XSim Examples

Release 1.0.0rc3

Tech-X Corporation

Apr 01, 2024

CONTENTS

CHAPTER

ONE

OVERVIEW

These are examples for illustrating the capabilities of XSim.

CHAPTER

TWO

XSIM FOR ELECTROMAGNETICS EXAMPLES

These examples demonstrate the basic solvers for simple, grid-aligned boundary conditions.

These examples can be run with any license.

2.1 Antennas

2.1.1 2.4 GHz Yagi Uda Antenna (YagiUda2p4.sdf)

Keywords:

`yagiUdaArrayWireModel, yagiT, far field, radiation`

Problem description

A Yagi-Uda array is a directional antenna consisting of several parallel dipole elements. Only one of these dipole elements is driven, the other elements being parasitic . Directionality is achieved by requiring that there be one longer element adjacent to the source element, which is referred to as the reflector. The rest of the elements being adjacent to the source but opposite to the reflector, and shorter than the source element, are referred to as directors. Yagi antennas are ubiquitous, and as such optimal parameters for dipole lengths and separations have been established. We go with values one would typically find in any text covering the matter. This example illustrates how to obtain the far field radiation pattern of a Yagi-Uda array.

Opening the Simulation

The Yagi-Uda example is accessed from within XSimComposer by the following actions:

- Select the *New* → *From Example…* menu item in the *File* menu.
- In the resulting *Examples* window expand the *XSim for Electromagnetics* option.
- Expand the *Antennas* option.
- Select *2.4 GHz Yagi Uda Antenna* and press the *Choose* button.
- In the resulting dialog, create a New Folder if desired, and press the *Save* button to create a copy of this example.

All of the properties and values that create the simulation are now available in the Setup Window as shown in Fig. 2.1. You can expand the tree elements and navigate through the various properties, making any changes you desire. The right pane shows a 3D view of the geometry, if any, as well as the grid, if actively shown. To show or hide the grid, expand the Grids element and select or deselect the box next to `Grid`.

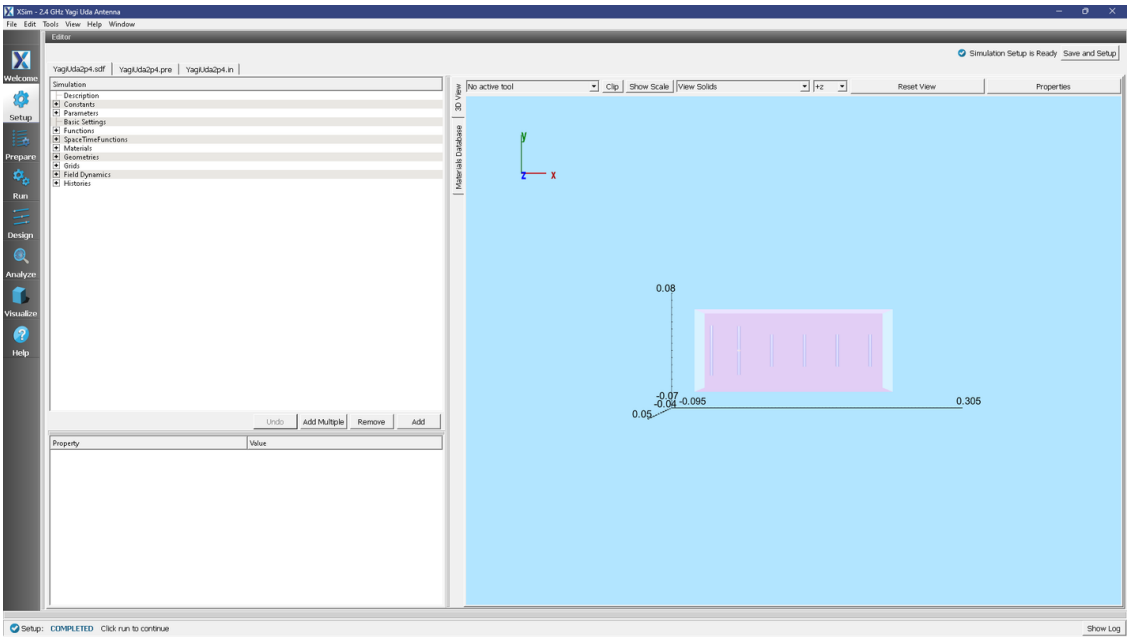

Fig. 2.1: Setup Window for the Yagi-Uda example.

Simulation Properties

This file allows the modification of the antenna operating frequency, antenna dimensions, and simulation domain size.

By adjusting the dimensions any sized Yagi-Uda array can be simulated.

Note: To obtain good far field resolution generally four or more antenna elements is desirable (One source, one reflector, two or more directors).

Running the Simulation

After performing the above actions, continue as follows:

- Proceed to the run window by pressing the Run button in the left column of buttons.
- Check that you are using these run parameters:
  - *Time Step*: 9.532874347655025e-13
  - *Number of Steps*: 6000
  - *Dump Periodicity*: 200
  - *Dump at Time Zero*: Checked
- Click on the *Run* button in the upper left corner of the right pane.

You will see the output of the run in the right pane. The run has completed when you see the output, “Engine completed successfully”. This result is shown in Fig. 2.2.

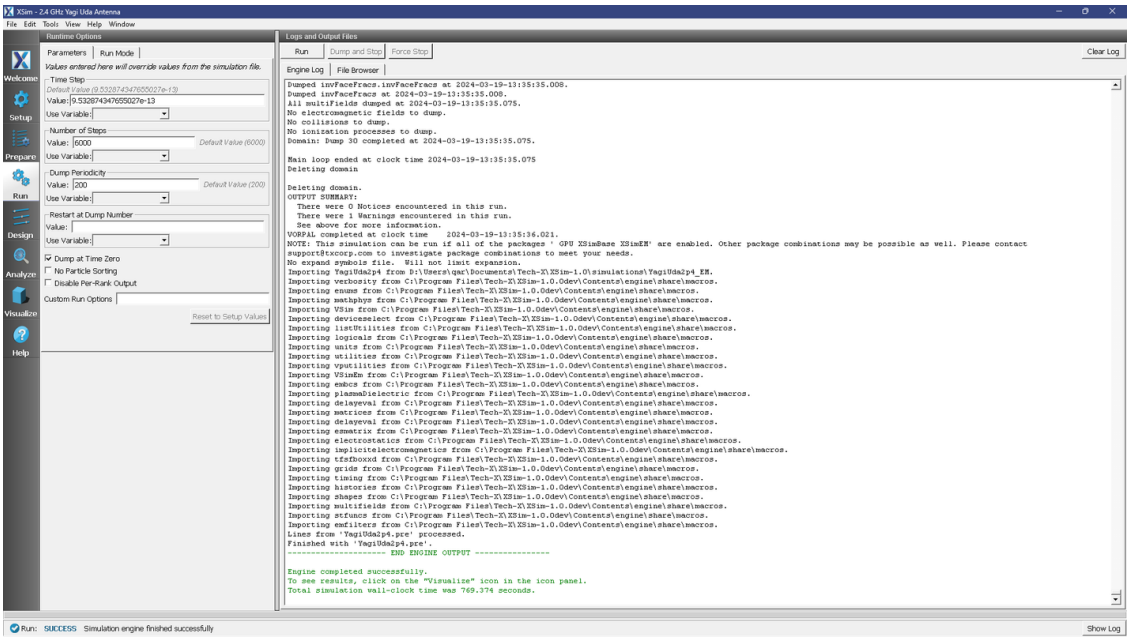

Fig. 2.2: The Run Window at the end of execution.

Analyzing the Results

- Proceed to the Analysis window by pressing the Analyze button in the left column of buttons.
- Select computeFarFieldFromKirchhoffBox.py from the list and select "Open" (Fig. 2.3)
- Input values for the analyzer parameters. The analyzer may be run multiple times, allowing the user to experiment with different values.
  - simulationName - yagiUda2p4
  - fieldLabel - E
  - farFieldRadius - 1024.0
  - backgroundEpsRel - 1.0
  - precision - double
  - numPeriods - 0.25
  - numFarFieldTimes - 2
  - frequency - 2.4e9
  - numTheta - 45
  - numPhi - 60
  - zeroThetaDirection - (0,1,0)
  - zeroPhiDirection - (0,0,1)
  - incidentWaveAmplitude - blank
  - incidentWaveDirection - (0,0,0)
  - varyingMeshMaxRadius - 1024.0

  - principalPlanesOnly - checked

- Click “Analyze”
- Depending on the values of numTheta, numPhi, and numFarFieldTimes, the script may need to run for several minutes.

