over the frequency from `fmin` to `fmax`.
- Create a general network:
  - Label it `MatchingNetwork`
  - The network name should then correspond to the internal network name used in `Match_circuit.cir`.
- Port 1 of this general network is excited using a voltage source. The second port is connected to the wire port in the centre of the wire.

The file `Match_circuit.cir` contains

```
Matching circuit

.SUBCKT MatchingNetwork n1 n2
c1 n1 0  2.43pF
l1 n1 n2 41.2nH
.ENDS NWN1

.end
```

Requesting calculations

No solution requests are required in CADFEKO.

Meshing information

Use the *standard* auto-mesh setting with wire segment radius `wireRadius`.

CEM validate

After the model has been meshed, run *CEM validate*. Take note of any warnings and errors. Correct any errors before running the FEKO solution kernel.

Save the file and run the solver.

E-3.2 Dipole matching using a general S-parameter network

Use the same model as for the SPICE matching network model.

Change the general network settings to refer to an S-matrix Touchstone file named `Matching.s2p`. This file defines the S-parameters of the matching network. Port 1 of this general network is excited using a voltage source. The second port is connected to the wire port in the centre of the wire.

Requesting calculations

No solution requests are required in CADFEKO.

Meshing information

Use the *standard* auto-mesh setting with wire segment radius `wireRadius`.

CEM validate

After the model has been meshed, run *CEM validate*. Take note of any warnings and errors. Correct any errors before running the FEKO solution kernel.

Save the file and run the solver.

E-3.3 Results

The reflection coefficient of the matched and unmatched dipole is shown in Figure E-3-2. (It may be difficult to see any variation of the reflection coefficient of the unmatched dipole since it is very close to 0 dB over the whole band).

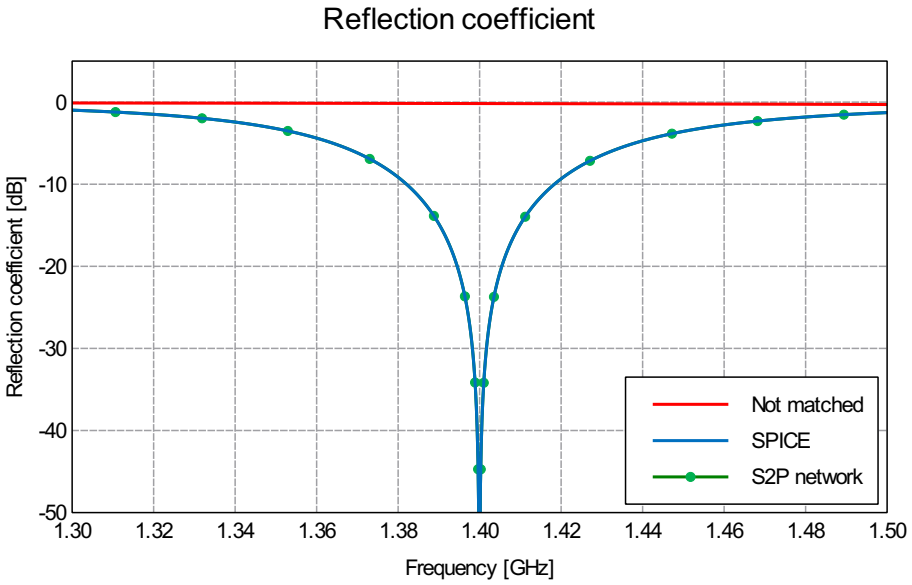


Figure E-3-2: The reflection coefficient of the dipole before and after application of the feed matching.

E-4 Subdividing a model using non-radiating networks

Keywords: network, S-parameters, Touchstone, input impedance, far field, patch, feed network, non-radiating network

A right hand circularly polarised patch antenna at 2.4 GHz is simulated in two ways. The problem is first divided so that the feed network is characterised (save S-parameters to a Touchstone file) and then the Touchstone file is used as a non-radiating network to feed the patch. The two models (feed network and patch antenna) are then combined so that the full simulation (model contains feed and patch) is performed. The input impedance as well as the simulation time and memory required for the two methods are compared.

We will see that subdividing the problem greatly reduces the required resources. However, when using this technique, the field coupling between the feed network and the patch is not taken into account which results in a variation in the results.

The steps that are required to create the model are not part of the this example. However, several important points regarding the creation process will be highlighted.

Figure E-4-1 is an illustration of RHC patch antenna with the feed network.

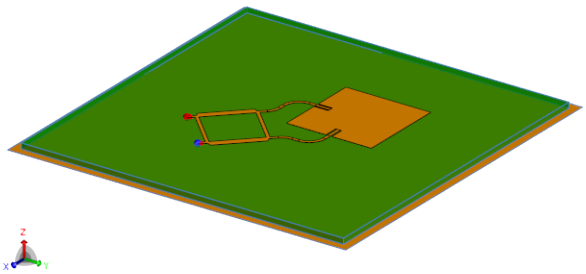


Figure E-4-1: The model of a RHC patch antenna with feed network.

E-4.1 Feed network

The feed network consists of a branch line coupler that divides the power evenly with 90 degree phase difference between the outputs. The output signals are then extended to the patch-feed interfaces using microstrip transmission lines. The entire system is designed in a 120 Ω system reference impedance.

Creating the model

The steps for setting up the feed network model are as follows:

- Define a new dielectric named `RogersDuroid5870`. (Relative dielectric constant of 2.2 and tanδ=0.0012)

- Add an planar multilayer substrate (infinite plane) with a height of 2.5 mm and dielectric material `RogersDuroid5870`. A perfect electric ground should be placed on the bottom of the substrate (this is the default).
- Create the branch coupler for an output impedance of 120 Ω. To do this, import the geometry from the Parasolid model file (`*.x_b`).
- Create the microstrip transmission line sections that connect the branch coupler to the patch antenna. (This model does not contain the antenna, but later this model is imported into the antenna model to do the complete simulation.)
- Create four microstrip ports on the four terminals of feed structure. (Name the ports by number (1 to 4) starting at the input port, then the two output ports that will connect to the patch, and then the last port that will be loaded with a resistance.)
- Add a 120 Ω load on fourth port.
- Set the solution frequency to be from `0.8*2.4e9` to `1.2*2.4e9`. Activate the *Specify sampling for exported data files* and set the value to `100`.

Requesting calculations

Add an S-parameter request for port one to three (not the port with the load connected). All ports should be active and the reference impedance should be set to 120 Ω.

Meshing information

Set the local mesh settings of the faces equal to `wl` equal to 1.4e-3.

CEM validate

After the model has been meshed, run *CEM validate*. Take note of any warnings, notes and errors. Please correct error before running the FEKO solution kernel.

Save the file as `feedNetwork.cfx` and run the solver. The S-parameters can be displayed in POSTEFEKO; this should illustrate that the branch coupler is working correctly (split power evenly and 90° phase difference between the output ports.) A Touchstone file containing the calculated S-parameters will be located in the project directory (named `feedNetwork_SParameter1.s3p`).

E-4.2 Patch with non-radiating feed network

We have simulated and characterised the feed network for the patch antenna in the previous section. The result (Touchstone file) from that simulation is now going to be combined with the patch antenna by using a general non-radiating network.

Creating the model

The steps for setting up the model are as follows:

- Define a new dielectric named `RogersDuroid5870`. (Relative dielectric constant of 2.2 and tanδ=0.0012)
- Add a planar multilayer substrate (infinite plane) with a height of 2.5 mm and dielectric material `RogersDuroid5870`. A perfect electric ground should be placed on the bottom of the substrate (this is the default).
- Create the rectangular patch antenna at the origin with a patch width of `39e-3`.
- Create the slots in the patch where the feed is connected by creating and then subtracting two. The length of the rectangular polygon is `6.5e-3` and the width `2.8e-3`.
- Create the inset microstrip feeds by creating rectangular polygons with length `6.5e-3` and width of `1.4e-3`. Union the structures to ensure connectivity.
- Create two microstrip ports on the two feed terminals.
- Create a new non-radiating general network with three ports that imports network properties for the Touchstone file created earlier (section E-4.1).
- Connect the correct microstrip ports to the corresponding network ports.
- Add a voltage source on the corresponding network port.
- Set the solution frequency to be from `0.8*2.4e9` to `1.2*2.4e9`.

Requesting calculations

The input impedance at the voltage source is available in POSTFEKO without any requests. Add a far field request for a vertical cut. Note that no field can exist below an infinite perfect electrically conducting plane. The far field request should only be for field points above the infinite plane -85°$\leq\theta\leq$85°, with ϕ=0° and 5° increments).

Meshing information

The faces of the two microstrip feeds have to be meshed finer than the patch. The required mesh size is determined by the size of the geometry. Set the local mesh size on these faces to `wl` equal to 1.4e-3. The global mesh is set on the *Create mesh* dialog and should use the *standard* auto-mesh setting.

Save the file as `touchstoneFedPatch.cfx` and run the solver. The input impedance and far field results can be viewed in POSTFEKO.

E-4.3 Patch with radiating feed network

The advantage of being able to model the feed as a non-radiating general network can only be seen when comparing the results and the required resources with the full 3D simulation.

Creating the model

The patch antenna model (`touchstoneFedPatch.cfx`) will be used as base model and the branch coupler model (`feedNetwork.cfx`) imported. The complete simulation is then performed.

The steps for setting up the model are as follows:

- Open the file `touchstoneFedPatch.cfx` and save it as `completePatch.cfx`.
- Delete the voltage source, remove the general network connections and then delete the general network and all the ports.
- Import the file `feedNetwork.cfx`:
  - Import only the geometry and mesh rules.
  - Merge identical variables and media.
- Delete `Port2` and `Port3`. Keep `Port1` and `Port4` (the outer ports).
- Union the two structures.
- Add a voltage source to `Port1`: (1 V, 0°, 50 Ω)
- Add a 120 Ω load to `Port4`.

Requesting calculations

Ensure that all faces of the feed network are set to have a local mesh size of `wl` equal to 1.4e-3. The model can the be meshed using the standard auto-mesh setting.

Save the file and run the solver.

E-4.4 Results

The difference in solution time and required main memory is tabled in Table E-4-1. We see that the solution time is reduced by subdividing the problem. Since the field coupling between the feed and the patch cannot be taken into account when substituting the feed with a general non-radiating network, the results are slightly different as can be seen in figure E-4-2.

The great advantage really becomes clear when the user has to design the antenna and cannot (or does not want to) change the feed network. This allows fast simulations during antenna development. Verification can then be done after development that includes a full 3D field solution including the patch and the feed network.

Table E-4-1: Comparison of resources for the simulations per frequency.

Model	*RAM*	*Time*	*Total Time*
Full model	4040 KB	537	537
Network only	891 KB	63	
Patch with general network	1545 KB	215	278

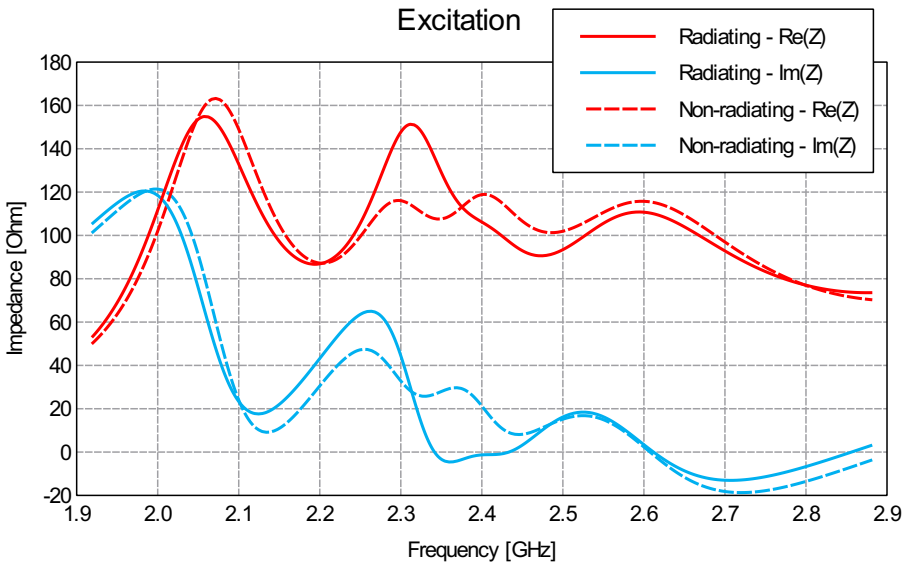


Figure E-4-2: Input impedance (real and imaginary) of the path with radiating and non-radiating feed.

E-5 Microstrip coupler

Keywords: microstrip coupler, FDTD, edge source, S-parameters, coupling

The coupling of a four port microstrip coupler is presented over a frequency band between 2.5 – 5.0 GHz. This example is based on the paper "On the design of planar microwave components using multilayer structures", by W. Schwab and W. Menzel, IEEE Trans. MTT, vol. 40, no. 1, Jan 1992, pp. 67-72, Fig 9.

Figure E-5-1 shows a top view of the model. Note that there are multiple layers visible due to transparency.

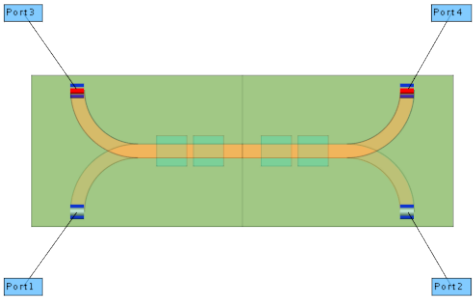


Figure E-5-1: A top view of the microstrip coupler.

E-5.1 Microstrip coupler

Creating the model

The steps for setting up the model are as follows:

- Set the model unit to millimetres.
- Define the following variables:
  - `d1 = 2.22` (Spacing 1 between apertures.)
  - `d2 = 12.51` (Spacing 2 between apertures.)
  - `epsr = 2.2` (Relative permittivity of substrate dielectric.)
  - `f_max = 5e9` (Highest simulation frequency.)
  - `f_min = 2.5e9` (Lowest simulation frequency.)
  - `s = 10` (Aperture side length.)
  - `strip_feed_arc_radius = 2*s` (Radius of curved microstrip line.)
  - `strip_length = 2*s+d2+d1` (Length of straight section for microstrip line.)
  - `substrate_depth = 50` (Depth of substrate.)
  - `substrate_height = 1.58` (Height of substrate.)
  - `substrate_width = 140` (Width of substrate.)

  - `w = 4.6` (Width of the microstrip lines.)

- Create a dielectric with a relative permittivity of `epsr` and label it `substrate`.
- Create a straight section of microstrip using a rectangle with a base corner at (0, `-w/2`, `substrate_height`), a width of `strip_length` and a depth of `w`.
- Create a rectangle that will be used for the feed. Rotate the local workplane around the *U* axis by 90° so that the plate is defined in the *XZ* plane. The origin of the workplane is at (`strip_length+strip_feed_arc_radius-w/2`, `strip_feed_arc_radius`, 0). The feed has a width of `w` and a depth of substrate_height.
- Create geometry that will be used for the bend of the microstrip by creating two ellipses:
  - The outer boundary of the arc is defined by a circular surface with a centre point at (`strip_length`, `strip_feed_arc_radius`, `substrate_height`) and radial dimensions of `strip_feed_arc_radius+w/2`.
  - The inner boundary of the arc is defined by a circular surface with a centre point at (`strip_length`, `strip_feed_arc_radius`, `substrate_height`) and radial dimensions of `strip_feed_arc_radius-w/2`.
  - Subtract the inner circle from the outer circle. A full 360° loop should now be visible in the view.
- Union all of the geometry.
- Delete the curved face that does not form part of the microstrip bend. Once deleted, the microstrip should consist of a straight section, a bend and a feed.
- Simplify the remaining microstrip geometry to remove any unwanted edges. The resulting geometry represents half of the top microstrip.
- Copy and rotate the microstrip by 180° around the *U* axis. The new part represents half of the bottom microstrip.
- Create a ground plate using a rectangle with a base corner at (0,`-substrate_depth/2`,0), a width of `substrate_width/2` and a depth of `substrate_depth`.
- Create the geometry for the apertures by creating two rectangles:
  - Place the base corner of `aperture_1` at (`d2/2`,`-s/2`,0). The aperture is a square with side length `s`.
  - Place the base corner of `aperture_2` at (`d2/2+s+d1`,`-s/2`,0). The aperture is a square with side length `s`.
- Subtract the two apertures from the ground plate. The result is a ground plane between the two microstrip lines that has two square holes cut into it.
- Copy and mirror all geometry around the *VN* plane.
- Union all geometry. Explicitly set the properties of all of the faces to be perfect electric conductors.
- Add edge ports to each of the feed faces:

  - `Port1`: Defined between the bottom microstrip feed in the negative X direction and the ground plate that it connects to.
  - `Port2`: Defined between the bottom microstrip feed in the positive X direction and the ground plate that it connects to.
  - `Port3`: Defined between the top microstrip feed in the negative X direction and the ground plate that it connects to.
  - `Port4`: Defined between the top microstrip feed in the positive X direction and the ground plate that it connects to.

- Create the substrate layers by using two cuboids:
  - The top layer has a base centre at the origin, a width of `substrate_width`, depth of substrate_depth and a height of `substrate_height`.
  - The bottom layer is the same, but has a base centre at (0,0,`-substrate_height`).
- Union the two cuboids and set their region properties to the `substrate` dielectric.
- Union the substrate layers and the rest of the geometry.

Solution settings and requests

The following solution setting changes and requests are required.

- Create an S-parameter request where all ports are included with a 50 Ω reference impedance. Only `Port1` is active. This will replace the *standard configuration* that was created by default.
- Set a linearly spaced frequency range from `f_min` to `f_max` with 101 frequency samples.
- Enable the finite difference time domain (FDTD) solver.

Meshing information

Use the *standard* auto-mesh setting.

CEM validate

After the model has been meshed, run *CEM validate*. Take note of any warnings and errors. Correct any errors before running the FEKO solution kernel.

E-5.2 Results

Figure E-5-2 shows the simulated coupling parameters are compared to the published data. The simulated results are in good agreement with the measured results. Any differences are most likely due to uncertainties regarding the model dimensions or dielectric properties.

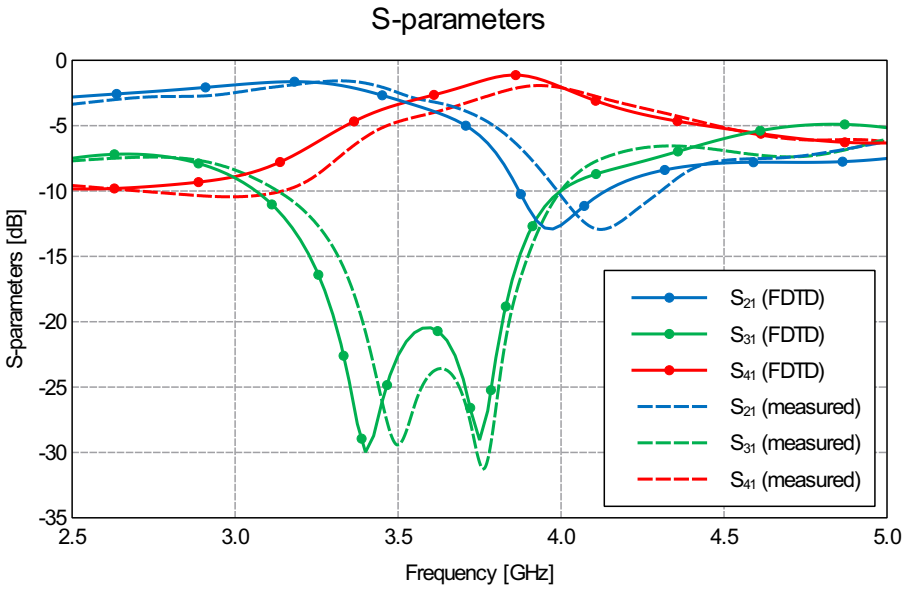


Figure E-5-2: Coupling to microstrip coupler ports.

Chapter F

Bio electromagnetics

F-1 Exposure of muscle tissue using MoM/FEM hybrid

Keywords: exposure analysis, FEM/MoM hybrid method, dielectric losses

This example considers the exposure of a sphere of muscle tissue to the field created by a dipole antenna between 0.1–1 GHz. The geometry of the example is shown in Figure F-1-1.

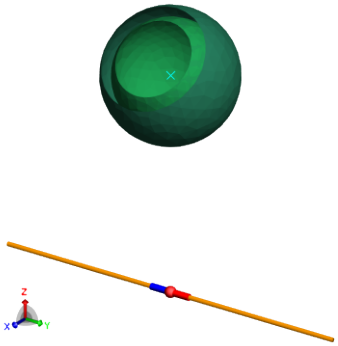


Figure F-1-1: Sphere of muscle tissue illuminated by a dipole antenna.

F-1.1 Dipole and muscle tissue

Note: There is an air layer used around the sphere of muscle tissue to reduce the number of triangle elements required on the boundary between the FEM and MoM regions. This is not strictly necessary, but helps to reduce the resource requirements without compromising on the accuracy of the results.

Creating the model

The steps for setting up the model are as follows:

- Define the following variables:
  - `f_min = 100e6` (Minimum simulation frequency.)
  - `freq = 900e6` (Operating frequency.)
  - `f_max = 1e9` (Maximum simulation frequency.)
  - `d = 0.1` (Distance between the dipole and muscle sphere.)
  - `rA = 0.03` (Radius of the outer sphere.)
  - `rM = 0.025` (Radius of the inner sphere.)
  - `lambda = c0/freq` (Free space wavelength.)
  - `wireRadius = 1e-3` (Radius of the dipole wire.)
- Create the media.

  - Create a dielectric named `Muscle_Parallel_Fibers_Ovine` - it is available in the media library.
  - Create a dielectric named `air` with a relative permittivity of 1 and dielectric loss tangent of zero.

- Create a sphere at the origin with a radius set to the defined variable `rM`. Set the label to *Muscle*.
- Create a sphere at the origin with a radius set to the defined variable `rA`. Set the label to *Air*.
- Union *Muscle* and *Air*.
- Use `Muscle_Parallel_Fibers_Ovine` for the region properties of the inside sphere.
- Set the region properties of the region between the inside and outside sphere to the dielectric called `air`.
- Set both regions to be solved using the finite element method (FEM).
- Create the line a distance of `d` away from the centre of the sphere. Set the *Start point* as (0,`-lambda/4`,`-d`) and the *End point* as (0,`lambda/4`, `-d`).
- Add wire vertex port on the middle of the wire.
- Add the voltage source on the port. (1 V, 0°, 50 Ω)
- Set the total source power (no mismatch) to 1 W.
- Set a continuous frequency range from `f_min` to `f_max`.

Requesting calculations

In the `X=0` plane, use magnetic symmetry. In the `Y=0` plane, use electric symmetry. No symmetry can be used in the `Z=0` plane.

The solution requests are: Create a near field request at `(0,0,0)` - a single request point.

Meshing information

Use the *standard* auto-mesh setting. Set the wire segment radius to `wireRadius`.

CEM validate

After the model has been meshed, run *CEM validate*. Take note of any warnings and errors. Correct any errors before running the FEKO solution kernel.

F-1.2 Results

The electric field strength as a function of frequency is illustrated in Figure F-1-2.