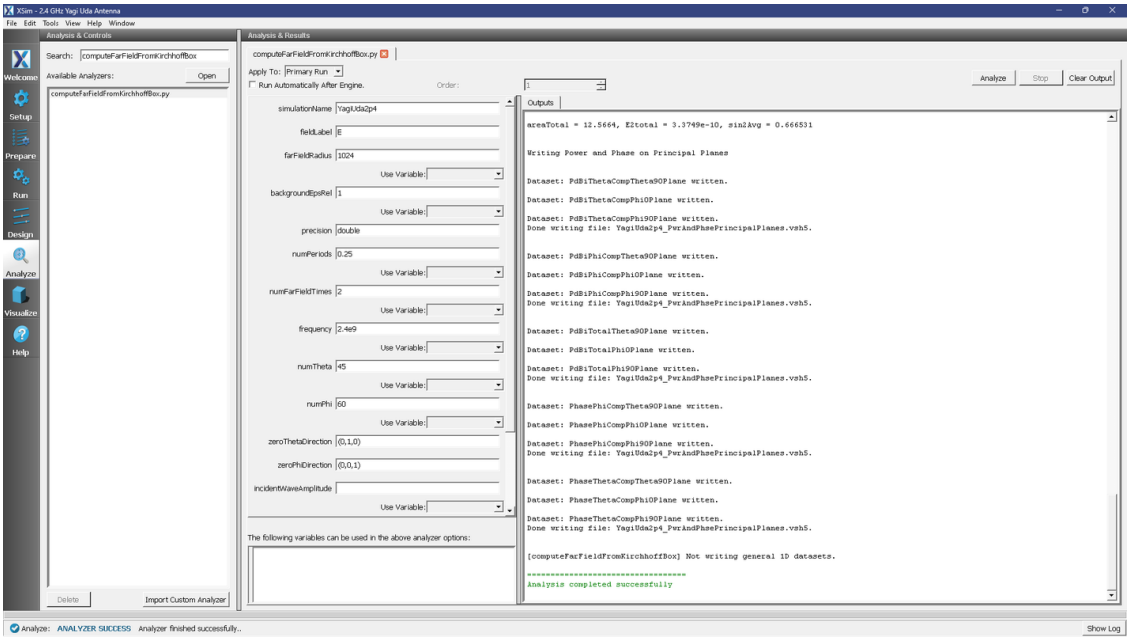

Fig. 2.3: The Analysis Window.

Visualizing the results

Proceed to the Visualize Window by pressing the Visualize button in the left column of buttons.

To view the near field pattern, do the following:

- Expand *Scalar Data*
- Expand *E*
- Select *E_x*
- Check the *Set Minimum* box and set the value to -0.1
- Check the *Set Maximum* box and set the value to 0.1
- Check the *Clip Plot* box
- Expand *Geometries*
- Select *YagiUda2p4PecShapesTriangles_surface*
- Move the dump slider forward in time

The far field radiation pattern can be found in the scalar data variables of the data overview tab underneath the farE field. Uncheck the E_x dataset and check the farE_magnitude box under *farE*.

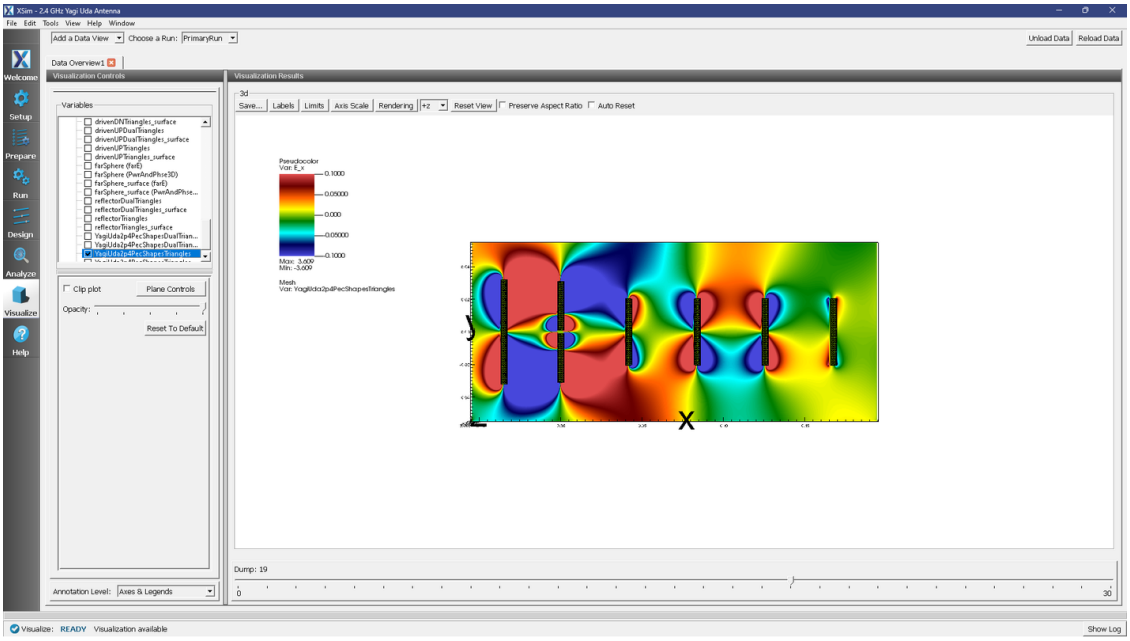

Fig. 2.4: The electric field near-field pattern.

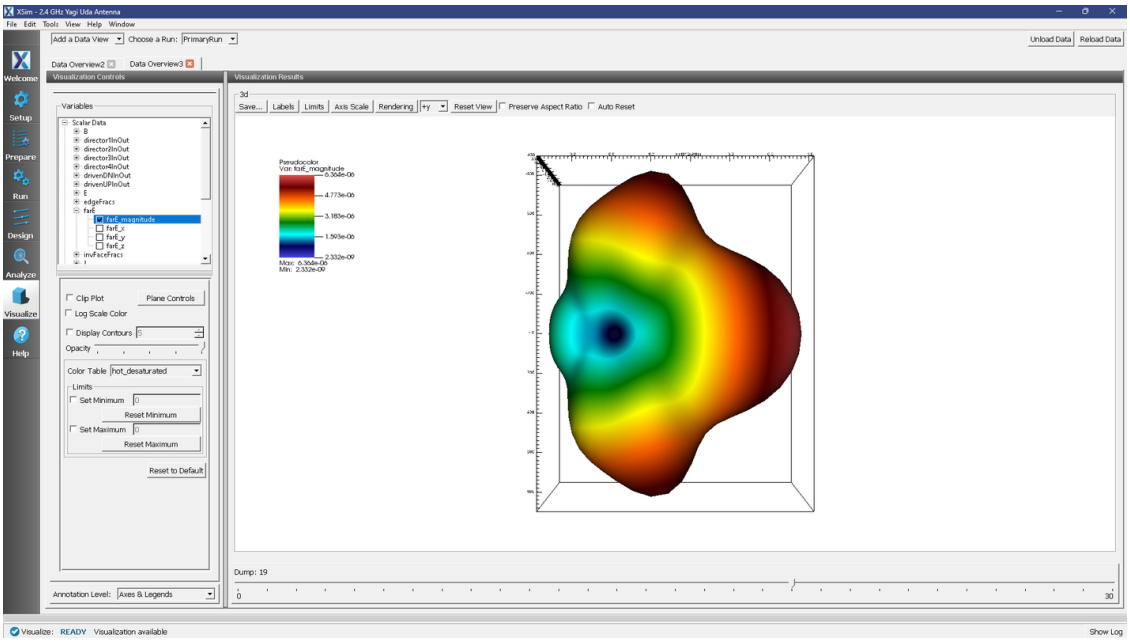

Fig. 2.5: The electric field manifestation of the far field pattern.

Further Experiments

Try adding more directors and changing their dimensions to see the effect on the far field pattern.

2.1.2 Antenna Array 2D (antennaArray2D.sdf)

Keywords:

`antennaArray2D, far field, radiation, s-parameters`

Problem Description

This set of 2-D XSimEM simulations shows how to obtain the far fields, S11 parameter, gain, and phase shift of a one-element antenna as well as the far fields, gain, S parameters, and phase shift of a multiple-element antenna array with one excited element. These simulations can be used as a basis for measuring coupling in phased array antennas. The analyzer compute2DantennaGainAndPhase.py is set up to calculate the S parameter for the excited element and any other reference element defined by the constant *S_PARAM_ELEM*.

Opening the Simulation

The Antenna Array 2D example is accessed from within XSimComposer by the following actions:

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *XSim for Electromagnetics* option.
- Expand the *Antennas* option.
- Select *Antenna Array 2D* and press the *Choose* button.
- In the resulting dialog, create a *New Folder* if desired, then press the *Save* button to create a copy of this example.

The resulting Setup Window is shown Fig. 2.6.

Simulation Properties

The antennas are waveguide apertures excited with a frequency of 1 GHz and the aperture width is 0.1λ (see the parameter GAP in the element tree). The distance between the gaps is 0.4λ.

A different array of geometries can be created using input parameters such as number of elements in the array (N_ELEM) and the distance between the elements in each direction. To recreate a different antenna array, expand *Geometries*, expand CSG, right-click on *gap* → Create Array. In the *Array Description* window, select the "Union elements" checkbox, type in the number of elements to the value under N_ELEM, and the distance between elements to the value under DIST_ELEM. Then select the CSG "metal", hold down Ctrl and select *gapElemUnion* located at the end of the gap array elements → Boolean Operation → select *metal_gapElemUnion*. Rename accordingly and assign the material PEC to the newly created geometry.

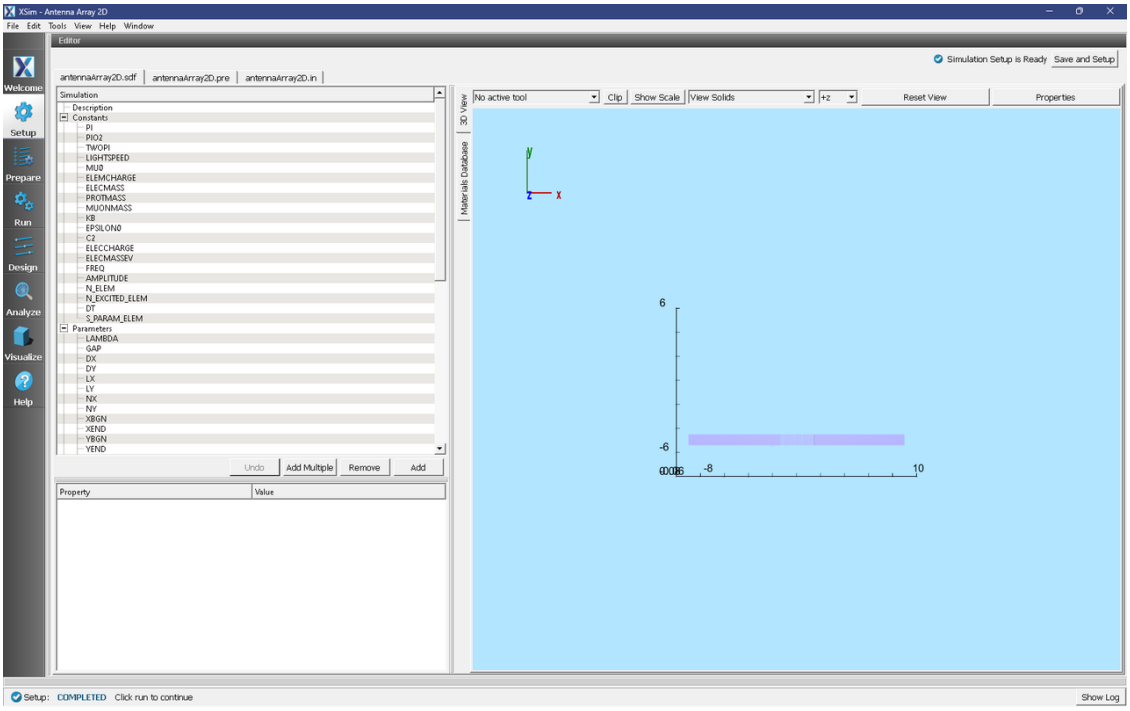

Fig. 2.6: Setup Window for the Antenna Array 2D example.

Running the Simulation

Once finished with the setup, continue as follows:

- Proceed to the Run Window by pressing the *Run* button in the navigation column out left.
- To run the file, click on the *Run* button in the upper left corner of the *Logs and Output Files* pane. You will see the output of the run in that pane. The run has completed successfully when you see the output, "Engine completed successfully."
- First run settings (default):
  - *Number of Steps*: 6000
  - *Dump Periodicity*: 3000
  - *Dump at Time Zero*: box checked

After the first run completes, proceed as follows:

- Second run settings:
  - *Number of Steps*: 1800
  - *Dump Periodicity*: 45 (Value taken from the parameter DUMP_PER_SECOND_RUN)
  - Set *Restart at Dump Number* to 2

Note: If the grid properties change, these values will have to be adjusted.

The end of the second run is shown in Fig. 2.7.

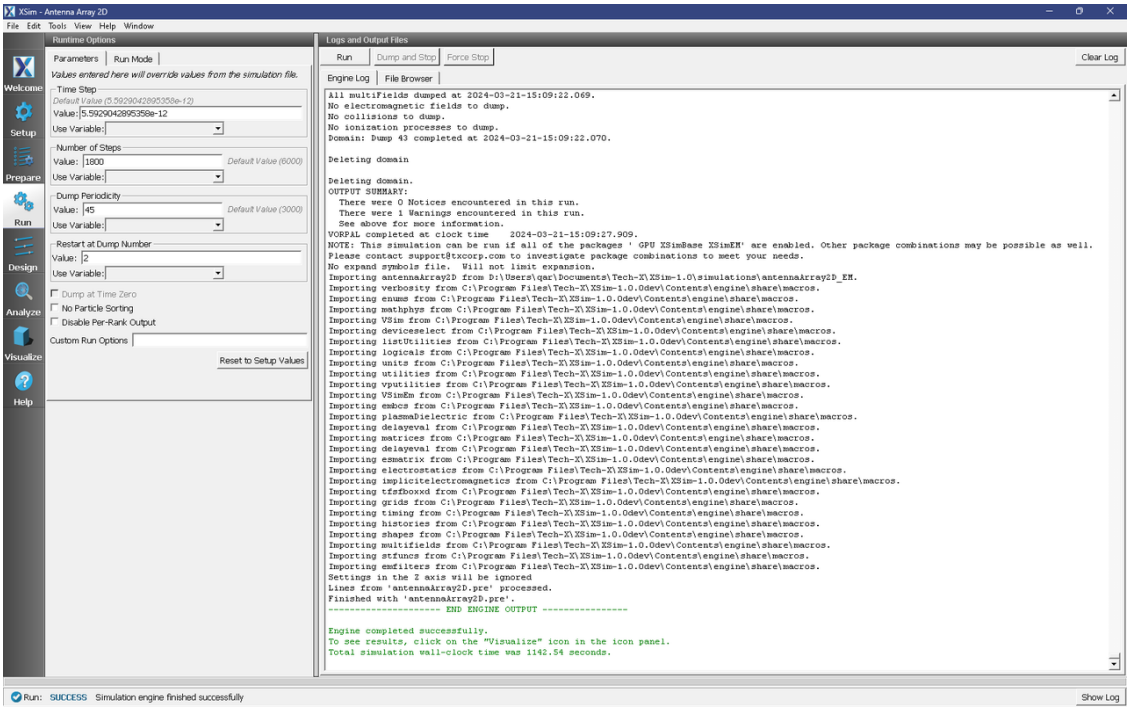

Fig. 2.7: The Run Window at the end of the second run execution.

Visualizing the Results

After performing the above actions, the results can be visualized as follows:

- Proceed to the Visualize Window by pressing the *Visualize* button in the navigation column
- Expand *Scalar Data* in the *Visualization Controls* pane
- Expand *E*
- Select *E_x*
- Check the box for *Set Minimum* and set it to -100
- Check the box for *Set Maximum* and set it to 100
- Select the dump slider and move it to higher dump numbers to see the evolution of the electric field in time.

The resulting visualization is shown in Fig. 2.8.

Figure Fig. 2.8 shows the near and far electric fields at the end of the simulation run. The dispersion of the electric field through the non-excited waveguides can also be seen.