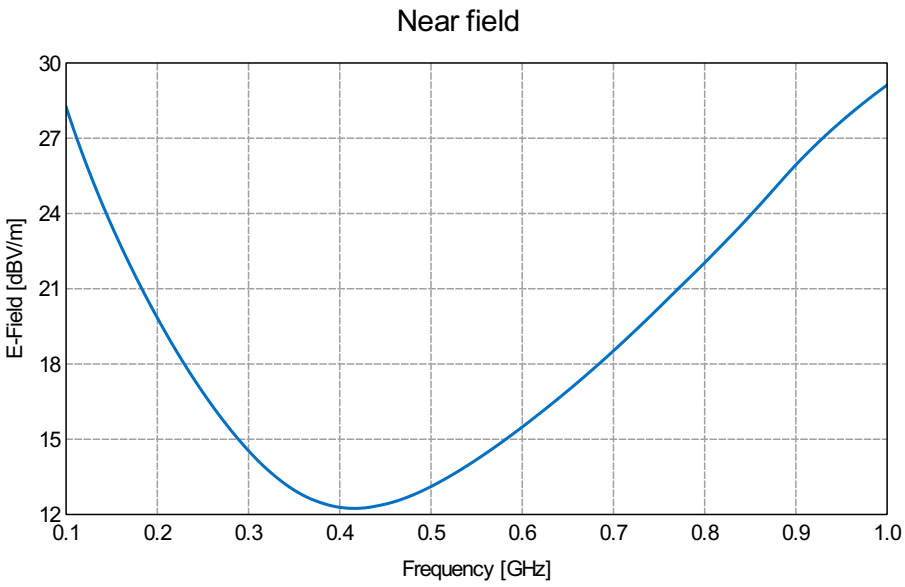


Figure F-1-2: Electric field at the centre of the sphere over frequency.

F-2 Magnetic Resonance Imaging (MRI) birdcage head coil example

Keywords: MRI, birdcage, S-parameters, Lua scripting, B1+

This example demonstrates the simulation of an MRI birdcage head coil with an elliptical phantom, shown in Figure F-2-1. The coil is a 7T highpass design with the tuning capacitors placed in the end-ring gaps between the 16 rungs.

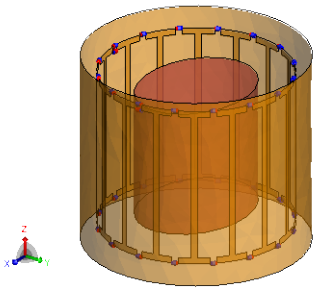


Figure F-2-1: Geometry for the 7T head coil with elliptical phantom.

F-2.1 Birdcage

Creating the model

This example consists of complicated geometry, the model of which is provided with the FEKO installation. The important details for this example are:

- The coil has an inner radius of 15 cm and an RF shield radius of 17.54 cm.
- The elliptical phantom has a major radius of 11 cm and a minor radius of 8.5 cm. Average head tissue properties are assigned to the phantom: $\varepsilon_r = 36$ and $\sigma = 0.657$.
- Capacitive loads (C = 4.15 pF) are added to the wire ports between the end-ring gaps on both sides of the birdcage.

NOTE: this specific example is solved with MoM (SEP), but MoM/FEM or FDTD could also be suitable choices.

Requested calculations

A standard configuration is used to calculate the fields and currents at 300 MHz.

- The frequency (specified *per configuration*) for the standard configuration is set to 300 MHz.
- 2 voltage sources are added to the I and Q feed ports for the quadrature excitation. The magnitude is set to 20 V for both and a 90° phase delay is set on the Q port.

- Currents and the near fields at z = 0 are requested.

A second S-parameter configuration is used to calculated the S-parameters for the coil.

- The frequency is set to continuous (interpolated) covering 290-310 MHz.
- Both the I and Q port are set to active for the calculation.

Meshing information

In order to resolve the geometry of the coil accurately, a custom mesh size is used. The triangle length is set to 4 cm and a local mesh size of 3 cm is set for the elliptical phantom region.

CEM validate

After the model has been meshed, run *CEM validate*. Take note of any warnings and errors. Correct any errors before running the FEKO solution kernel.

F-2.2 Results

The S-parameters for the head coil are shown in Figure F-2-2 as a function of frequency.

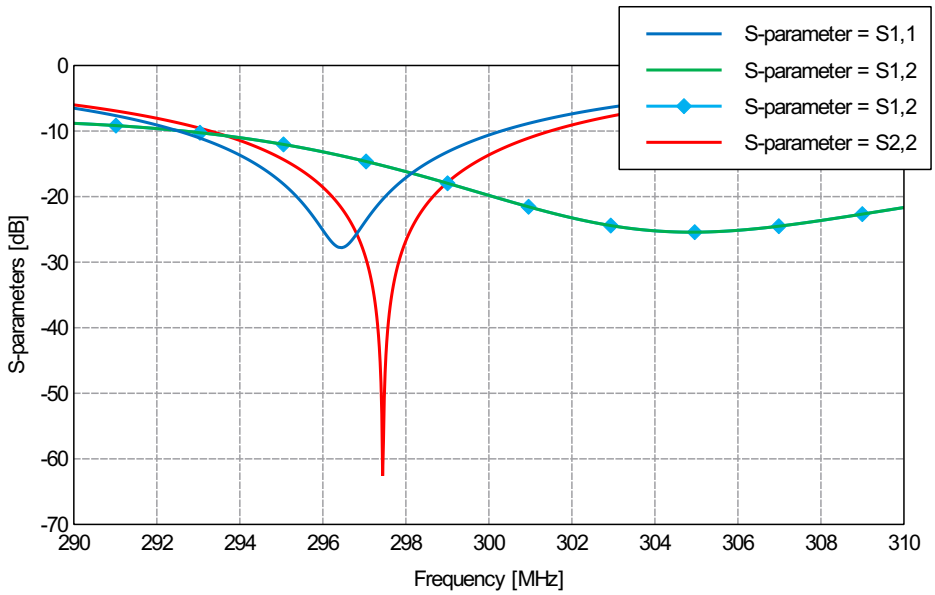


Figure F-2-2: S-parameters for the I and Q feed ports of the coil.

Figure F-2-3 shows the current and near field results at 300 MHz. The B1+ and ratio (B1+/B1-) plots are obtained using the MRI quantities POSTFEKO automation script, which can be found in the same folder as the other files for this example. Other scripts and updates can be downloaded from www.feko.info/support/lua-scripts.

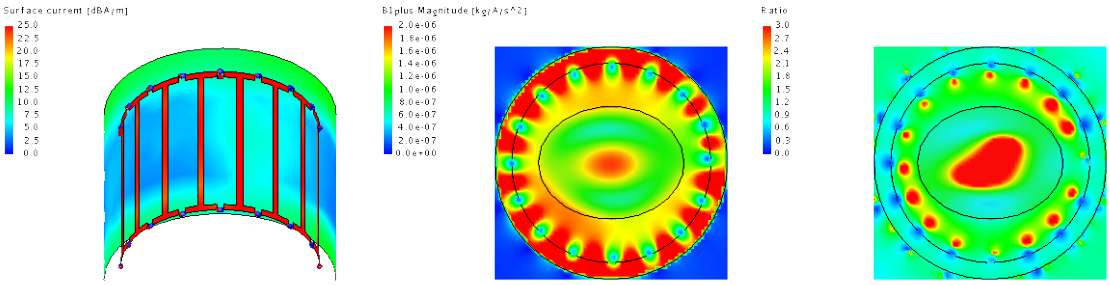


Figure F-2-3: Surface currents on the coil rungs with the phantom hidden (left), B1+ field distribution at z=0 (center) and the ratio of (B1+/B1-) (right) are shown.

Chapter G

Time domain examples

G-1 Time analysis of the effect of an incident plane wave on an obstacle

Keywords: time domain, Fourier transform

A conducting body is placed in the path of an incident plane wave. The effect of this obstacle on the plane wave is analysed in the time domain. The frequency domain results are first obtained using a wide-band simulation using traditional MoM techniques. Post processing analysis of the frequency domain data is then performed to obtain a time response. Figure G-1-1 shows the obstacle along with the time response of the field and current components at a given time step.

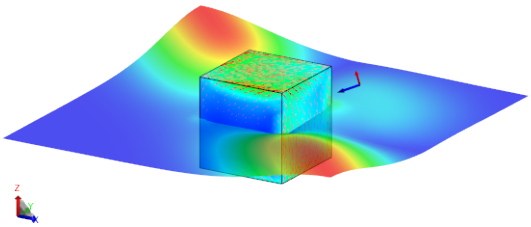


Figure G-1-1: Obstacle and time domain results in the 3D view.

G-1.1 CADFEKO

Creating the model

The steps for setting up the model are as follows:

- Define the following variables:
  - `d = 1` (Length of the side of the cube obstacle.)
- Create a cuboid whose centre is at the origin and has a side length `d`.
- Define a plane wave source with an incident direction of θ=75° and ϕ=45°.
- A list of discrete frequency points will be used during the simulation. Import the list of discrete frequency points from the file called `frequency_list.txt`; this will define an arbitrarily sampled frequency span between 2.5 – 300 MHz.

Requesting calculations

Geometric symmetry may be applied to all three symmetry planes.

All surface currents should be calculated.

A near field with the following properties should be requested:

- *Start*: `(-2*d, -2*d, 0)`
- *End*: `(2*d, 2*d, 0)`
- Request 31 field points in both directions.

Meshing information

Use the *Coarse* auto-mesh setting.

CEM validate

After the model has been meshed, run *CEM validate*. Take note of any warnings and errors. Correct any errors before running the FEKO solution kernel.

G-1.2 POSTFEKO

The currents and fields on and around the obstacle have been calculated within a predetermined frequency range. Any time signal whose spectral content falls within this band may be used to analyse the time response of the system. To illustrate this, the effect of the two signals depicted in figure G-1-2 will be analysed.

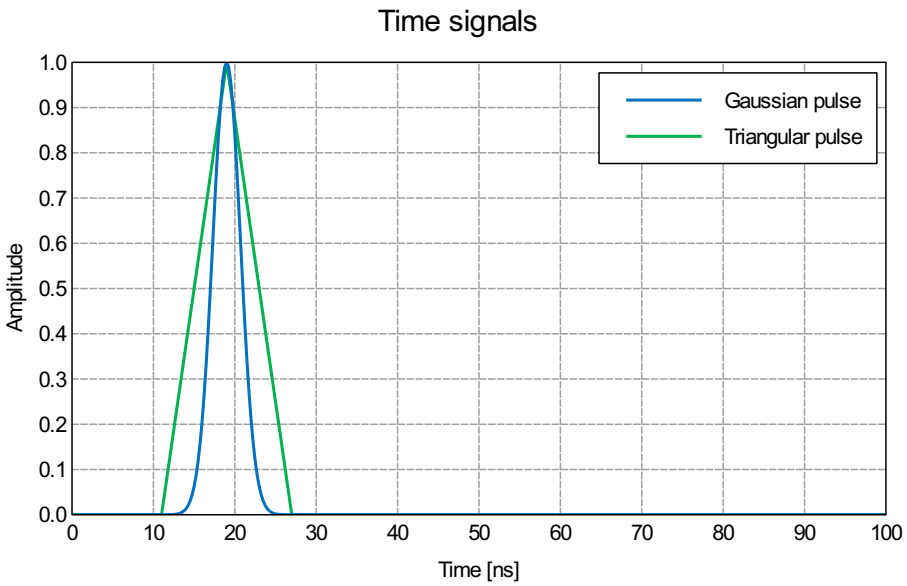


Figure G-1-2: Input time signals.

The parameters required to define the input time signals are provided in Table G-1-1.

Property	*Gaussian pulse*	*Triangular pulse*
Time axis unit	ns	ns
Total signal duration	100	100
Amplitude	1	1
Pulse delay	19	19
Pulse width	4	8
Number of samples	400	400

Table G-1-1: Input time signal properties

G-1.3 Results

The 3D view image depicted in figure G-1-1 shows the response of the near field and currents when the triangular pulse is applied to the system. Animating over time is a useful means of gaining insight into the time domain behaviour of a system.

Figure G-1-3 shows the time response of the system to the Gaussian and triangular pulses after 19 ns.

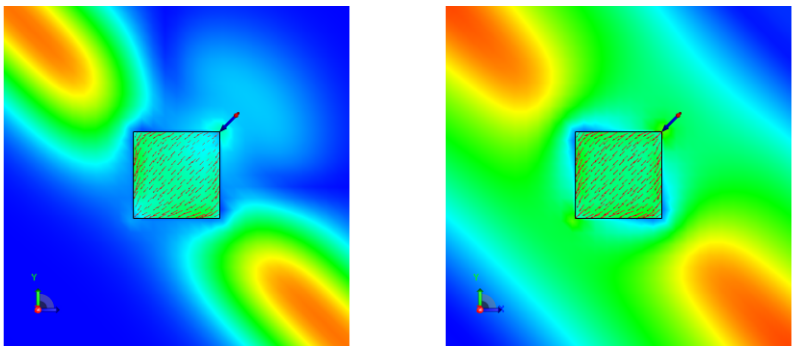


Figure G-1-3: Time response to the Gaussian pulse (left) and triangular pulse (right) after 19 ns.

Time results can also be analysed on graphs. Figure G-1-4 shows the time response of the near field magnitude at (-2,-2,0) m to both input signals.

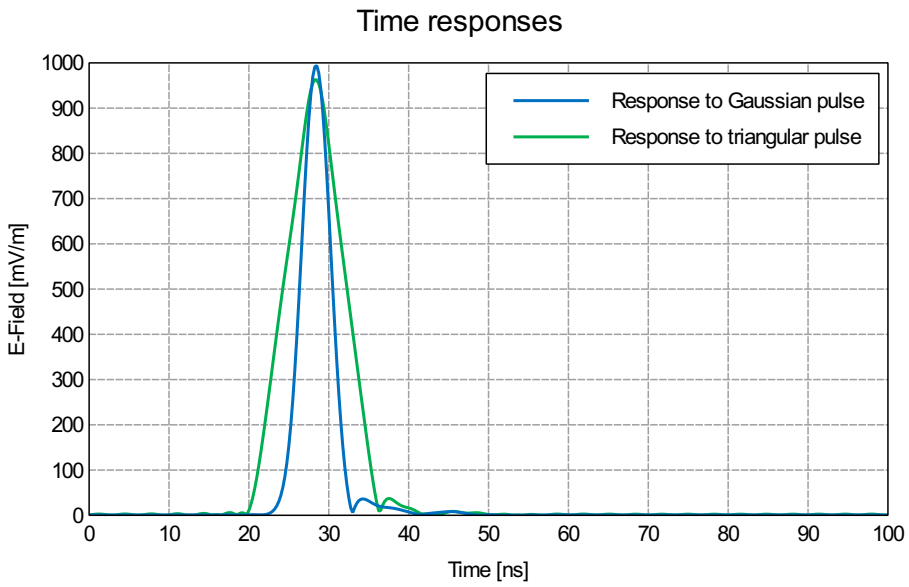


Figure G-1-4: E-Field Magnitude (X position = -2 m; Y position = -2 m; Z position = 0 m)

Chapter H

Special solution methods

H-1 Forked dipole antenna (continuous frequency range)

Keywords: ADAPTFEKO, continuous frequency, adaptive sampling

We will consider the input admittance of a simple forked dipole as shown in Figure H-1-1.

This example is based on the paper "Efficient wide–band evaluation of mobile communications antennas using [Z] or [Y] matrix interpolation with the method of moments", by K. L. Virga and Y. Rahmat-Samii, in the *IEEE Transactions on Antennas and Propagation*, vol. 47, pp. 65–76, January 1999, where the input admittance of a forked monopole is considered.

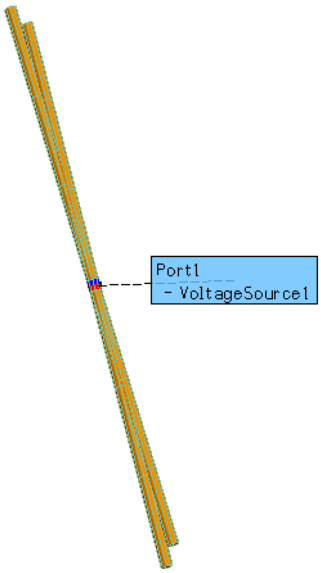


Figure H-1-1: The forked dipole geometry.

H-1.1 Forked dipole model

Creating the model

The model is simple, and can be created as follows:

- Create the following variables
  - `fmin = 100e6` (The minimum calculation frequency.)
  - `fmax = 300e6` (The maximum calculation frequency.)
  - `wireRadius = 1e-3` (The radius of the wire segments.)
- Create the following named points:
  - `point1` (-0.01,0,0.5)
  - `point2` (0,0,0.01)
  - `point3` (0.01,0,0.466)

  - `point4` (0,0,-0.01)

- Create 2 line primitives. One from `point1` to `point2`, and a second from `point2` to `point3`.
- Copy and mirror the two lines around the *UV* plane.
- Create a line primitive between the named points `point2` and `point4`. Label this line as `feed`.
- Union all of the lines into a single part.
- Add a wire port to the middle of the feed wire.
- Apply a voltage source (1V, 0°, 50 Ω) to the port.
- Set the solution frequency settings to *Continuous (interpolated) range* between `fmin` and `fmax`.

Requesting calculations

For this example we only wish to view the input impedance of the forked dipole. No calculations therefore need be specifically requested.

Meshing information

Use the *standard* auto-mesh setting with the wire segment radius equal to `wireRadius`.

CEM validate

After the model has been meshed, run *CEM validate*. Take note of any warnings and errors. Correct any errors before running the FEKO solution kernel.

H-1.2 Results

In order to view the results for this example, we create a Cartesian graph and plot the real and imaginary parts of the input impedance of the voltage source. The input impedance is plotted in Figure H-1-2. Figure H-1-3 shows the same results over a smaller frequency band.

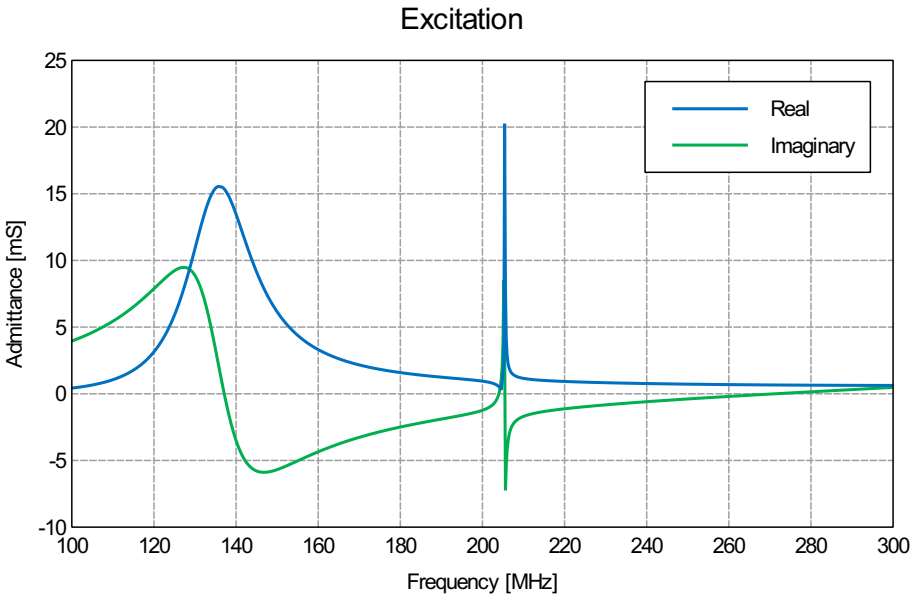


Figure H-1-2: Real and imaginary parts of the input admittance of the forked dipole.

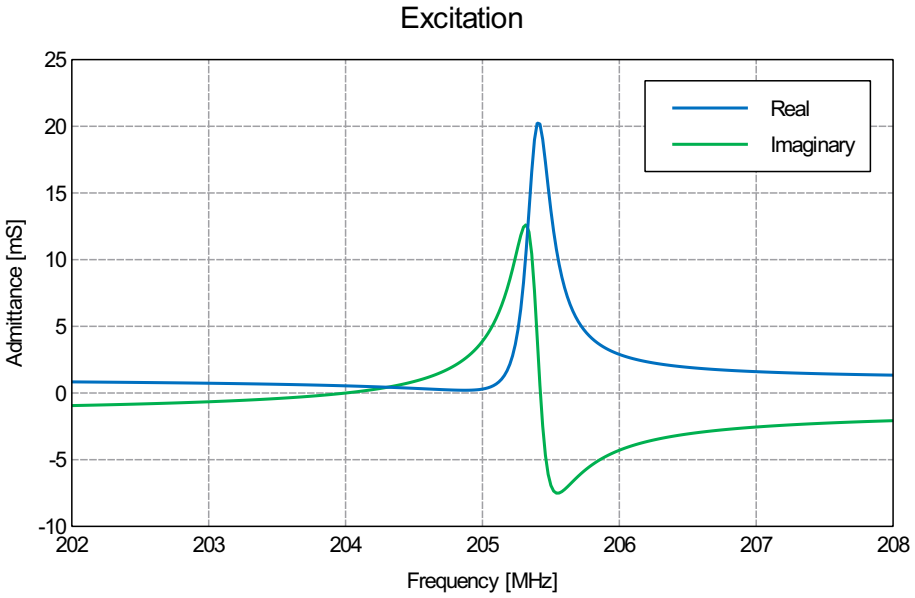


Figure H-1-3: Input admittance of the forked dipole around the resonance point.

H-2 Using the MLFMM for electrically large models

Keywords: MLFMM, large model, Radar cross section, trihedral

In this example we consider a single plane wave incident (from θ=60° and ϕ=0°) on a large trihedral. The size of the trihedral ($13.5\lambda^2$ surface area) was chosen such that it can still be solved using the standard MoM. This example is large enough to demonstrate the advantage of using the MLFMM, but larger examples will show a proportionally larger resource saving.

Figure H-2-1 shows an illustration of the trihedral with a plane wave source.

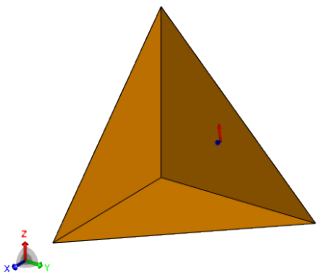


Figure H-2-1: Plane wave incident on an electrically large trihedral.

H-2.1 Large trihedral

Creating the model

The steps for setting up the model are as follows:

- Define the following variables:
  - `lambda = 1` (Free space wavelength.)
  - `freq = c0/lambda` (The operating frequency.)
  - `s = 3*lambda` (Side lengths of the trihedral.)
- Create the first polygonal plate. The three corner points are (0,0,0), (s,0,0) and (0,s,0).
- Create the second polygonal plate. The three corner points are (0,0,0), (0,0,s) and (s,0,0).
- Create the third polygonal plate. The three corner points are (0,0,0), (0,s,0) and (0,0,s).
- Union the plates.
- Define a linear plane wave source at θ=60° and ϕ=45°.
- Set the frequency to `freq`.

The model is now set up to be solved with the default MoM. The model should be set to use the MLFMM with default values. All solution method settings, including MLFMM, are set on the *Solver settings* dialog.

Requesting calculations

The solution requests are:

- Create a 180° vertical far field request. (-180°$\leq\theta\leq$180°, with ϕ=45° and 2° steps)

Meshing information

Use the *standard* auto-mesh setting.

CEM validate

After the model has been meshed, run *CEM validate*. Take note of any warnings, notes and errors. Please correct error before running the FEKO solution kernel.

H-2.2 Results

The runtime and memory that was used for both the MoM and MLFMM models are given in Table H-2-1. As the problem size increases, the difference will become increasingly significant. Memory and other detailed information is available in the `*.out` file as well as in POSTFEKO's *Details Browser*.

Solution method	*Memory (MBytes)*	*Runtime (seconds)*
MoM	487	146
MLFMM	165	53

Table H-2-1: Memory and runtime requirements for the MoM and MLFMM models

Figure H-2-2 compares the results obtained with the MLFMM with those obtained with the MoM.

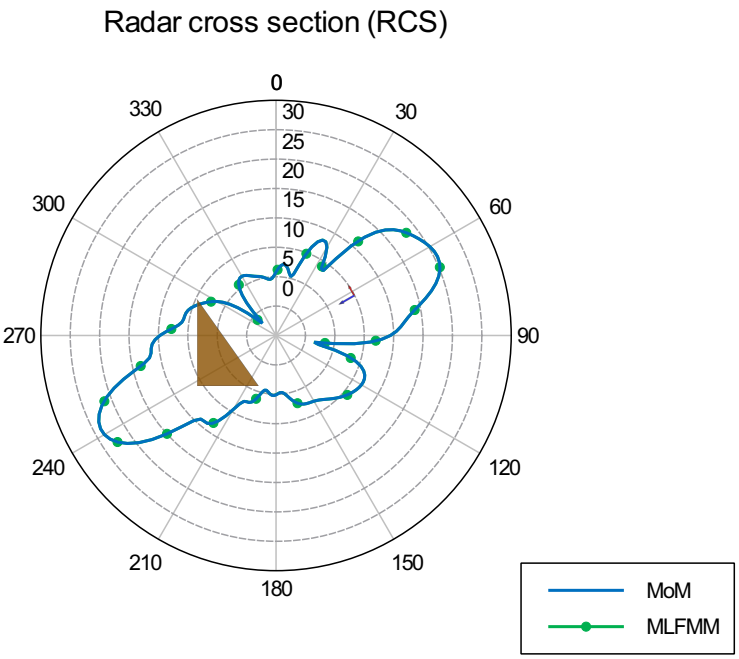


Figure H-2-2: Bistatic RCS of a trihedral. Comparison of the MLFMM and MoM results.

H-3 Horn feeding a large reflector

Keywords: Waveguide, horn, reflector, PO, MLFMM, near field source, spherical modes source, equivalent source, decouple, far field

A cylindrical horn is excited with a waveguide port is used to feed a parabolic reflector at 12.5 GHz. The reflector is electrically large (diameter of 36 wavelengths) and well separated from the horn. An illustration of the model is shown in figure H-3-1. This example illustrates several of the techniques available in FEKO to reduce the required resources for electrically large models.

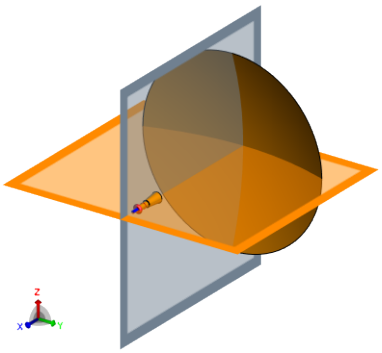


Figure H-3-1: Illustration of a circular horn and parabolic dish reflector.

It is important to understand the problem to be solved and the approximations being made to reduce the required resources. Employ the following techniques to reduce the required resources:

- Use the multilevel fast multipole method (MLFMM) instead of the method of moments (MoM) for electrically large models. The required memory are reduced considerably by using MLFMM. The MLFMM solution is used as the reference solution in the results section (H-3.7).
- Large element physical optics (LE-PO) used on sub parts of the model.
- Subdivide the problem and use an equivalent source. Possible equivalent sources are:
  - Near field source - using the equivalence principle, a region can be replaced by equivalent electric and magnetic field sources on the boundary of the region.
  - Spherical modes source - the far field can also be used as an impressed source.

H-3.1 MoM horn and LE-PO reflector

The first example creates the horn and the dish. Simulate the horn using the method of moments (MoM) and the dish reflector using large element physical optics (LE-PO).

Creating the model

The model is created in two parts; create the horn first and second the parabolic dish. Start by defining the following variables.

- freq = `12.5e9` (The operating frequency.)
- lam = `c0/freq` (Free space wavelength.)
- lam_w = `0.0293` (The guide wavelength.)
- h_a = `0.51*lam` (The waveguide radius.)
- h_b0 = `0.65*lam` (Flare base radius.)
- h_b = `lam` (Flare top radius.)
- h_l = `3.05*lam` (Flare length.)
- ph_centre = `-2.6821e-3` (Horn phase centre.)
- R = `18*lam` (Reflector radius.)
- F = `25*lam` (Reflector focal length.)
- w_l = `2*lam_w` (The waveguide length.)

The steps for creating the *horn* are as follows:

- Create a cylinder along the *Z* axis with the base centre at (0,0,`-w_l-h_l`), a radius `h_a` and a height `w_l`. Label the cylinder `waveguide`.
- Create a cone with a base centre (0,0,`-h_l`), a base radius `h_b0`, a height `h_l` and a top radius `h_b`. Label the cone `flare`.
- Union the two parts and then simplify the resulting union. Rename the new part to `horn`.
- Delete the face on the end of the horn.
- Rotate the `horn` with -90° after setting the *axis direction* to (0,1,0).
- Set a local mesh size of `lam/20` on the face at the back of the waveguide section. Create a waveguide port on the same face.
- Add a waveguide source on the waveguide port (Excite the fundamental mode - use the default settings).

The horn is now complete. The next step is to create the *parabolic reflector*.

- Create a paraboloid at (0,0,`F`) with *radius* `R` and *depth* `-F`.

  Label the paraboloid `reflector`.
- Rotate the `reflector` with -90° after setting the *axis direction* to (0,1,0).

- Set the face properties of the reflector to use *Large element PO - always illuminated*, during the solution.
- Decouple the MoM and LE-PO by enabling the *Decouple PO and MoM solutions* option on the *High frequency* tab under *Solve/Run → Solver settings*.
- Set a magnetic plane of symmetry at Z=0 and an electric plane of symmetry at Y=0.
- Set the frequency to `freq`.

Requesting calculations

Create a full 3D far field request with an increment of 5° for θ over the range -180°$\leq\theta\leq$180° and an increment of 5° for ϕ over the range 0°$\leq\phi\leq$180°. On the *Advanced* tab, select continuous far to be calculated. Spatially continuous far field allow the far field to be re-sampled to any resolution in POSTFEKO.

Meshing information

Use the *coarse* auto-mesh setting. Reduce simulation time in this example by using the coarse mesh setting. The *standard* mesh setting is recommended in general.

CEM validate

After the model has been meshed, run *CEM validate*. Take note of any warnings and errors. Correct any errors before running the FEKO solution kernel.

Save the file and run the solver. Note that this is a large simulation and may take a relatively long time to complete. The model that has been created will be referred to as the "original" model throughout the rest of this example.

H-3.2 MLFMM horn and PO reflector

For the second example, simulate the horn using the MLFMM and the dish reflector using physical optics (PO).

Creating the model

The steps for setting up the model are as follows:

- Open the original model and save it under a new name.
- Solve the model with the multilevel fast multipole method (MLFMM).
- Couple the MLFMM and PO by enabling the *Couple PO and MoM/MLFMM solutions (iterative technique)* option on the *High frequency* tab under *Solve/Run → Solver settings*.

- Set the face properties of the reflector to use *PO - always illuminated*, during the solution.

Note that a notice will be encountered when running the solution as result of using symmetry in conjunction with the MLFMM. Using symmetry will not lead to reduction of memory or run-time, as only the geometry is mirrored for a model solved by the MLFMM.

Requesting calculations

Ensure that the 3D far field is still requested from the previously generated model. Save the file and run the solver.

H-3.3 MLFMM horn and LE-PO reflector

For the third example, simulate the horn using the MLFMM and the dish reflector using large element physical optics (LE-PO).

Creating the model

The steps for setting up the model are as follows:

- Open the original model and save it under a new name.
- Solve the model with the multilevel fast multipole method (MLFMM).
- Couple the MLFMM and LE-PO by enabling the *Couple PO and MoM/MLFMM solutions (iterative technique)* option on the *High frequency* tab under *Solve/Run → Solver settings*.
- Set the face properties of the reflector to use *Large element PO - always illuminated*, during the solution.

Note that a notice will be encountered when running the solution as result of using symmetry in conjunction with the MLFMM. Using symmetry will not lead to reduction of memory or run-time, as only the geometry is mirrored for a model solved by the MLFMM.

Requesting calculations

Ensure that the 3D far field is still requested from the previously generated model. Save the file and run the solver.

H-3.4 Generate equivalent near field and spherical modes sources using only the horn

The model is now simplified by simulating the horn by itself. A set of near field points are calculated around the horn and then used as a source for the reflector.

Creating the model

The model is created by saving the previous model with a new name and making the required changes. First we delete the dish from the original model and create a model containing only the horn. The near and far field information is calculated and saved to a file.

The steps for setting up the model containing only *the horn* are as follows:

- Open the original model and save it under a new name.
- Remove the reflector from the model.

Requesting calculations

Request a 3D far field with an origin at (0, 0, 0); -180°$\leq\theta\leq$180° in θ=5° increments; 0°$\leq\phi\leq$180° in ϕ=5° increments. This is the default set of values when *3D pattern* is selected on the dialog. On the *Advanced* tab, enable the option to *Calculate continuous far field data* and also *Calculate spherical expansion mode coefficients*. This file will be stored with a `*.sph` extension.

Create a spherical near field request with its origin at (`w_l`,0,0), radius of `1.3*w_l`, 0°$\leq\theta\leq$180°, 0°$\leq\phi\leq$360° and an increment of 5° for θ and ϕ. Change the check box setting so that the near field is not sampled on the edges. Sampling on the edges would create duplicate request points at 0° and 360° and also at the poles of the spherical coordinate system. Ensure that the *Export fields to ASCII file* is checked on the *Advanced* tab of the *Near field request* dialog. This saves the electric near fields to an `*.efe` file and the magnetic near fields to an `*.hfe` file.

Meshing has already been set up and nothing should be changed. Save the file and run the solver. Once the simulation has completed, the model containing on the reflector can be constructed.

H-3.5 Near field source and LE-PO reflector

The steps for setting up the model containing the *reflector* and equivalent near field source are as follows:

- Open the original model and save it under a new name.
- Remove the waveguide source and port.
- Remove the horn from the model.
- Create a new near field data definition. Set the position of the workplane equal to (`w_l`,0,0). Enter the name of the `*.efe` and `*.hfe` in the *Source* group box. The coordinate system is a spherical coordinate system with radius `1.3*w_l` and the number of θ and ϕ points is equal to 36 and 71 respectively.
- Create a new near field source that uses the near field data definition.

Requesting calculations

Ensure that the 3D far field is still requested from the previously generated model.

Save the file and run the solver.

H-3.6 Spherical modes source and LE-PO reflector

In this model, the far field that was stored in the `*.sph` file from the horn model is used. As for the previous model when the near fields were used, the horn is removed from the full model and replaced by an equivalent source.

Creating the model

The original model is opened again and saved with a different name. The horn is removed and a new source is created for the dish.

The steps for setting up the model containing the *reflector* and equivalent spherical modes source are as follows:

- Open the original model and save it under a new name.
- Remove the waveguide source and port.
- Remove the horn from the model.
- Create a new spherical modes data definition at the origin. Select the `*.sph` file that has been created during the previous simulation.
- Create a new spherical modes source that uses the spherical modes data definition.

Requesting calculations

No calculation requests need to be created since the far field request was created in the original model and it has not been removed.

Save the file and run the solver.

H-3.7 Comparative results

The required resources (memory and CPU time) is listed in Table H-3-1. It is clear that the required resources are decreased by using approximations. By simply using LE-PO as solution method for the reflector, the memory requirement and solution time is reduced by several orders of magnitude. By sub-dividing the model into equivalent source models, the resource requirements can be reduced even further.

The differences in the results is shown in figure H-3-2 and H-3-3 respectively.

Table H-3-1: Comparison of resources using different techniques for large models.

Model	*RAM [MB]*	*Time [s]*	*Total Time [s]*
MLFMM (benchmark)	4910	1403	1403
MLFMM Horn + PO Reflector	700	3663	3663
MLFMM Horn + LE-PO Reflector	285	610	610
MoM Horn + LE-PO Reflector	196	84	84
Generate equivalent source data	8	36	
Near field source + LE-PO Reflector	41	147	183
Spherical modes source + LE-PO Reflector	32	6	42

We can see there is excellent comparison between the results. Difference observed in the results are due to coupling between the horn and reflector only taken into account for MLFMM solution. Although there is no restriction on the size of LE-PO triangles, the geometry must be accurately meshed. For example, had a flat plate been used, only two triangles would have been required to obtain the same results.

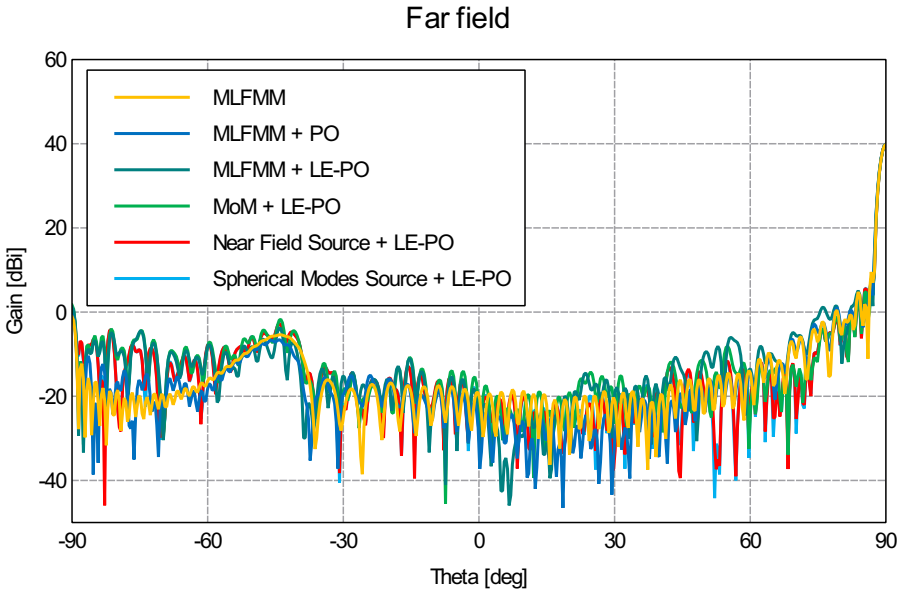


Figure H-3-2: Gain of the reflector antenna calculated using different techniques over a 180 degree angle.

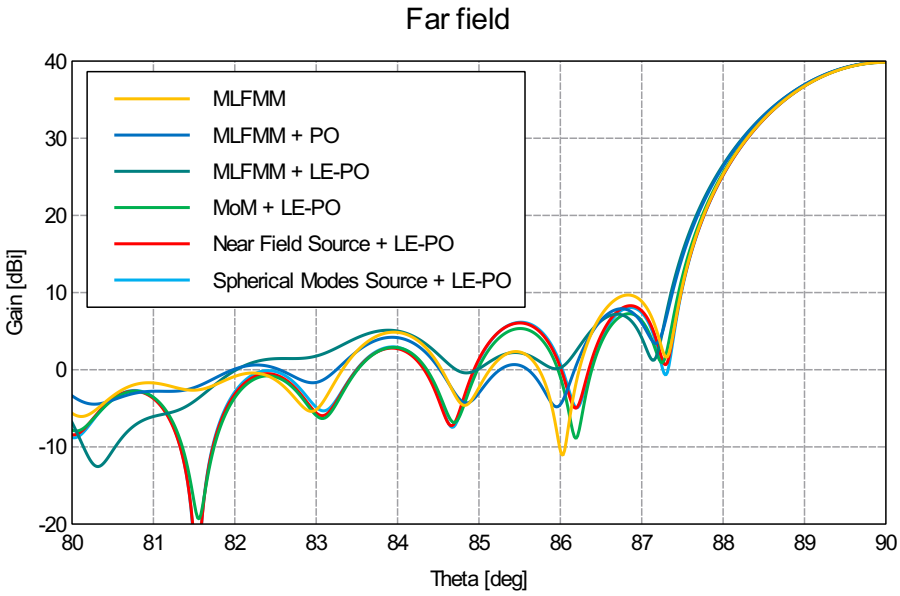


Figure H-3-3: Gain of the reflector antenna calculated using different techniques - main lobe.

H-4 Optimise waveguide pin feed location

Keywords: waveguide, NGF, optimisation, grid search

In practice, a waveguide can be fed using a pin placed a quarter of a waveguide wavelength from a terminated end of the waveguide. This can be a difficult position to determine for arbitrary waveguide cross-sections, since the waveguide wavelength is not always known. This example will demonstrate how to use an optimisation grid search and the NGF solution settings to efficiently analyse the effect of the pin offset on the reflection coefficient.

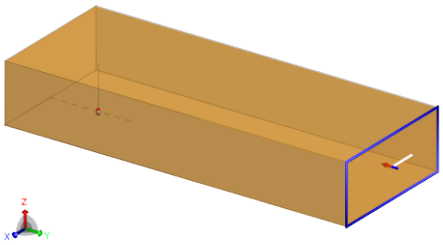


Figure H-4-1: 3D view of a waveguide fed with a pin feed.

H-4.1 Pin fed waveguide

Creating the model

The steps for setting up the model are as follows:

- Set the model unit to millimetres.
- Define the following variables:
  - `freq = 10e9` (Centre frequency.)
  - `lambda = c0/freq*1e3` (Free space wavelength in millimetres.)
  - `n = 1` (Feed pin position index.)
  - `pin_step_size = lambda/32` (Distance between pin positions.)
  - `pin_length = 0.9*lambda/4` (Length of pin feed monopole.)
  - `pin_offset = pin_step_size*n` (Pin offset from waveguide tip.)
  - `radius = 0.1` (Radius of pin wires.)
  - `waveguide_length = lambda*2` (Length of waveguide section.)
  - `wr90_height = 10.16` (Waveguide height for WR90 (X-Band).)
  - `wr90_width = 22.86` (Waveguide width for WR90 (X-Band).)
- Create a cuboid labelled `waveguide`. It is created with the *Base centre, width, depth, height* definition method. The base centre is at (0,0,0) and with

  - *width*: `wr90_width`
  - *depth*: `waveguide_length`
  - *height*: `wr90_height`

- Set the region of the waveguide to free space.
- Create lines that will be used to imprint vertices on the mesh.
  - Create a line labelled `imprinted_edge_1` starting at (0, `-waveguide_length/2`, 0) and ending at (0, `-waveguide_length/2+pin_step_size`, 0).
  - Copy and translate the line from (0,0,0) to (0, `2*pin_step_size`, 0) a number of 7 times.
- Union all geometry and label it `waveguide_perforated`.
- Activate NGF and set `waveguide_perforated` as a static part. The NGF icon will appear in the tree next to the part. This will prevent any changes to the static part of the model.
- Create the feed pin using a line:
  - *start point*: (0, `pin_offset`, 0)
  - *end point*: (0, `pin_offset`, `pin_length`)
  - *workplane origin*: (0, `-waveguide_length/2`, 0)
- Add a segment port at the connection point between the feed and the waveguide floor.
- Place a waveguide port at the other end of the waveguide. This will absorb the power injected by the pin feed.
- Apply a voltage source (1V, 0°, 50 Ω) to the wire port.
- Set the frequency to `freq`.

Requesting calculations

Magnetic symmetry exists in the X=0 plane, but no performance gain is experienced when used in conjunction with the NGF.

No solution requests are required since the input impedance and reflection coefficient is always available as output for all voltage sources in the model.

Setting up optimisation

- An optimisation search is added with the `Grid search` method and 15 default points.
- The following parameters are set:
  - `n` (min 1; max 15; grid points 15)
- Set an *Impedance* goal to minimise the magnitude of the reflection coefficient.

Meshing information

Use the *fine* auto-mesh setting and set the wire segment radius to `radius`.

CEM validate

After the model has been meshed, run *CEM validate*. Take note of any warnings and errors. Correct any errors before running OPTFEKO.

H-4.2 Results

The first time the solver is run, the full calculation needs to be performed. The solution to the static domain is stored in a file, which can be reused in subsequent simulations. By storing the solution to the static domain, which in this case is the waveguide, FEKO only needs to calculate the effect of the feed pin location in further simulations.

The resource requirements for each iteration are given in Table H-4-1. Note that the first run took significantly longer than the rest. This effect will become more pronounced as the size of the static domain increases relative to the dynamic domain.

Iteration	*Memory (MBytes)*	*Runtime (seconds)*
1	214.627	52
2	214.708	6
3	214.708	5
4	214.708	5
5	214.708	5
6	214.708	5
7	214.708	5
8	214.708	5
9	214.708	5
10	214.708	5
11	214.708	5
12	214.708	5
13	214.708	5
14	214.708	5
15	214.708	5

Table H-4-1: Comparison of resource requirements for each iteration.

The port is optimally matched when the magnitude of the reflection coefficient is as small as possible or the input impedance is equal to 50 Ω. On the Smith chart depicted in Figure H-4-2, we can see this condition is met roughly at iteration 6. This corresponds to a feed pin position given by Eq. H-4-1

$$pin\_offset \approx 6 \times pin\_step\_size = \frac{6}{32}\lambda. \tag{H-4-1}$$

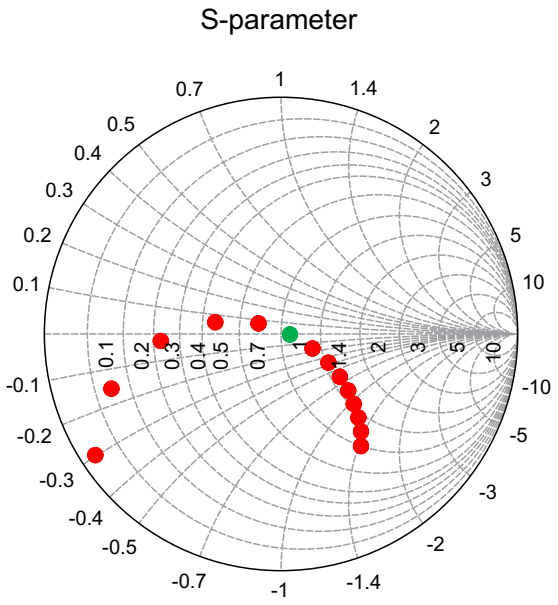


Figure H-4-2: Smith chart showing the reflection coefficient for each feed pin position.

Chapter I

User interface tools

I-1 Introduction to application automation

Keywords: application automation, API, CADFEKO, POSTFEKO, scripting

I-1.1 Automation

Automation is the execution of a task by means of a script (where either some or no user interaction is required). Typical scenarios where automation is used include repetitive tasks or tasks that require several steps, calculations or modifications for the user to complete. Automation is supported in both CADFEKO and POSTFEKO.

I-1.2 Automation terminology

The terminology of automation will be briefly described. The terminology will be used extensively in this example.

Lua *Lua* is a lightweight multi-paradigm programming language designed as a scripting language. Both CADFEKO and POSTFEKO feature a scripting interface based on the Lua language.

API The FEKO *application programming interface* (API) defines the namespaces, objects, methods and functions used to access and control the FEKO applications.

handle A *handle* is a variable that refers to an object. Instead of referring to the object itself, the handle is used to refer to the object.

object An *object* is an entity within an object oriented programming language with two main characteristics: a state and a behaviour. The settings of the object are stored in its *properties* and its abilities are accessed through *methods*.

app The *Application* is either CADFEKO or POSTFEKO.

cf and pf Each application has a "namespace" that groups all objects and properties for that particular application. The CADFEKO namespace is `cf` and the POSTFEKO namespace is `pf`. This means that all objects, static functions and nested namespaces will be accessed using the application's namespace. As an example, `cf.GetApplication()` calls the `GetApplication()` static function in the `cf` namespace.

project The *project* is the specific CADFEKO model the user is working on, for example, `patch_antenna.cfx`. Note that *project* is specific to CADFEKO.

type Each object has a type. Lua supports the following standard types: number, boolean, string, table, function, userdata and nil. Objects created in CADFEKO or POSTFEKO will usually have a "Type" property. It is used to determine the object's type since these are of "userdata" type for standard Lua.

collection A *collection* is a special object that contains objects of which there can be more than one. For example, there can be multiple sources, farfields, geometry parts and so on. When referencing an item in a collection, an index must always be specified, for example `farfield[1]` or `farfield[''FarField'']`.

enum An *enumerated list* or *enum* is a set of options. In the graphical user interface these options relate to grouped options in a dropdown box or a radio button group. The user is unable to modify the list of options and must select one of the options.