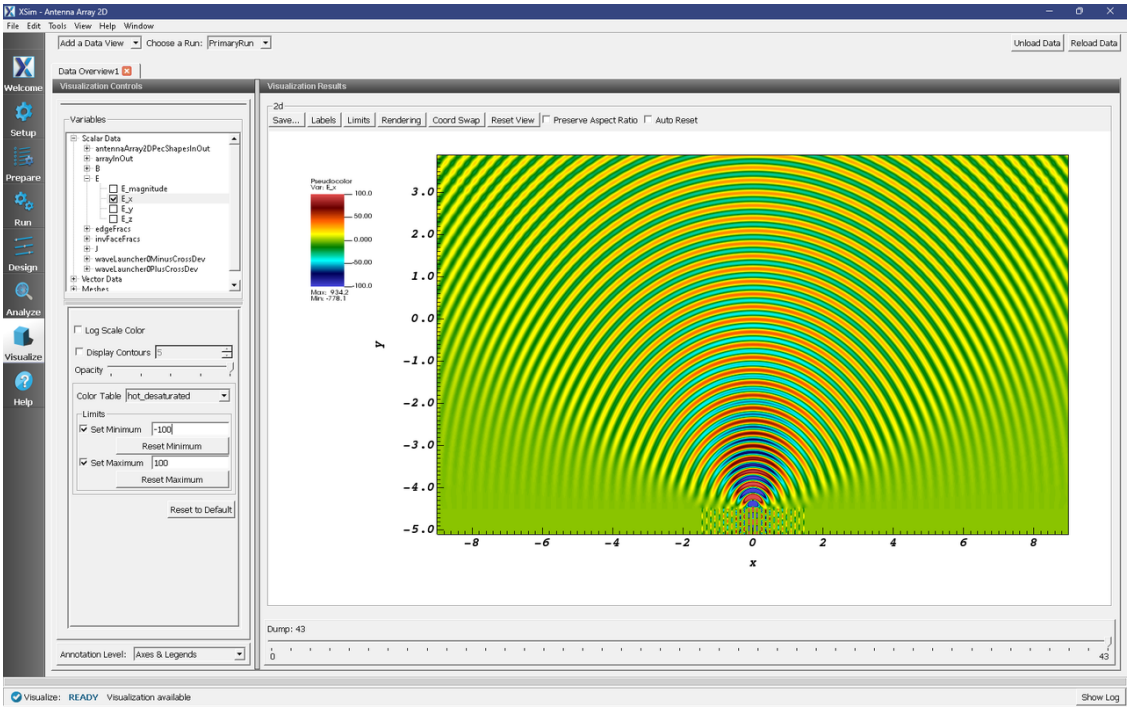

Fig. 2.8: The near and far electric fields in the x-direction at the end of the simulation.

Single Element Antenna

- Expand Constants
- Change constants N_ELEM to 1
- Change N_EXCITED_ELEM to 1
- Expand Geometries
- Expand CSG
- Deactivate *array*
- Deactivate *gapArray*
- Select *metal*, hold Ctrl, select *gap*, right click → *Boolean Operation* → select *metal - gap*
- Select *metalMinusgap*
- For *material* select *PEC* from the drop-down menu

You can now assign any name of your choice to the *metalMinusgap* geometry (e.g., aperture). Save and proceed to the Run tab. Follow the same run steps as described above in the section *Running the Simulation*.

Second vizualization results are shown in Fig. 2.9.

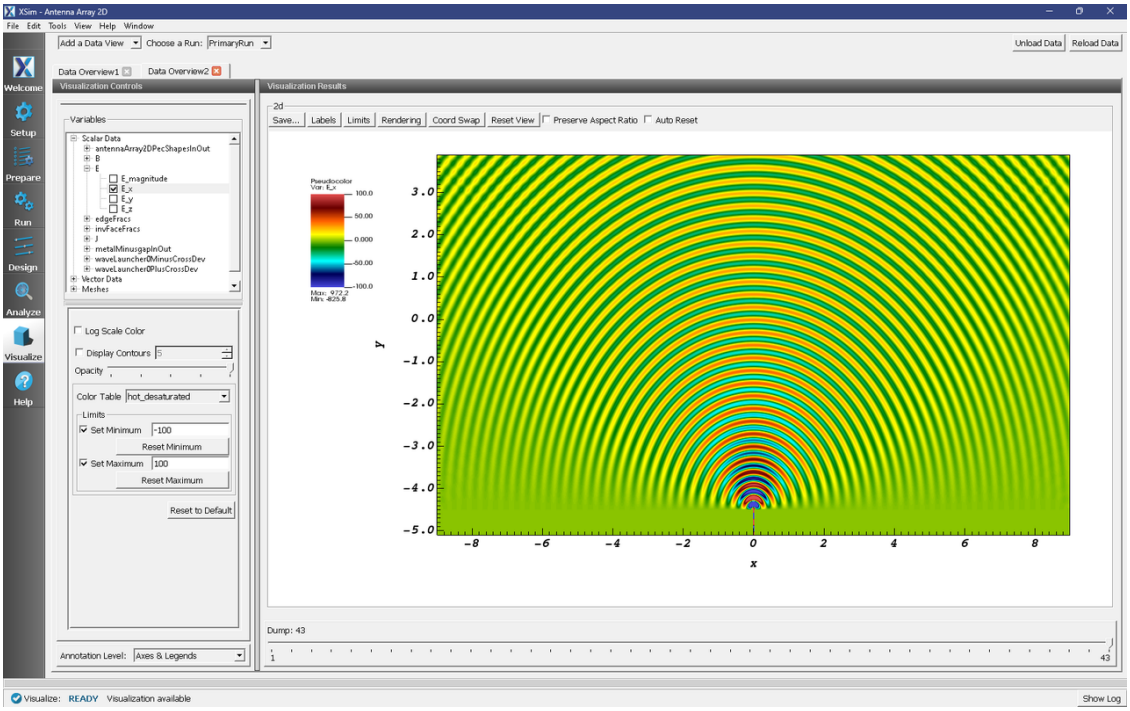

Fig. 2.9: The near and far electric fields in the x-direction for a 1-element antenna.

Calibration Runs

For both the multiple-element and single-element antenna simulations, calibration runs are needed for the analyzer.

For the original multiple-element array setup, proceed as follows:

- Proceed to the Setup Window
- In the top left corner, select *File* → *Save Simulation As* . . .
- Rename the simulation to *antennaArray2DCalibration.sdf*

Note: If your simulation has a different name, add the word *Calibration* before *.sdf*

- Click *Save*
- Expand *Geometries*
- Expand *CSG*
- Deactivate *array*
- Deactivate *gapArray*
- Select *gap*
- Change the height to HEIGHT_METAL_CALIB
- Change the x position setting to XBGN_EXCITED_GAP
- Select *metal*
- Change the height to HEIGHT_METAL_CALIB

- Click on *metal*, hold down Ctrl button and select *gap* right click → Boolean Operation
- Select metal_gap
- Select metalMinusgap
- Select *PEC* under *material* from the drop-down menu.

You can now assign any name of your choice to the metalMinusgap geometry (e.g., myWaveguide).

- Expand *Field Dynamics*
- Expand *FieldBoundaryConditions*
- Remove *malUpperY*
- Right-click *FieldBoundaryConditions* → Add FIeldBoundaryCondition → select *Port*
- Select *upper y* for the boundary surface from the drop-down menu
- Save and proceed to the Run tab.
- Change *Number of Steps* to 7800

Note: The calibration number of steps must equal the total number of steps that the simulation ran for during the regular run.

Repeat the same steps for the single-element antenna simulation setup.

Analyzing the Results

After performing the above actions, continue as follows:

- Proceed to the Analysis Window by pressing the *Analyze* button in the navigation column.
- Open the *compute2DantennaGainAndPhase.py* analyzer by selecting it and selecting "open".
- The default analyzer fields are the following:
  - *simulationName*: antennaArray2D
  - *dumpNr*: 30
  - *nlambda*: 15.0
  - *gapWidth*: 0.03
  - *center*: 0.0,-4.4969
  - *dt*: 5.59290428954e-12
  - *freq*: 1000000000.0
- The *overwrite* box should be checked
- Click *Analyze* in the top right corner.
- The analysis is completed when you see the output shown in Fig. 2.10.

The S-parameters for the excited element as well as the reference element associated with the constant S_PARAM_ELEM in the simulation setup are shown at the end of the analyzer run.

This analyzer creates a text file with 5 columns. The first column is the theta direction in degrees, the second column is the analytical gain of the ISOLATED excited element in dB, the third column is the gain measured by XSim in dB, the

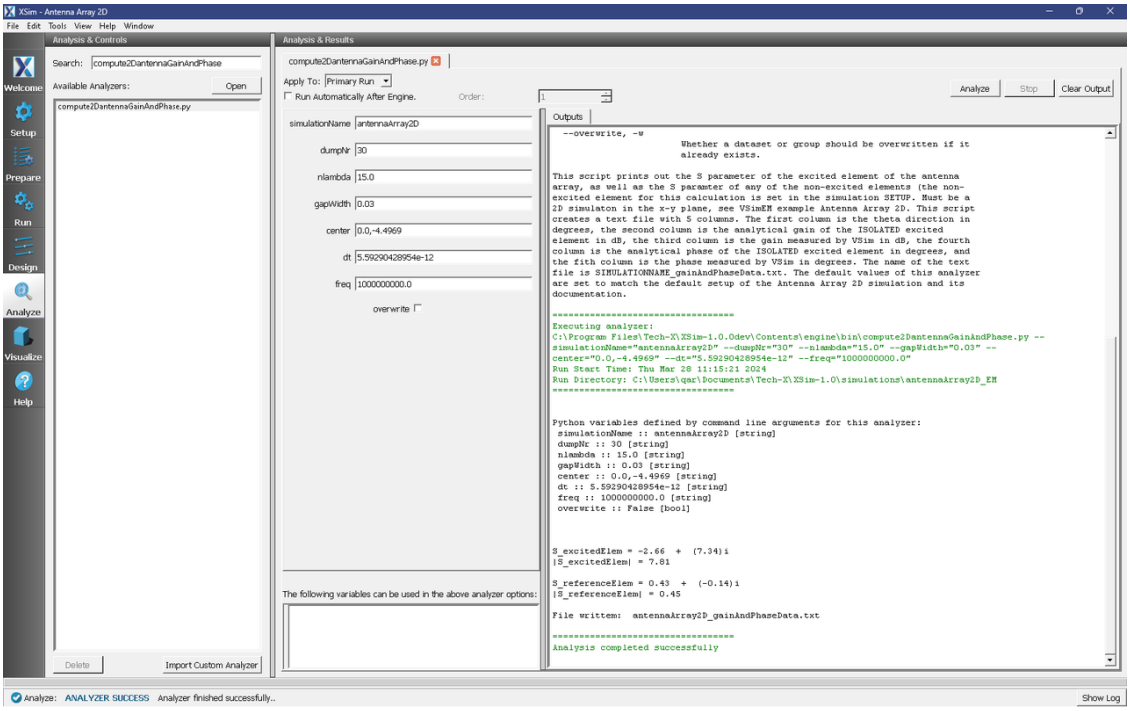

Fig. 2.10: The S-parameters for the excited element as well as the reference element associated with the constant S_PARAM_ELEM in the simulation setup are shown at the end of the analyzer run.

fourth column is the analytical phase of the ISOLATED excited element in degrees, and the fith column is the phase measured by XSim in degrees. The name of the text file is SIMULATIONNAME_gainAndPhaseData.txt.

For the default simulation settings (i.e., the center element of a 25-element array is excited while the other elements are turned off), plotting the second and third columns (analytical and measured gains) against the first column (as a function of theta) will give the results shown in Fig. 2.11.

Plotting the fourth and third columns (analytical and measured field phases) against the first column (as a function of theta) will give the results shown in Fig. 2.12.

Further Experiments

A different array of geometries can be created changing input parameters such as number of elements in the array (N_ELEM) and the distance between the elements in each direction (DIST_ELEM). After changing these *Constants*, to create a different antenna array, proceed as follows:

- Expand *Geometries*
- Expand CSG
- Right-click on *gap* → Create Array

In the *Array Description* window, select the "Union elements" checkbox, type in the number of elements to the value under N_ELEM, and the distance between elements to the value under DIST_ELEM. Then select the CSG "metal", hold down Ctrl and select *gapElemUnion* located at the end of the gap array elements → Boolean Operation → select *metal_gapElemUnion*. Rename accordingly and assign the material PEC to the newly created geometry.

Repeating the analysis steps for a 1-element antenna (N_ELEM = 1 in the simulation setup) will give the results shown in Fig. 2.13 and Fig. 2.14.

A different element can be excited by changing input parameter N_EXCITED_ELEM.

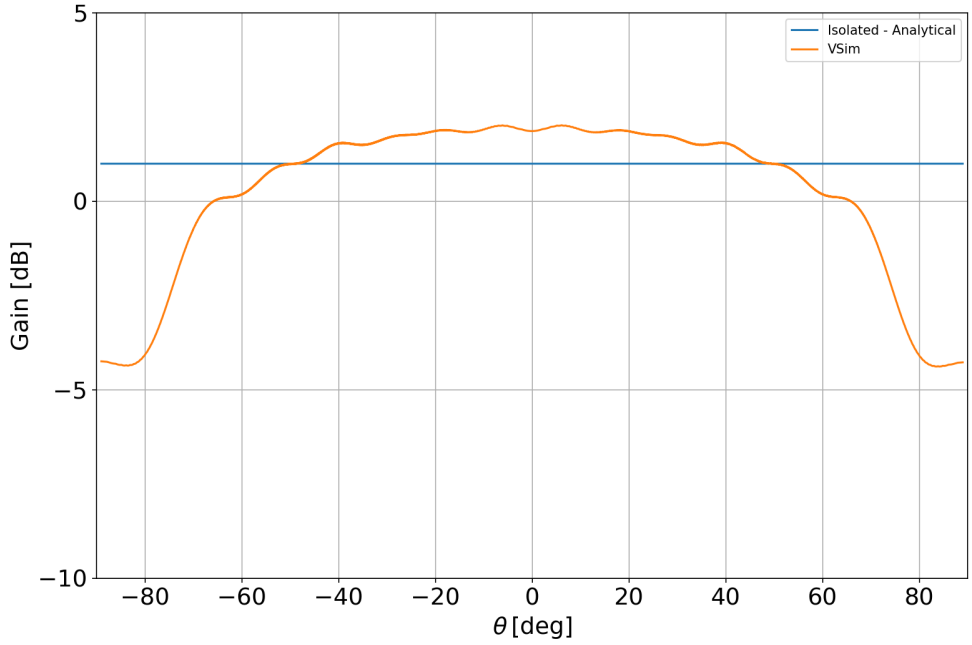

Fig. 2.11: The gain pattern of a 25-element array with the center excited element.

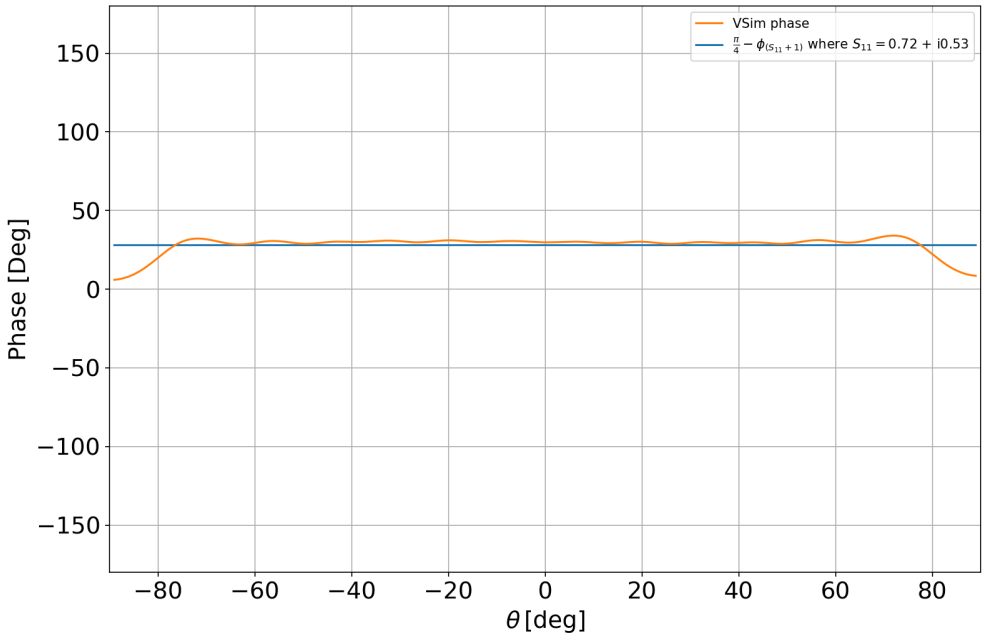

Fig. 2.12: The phase pattern of a 25-element array with the center excited element.