Examples of enums:

- Select either the *Frequency independent*, *Debye relaxation*, *Cole-Cole*, *Havrillak-Negami*, *Djordevic-Sarkar* or the *Frequency list* option for the *Definition method* (*Create dielectric medium* dialog).
- Select either the *Fine*, *Standard*, *Coarse* or *Custom* option for the *Mesh size* (*Create mesh* dialog).

method A *method* is a function that acts on a particular object. Methods are called using the ":" after the object to which the method belongs.

For example, `Cuboid:ConvertToPrimitive()` returns the geometry in its primitive base form, whereas `FarField:Duplicate()` returns a duplicate far field solution entity.

`Cuboid` and `FarField` are the objects and `ConvertToPrimitive` and `Duplicate` are the methods. Note the use of ":" instead of a "." to distinguish between properties and methods.

static function A *static function* is a function that is not associated with an object, as opposed to a method that works on a particular object. Static functions are called using ".".

For example, `cf.Cuboid.GetDefaultProperties()` is a static function that creates the properties table for a cuboid. `cf.GetApplication()` is a static function that returns the application object. Note the use of "." instead of a ":" after `cf`.

I-1.3 Navigating the API documentation

The application programming interface (API) documentation is contained in the *Working with CADFEKO* and *Working with POSTFEKO* chapters of the User Manual. The integrated help is simple to navigate due to its *Back* and *Forward* functionality and the *Index* or *Search* tab is useful to find a specific item.

For this example we will make use of the *Index* tab in the integrated help. For conciseness we will use the terminology, *search*, when we refer to the *Look for* an item on the *Index* tab.

Hyperlinks are indicated by blue text. Clicking on a hyperlink will navigate to other sections in the documentation.

The following example illustrates how to navigate the documentation and the use of correct syntax. The example contains steps to create a wire-fed patch antenna on a planar multilayer substrate.

The example can be found in the

```
examples\ExampleGuide_models\Example-I01-Introduction_to_Application_Automation
```

directory of the FEKO installation.

I-1.4 Example: Patch antenna on planar multilayer substrate

This example demonstrates how application automation can be used to create a patch antenna on a planar multilayer substrate. The Lua script used to create the model is included with this example.

As a first step we need a "handle" on the CADFEKO application. We open `CADFEKO` in the background. Through the use of the handle, *app*, we can do subsequent tasks in the same instance of CADFEKO. One could say that the open CADFEKO window is given the label *app*.

```
app = cf.GetApplication()
```

Next we start a new empty project. This is the same as going to *Application menu* → *New project* in CADFEKO.

```
my_project = app:NewProject()
```

One could also continue to work in an existing project by using `app.Project` or open a file by using `app:OpenFile()`.

Note that we will use the integrated help extensively during the example. Most of the syntax we require is contained in the section entitled *Object reference (API)* under the *Working with CADFEKO* and *Working with POSTFEKO* sections in the User Manual (PDF) or integrated help.

Create a patch or rectangle

For information on how to create a rectangle, search[1] for *rectangle* in the integrated help.

The results returned in response to our *rectangle* query includes *Rectangle (object)*. Double-click *Rectangle (object)*.

The *Rectangle (object)* is part of the geometry collection since there can be more than one geometry part in the model tree.

Note that no information is given at the *Rectangle (object)* description regarding the creation of a rectangle. To find information on how to create the rectangle, we click the *GeometryCollection* hyperlink (see Figure I-1-1) to navigate to the *GeometryCollection*, see Figure I-1-2.

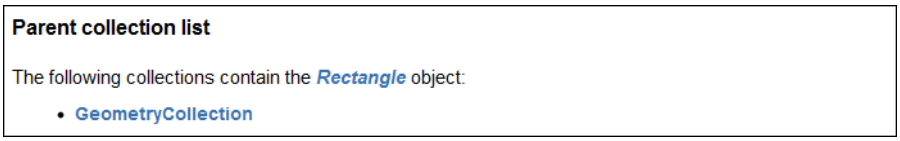


Figure I-1-1: A snippet of the *Rectangle* object which shows the *Parent collection list*.

Figure I-1-2: A snippet of the *GeometryCollection* which shows the *Usage locations*.

Under the *GeometryCollection* we see the *Method list*, see Figure I-1-3. It indicates that we can create many other geometry primitives including a rectangle.

[1] Use the "*Look for:*" functionality on the *Index* tab.

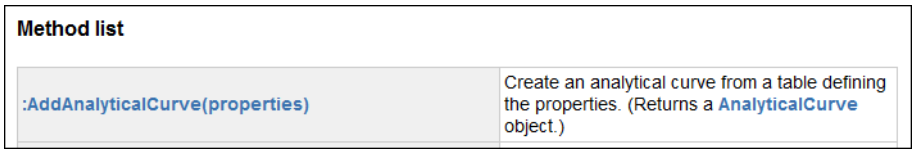


Figure I-1-3: A snippet of the *GeometryCollection* which shows the first method (of many) in the *Method list*.

In the *Method list* (see Figure I-1-3), search for `:AddRectangle(cornerpoint, width, depth)`.

Click the `:AddRectangle(cornerpoint, width, depth)` hyperlink to navigate to a short description for each parameter, see Figure I-1-4.

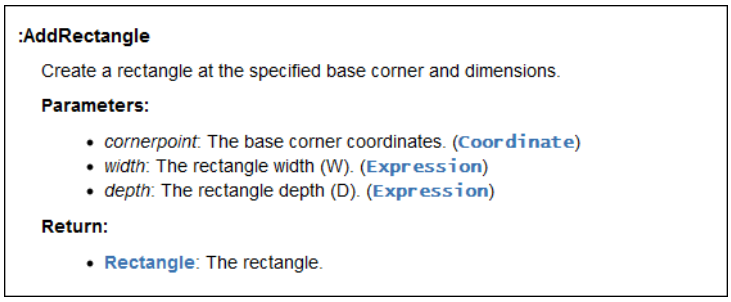


Figure I-1-4: A short description for the `AddRectangle` method and its parameters.

We see *width* and *depth* are both expressions and *cornerpoint* is a *Coordinate* type.

For more information regarding coordinates, click the *Coordinate* hyperlink, see Figure I-1-4.

Click *Point* (see Figure I-1-5) and navigate to the *Point* description.

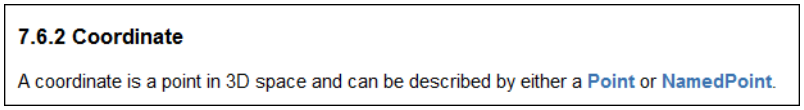


Figure I-1-5: A short description for the parameter, *Coordinate*.

Here an example is given on how to define a point, see Figure I-1-6.

```
    -- Create a default 'Point' at (0,0,0)

p1 = cf.Point.New()

    -- Assign values to each component of the point

p1.x = 1
p1.y = 1
p1.z = 1

    -- Create a 'Point' with number values

p2 = cf.Point(2,2,2)
```

Figure I-1-6: An example indicating the usage of `.Point`.

On the same page we see the *Property list* containing: `.Type`, `.X`, `.Y` and `.Z`; the *Method list* containing `:DistanceTo(point)` and the *Static function list* containing `.New(x,y,z)` and `.New()`.

We now create the corner point using the following syntax:

```
corner1 = cf.Point(-0.25, -0.25, 0)
```

Now for the rectangle, we still require the syntax to prepend to the `:AddRectangle` method. In Figure I-1-2 we noted `Project(.Geometry)`. Since *Project* is the highest level (our `*.cfx` file), the syntax is complete.

```
my_rectangle = my_project.Geometry:AddRectangle(corner1, 0.5, 0.5)
```

The *my_rectangle* part of the code is our handle or reference in Lua to the newly created geometry part. If we created the rectangle manually ourselves in the graphical user interface (GUI), we would have seen *Rectangle1* appear in the geometry tree. If we run the automation script up to this point, we would also see *Rectangle1* in the GUI, see Figure I-1-7.

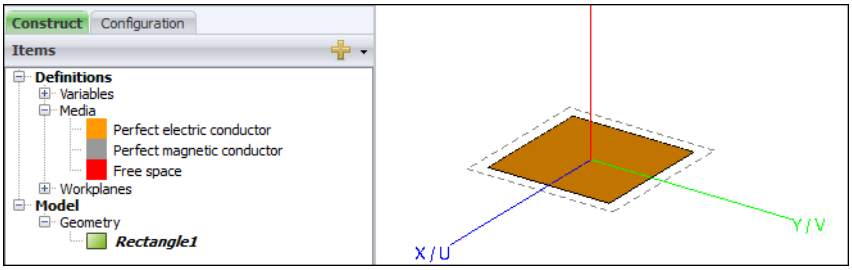


Figure I-1-7: The result in the CADFEKO GUI after running the script.

We can change the handle to *Rectangle1*, but in the automation, *Rectangle1* refers to the label of the rectangle object. For this example, in order to emphasise the difference between the handle and label of an object, we will create handle names with the "my_" prepending syntax.

Create dielectric media

Next we will add two dielectric media. There can be multiple media, as a result we have a collection again. In the integrated help, search for *dielectric*. The results returned in response to our *dielectric* query includes *Dielectric (object)*. Double-click *Dielectric (object)*.

To find more information on how to create a dielectric medium, we click the *MediaCollection* hyperlink, see Figure I-1-8.

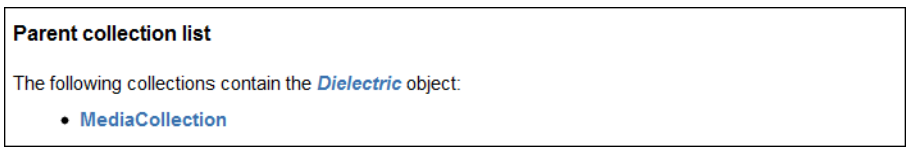
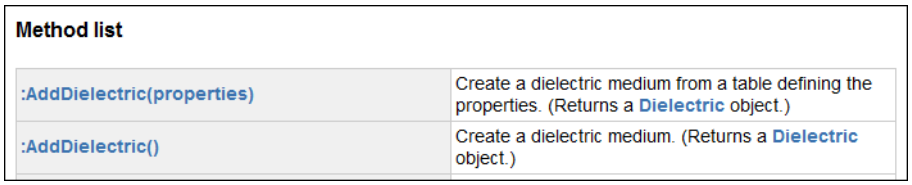


Figure I-1-8: A snippet of the *Dielectric* object which shows the *Parent collection list*.

We note the method `:AddDielectric()`, see Figure I-1-9.



Method list

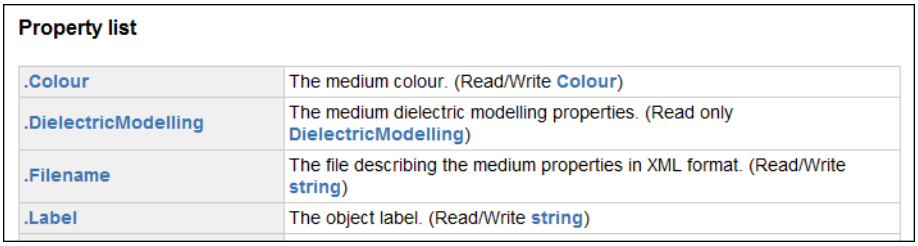
:AddDielectric(properties)	Create a dielectric medium from a table defining the properties. (Returns a Dielectric object.)
:AddDielectric()	Create a dielectric medium. (Returns a Dielectric object.)

Figure I-1-9: A snippet of the *Dielectric* object which shows the *AddDielectric* methods.

We will use this method to create a default dielectric. We get a handle on the dielectric by defining the handle *my_diel*.

```
my_diel = my_project.Media:AddDielectric()
```

Click *Dielectric* (see Figure I-1-9) to navigate to the *Dielectric* object. To view the available properties for the *Dielectric* object, look under the *Property list*, see Figure I-1-10.



Property list

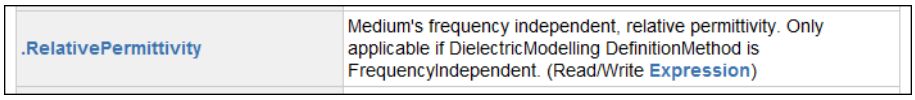
.Colour	The medium colour. (Read/Write Colour)
.DielectricModelling	The medium dielectric modelling properties. (Read only DielectricModelling)
.Filename	The file describing the medium properties in XML format. (Read/Write string)
.Label	The object label. (Read/Write string)

Figure I-1-10: A snippet of the *Dielectric* object which shows the *Property list*.

We now modify the property, `.Label` (see Figure I-1-10).

```
my_diel.Label = "substrate"
```

Next we want to modify the relative permittivity of the dielectric. Click *DielectricModelling* (see Figure I-1-10) and navigate to the property, `.RelativePermittivity` (see Figure I-1-11).



.RelativePermittivity	Medium's frequency independent, relative permittivity. Only applicable if DielectricModelling DefinitionMethod is FrequencyIndependent. (Read/Write Expression)

Figure I-1-11: A snippet of the *DielectricModelling* object which shows the *RelativePermittivity* property.

We now modify the relative permittivity.

```
my_diel.DielectricModelling.RelativePermittivity = 1.5
```

Note in the above the `Label` property is accessed directly but not the `RelativePermittivity`. Similar properties are grouped together, for example in this case `DielectricModelling`. Another example of a property grouping is `SolutionSettings`.

As a result we prepend `.RelativePermittivity` with `.DielectricModelling`.

We now get a handle on the second dielectric by defining the handle, *my_diel2*. The second dielectric is created using the *properties* method, `:AddDielectric(properties)` (see Figure I-1-9).

With this method a temporary table is first created with all the (minimum) required settings for the dielectric.

Note that *properties* refers to a Lua table of properties. An example is the following (only for demonstration and not part of the example script):

```
Luatable1 = {["cheese"] = "blue",["bread"] = "brown", ["nr_of_sandwiches"] = "2"}
```

After creating/populating the table, it is used in the `:AddDielectric` method. We will define a conductivity for the second dielectric. To know which properties we must set, we can use the static function `GetDefaultProperties()` to get all the properties which can be set or modified, see Figure I-1-12. Note that the documentation always gives the exact syntax for accessing the static function, for example `cf.Dielectric.GetDefaultProperties`.

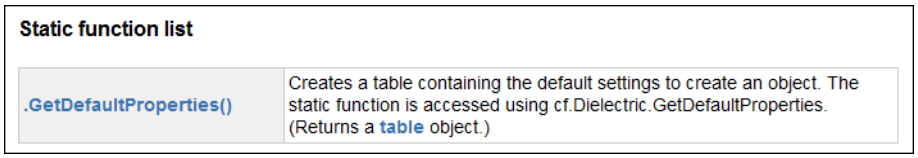


Figure I-1-12: A snippet of the *Dielectric* object which shows the *Static function list*.

```
my_diel2_properties = cf.Dielectric.GetDefaultProperties()
```

Next we use the inspect command to print to screen all the properties.

```
inspect(my_diel2_properties)
```

The screen output is as follows:

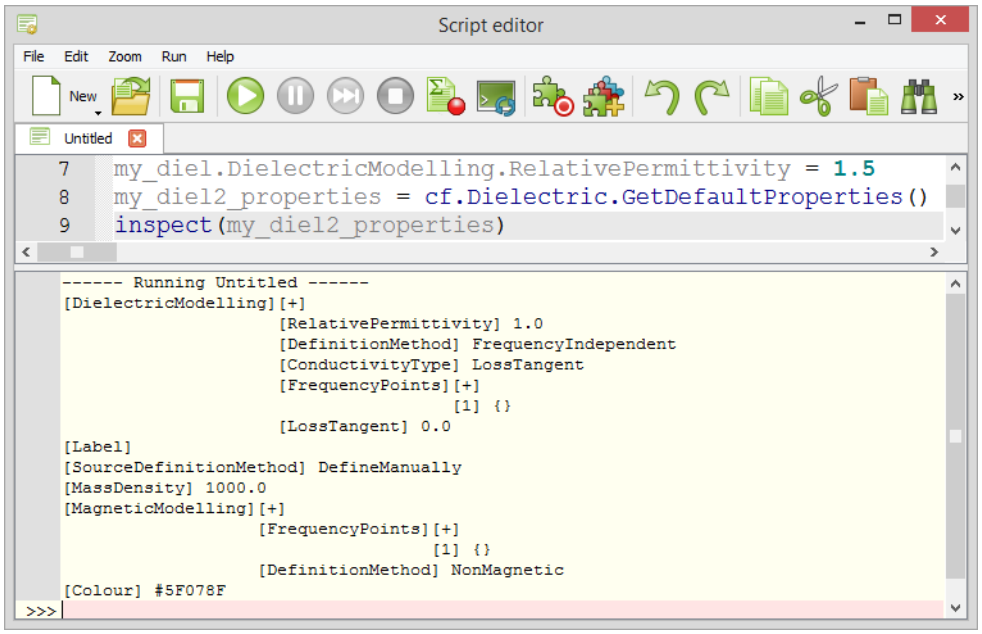


Figure I-1-13: The screen output in the script editor when *my_diel2_properties* is inspected.

Next we only modify the properties we need and apply the modified table to our handle, *my_diel2*. For the second dielectric, we decided to use the conductivity definition instead of the loss tangent. In the CADFEKO GUI the conductivity definition is available on the *Dielectric modelling* tab.

Click the *DielectricModelling* hyperlink (see Figure I-1-14) to navigate to the `DielectricModelling` object.

.DielectricModelling	The medium dielectric modelling properties. (Read only DielectricModelling)

Figure I-1-14: A snippet of the *Dielectric* object which shows the *DielectricModelling* property.

Here we see the `.ConductivityType` property, see Figure I-1-15. To see the available properties listed under `ConductivityType`, we click the *MediumDielectricConductivityTypeEnum* hyperlink, see Figure I-1-15.

.ConductivityType	Medium's conductivity type. Only applicable if DielectricModelling DefinitionMethod is FrequencyIndependent or FrequencyList. (Read/Write MediumDielectricConductivityTypeEnum)

Figure I-1-15: A snippet of the *DielectricModelling* object which shows the *ConductivityType* property.

This navigates us to the *MediumDielectricConductivityTypeEnum* where we must choose one of the given options, see Figure I-1-16.

7.5.71 MediumDielectricConductivityTypeEnum

Enumerations are lists of values that can be used. The enumerations for CADFEKO are available under the `cf` namespace and grouped together under `enums`. This means that the MediumDielectricConductivityTypeEnum enumeration is accessed as illustrated below.

`cf.Enums.MediumDielectricConductivityTypeEnum.<enum option>`

LossTangent	LossTangent
Conductivity	Conductivity

Figure I-1-16: A snippet of the *MediumDielectricConductivityTypeEnum*.

```
my_diel2_properties.DielectricModelling.ConductivityType =
     cf.Enums.MediumDielectricConductivityTypeEnum.Conductivity
my_diel2_properties.DielectricModelling.Conductivity = "1e-2"
my_diel2_properties.DielectricModelling.RelativePermittivity = "2.5"
my_diel2_properties.Label = "substrate2"
my_diel2 = my_project.Media:AddDielectric(my_diel2_properties)
```

Run the script and in the CADFEKO GUI we note the two dielectrics, see Figure I-1-17

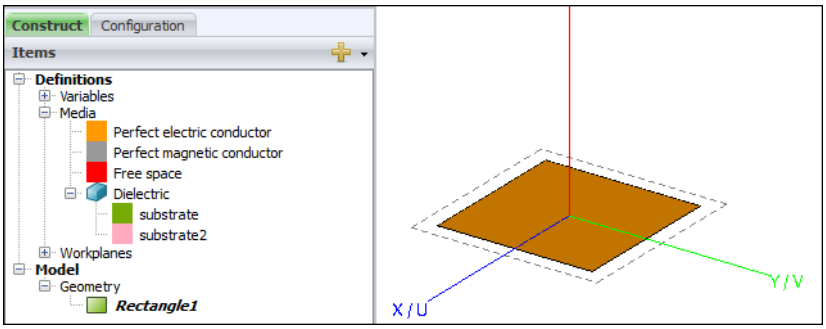


Figure I-1-17: The result in the CADFEKO GUI after running the script.

Create a planar multilayer substrate

If we want to add a planar multilayer substrate in the CADFEKO GUI, we would select the *Construct* tab and click *Plane / ground*.

For information on how to create a planar multilayer substrate, search for *ground* in the integrated help. The results returned in response to our *ground* query includes the *GroundPlane(object)*. Since there can be only one ground plane in a model, this object does not reside in a *Collection*. Similarly we note in the documentation the absence of a *Parent collection list*. It therefore cannot be created using an `:Add...` method but is merely a property of the project.

At the *GroundPlane* object, we see the `.DefinitionMethod` property. To see the available properties listed under `DefinitionMethod`, we click the *GroundPlaneDefinitionMethodEnum*, see Figure I-1-18.

.DefinitionMethod	Infinite plane/ground definition method (environment type). (Read/Write GroundPlaneDefinitionMethodEnum)

Figure I-1-18: A snippet of the *GroundPlane* object which shows the *DefinitionMethod* property.

This navigates us to the *GroundPlaneDefinitionMethodEnum* where we must choose one of the given options, see Figure I-1-19.

7.5.61 GroundPlaneDefinitionMethodEnum

Enumerations are lists of values that can be used. The enumerations for CADFEKO are available under the `cf` namespace and grouped together under `enums`. This means that the GroundPlaneDefinitionMethodEnum enumeration is accessed as illustrated below.

```
cf.Enums.GroundPlaneDefinitionMethodEnum.<enum option>
```

Homogeneous	Homogeneous
PEC	PEC
PMC	PMC
HalfspaceReflectionCoefficient	HalfspaceReflectionCoefficient
HalfspaceSommerfeld	HalfspaceSommerfeld
MultilayerSubstrate	MultilayerSubstrate

Figure I-1-19: A snippet of the *GroundPlaneDefinitionMethodEnum*.

```
my_project.GroundPlane.DefinitionMethod = cf.Enums.GroundPlaneDefinitionMethodEnum.MultilayerSubstrate
```

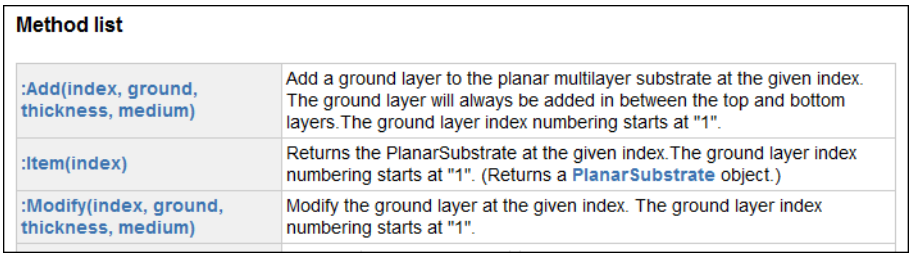
We must now add a layer which will form part of a collection of layers for this object. Return to the *GroundPlane* object in the integrated help, see Figure I-1-20. For more information on how to add a layer, click the *PlanarSubstrateCollection* hyperlink.

Collection list

.Layers	The collection of planar layers for the multilayer substrate. Only applies when the DefinitionMethod is MultilayerSubstrate. By default three layers are created where the top and bottom layers are infinitely thick. (PlanarSubstrateCollection of PlanarSubstrate)

Figure I-1-20: A snippet of the *GroundPlane* object which shows the *Collection list*.