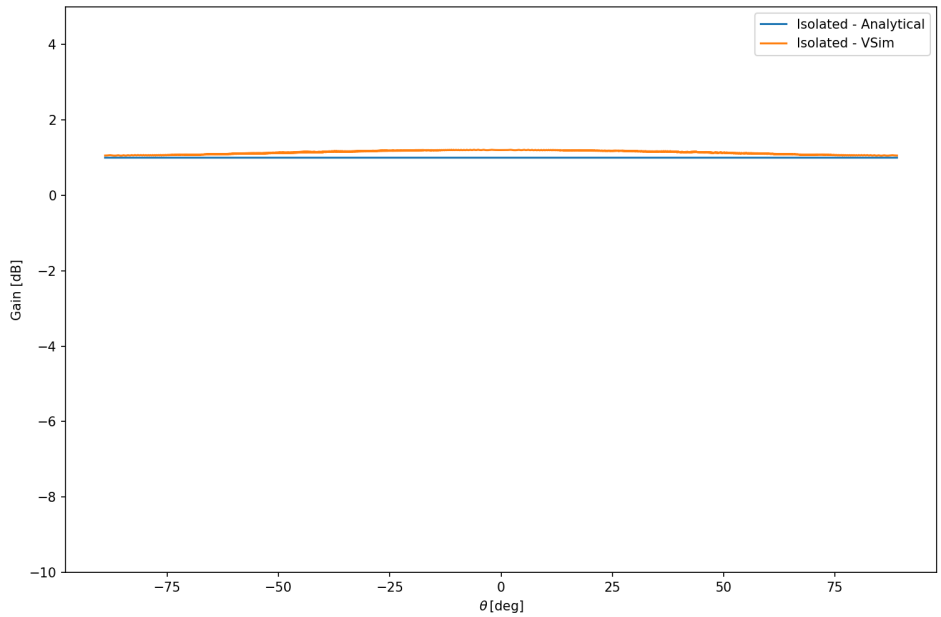

Fig. 2.13: The gain pattern of a 1-element array.

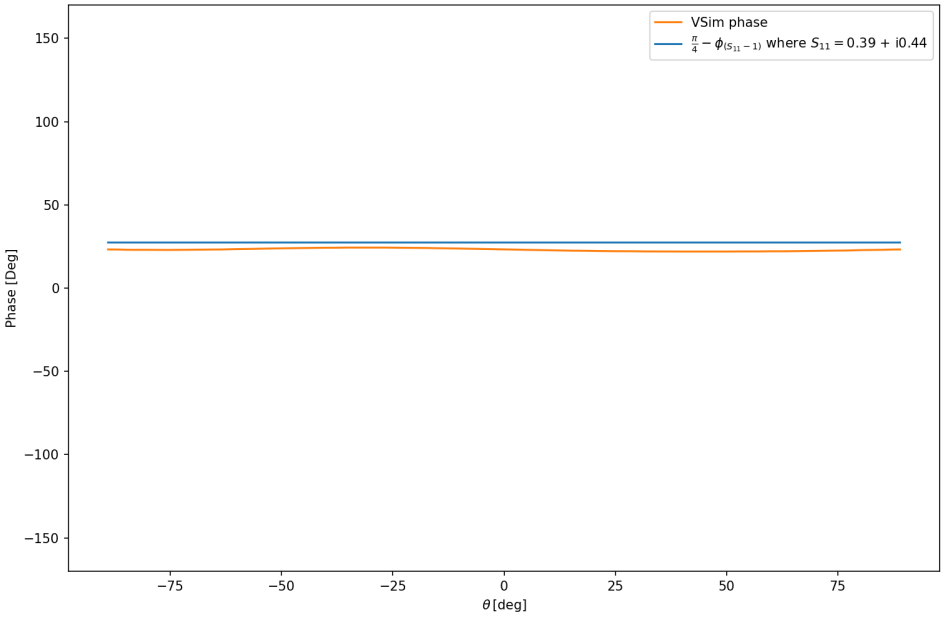

Fig. 2.14: The phase pattern of a 1-element array.

Repeating the analysis steps for a 25-element antenna with the edge element excited (N_EXCITED_ELEM = 25 in the simulation setup) will give the results shown in Fig. 2.15 and Fig. 2.16.

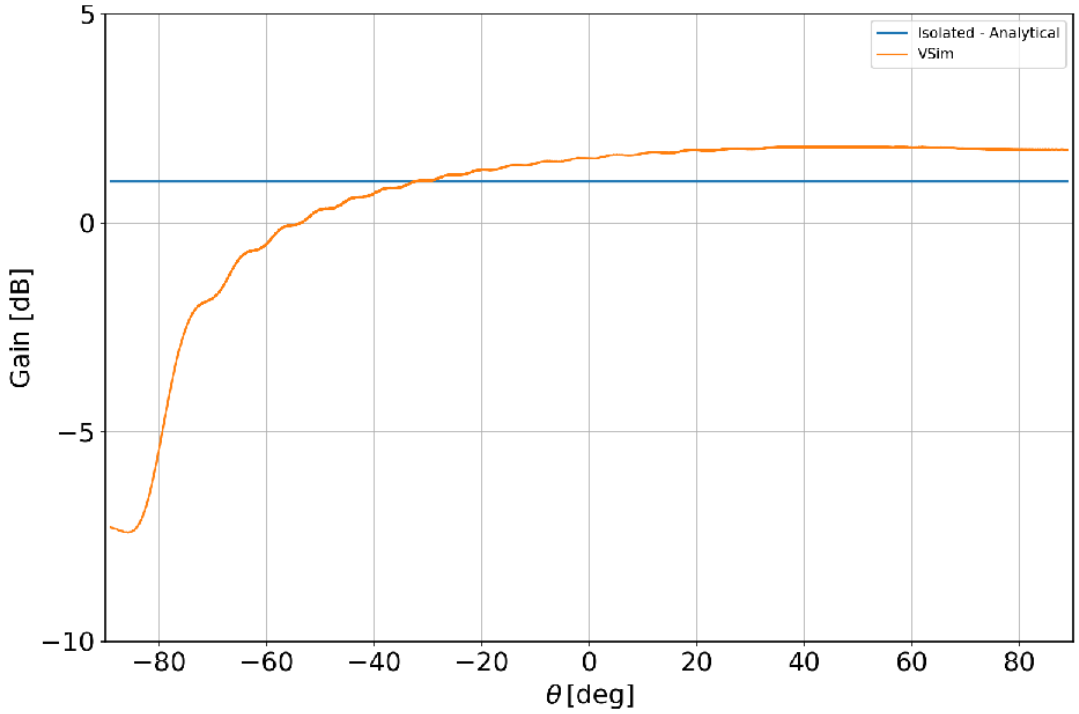

Fig. 2.15: The gain pattern of a 25-element array with the edge excited element.

2.1.3 Antenna on Human Hand with Dielectric (antennaOnHand.sdf)

Keywords:

antennaOnHand, far field, radiation

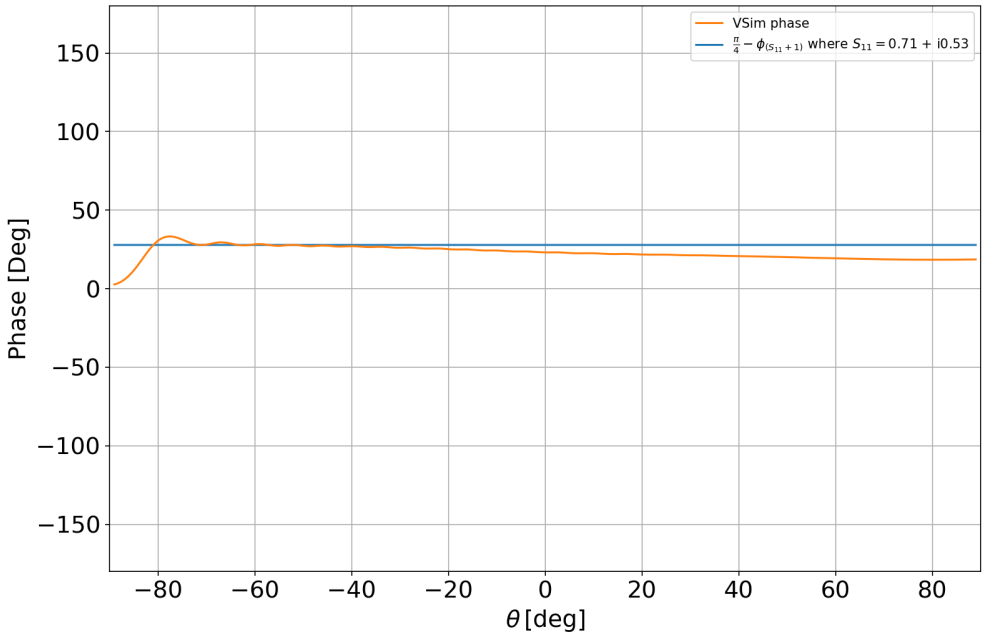

Fig. 2.16: The phase pattern of a 25-element array with the edge excited element.