Here we note the `:Add` and `:Modify` methods, see Figure I-1-21.

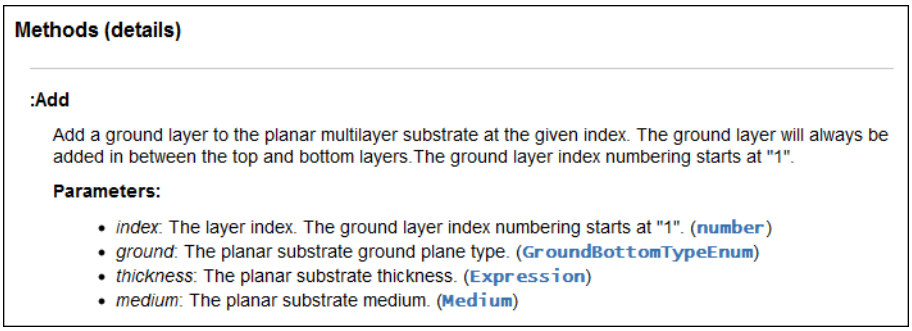


Method list

:Add(index, ground, thickness, medium)	Add a ground layer to the planar multilayer substrate at the given index. The ground layer will always be added in between the top and bottom layers.The ground layer index numbering starts at "1".
:Item(index)	Returns the PlanarSubstrate at the given index.The ground layer index numbering starts at "1". (Returns a PlanarSubstrate object.)
:Modify(index, ground, thickness, medium)	Modify the ground layer at the given index. The ground layer index numbering starts at "1".

Figure I-1-21: A snippet of the *PlanarSubstrateCollection* which shows the *Method list*.

And if we click the `:Add(index, ground, thickness, medium)` method we see a short description for each parameter, see Figure I-1-22.



Methods (details)

:Add

Add a ground layer to the planar multilayer substrate at the given index. The ground layer will always be added in between the top and bottom layers.The ground layer index numbering starts at "1".

Parameters:

- *index*: The layer index. The ground layer index numbering starts at "1". (**number**)
- *ground*: The planar substrate ground plane type. (**GroundBottomTypeEnum**)
- *thickness*: The planar substrate thickness. (**Expression**)
- *medium*: The planar substrate medium. (**Medium**)

Figure I-1-22: A snippet of the `:Add` method.

```
my_project.GroundPlane.Layers:Add(2,"None","0.351",my_diel2)
```

Since there will automatically be two layers created when choosing a multilayer substrate and we have added another, we just modify the third layer. Note that the first layer is indexed as one in the automation which differs from the CADFEKO GUI.

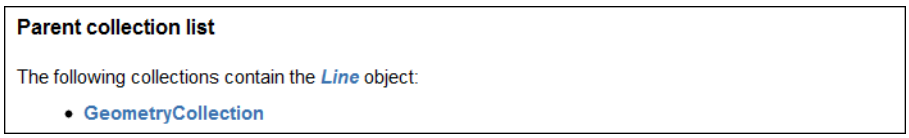
```
my_project.GroundPlane.Layers:Modify(3,"PEC","0.2",my_diel)
```

Create the feed pin

The patch antenna contains a feed pin where the source will be placed. In the integrated help, search for *line*. The results returned in response to our *line* query includes *Line (object)*. Double-click *Line (object)*.

Click the *GeometryCollection* hyperlink (see Figure I-1-23) to navigate to the

`:AddLine(startpoint, endpoint)` method, see Figure I-1-24.



Parent collection list

The following collections contain the *Line* object:

- GeometryCollection

Figure I-1-23: A snippet of the *Line* object which shows the *Parent collection list*.

:AddLine(startpoint, endpoint)	Create a straight line between the given start and end coordinates. (Returns a Line object.)

Figure I-1-24: A snippet of the *GeometryCollection* which shows the *AddLine* method.

Next we add a line (we will add a wire port to this line in later steps). A line needs two points:

```
startPoint = cf.Point(0,0,0)
endPoint = cf.Point(0,0,-0.551)
my_line = my_project.Geometry:AddLine(startPoint,endPoint)
```

We note after running the script that the geometry tree in the GUI contains two parts: *Line1* and *Rectangle1*, see Figure I-1-25.

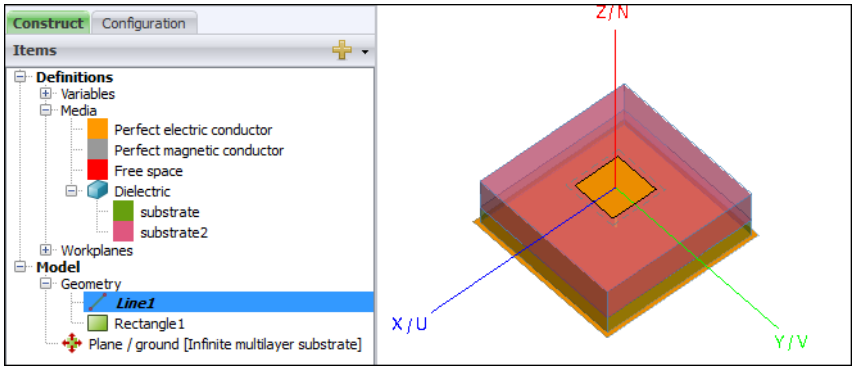


Figure I-1-25: The result of the CADFEKO GUI after running the script

These are default labels created by CADFEKO. The labels would be identical if the parts were created manually in CADFEKO. It is emphasised that *my_line* and *my_rectangle* are *handles* and not *labels*. We can modify *Line1* to have the same label as the handle variable (or any unique label of our choice) as follows:

```
my_line.Label = "my_line_Line1"
```

Union the line and the rectangle

Next we must union[2] the line and rectangle. For information on how to union, search for *union* in the integrated help. The results returned to our *union* query includes *Union(method)*. Double-click *Union(method)*.

We will obtain a *Union (object)* after applying the method to the two parts, see Figure I-1-26.

:Union()	Union all the geometry. (Returns a Union object.)

Figure I-1-26: A snippet of the *GeometryCollection* which shows the *Union* method.

As usual we get a handle on the union object by equating it to something:

```
my_Union = my_project.Geometry:Union()
```

[2]Only applicable to geometry.

Of course we could have used `Union1 = my_project.Geometry:Union()`. The handle name will then be identical to the label.

By using `:Union()` all geometry in the model is unioned. Specific geometry can be unioned by using the syntax `:Union({geometry1, geometry2})`, see Figure I-1-27.

Figure I-1-27: A snippet of the *GeometryCollection* which shows the *Union* method.

Wire port

Next we will add a wire port on the line. Search for *wireport* in the integrated help. The results returned to our *wireport* query includes *WirePort(object)*. Double-click *WirePort(object)*.

There can be multiple ports in a model, as a result the port will be added to the *PortCollection*. To view the method click the *PortCollection* hyperlink in the integrated help see Figure I-1-28.

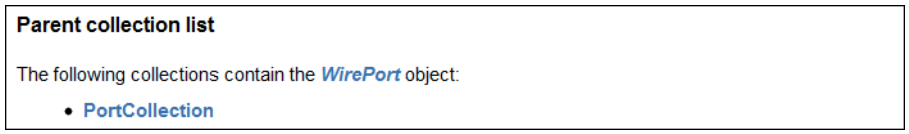


Figure I-1-28: A snippet of the *WirePort* object which shows the *Parent collection list*.

We use the method `:AddWirePort(wire)`, see Figure I-1-29.

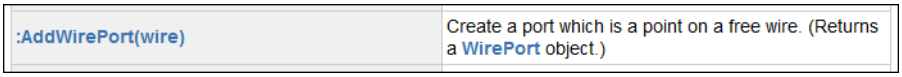


Figure I-1-29: A snippet of the *PortCollection* which shows the *AddWirePort* method.

If we did this in CADFEKO manually, we would open the *Create wire port(geometry)* dialog and for *Edge* enter the label or click on the free wire, *Union1.Wire5*. In the automation we do not know the number of the *Wire*. As a result we use indexing to gain access to the wire in the details tree for *my_line*. Note that indexing can be done either by number or a string.

```
my_port = my_project.Ports:AddWirePort(my_Union.Wires[1])
```

In the above we know the label to be *.Wires[1]* since its is the first and only wire in the *Wires* collection for *Union1*.

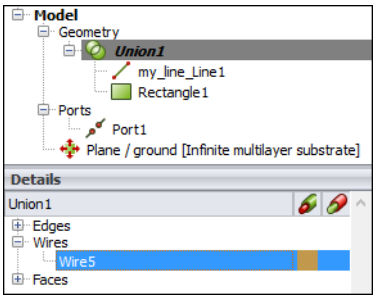


Figure I-1-30: The result in the CADFEKO GUI after running the script

Click *WirePort* to view its properties, see Figure I-1-29. The default `.Location` property for the wire port is always `Start`. The start point of the wire is at the origin. We want to add the port at the end point of the wire. As a result we must select `End` as the location for the port. To see the available properties listed under `Location`, we click `WirePortLocationEnum`, see Figure I-1-31.

.Location	The port location on the wire. (Read/Write WirePortLocationEnum)

Figure I-1-31: A snippet of the *WirePort* object which shows the *Location* property.

This navigates us to the `WirePortLocationEnum` where we must choose `End`, see Figure I-1-32.

7.5.147 WirePortLocationEnum

Enumerations are lists of values that can be used. The enumerations for CADFEKO are available under the `cf` namespace and grouped together under `enums`. This means that the WirePortLocationEnum enumeration is accessed as illustrated below.

```
cf.Enums.WirePortLocationEnum.<enum option>
```

Start	Start
Middle	Middle
End	End
SpecifiedManually	SpecifiedManually

Figure I-1-32: A snippet of the *WirePortLocationEnum*.

```
my_port.Location = cf.Enums.WirePortLocationEnum.End
```

Voltage source – part of a configuration

Next we will add a voltage source to this port. There can be multiple voltage sources, as a result the voltage source will be in a collection. Due to voltage sources being configuration specific, we need to specify the configuration in which the source must be placed. First we get a handle on the default configuration. We also know the default configuration in CADFEKO is *StandardConfiguration*.

For information regarding standard configurations, search for *StandardConfiguration* in the integrated help. The results returned in response to our *StandardConfiguration* query includes *StandardConfiguration(object)*. Double-click *StandardConfiguration(object)*.

Figure I-1-33: A snippet of the *StandardConfiguration* object which shows the *Parent collection list*.

Click the *SolutionConfigurationCollection* hyperlink (see Figure I-1-33) to navigate to the *SolutionConfigurationCollection*.

Here we see the *Project(.SolutionConfigurations)* object contains the *StandardConfiguration*. So the prepending syntax is `Project.SolutionConfigurations`.

Figure I-1-34: A snippet of the *SolutionConfigurationCollection* which shows the *Usage locations*.

The *Index list* indicates we can reference our *StandardConfiguration* through indexing with the use of a number or a string, see Figure I-1-35.

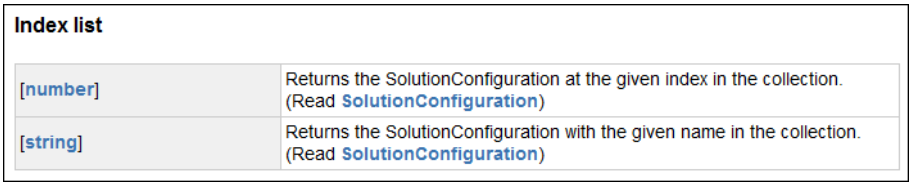


Figure I-1-35: A snippet of the *SolutionConfigurationCollection* which shows the *Index list*.

```
this_config = my_project.SolutionConfigurations[1]
```

Next we specify the voltage source. For information regarding voltage sources, search for *voltage-source*. The results returned in the response to our *voltagesource* query includes *VoltageSource (object)*. Double-click *VoltageSource (object)*.

Next, navigate to the *VoltageSource(object)* in the integrated help. Its *Parent* collection is the *SourceCollection* (remember there can be multiple sources in CADFEKO), see Figure I-1-36.

Figure I-1-36: A snippet of the *VoltageSource* object which shows the *Parent Collection List*.

Navigating to the *SourceCollection* we see that *VoltageSource* can be added to the listed Configuration types, see Figure I-1-37.

Figure I-1-37: A snippet of the *SourceCollection* which shows the *Usage locations*.

We already have a handle on the *StandardConfiguration* (*this_config*) meaning that we can add the voltage source as follows:

```
my_voltage_source = this_config.Sources:AddVoltageSource(my_port)
```

Frequency

Next we set the frequency. For information regarding *frequency*, search for *frequency* in the integrated help. The results returned in response to our *frequency* query includes the *Frequency (object)*. Double-click *Frequency (object)*.

Similar to *VoltageSource* we note *Frequency* is also configuration specific, see Figure I-1-38.

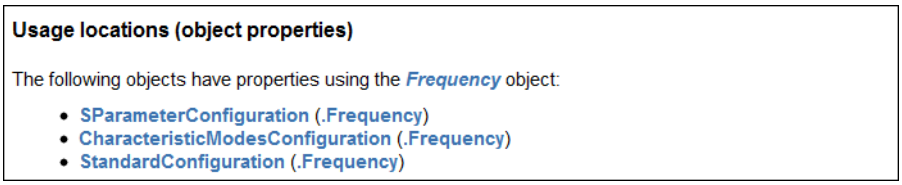


Figure I-1-38: A snippet of the *Frequency* object which shows the *Usage locations*.

To view the available properties for the *Frequency* object, look under *Property list*. Note the `.Start` and `.End` properties.

```
this_config.Frequency.Start = "3e8"
```

Note that for a single frequency solution, we only need to specify `.Start`.

Mesh

Next we want to mesh the model. In the integrated help, search for *Mesh*. The results returned in response to our query includes:

Mesh (method) is the action of meshing the model. The type of mesh created depends on the solver settings.

Mesh (object) is a meshed part. A meshed part can be copied, modified or deleted.

We want to create a mesh, as a result we choose the *Mesh (method)* from the integrated help.

If we think of the manual way of creating a mesh, simply using the keyboard shortcut <CTRL><SHIFT><M> would create a *Standard* mesh. However, the wire radius must first be specified with the result that the keyboard shortcut will not initially create a mesh. Therefore we

look for the wire radius setting in the integrated help. We find the *WireRadius(property)* in the integrated help with two topics listed: *MeshSettings* or *VoxelSettings*.

As this example does not include the FDTD solver, we choose *MeshSettings*. Click *MeshSettings* and navigate to the property, `.WireRadius`, see Figure I-1-39.

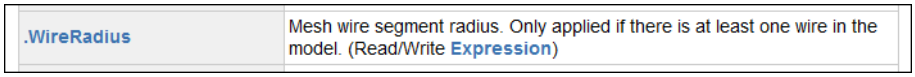


Figure I-1-39: A snippet of the *MeshSettings* object which shows the *WireRadius* property.

We click on the *Mesher* hyperlink (see Figure I-1-40) to see that the *Mesher* object is part of the *Project* object, see Figure I-1-41.

Figure I-1-40: A snippet of the *MeshSettings* object which shows the *Usage locations*.

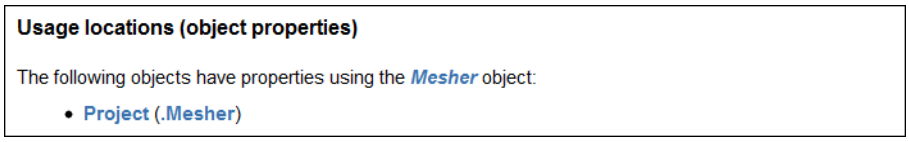


Figure I-1-41: A snippet of the *Mesher* object which shows the *Usage locations*.

```
my_project.Mesher.Settings.WireRadius = "0.001"
```

Now create the mesh using the *Mesh* method of the *Mesher* object.

```
my_project.Mesher:Mesh()
```

Far field request

Next we need output requests, so we create a far field request. For information regarding far fields, search for *farfield* in the integrated help. The results returned in response to our *farfield* query includes *FarField (object)*. Double-click *FarField (object)*.

FarField is part of a collection, as a result we click *FarFieldCollection* to find information on how to create it. It is added to the *FarFields* collection in the configuration, see Figure I-1-42.

Figure I-1-42: A snippet of the *FarFieldCollection* which shows the *Usage locations*.

```
my_farfield = this_config.FarFields:Add3DPattern()
```

We created a 3D pattern, but for planar multilayer substrates any field calculations below ground will result in zero. As a result we modify the far field request to exclude angles of $90<\theta<180$

degrees. We go back to the *FarField(object)* in the integrated help to see the method to get its properties.

```
properties = my_farfield:GetProperties()
inspect(properties)
```

Using `inspect(properties)` shows us which property to modify:

```
properties.Theta.End = "89"
properties.Theta.Increment = "1"
```

Then we apply *properties* back to our far field object:

```
my_farfield:SetProperties(properties)
```

Current request

Lastly we want to add a current request. For information regarding currents, search for *currents* in the integrated help. The results returned in the response to our *currents* query includes *Currents (object)*.

Currents is part of a collection, as a result we click *CurrentsCollection* to find information on how to create it. It is added to the *Currents* collection in the configuration, see Figure I-1-43

Figure I-1-43: A snippet of the *CurrentsCollection* which shows the *Usage locations*.

We use the method `:Add()`, see Figure I-1-44

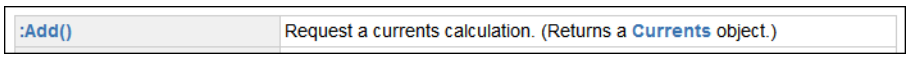


Figure I-1-44: A snippet of the *CurrentsCollection* which shows the *Add* method.

```
my_currents = this_config.Currents:Add()
```

Zoom to extents

So far we have not worked much with the application, we have mostly worked with the project. Often a user wants to zoom to extents to see the entire model. We search for *zoom* in the integrated help and find the *API method, ZoomToExtents*. Double-click *API method, ZoomToExtents*, select *View* as a topic and we are navigated to the *View* object.

There can be more than one 3D view in CADFEKO, so our view is part of a collection. Clicking *ViewCollection* (see Figure I-1-45) shows that the *Views* collection is part of *Application*, see Figure I-1-46.

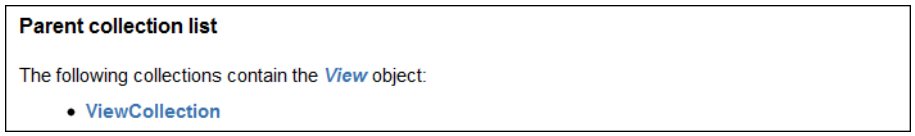


Figure I-1-45: A snippet of the *View* object which shows the *Parent collection list*.

Figure I-1-46: A snippet of the *ViewCollection* which shows the *Usage locations*.

We must tell CADFEKO which view the *ZoomToExtents* operation must be applied on. We use indexing to access the first view (and for this example the only available view).

```
app.Views[1]:ZoomToExtents()
```

Enable parallel FEKO solver

Assume for a moment the model was large and we wanted to invoke the parallel solver. This is also an application specific setting. Enabling parallel would enable parallel for all future models. For information regarding the parallel FEKO solver, search in the integrated help. The results returned in response to our *parallel* query includes *Parallel (property)*. Double-click *Parallel (property)*.

The `.Parallel` property (see Figure I-1-47) lives under the `.FEKO` object. As a result `.Parallel` must be prepended with `.FEKO`.

Figure I-1-47: A snippet of the *FEKOLaunchOptions* object which shows the *Parallel* property.

The `.FEKO` object is part of the *ComponentLaunchOptions* object. To see what object uses the *ComponentLaunchOptions* type, we click *ComponentLaunchOptions*, see Figure I-1-48.

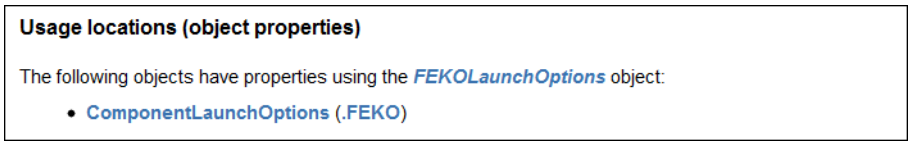


Figure I-1-48: A snippet of the *FEKOLaunchOptions* object showing the *Usage locations (object properties)*.

We see the `.Settings` property on the `Launcher` object is the owner of the `.FEKO` property, see Figure I-1-49. As a result `.FEKO` must be prepended with `.Settings`.

Figure I-1-49: A snippet of the *ComponentLaunchOptions* object showing the *Usage locations (object properties)*.

If we click *Launcher* we see it is part of our *Project*, see Figure I-1-50. As a result `.Launcher` must be prepended with `Project`.

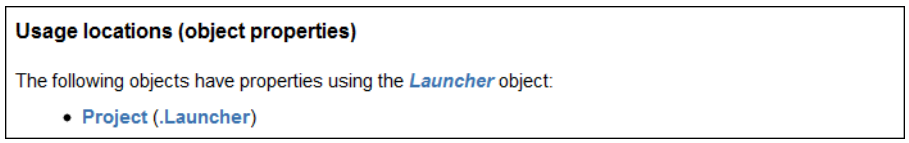


Figure I-1-50: A snippet of the *Launcher* object showing the *Usage locations (object properties)*.

Note however that *Parallel* also has properties of its own, so if we go back to the *Parallel (property)* in the integrated help, we click *FEKOParallelExecutionOptions* (see Figure I-1-47) and there we find the *boolean* property, `.Enabled`. Therefore `.Parallel` must be appended with `.Enabled`.

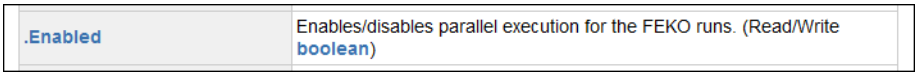


Figure I-1-51: A snippet of the *FEKOParallelExecutionOptions* which shows the *Enabled* property.

```
my_project.Launcher.Settings.FEKO.Parallel.Enabled = true
```

The relationship between the objects and its properties to enable the parallel FEKO solver is shown in Figure I-1-52.

Save the *.cfx file

Next we want to save the `*.cfx` file. For information on how to save the model with a specified file name, search for *saveas* in the integrated help. The results returned to our *saveas* query includes *SaveAs (method)*, see Figure I-1-53. Click *SaveAs (method)*.

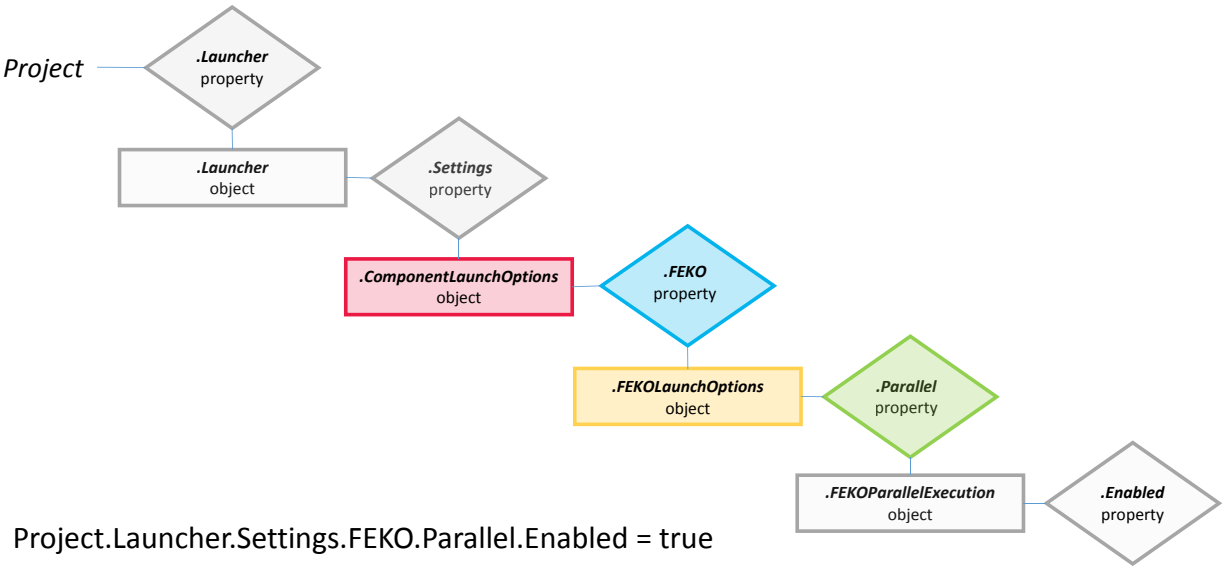


Figure I-1-52: A tree diagram indicating the relationship between the objects and properties. Note that the *.FEKO* property makes use of the *FEKOLaunchOptions* object. The *.FEKOLaunchOptions* object has a number of properties of which *.Parallel* is one.

:SaveAs(filename)	Saves the current model with the given name.

Figure I-1-53: A snippet of the *CADFEKO application* object which shows the *SaveAs* method.

We already have a "handle", *app*, on the CADFEKO application. The `*.cfx` file is saved as `patch_antenna_scripted_model.cfx` in the current folder where the `*.lua` file is located.

```
app:SaveAs([[patch_antenna_scripted_model.cfx]])
```

Note that "[[" and "]]" indicates to the Lua parser that `patch_antenna_scripted_model.cfx` must be use as is and that it is not an object `patch_antenna_scripted_model` with the property `cfx`.

Run the FEKO solver

As a final step we want to run the FEKO solver. For information on how to launch the FEKO solver, search for *runfeko* in the integrated help. The results returned to our *runfeko* query includes *RunFEKO (method)*, see Figure I-1-54. Double-click *RunFEKO (method)*.

:RunFEKO()	Run FEKO. (Returns a LaunchResult object.)

Figure I-1-54: A snippet of the *Launcher* object which shows the *RunFEKO* method.

```
my_project.Launcher:RunFEKO()
```

The CADFEKO application automation script is now complete. Run the script and observe the result in the CADFEKO GUI.

The completed script

Create the model geometry:

```
--[[
PATCH ANTENNA ON PLANAR MULTILAYER SUBSTRATE
=====================================================================
This script creates a patch antenna on a planar multilayer substrate

NOTE: This example is only to be used as an example to demonstrate
how to automate tasks that are often required in CADFEKO.
An electromagnetic solution is only performed to obtain results to
demonstrate how to automate tasks often required in POSTFEKO.

]]--

app = cf.GetApplication()
my_project = app:NewProject()

-- Create a rectangle
corner1 = cf.Point(-0.25, -0.25, 0)
my_rectangle = my_project.Geometry:AddRectangle(corner1, 0.5, 0.5)

-- Create dielectrics
my_diel = my_project.Media:AddDielectric()
my_diel.Label = "substrate"
my_diel.DielectricModelling.RelativePermittivity = 1.5

my_diel2_properties = cf.Dielectric.GetDefaultProperties()
inspect(my_diel2_properties)
my_diel2_properties.DielectricModelling.ConductivityType =
    cf.Enums.MediumDielectricConductivityTypeEnum.Conductivity
my_diel2_properties.DielectricModelling.Conductivity = "1e-2"
my_diel2_properties.DielectricModelling.RelativePermittivity = "2.5"
my_diel2_properties.Label = "substrate2"
my_diel2 = my_project.Media:AddDielectric(my_diel2_properties)

-- Create a planar multilayer substrate
my_project.GroundPlane.DefinitionMethod =
    cf.Enums.GroundPlaneDefinitionMethodEnum.MultilayerSubstrate
my_project.GroundPlane.Layers:Add(2,"None","0.351",my_diel2)
my_project.GroundPlane.Layers:Modify(3,"PEC","0.2",my_diel)

-- Create the feed pin
startPoint = cf.Point(0,0,0)
endPoint = cf.Point(0,0,-0.551)
my_line = my_project.Geometry:AddLine(startPoint,endPoint)
my_line.Label = "my_line_Line1"

-- Union the line and rectangle
my_Union = my_project.Geometry:Union()
```

Create the wire port, add voltage source, define the frequency, mesh the model, add requests:

```
-- Create a wire port
my_port = my_project.Ports:AddWirePort(my_Union.Wires[1])
my_port.Location = cf.Enums.WirePortLocationEnum.End

-- Add a voltage source as part of configuration
this_config = my_project.SolutionConfigurations[1]
my_voltage_source = this_config.Sources:AddVoltageSource(my_port)

-- Set the frequency
this_config.Frequency.Start = "3e8"

-- Create the mesh
my_project.Mesher.Settings.WireRadius = "0.001"
my_project.Mesher:Mesh()

-- Add a far field request
my_farfield = this_config.FarFields:Add3DPattern()
properties = my_farfield:GetProperties()
inspect(properties)
properties.Theta.End = "89"
properties.Theta.Increment = "1"
my_farfield:SetProperties(properties)

-- Add all currents request
my_currents = this_config.Currents:Add()

-- Zoom to extents in the 3D view
app.Views[1]:ZoomToExtents()
```

Launch the FEKO solver:

```
-- Enable the parallel FEKO solver
my_project.Launcher.Settings.FEKO.Parallel.Enabled = true

-- Save the *.cfx file
app:SaveAs([[patch_antenna_scripted_model.cfx]])

-- Run the FEKO solver
my_project.Launcher:RunFEKO()
```

Macro recording

Macro recording allows scripts to be created by starting the recording and then manually performing the actions in CADFEKO. Once all the actions have been performed and script recording stopped, the script can be modified to perform more advanced or repetitive actions. Script recording enables users to learn by analysing generated scripts.

I-1.5 Example: Post-processing the results of a patch antenna on a planar multilayer substrate

This example demonstrates how application automation can be used to post-process the calculated results of our patch antenna placed on a planar multilayer substrate. The Lua script used to process the results is included with this example.

As a first step we need a "handle" on the POSTFEKO application. We open `POSTFEKO` in the background. Through the use of the handle, `app`, we can do subsequent tasks in the same instance of POSTFEKO. One could say that the open POSFEKO window is given the label `app`.

```
app = pf.GetApplication()
```

Next we start a new (blank) session.

```
my_project = app:NewProject()
```

Next we open the `*.fek` file of the patch antenna already simulated.

```
app:OpenFile("patch_antenna_scripted_model.fek")
```

Cartesian graph

We want to view the two radiation pattern cuts on a Cartesian graph. We know that in the POSTFEKO GUI we would first create a new Cartesian graph. We search for the *CartesianGraph (object)* in the integrated help.

As there can be more than one *CartesianGraph*, it forms part of a collection. On the *CartesianGraph* page, click the hyperlink *CartesianGraphCollection*, see Figure I-1-55.

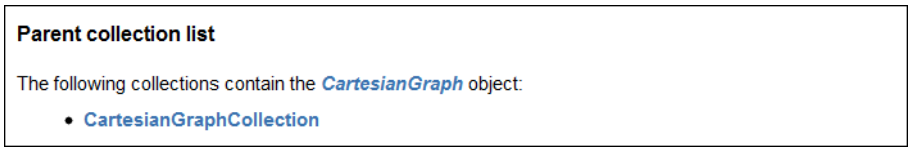


Figure I-1-55: A snippet of the *CartesianGraph* object which shows the *Parent collection list*.

We note the `:Add()` method for the *CartesianGraphCollection*, see Figure I-1-56.

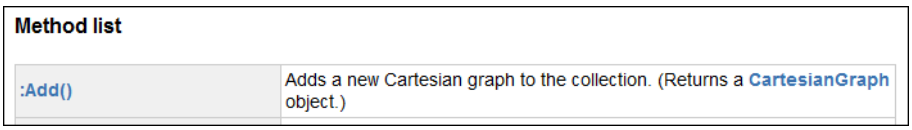


Figure I-1-56: A snippet of the*CartesianGraphCollection* object which shows the *Add()* method.

We see that the *CartesianGraphCollection* resides under the *Application* object (see Figure I-1-57), therefore we have all the information we need:

Usage locations (collections)

The following objects contain the *CartesianGraphCollection* collection:

- Application (.CartesianGraphs)

Figure I-1-57: A snippet of the*CartesianGraphCollection* object which shows the *Usage locations*.

```
my_graph = app.CartesianGraphs:Add()
```

Running the script up to this point will yield the GUI as shown in Figure I-1-58.

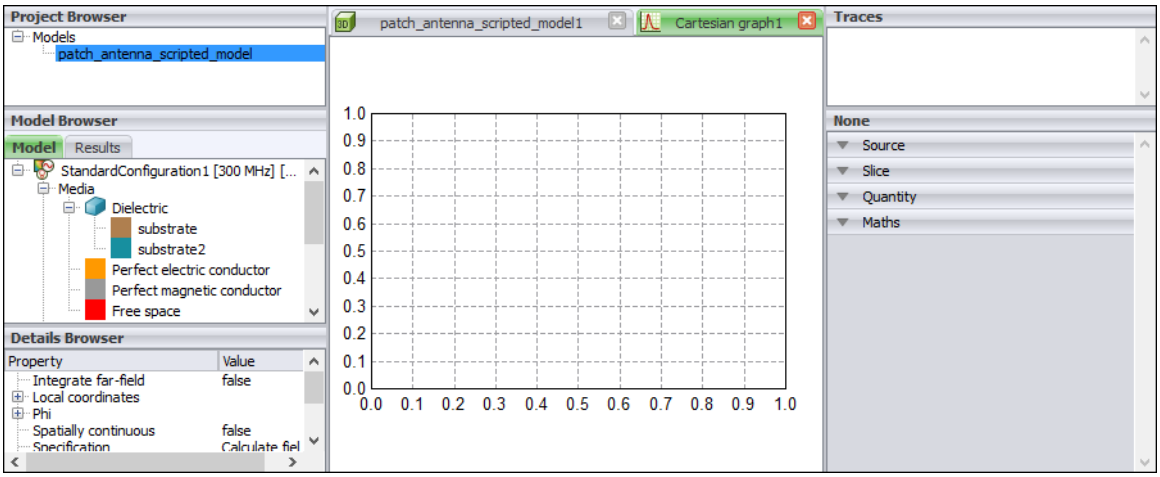


Figure I-1-58: The result in the POSTFEKO GUI after running the script.

Adding far field traces to the Cartesian graph

Next we want to add the far field traces to the graph. If we use *FarField* as a search term in the integrated help, we find several matches.

If there are too many matches in the integrated help we can find more clues by looking in POSTFEKO itself. In POSTFEKO's result palette we note the heading, *Traces* (top panel).

A graph may have more than one trace, therefore traces form part of a collection. We search for *traces* in the integrated help. The results returned in response to our *traces* query includes *Traces (collection)*. When we double-click *Traces (collection)*, we see the following four topics on the *Choose Topic* dialog: *PolarGraph*, *SmithChart*, *CartesianGraph* and *Graph*. Click the topic, *CartesianGraph*, to navigate to the *CartesianGraph* object. We note that *Traces* are added to the *ResultTraceCollection*, see Figure I-1-59.

Collection list

.Traces	The collection of 2D traces on the graph. (ResultTraceCollection of ResultTrace)

Figure I-1-59: A snippet of the*CartesianGraph* object.

Therefore we click *ResultTraceCollection* hyperlinks which navigates us to *ResultTraceCollection*. Here we see under *Method list* the method `:Add(result)`.

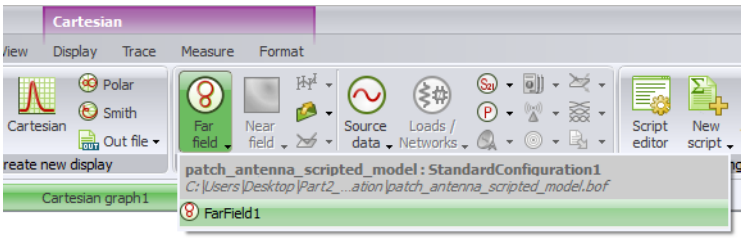


Figure I-1-60: The *FarField1* result available on the *Home* tab (*Add results* group).

We want to add the calculated far field result to a Cartesian graph. This far field result is available in the GUI on the *Home* tab (*Add results* group), see Figure I-1-60.

How do we access this result? A clue is given in the image above. The result resides under the *StandardConfiguration1* and inside the model file `patch_antenna_scripted_model.bof`. The user can load more than one model into POSTFEKO. In addition every model can have more than one configuration. Lastly the user can specify more than one far field request. Therefore the model file, configuration and farfield must be specified. We must also remember that there can be more than one graph in POSTFEKO, so the graph to add the far field to must be specified as well. The *ResultTraceCollection* section in the integrated help provides a small example as well.

```
my_farfield_trace = my_graph.Traces:Add(app.Models[1].Configurations[1].FarFields[1])
```

Now we want to display the farfield in dB. Entering *dB* in the documentation will yield a number of search results. The key is to look on the result palette in POSTFEKO. The dB checkbox resides under a group named *Quantity*, see Figure I-1-61.

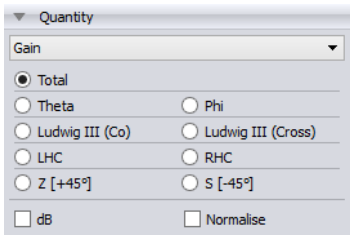


Figure I-1-61: The *Quantity* group in the result palette.

Search for *quantity* in the integrated help. The search will yield a number of results. Double-click the *Quantity (property)* entry and select the *FarFieldTrace* topic on the *Choose Topic* dialog.

On the *FarFieldTrace* object page, we see that one of the properties of *FarFieldTrace* is *Quantity*, see Figure I-1-62.

.Quantity	The far field trace quantity properties. (Read only FarFieldQuantity)

Figure I-1-62: A snippet of the *FarFieldTrace* object which shows the *Quantity* property.

When we click the *FarFieldQuantity* hyperlink, it will navigate us to the *FarFieldQuantity* object in the integrated help with the property, `.ValuesScaledToDB`, see Figure I-1-63.

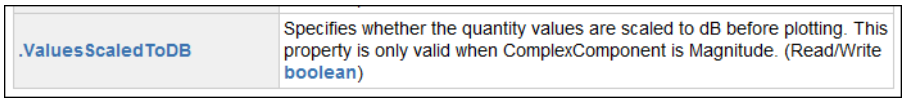


Figure I-1-63: A snippet of the *FarFieldQuantity* object which shows the *ValuesScaledToDB* property.

```
my_farfield_trace.Quantity.ValuesScaledToDB = true
```

Next we want to change the *Independent Axis* to phi. As a second step we want change the *Fixed axis* to 30 degrees.

Search for *axis* in the integrated help. The search will yield a number of results. Double-click the *IndependentAxis (property)* and select the *FarFieldTrace* topic on the *Choose Topic* dialog. This navigates us to the *FarFieldTrace (object)* with the property, `.IndependentAxis`, see Figure I-1-64.

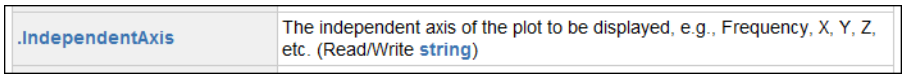


Figure I-1-64: A snippet of the *FarFieldQuantity* object which shows the *IndependentAxis* property.

```
my_farfield_trace.IndependentAxis = "phi"
```

Repeat the search for *axis* in the integrated help. The search will yield a number of results. The *Fixed axis* is changed by means of the method `:SetFixedAxisValue(axis, value, unit)`. Double-click the *SetFixedAxisValue (method)* and select the *FarFieldTrace* topic on the *Choose Topic* dialog. This navigates us to the *FarFieldTrace (object)* with the method, `SetFixedAxisValue`, see Figure I-1-65.

Figure I-1-65: A snippet of the *FarFieldQuantity* object which shows the *SetFixedAxisValue*.

```
my_farfield_trace:SetFixedAxisValue("theta",30,"deg")
```

Adding a current result to a 3D view

Next we want to add a current result to the 3D view. There can be multiple 3D views, as a result the 3D view will be in a collection. When loading a `*.fek` file, note that a 3D view is created from the first configuration by default. We therefore do not need to add a 3D view.

For information regarding 3D views, search for *view*. The results returned in response to our *view* query includes *ViewCollection (object)*. Double-click *ViewCollection (object)*.

From the *Usage locations* we see that the *ViewCollection* resides under the *Application* object (see Figure I-1-66).

Figure I-1-66: A snippet of the *ViewCollection* which shows the *Usage locations*.

Next we require a handle on the 3D view to which we will add the 3D current result. We use indexing to access the first 3D view (and for this example the only available 3D view).

```
this_3Dview = app.Views[1]
```

We now want to add the current result to the 3D view. As one can add multiple results from different requests (for example far field results and near field results) to the same 3D view, we are adding results to a collection.

We search for *result* in the integrated help. The results returned in response to our *result* query includes *Result3DPlotCollection (object)*. Double-click *Result3DPlotCollection (object)*.

On the same page we see the *Usage locations (collections)* and note that a result can be added to a plot, see Figure I-1-67.

Figure I-1-67: A snippet of the *Result3DPlotCollection* object which shows the *Usage locations*.

Similar to adding the farfield trace we need to specify the index of the following (since there can be more than one of each): model, configuration, current request and the view.

We also know there are two types of currents to display, namely surface currents and wire currents.

Another way to find the syntax for the surface currents is to look at the returned objects list for the *Results3DPlotCollection*. We click the *ResultPlot* hyperlink, see Figure I-1-68.

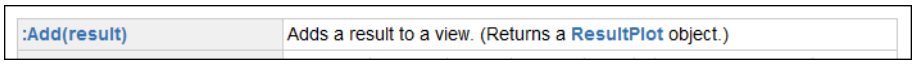


Figure I-1-68: A snippet of the *Result3DPlotCollection* object which shows the *Add* method.

This gives the following list of objects, see Figure I-1-69.

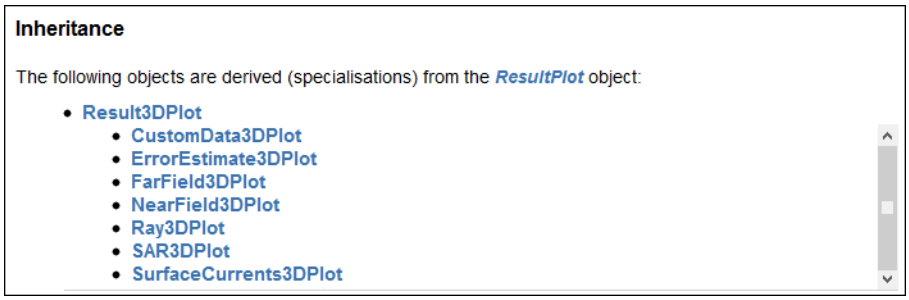


Figure I-1-69: A snippet of the *ResultPlot* object which shows the derived objects.

```
my_3Dview_currents_plot = this_3Dview.Plots:Add(app.Models[1].Configurations[1].SurfaceCurrents[1])
```

The above line of code gives a *SurfaceCurrents3Dplot* object (see Figure I-1-70), this is similar to a *Trace* for a graph.

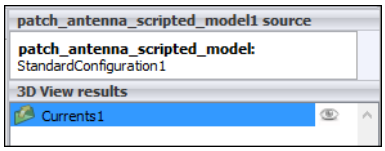


Figure I-1-70: *Currents1* in the result palette is equal to a *SurfaceCurrents3Dplot* object.

Similar to the far field trace, we change the currents to dB.

```
my_3Dview_currents_plot.Quantity.ValuesScaledToDB = true
```

Next we want to change the font size of the vertical and horizontal axis as well as the axes labels on the Cartesian graph. We search for *Font* in the integrated help and find *Font (property)*.

Double-click the *Font (property)* and select the *GraphAxisTitle* topic on the *Choose Topic* dialog. This navigates us to the *GraphAxisTitle (object)*, see Figure I-1-71.

Figure I-1-71: A snippet of the *GraphAxisTitle* object which shows the *Usage locations*.

From the *Usage locations* we note the prepending syntax is *.Title*. We click the first *.Title* hyperlink (see Figure I-1-71) and navigate to the *VerticalGraphAxis* object, see Figure I-1-72.

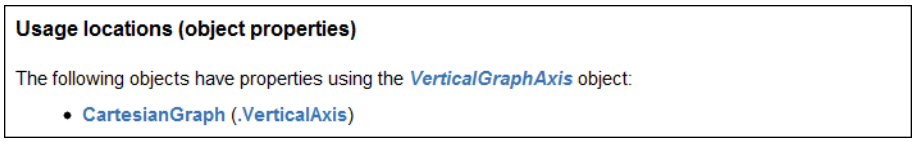


Figure I-1-72: A snippet of the *VerticalGraphAxis* object which shows the *Usage locations*.

We click *.VerticalAxis* at *Usage locations* and navigate to the section entitled *CartesianGraph*.

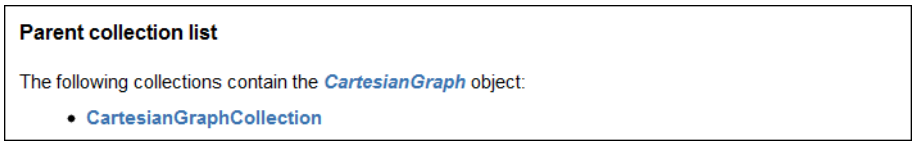


Figure I-1-73: A snippet of the *CartesianGraph* object.

There can be more than one graph, so they form part of the collection of *CartesianGraphCollection*. We click *CartesianGraphCollection* and see the graphs reside under the *Application*. The preceding section shows that the graph can be accessed with `Application.CartesianGraphs`.

We have now found all the prepending syntax as follows:

`my_graph.VerticalAxis.Title.Font`. Note we already have a handle on the *CartesianGraph* (`my_graph`).

Going back to the `GraphAxisTitle`, we see `.Font` under the *Property list*.

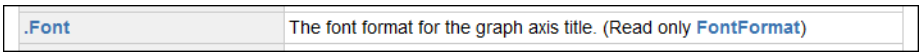


Figure I-1-74: A snippet of the *GraphAxisTitle* object which shows the *Font* property.

We click `.FontFormat` in the integrated help and see the following property (see Figure I-1-75):

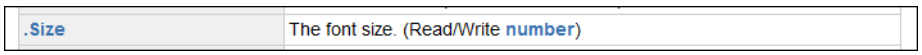


Figure I-1-75: A snippet of the *FontFormat* object which shows the *Size* property.

We will use the `.Size` property. Therefore we have all the syntax constructed in the correct sequence as follows:

```
my_graph.VerticalAxis.Title.Font.Size = 14
my_graph.HorizontalAxis.Title.Font.Size = 14
```

To set the labels of the axes, we use *GraphAxisLabels*, see Figure I-1-76.

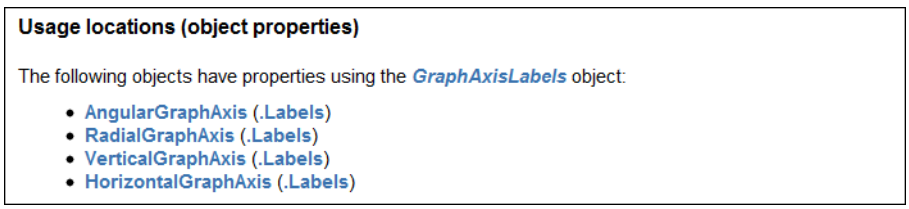


Figure I-1-76: A snippet of the *GraphAxisLabels* object which shows the *Usage locations*.

```
my_graph.HorizontalAxis.Labels.Font.Size = 14
my_graph.VerticalAxis.Labels.Font.Size = 14
```

The POSTFEKO application automation script is now complete. Run the script and observe the result in the POSTFEKO GUI.