Problem Description

This problem calculates the far-field radiation pattern of a small wifi antenna. The fields interact with the human hand for which the bone structure was approximated by long thin cylinders. The antenna frequency can be either 2.4 or 5 GHz, the two most common wifi bands.

Opening the Simulation

The Antenna on Human Hand with Dielectric example is accessed from within XSimComposer by the following actions:

- Select the *New* → *From Example…* menu item in the *File* menu.
- In the resulting *Examples* window expand the *XSim for Electromagnetics* option.
- Expand the *Antennas* option.
- Select "Antenna on Human Hand with Dielectric" and press the *Choose* button.
- In the resulting dialog, create a new folder if desired, and press the *Save* button to create a copy of this example.

The **Setup** window is now shown with all the implemented physics and geometries. See Fig. 2.17.

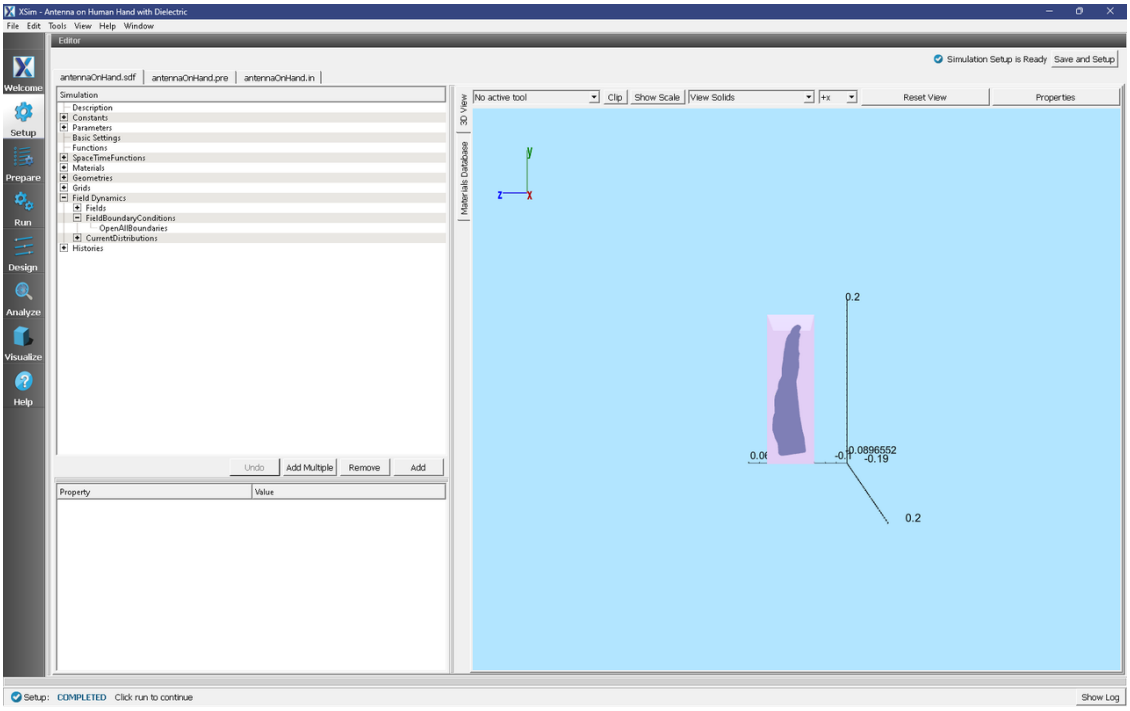

Fig. 2.17: Setup Window for the Antenna on Human Hand with Dielectric example, with Grid and farFieldBox History hidden.

Simulation Properties

This file allows the modification of antenna operating frequency, dimensions, orientation, and simulation domain size.

Running the Simulation

- Proceed to the run window by pressing the Run button in the left column of buttons.
- Check that you are using these run parameters:
  - *Time Step*: 3.659083082938294e-12
  - *Number of Steps*: 3000
  - *Dump Periodicity*: 300
  - *Dump at Time Zero*: Checked
- Click on the *Run* button in the upper left corner of the right pane.

You will see the output of the run in the right pane. The run has completed when you see the output, "Engine completed successfully". This result is shown in Fig. 2.18.

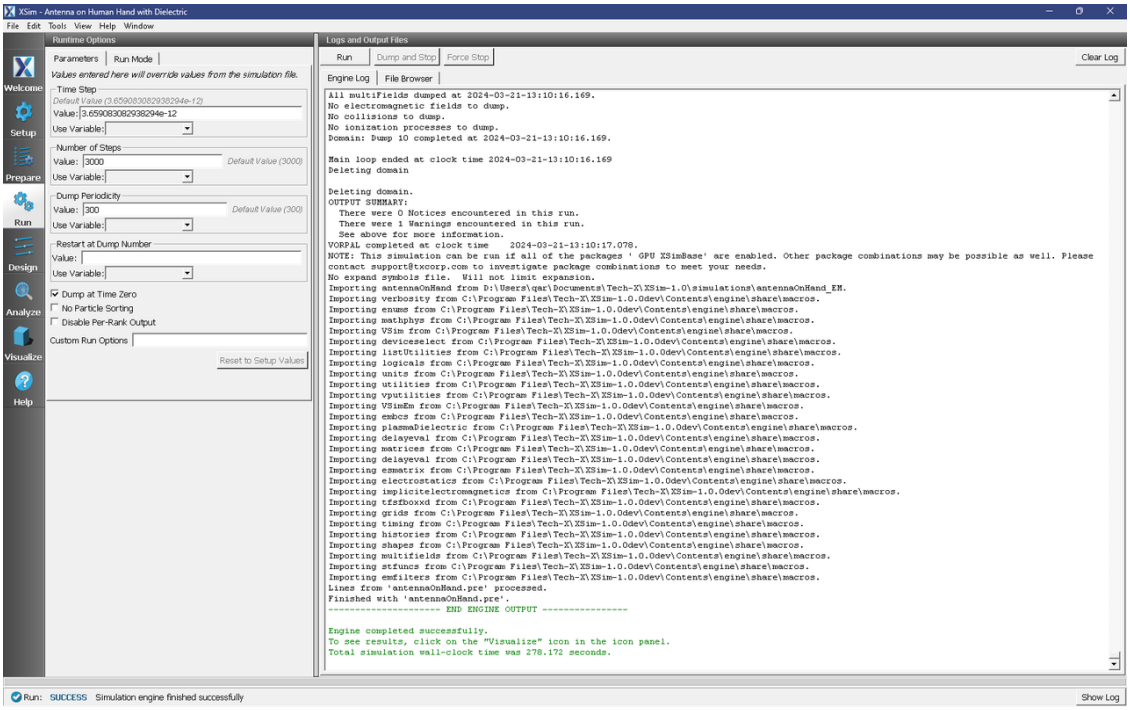

Fig. 2.18: The Run Window at the end of execution.

Analyzing the Results

After performing the above actions, continue as follows:

- Proceed to the Analysis window by pressing the Analyze button in the left column of buttons.
- Select computeFarFieldFromKirchhoffBox.py from the list and select “Open” (Fig. 2.19)
- Input values for the analyzer parameters. The analyzer may be run multiple times, allowing the user to experiment with different values.
  - *simulationName*: antennaOnHand
  - *fieldLabel*: E
  - *farFieldRadius*: 1024.0
  - *backgroundEpsRel*: 1
  - *precision*: double
  - *numPeriods*: 0.25
  - *numFarFieldTimes*: 2
  - *frequency*: 5e9
  - *numTheta*: 45
  - *numPhi*: 60
  - *zeroThetaDirection*: (1,0,0)
  - *zeroPhiDirection*: (0,0,1)
  - *incidentWaveAmplitude*: blank

  - *incidentWaveDirection*: (0,0,0)
  - *varyingMeshMaxRadius*: 0.05
  - *principalPlanesOnly*: checked
- Click “Analyze”
- Depending on the values of numTheta, numPhi, and numFarFieldTimes, the script may need to run for several minutes or longer.