The completed script

```
--[[
PATCH ANTENNA ON PLANAR MULTILAYER SUBSTRATE
====================================================================
This script post-process the calculated results of the patch antenna
placed on a planar multilayer substrate.
]]--
app = pf.GetApplication()
app:NewProject()
app:OpenFile("patch_antenna_scripted_model.fek")

-- Add a Cartesian graph
my_graph = app.CartesianGraphs:Add()

-- Add a far field result to a Cartesian graph
my_farfield_trace = my_graph.Traces:Add(app.Models[1].Configurations[1].FarFields[1])

-- Scale the quantity to dB
my_farfield_trace.Quantity.ValuesScaledToDB = true

-- Set the independent axis to Phi
my_farfield_trace.IndependentAxis = "phi"

-- Set the fixed axis to Theta = 30 deg
my_farfield_trace:SetFixedAxisValue("theta",30, "deg")

-- Add the surface currents to the 3D view
this_3Dview = app.Views[1]
my_3Dview_currents_plot = this_3Dview.Plots:Add(app.Models[1].Configurations[1].SurfaceCurrents[1])

-- Scale the quantity to dB
my_3Dview_currents_plot.Quantity.ValuesScaledToDB = true

-- Set the font size of the title
my_graph.VerticalAxis.Title.Font.Size = 14
my_graph.HorizontalAxis.Title.Font.Size = 14

-- Set the font size of the labels
my_graph.HorizontalAxis.Labels.Font.Size = 14
my_graph.VerticalAxis.Labels.Font.Size = 14
```

I-2 POSTFEKO application automation

Keywords: application automation, API, POSTFEKO, scripting, forms, reporting

Application automation is a tool that can increase productivity when dealing with predictable and repeatable POSTFEKO sessions. This example will illustrate how an automation script can be used to configure a session and export a report that highlights the antenna properties of the model.

It will be illustrated that a script can be used on various models with repeatable results. Figure I-2-1 shows equivalent pages for two different antenna types.

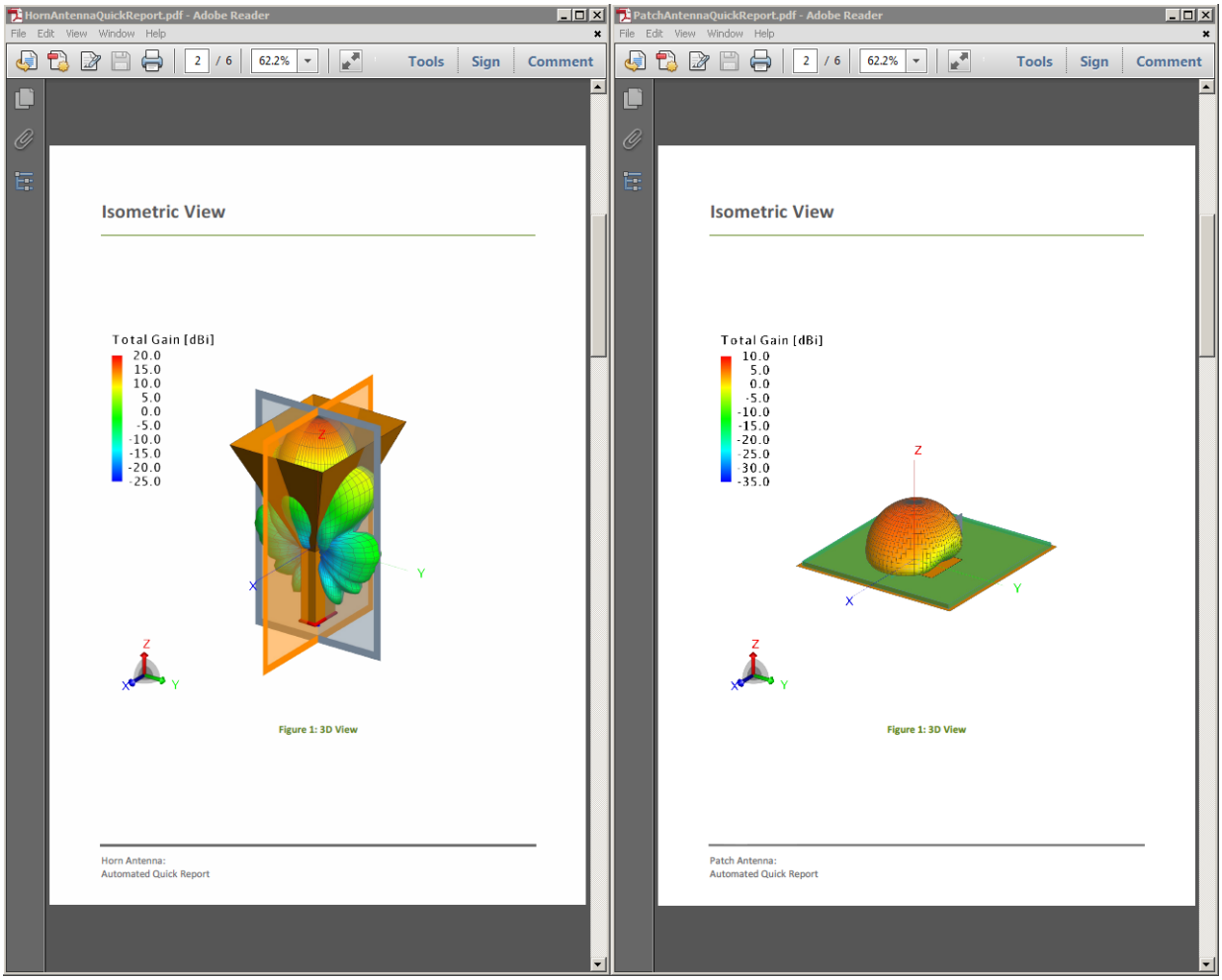


Figure I-2-1: Results of the automatic report generation.

Note that the base language for the automation interface is Lua. For a complete guide on the Lua language, we refer you to the official Lua Reference Manual (www.lua.org/manual/5.1/) and Programming introduction (www.lua.org/pil/). Another useful resource is the community maintained Wiki that is available on the internet at lua-users.org/wiki/.

For documentation and examples on how to use the automation API, see the integrated help by pressing on the help button in the script editor.

I-2.1 The models

Two models are included with this example that will provide the content of the POSTFEKO sessions. Assumptions are made about both models. For instance:

- It is assumed that the first configuration will contain 3D far field data.
- It is assumed that the far field can be wrapped in the θ direction.
- It is assumed that there is a ϕ angle calculated at ϕ=0° and at ϕ=90°. This also implies that the main direction of radiation is in the positive Z axis.

Apart from these assumptions, any antenna geometry can be used as an input. The script can be adapted to iterate over multiple models or to display different properties of interest.

I-2.2 Exercises

As an exercise, modify the script from section I-2.3 to:

- change the rendering of the mesh to be 60% opaque.
- Automatically generate reports for all of the models in the `modelName` list.
- Save the sessions under unique names.

Another exercise that could be useful to try is to reproduce figure I-2-2. It is recommended to write a new script and make use of the documentation for this exercise. Export the graph to a `*.pdf` file and save the session.

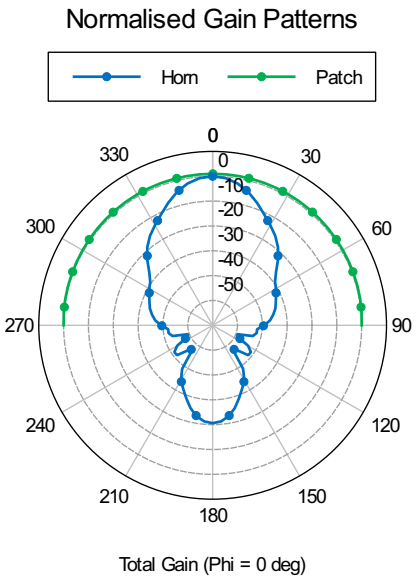


Figure I-2-2: Gain patterns for both antenna models.

I-2.3 The script

generate_antenna_report.lua

```
--[[
AUTOMATIC QUICK REPORT GENERATION FOR ANTENNA PATTERN ANALYSIS
=============================================================
This script loads the specified model.  It then creates various
views and graphs that display the far field pattern.  These
views are then exported to a PDF report
]]--

modelName = {}
modelName[1] = "Horn"
modelName[2] = "Patch"

-- Get user input through Forms
form = pf.Form.New("Select model")
comboBox = pf.FormComboBox.New("Model name", modelName)
form:Add(comboBox)
form:Run()
index = comboBox.Index

app = pf.GetApplication()
app:NewProject()
app:OpenFile(modelName[index]..".fek")

selectedModel   = app.Models[modelName[index]]
selectedConfig1 = selectedModel.Configurations[1]

ffData      = selectedConfig1.FarFields[1] -- This is a handle on the far field data itself

view3D = app.Views[1]
ffPlot = view3D.Plots:Add(ffData)
ffPlot.Label = "ff3D"
ffPlot.Quantity.ValuesScaledToDB = true

view3D_top = view3D:Duplicate()
view3D_top:SetViewDirection(pf.Enums.ViewDirectionEnum.Top)

view3D_right = view3D:Duplicate()
view3D_right:SetViewDirection(pf.Enums.ViewDirectionEnum.Right)

view3D_front = view3D:Duplicate()
view3D_front:SetViewDirection(pf.Enums.ViewDirectionEnum.Front)

polarGraph = app.PolarGraphs:Add()
ffTracePhi_00 = polarGraph.Traces:Add(ffData)
ffTracePhi_00.IndependentAxis = "Theta (wrapped)"
ffTracePhi_00.Quantity.ValuesScaledToDB = true
ffTracePhi_90 = polarGraph.Traces:Add(ffData)
ffTracePhi_90.IndependentAxis = "Theta (wrapped)"
ffTracePhi_90.Quantity.ValuesScaledToDB = true
ffTracePhi_90:SetFixedAxisValue(ffTracePhi_90.FixedAxes[2],90,"deg")
polarGraph:ZoomToExtents()
polarGraph.Title.Text = "Gain"
polarGraph.Legend.Position = pf.Enums.GraphLegendPositionEnum.OverlayTopRight
polarGraph.BackColour = pf.Enums.ColourEnum.LightGrey
polarGraph:Restore()

quickReport = app:CreateQuickReport(modelName[index].."AntennaQuickReport",
                                    pf.Enums.ReportDocumentTypeEnum.PDF)
quickReport.DocumentHeading = modelName[index]..[[ Antenna:
Automated Quick Report]]
quickReport:SetPageTitle(view3D.WindowTitle, "Isometric View")
quickReport:SetPageTitle(view3D_top.WindowTitle, "Top View")
quickReport:SetPageTitle(view3D_right.WindowTitle, "Right View")
quickReport:SetPageTitle(view3D_front.WindowTitle, "Front View")
quickReport:SetPageTitle(polarGraph.WindowTitle, "Theta Cuts")
quickReport:Generate()
```

I-3 Matching circuits generation with Optenni Lab

Keywords: multiband, S-parameter, matching circuit

This example demonstrates how Optenni Lab can be used in conjunction with FEKO for designing antenna matching circuits. Two different antenna geometries will be simulated and matched. The first is a single PIFA antenna design for GSM 1800 band (1710-1880 MHz) and the second is a dual PIFA antenna configuration designed to operate at the WCDMA I-III (1710-2170 MHz) and WLAN bands (2400-2483 MHz). The geometries are shown in Figure I-3-1.

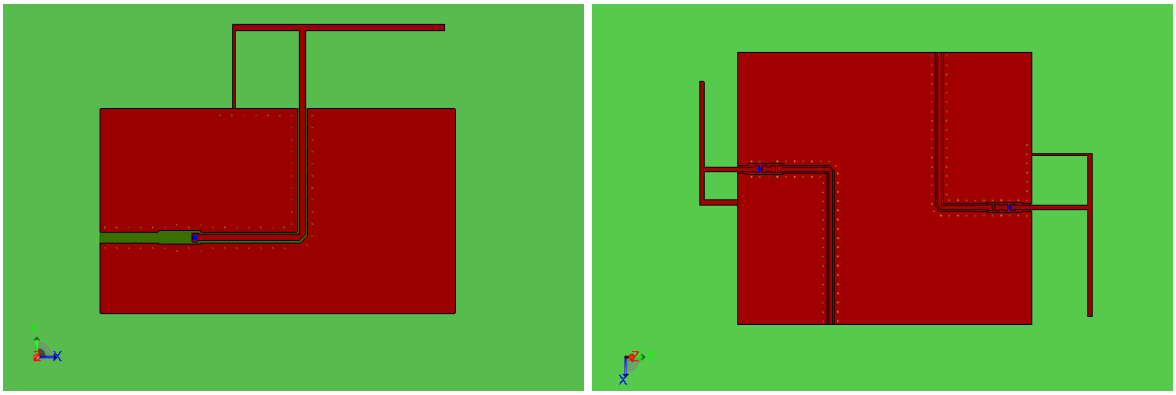


Figure I-3-1: A 3D view of the single (left) and dual (right) PIFA antenna geometries.

I-3.1 Single antenna model

The geometry for this antenna is created from a `*.dxf` file, although a previously prepared `*.cfx` model is also provided for convenience. The steps for setting up the model are as follows:

- Set the model unit to millimetres.
- Import `single_antenna.dxf`, the geometry of the antenna is added as a number of polylines.

There are several different ways to convert the polylines into the metal surfaces of the antenna. For example, the *Fill hole* tool can be applied to create a face that fills a closed polyline. Alternatively, a *Polygon* can be created by snapping to the vertices of the polylines.

- Create the surfaces for the antenna, feedline and other metal surfaces on the top layer.
- Create the ground plane *Rectangle* at Z = *-0.508*.
- Add a *Line* between the center of the feedpad and the ground plane. Add a *Wire port* to the wire segment.
- The vias are also added using *Line* geometry. Multiple vias can be added quickly using *Copy special and translate...*.

- Once the geometry has been created, the imported polylines can be deleted.
- Union the remaining geometry.
- Open the *Media library* and add *Copper* to the model. Select all the faces of the unioned geometry and change the *Face medium* properties to *Copper* with *Thickness = 35e-3*.

For simplicity, an infinite substrate will be used when simulating the antenna:

- Add a new dielectric medium called *Substrate* with $\epsilon_r = 3.55$, $\tan\delta = 0.003$.
- Add a *Planar multilayer substrate*, set *Layer 1 Ground plane* to *None*, *Thickness = 0.508* and *Medium* to *Substrate*.

Solution settings and requests

In order to match the antenna, the S-parameters of the unmatched antenna must first be calculated. The following solution setting changes and requests are required.

- Create an S-parameter request where all ports are included and active with a 50 Ω reference impedance. This will replace the standard configuration that was created by default. Check the *Export S-parameters to Touchstone file* checkbox.
- For the configuration frequency choose *Continuous (interpolated) range* and set the start and end frequencies to 1 GHz and 3 GHz respectively. On the *Export* tab check the checkbox and set 101 for the number of frequency samples used in the exported Touchstone file.

Meshing information

The mesh settings need to be refined slightly. Select the metallic face of the antenna arms and set the *Local mesh = 1.5*. Mesh the geometry using the *standard* mesh size and wire segment radius equal to 0.15.

I-3.2 Dual antenna model

Create a new CADFEKO model and follow the same steps as the previous model, importing `dual_antenna.dxf`, using the same approach and parameters as before. A *Wire port* is placed at the feed pad of each of the two antennas. When setting up the S-parameter configuration ensure that both of the ports are active in the S-parameter request.

CEM validate

After both the models have been meshed, run *CEM validate*. Take note of any warnings and errors. If necessary, correct any errors before running the FEKO solution kernel.

I-3.3 Unmatched S-parameter results

The S-parameter results for the unmatched antennas are shown in Figure I-3-2. The single antenna resonance is close to the requirement but the bandwidth needs to be increased to meet GSM 1800 band (1710-1880 MHz). For the dual antenna the bandwidth also needs to be increased to meet the WCDMA I-III band (1710-2170 MHz). In addition, the coupling between the two antennas should be reduced below -15 dB. In both cases, Optenni Lab will be used to design the matching circuit.

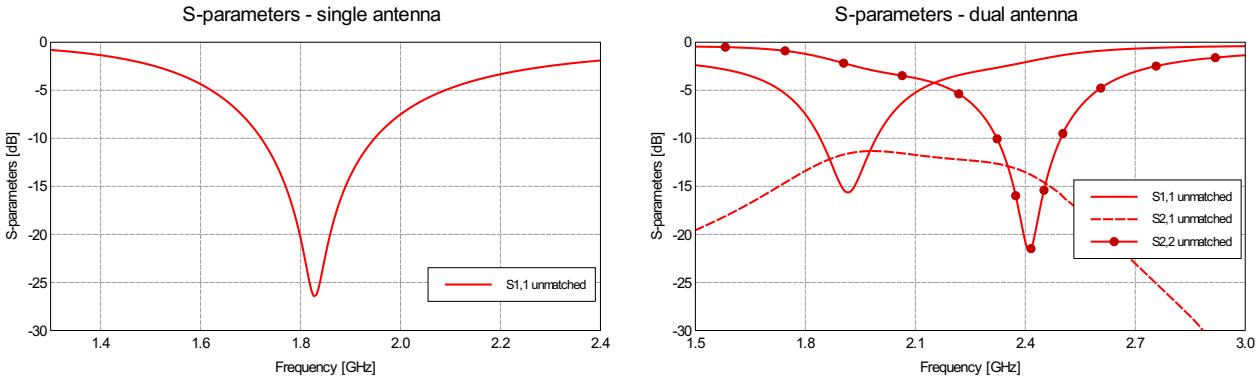


Figure I-3-2: S-parameters for the single and dual unmatched antennas.

I-3.4 Matching the antennas with Optenni Lab

This section describes the functionality of the FEKO-Optenni Lab linking to automate matching circuit generation. Note that these examples are also available as tutorials in the Optenni Lab installation. More detail on the different matching options in Optenni Lab is provided there including the use of discrete components from the component library. The single antenna model corresponds to Optenni Lab "Basic Tutorial A" and the dual antenna model corresponds to Optenni Lab "Multiantenna Tutorial B".

The Optenni Lab: Port matching link

From the *Application macro* menu choose *Optenni Lab: Port matching*. This will launch the dialog shown in Figure I-3-3

If the first run mode is chosen, a new `*.cfx` file will be created and the unmatched S-parameter simulation will be run. The second run mode (*Use existing Touchstone data*) is used in this example because the unmatched simulations have already been run and the Touchstone files were exported. Press *Next* and the Touchstone results will be sent to Optenni Lab to design the matching circuit.

The steps for generating the matching circuit are described in detail in the Optenni Lab tutorials. The matching circuits configurations generated in Optenni Lab, the minimum efficiency and isolation are shown in Figure I-3-4.

Once the circuit has been generated, right-click in the circuit view and choose *Transfer circuit to*

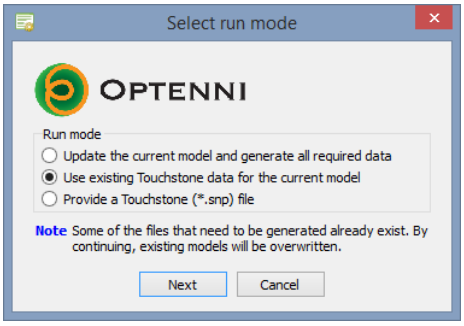


Figure I-3-3: Options for running the FEKO-Optenni Lab link.

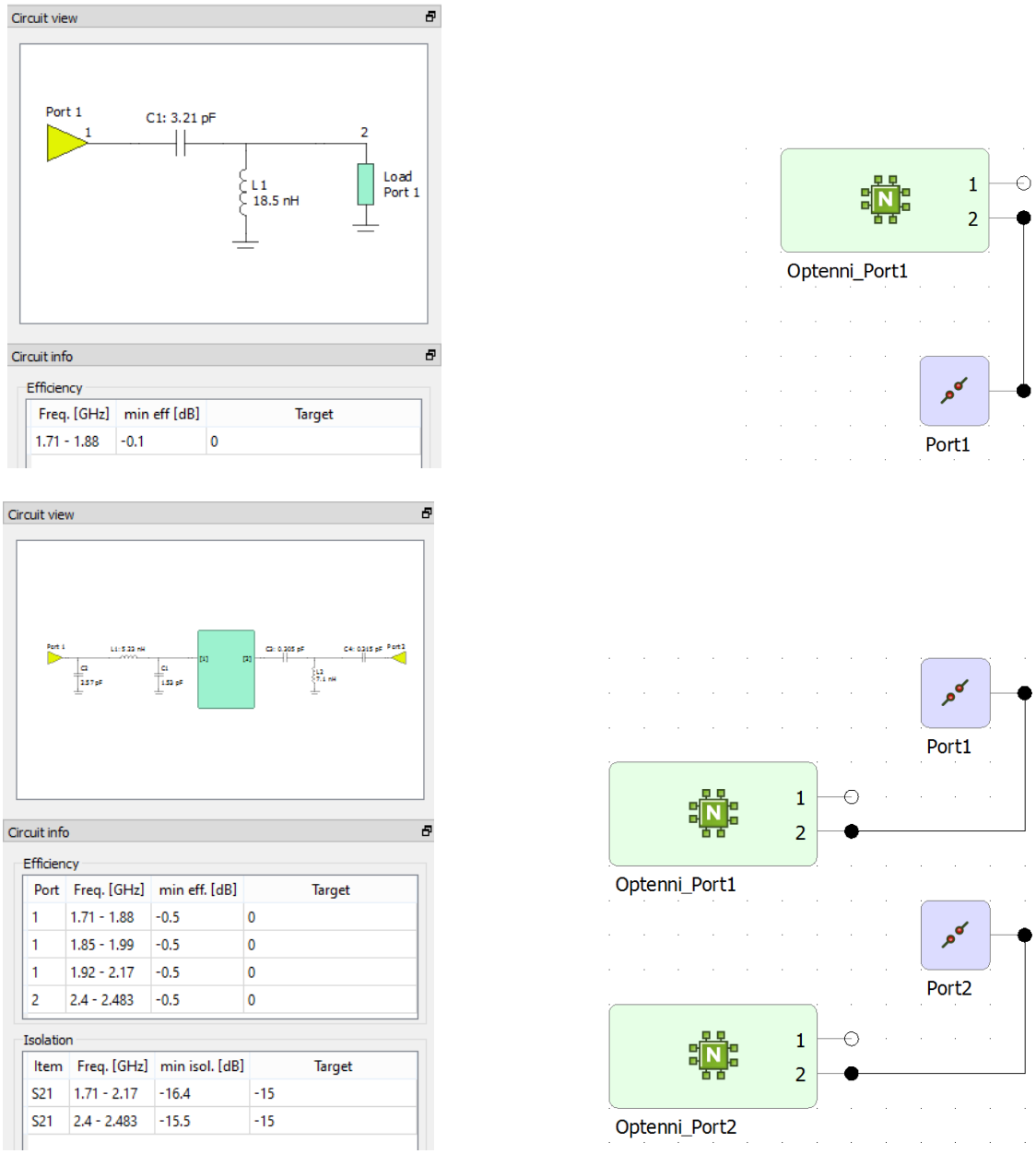


Figure I-3-4: Matching circuits for the two antennas using generic components and the corresponding schematics in FEKO.

FEKO and quit. A new `*.cfx` file is automatically created with the matching circuit added as a general network (Figure I-3-4). Run the simulation to obtain the matched antenna S-parameters. Figure I-3-5 shows the unmatched and matched S-parameters for the two antennas. In both cases, the bandwidth has been improved and in the case of the dual antenna the coupling has been reduced below -15 dB in both bands.

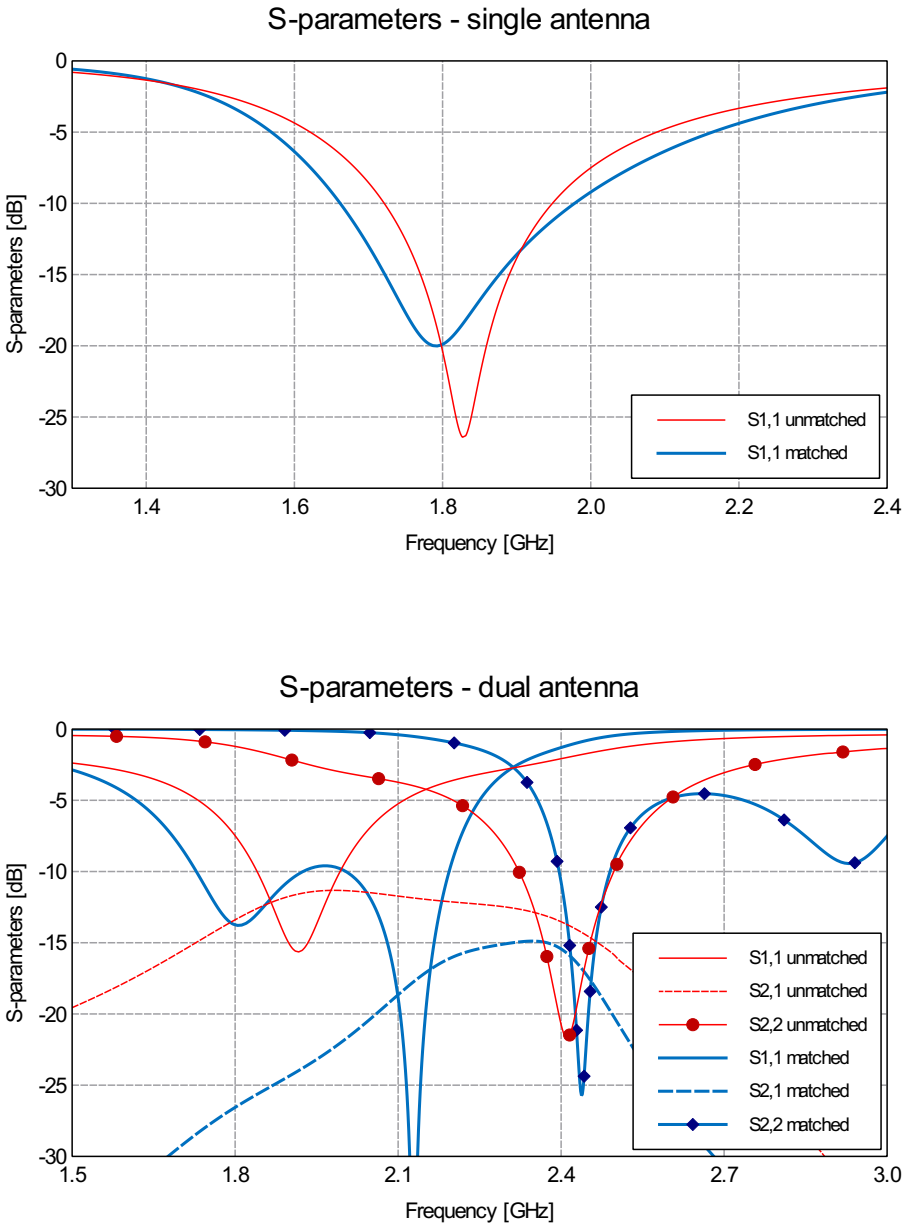


Figure I-3-5: S-parameters for the matched single and dual antennas.

I-4 Using HyperStudy with FEKO to optimise a bandpass filter

Keywords: application automation, bandpass filter, HyperStudy, optimisation

Microstrip filters are frequently encountered in communication networks. A bandpass filter is designed by using HyperStudy as an optimisation engine with FEKO as the computational solver. The design will aim to obtain a value for spacing S1 – S3 so as to maximise the coupling between input and output ports within the operating frequency range. The reflection coefficients over a frequency range will then be analysed. Figure I-4-1 shows a top view of the model.

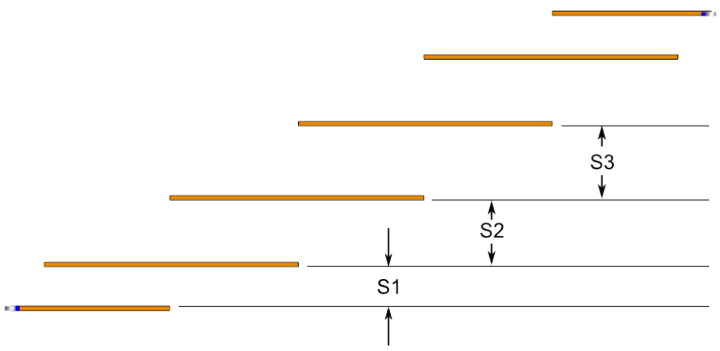


Figure I-4-1: Top view of bandpass filter.

The parametric model is provided for convenience. For the sake of simplicity, the cutoff frequencies and out-of-band performance will not be considered for the design.

I-4.1 Workflow

A typical workflow for HyperStudy is to:

- Configure a study setup. This includes defining which models need to be included in a study, which solvers need to be used, which design variables may be altered and which responses to analyse.
- Once configured, HyperStudy will know how to modify a model and how to interpret the results. This knowledge can be applied to study a problem using one of four approaches:

    1. The “Design of Experiments” (DOE) approach can be defined as a test or a series of tests in which purposeful changes are made to the input variables of a process or system so that the reasons for changes in the output response can be identified and observed. Responses can be extracted, but will not affect how the model permutations are generated.
    2. A “Fit” approach is used to approximate the response of a model by creating a mathematical equivalent of the model. This approach uses previous approaches as inputs that are used to predict how a model would behave for a change in design variables. This is useful in situations where resources are scarce.

3. "Optimisation" approaches are used to generate a model that behaves in the desired manner. The responses that are extracted from simulations are used to determine what the next model permutation should be.
4. "Stochastic" approaches are used to analyse the effect of tolerances in the design variables (e.g. from material properties, manufacturing tolerances, etc.). These approaches can help identify the probability of responses adhering to defined specification.

HyperStudy has FEKO registered as a supported solver, meaning that the majority of the workflow is integrated by default. The following phases form part of the configuration process.

Define models With this phase, the solver input file is provided. HyperStudy will create this file for each run from the initial template file, using the current value of the design variables. Once the model has been added, the *Import Variables* button will analyse the model and identify design variables that may be modified. These variables can be used in subsequent approaches to generate different model permutations.

Define design variables A table of variables have been imported in the model definition phase. These variables may now be edited, deactivated and the value ranges can be set.

Evaluate This phase is responsible for writing files, executing the solver and extracting values from the results.

Write During the write phase, HyperStudy will create subdirectories with a copy of the model and all of the files that the model depends on to be executed. The model variables will be updated before executing the solver.

Execute During the execution phase, the solver will be run and output data will be generated. Note that this includes any post-processing of the raw simulation data (e.g. by running a script on output data).

Extract The extraction phase is responsible for identifying output response values and pulling them back into HyperStudy .

Figure I-4-2 depicts these generalised steps and will be discussed by way of the bandpass filter. Note that in general, ASCII files can be processed during the extraction phase.

I-4.2 Bandpass Filter

Configuring the HyperStudy setup

- **Define models**
  - Create a new study with a convenient name and location.
  - Add a new FEKO model.
  - Select `bandpassfilter.cfx` as the model resource. Note that the provided file `bandpassfilter.cfx_extract.lua` is expected to be saved in the same location as the model file.

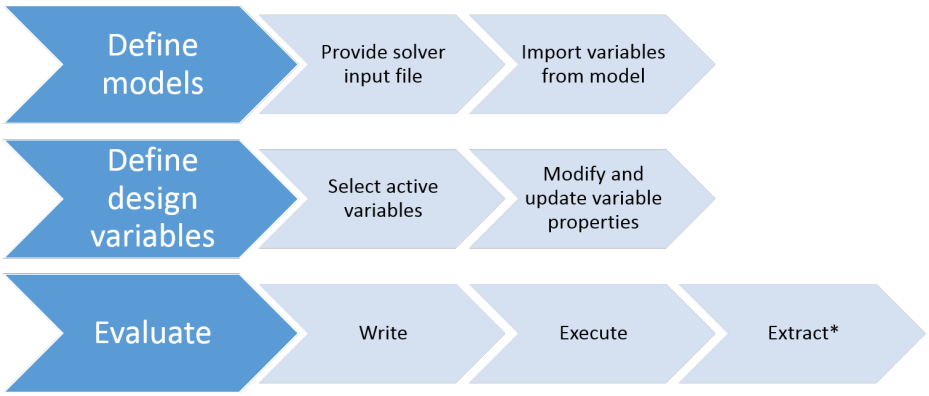


Figure I-4-2: HyperStudy and FEKO integration workflow.

  - Ensure that the FEKO solver execution script is selected. Note that additional solver input arguments can be provided in addition to the filename to affect how the solver is utilised. For instance, a parallel run can be launched by modifying the input arguments from `$filebasename` to `$filebasename -np 4 --priority 1`
  - Import CADFEKO model variables by pressing the *Import Variables* button.
  - Click *Next*.

- **Define design variables**
  - Choose which variables to include in the study. Only `S1` – `S3` should be activated.
  - The default value ranges are used.
  - Click *Next*.

- **Specifications**
  - Select a nominal run for this phase of the configuration. It will run the model with the original model values so that output data can be processed.
  - Click on *Apply* and *Next*.

- **Evaluate**
  - This phase will do the bulk of the processing.
  - Click on *Evaluate Tasks*.
  - Note that additional feedback can bee seen by activating the message window. Right-clicking on the message window and activating the console will show all FEKO solver output.
  - Click on *Next*.

- **Define responses**
  - During the *Evaluate* phase, the `bandpassfilter.cfx_extract.lua` file was copied to the run directory and executed after the FEKO solver has been run. This generated an output file that HyperStudy can easily process.
  - Click on *File Assistant*.

  - Choose the `./m_1/hst_output.hstp` file in the browser under the default run directory. Keep the *Altair® HyperWorks® (HstReaderPdd)* option selected and click *Next*.
  - Keep *Single serial or time series* selected and click *Next*.
  - Keep `s11avg` selected and click *Next*.
  - Click *Finish* to close the dialog.
  - Change the label of the response to `s11avg`.
  - Click *Evaluate Expressions* to extract the value from the output file.

- HyperStudy has now been configured to understand which model to use, which variables are available for modification and how to process the output.

bandpassfilter.cfx_extract.lua

```
--
-- Run into POSTFEKO to fetch results and push the scalar values to a HyperStudy .hstp file
--

HstUtl = require "hst.hstutl"

local app = pf.GetApplication()
local config = app.Models[1].Configurations[1]

-- Get first s-parameter result
local sparam = pf.SParameter.GetDataSet( pf.SParameter.GetNames()[1] )
local s11 = sparam[1][1].SParameter

-- Calculate the average reflection coefficient across the frequency band
local s11avg = 0
local num_frequencies = sparam.Axes[pf.Enums.DataSetAxisEnum.Frequency].Count
for findex = 1,num_frequencies do

    s11avg = s11avg + sparam[findex][1].SParameter:Abs()

end
s11avg = s11avg/num_frequencies

-- Create the output file
file = HstUtl.NewOutputFile( )
HstUtl.StoreScalarValue( file, "s11avg", s11avg )
HstUtl.WriteFile( file )
-- End of file
```

Performing an optimisation

The goal is to obtain an average reflection coefficient close to zero. Energy that doesn't reflect back to the input port is assumed to be transmitted to the output port.

- Add an optimisation approach by right-clicking on the defined study in the *Explorer* tab and selecting *Add Approach...*.
  - Keep the design variables `S1` – `S3` selected with their default settings and click on *Next*.
  - Add an objective under *Select responses*. The objective should be configured such that `s11avg` is minimised.

– Click on *Apply* and *Next*.

– Choose the *Adaptive Response Surface Method* (or *ARSM*) optimiser.

– Evaluate the tasks. Each of the input variables will be randomly altered and its effect on the response analysed. Figure I-4-3 shows typical data for the evolutions.

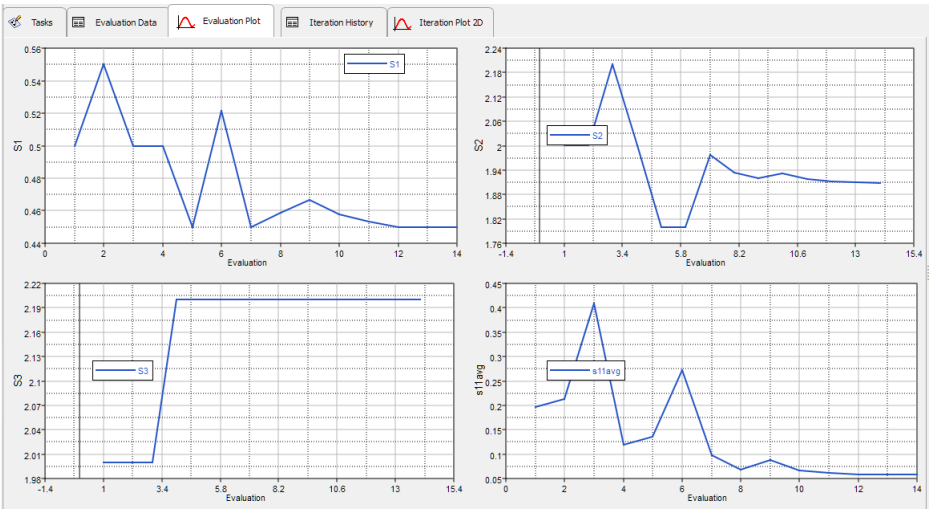


Figure I-4-3: Typical evolution data for an ARSM optimisation.

– The optimum value can be found in the *Iteration History* as the row highlighted in green. For this example, an optimum value was found with:

```
S1 = 0.4500000
S2 = 1.9087706
S3 = 2.2000000
```

I-4.3 FEKO modelling and performance

Generating a model with the optimum values in FEKO

The optimisation relied on five samples within the frequency range of interest, namely 3.96 – 4.00 GHz. To get a better sense of the frequency response for the optimum model, it is run in FEKO.

- Save a copy of the original CADFEKO model.
- Update the following variables:
  - `S1 = 0.45`
  - `S2 = 1.91`
  - `S3 = 2.20`
- Include the S-parameter configuration called `SParamBand` that is defined over the frequency range.
- Mesh the model and run the FEKO solver.

Results

Figure I-4-4 depicts the magnitudes of S_{11} and S_{21}. Within the design frequency range, a reflection coefficient of less than -20 dB is predicted.

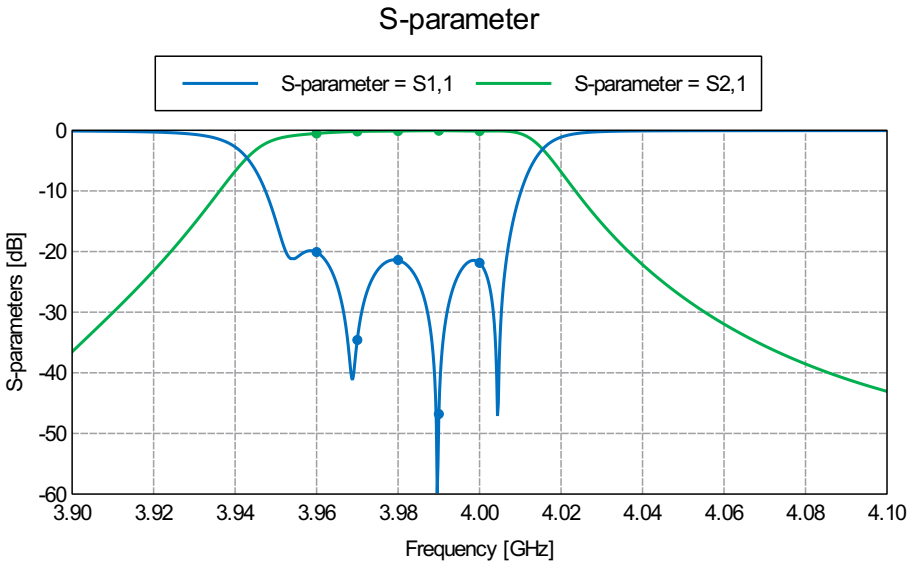


Figure I-4-4: S-parameters for the bandpass filter between 3.9 – 4.1 GHz.

Chapter J

Index

Index

A

ADAPTFEKO H-1-1
adaptive sampling H-1-1
antenna placement B-1-1
aperture A-10-1
aperture coupled A-10-1
API I-1-1, I-2-1
application
  antenna analysis I-3-1, A-1-1, A-4-1, A-5-1, A-7-1, A-8-1, A-9-1, A-10-1, A-11-1, A-12-1, D-4-1, E-3-1, H-1-1, H-3-1
  antenna optimisation A-6-1
  antenna placement B-1-1
  cable analysis D-2-1
  EMC D-1-1, D-2-1, D-3-1
  exposure analysis F-1-1
  lens antenna A-13-1
  matching circuit I-3-1
  microstrip coupler analysis E-5-1
  microstrip filter analysis E-1-1
  time domain analysis G-1-1
  waveguide analysis E-2-1, H-4-1
application automation
  CADFEKO I-1-1
  Forms I-2-1
  POSTFEKO I-1-1, I-2-1, I-4-1
appplication
  multiband I-3-1
array A-7-1

B

bandpass filter I-4-1
birdcage F-2-1

C

cable analysis D-2-1
cable modelling D-2-1
CFIE B-1-1
characteristic modes A-15-1
continuous frequency H-1-1
coupling B-1-1, B-2-1, B-3-1, D-2-1
current A-4-1

D
dielectric resonator antenna (DRA) A-12-1
dielectric solid A-2-1, C-2-1
dielectric substrate A-8-1, A-9-1, A-10-1
dipole A-1-1, A-7-1
  forked H-1-1
  ideal lumped element matching E-3-1
  near a cube A-2-1
  near a dielectric sphere F-1-1
  near a large metal plate A-3-1
DXF import I-3-1

E
edge port A-9-1, A-10-1
electrically large model B-1-1, B-2-1, B-3-1, H-2-1
EMC D-1-1, D-2-1, D-3-1
equivalent source A-13-1, B-3-1
excitation ..... *see* source
exposure analysis F-1-1

F
far field A-1-1, A-3-1, A-4-1, A-7-1, A-11-1, A-12-1, A-13-1, A-16-1, A-17-1
far field data file B-2-1, B-3-1
FDTD A-3-1, A-8-1, E-5-1
feed ..... *see* source
feed network E-4-1
FEM current source A-12-1
FEM modal source A-12-1, A-12-3
FEM/MoM hybrid F-1-1
finite array A-17-1
finite conductivity D-1-1
finite ground plane A-4-1, A-10-1

G
geometrical optics A-13-1

H
half-wavelength dipole A-1-1
helix antenna B-2-1
HOBF A-3-1
horn antenna A-11-1, B-3-1
HyperStudy I-4-1

I
ideal matching E-3-1
ideal receiving antenna B-2-1, B-3-1
infinite ground A-5-1, A-10-1, A-12-1
infinite planar Green's function A-5-1, A-9-1, A-10-1, A-12-1
input impedance A-1-1, A-12-1, E-1-1

L
LE-PO A-3-1
lens antenna A-13-1
lossy metal A-2-1

M
macro recording I-1-22
magnetic field probe D-3-1
Magnetic Resonance Imaging F-2-1
matching circuit I-3-1
materials
    dielectric solid A-2-1, A-13-1, C-2-1
    lossy metal A-2-1, D-1-1
    PEC A-2-1
method of moments I-3-1, A-1-1, A-2-1, D-1-1, D-4-1, F-2-1
microstrip A-8-1, A-9-1, A-10-1
    feed line A-9-1, A-10-1
microstrip coupler E-5-1
microstrip feed A-8-1
microstrip filter E-1-1
MIMO A-15-1
MLFMM B-1-1, H-2-1
MoM A-3-1
monopole antenna A-4-1
MRI F-2-1
multiband I-3-1

N
near fields D-1-1
non-radiating network A-7-1, E-3-1, E-4-1

O
optimisation I-4-1, A-6-1, H-4-1

P
patch E-4-1
patch antenna A-8-1, A-9-1, A-10-1
PEC A-2-1
PIFA antenna I-3-1
pin feed A-8-1, A-11-1
planar Green's function aperture A-10-1
planar multilayer substrate (Green's function) I-3-1, A-8-1, A-10-1
plane wave C-1-1, C-2-1, D-1-1
PO A-3-1
polygon plate C-1-1
proximity coupling A-9-1, A-10-1
pulse shape G-1-1

R
radiation pattern A-1-1, A-4-1, A-11-1, A-13-1
ray tracing A-13-1
RCS (radar cross section) C-2-1
real ground A-5-1
reporting I-2-1
resource requirements H-2-1
results
    continuous frequency H-1-1
    current A-4-1, D-3-1

far field A-1-1, A-3-1, A-4-1, A-11-1, A-12-1
gain A-5-1
input impedance A-1-1, A-12-1, A-14-1, E-1-1
RADHAZ D-4-1
radiation pattern A-1-1, A-4-1, A-6-1
RCS (Radar cross section)
bistatic C-2-1
monostatic C-2-1
RCS (radar cross section)
bistatic C-1-1
monostatic H-2-1
reflection coefficient A-9-1, A-10-1, E-1-1
S-parameters I-3-1, B-1-1, E-1-1, E-2-1, E-3-1, E-4-1, E-5-1, H-4-1
RL-GO A-3-1

S

S-parameters I-3-1, E-3-1, E-4-1
scripting I-1-1, I-2-1
shielded cable D-2-1
shielding D-1-1, D-3-1
skin effect D-1-1
slot A-10-1
Smith chart A-9-1, A-10-1
solution method
DGFM A-17-1
FDTD A-3-1, A-8-1, E-5-1
FEM/MoM F-1-1
geometrical optics A-13-1
infinite planar Green's function A-5-1, A-9-1, A-10-1, A-12-1
method of moments I-3-1, A-1-1, A-2-1, D-1-1, F-2-1
MLFMM B-1-1, H-2-1
MoM A-3-1
MoM/PO A-3-1
MoM/RL-GO A-3-1
MoM/UTD A-3-1
NGF H-4-1
periodic boundary conditions A-16-1, C-3-1, C-4-1
planar multilayer substrate (Green's function) A-8-1
surface equivalence principle A-2-1, A-8-1, A-10-1, C-2-1, E-1-1
symmetry A-9-1, A-10-1
thin dielectric sheet approximation C-1-1
UTD B-2-1
source
edge E-1-1, E-5-1
far field A-13-1, B-3-1
FEM current A-12-1, E-1-1
FEM modal A-12-1
impressed field distribution A-11-1
microstrip feed A-9-1, A-10-1, E-1-1
pin feed A-11-1
plane wave C-1-1, C-2-1, C-3-1, C-4-1, D-1-1, D-3-1, H-2-1
voltage A-16-1, A-17-1

voltage on an edge A-9-1, A-10-1
waveguide feed A-11-1, E-2-1
sphere creation C-2-1
surface equivalence principle A-2-1, C-2-1

T
thin dielectric sheet approximation C-1-1
time domain G-1-1
Touchstone E-3-1, E-4-1
transmission line A-7-1
trihedral reflector H-2-1

U
UTD A-3-1, B-2-1

W
waveguide A-11-1, E-2-1, H-4-1
waveguide modes E-2-1
windscreen A-14-1
wires A-1-1

Y
Y-parameters E-3-1
Yagi-Uda antenna A-5-1, A-6-1, B-2-1

Z
Z-parameters E-3-1