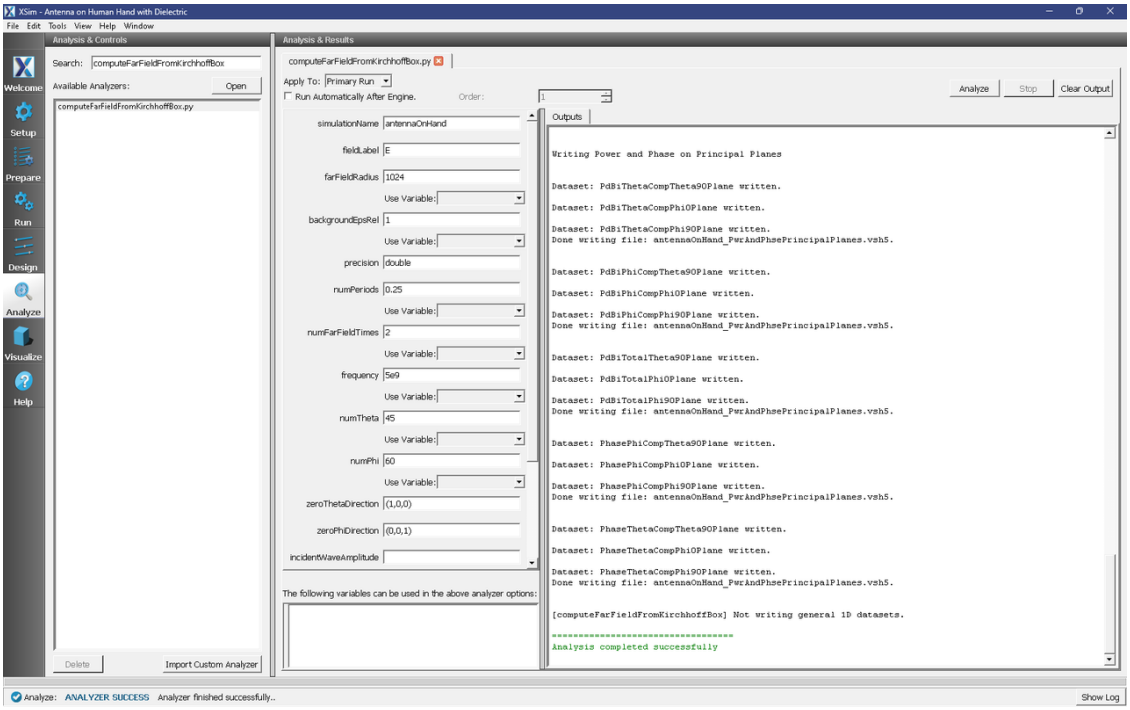

Fig. 2.19: The **Analyze** panel after running computeFarFieldFromKirchhoffBox.py.

Visualizing the Results

- Proceed to the Visualize Window by pressing the Visualize button in the left column of buttons.
- Expand *Scalar Data*
- Expand *farE*,
- Select *farE_Magnitude*.
- Expand *Geometries*
- Check *HandGeomSolidTriangles_surface*

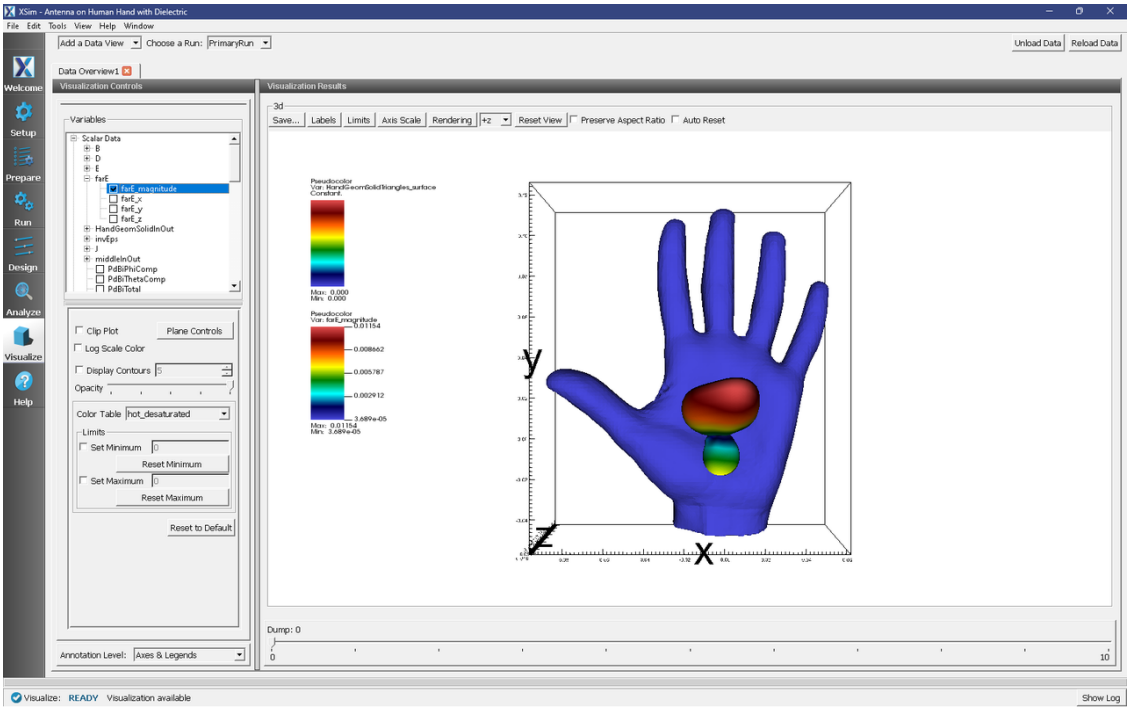

Fig. 2.20: The Far Field Radiation Pattern.

Further Experiments

The skin can be included as an additional geometry by simply importing the hand geometry a second time within the same set-up, but with a very slightly higher scaling factor and setting the *Skin* material for the hand geometry with the higher scaling factor. Some "by eye" adjustments of the x-, y-, and z- translation values may be needed.

2.1.4 Antenna on Predator Drone (predatorDrone.sdf)

Keywords:

predatorDrone, far field, radiation

Problem Description

This problem illustrates how to obtain the far field radiation patterns of a current source antenna mounted on a Predator Drone.

Opening the Simulation

The Predator Drone example is accessed from within XSimComposer by the following actions:

- Select the *New* → *From Example…* menu item in the *File* menu.
- In the resulting *Examples* window expand the *XSim for Electromagnetics* option.
- Expand the *Antennas* option.
- Select "Antenna on Predator Drone" and press the *Choose* button.
- In the resulting dialog, create a New Folder if desired, and press the *Save* button to create a copy of this example.

The *Setup Window* is now shown with the CAD imported geometry and antenna current distribution accessible to the user. See Fig. 2.21.

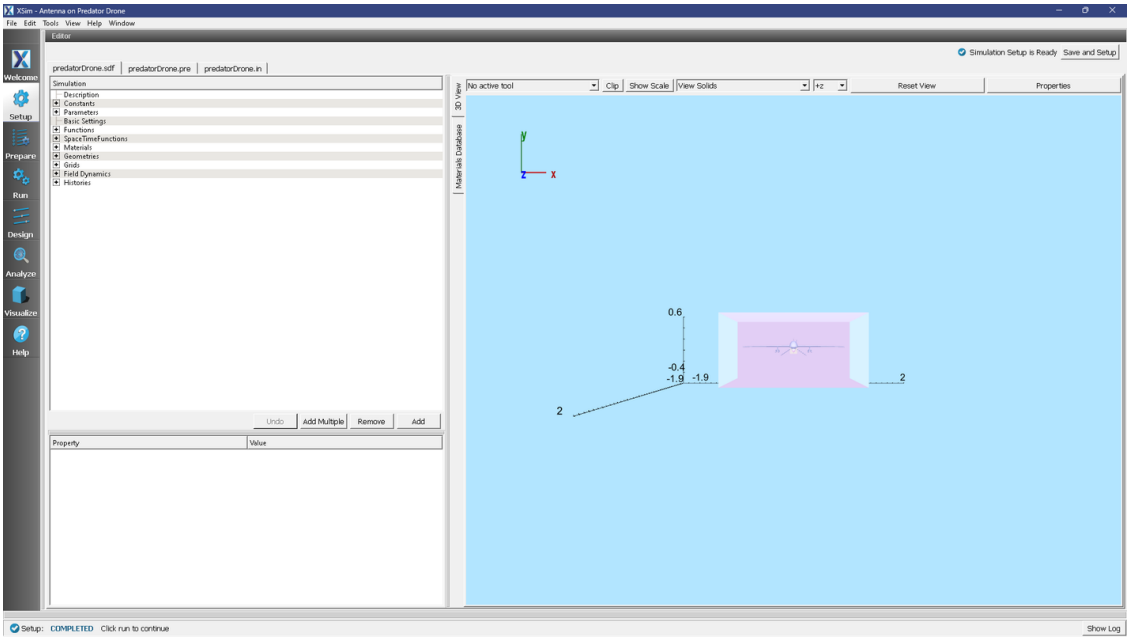

Fig. 2.21: Setup Window for the Predator Drone example.

One can click and unclick the grid, the farFieldBox0 in the histories, the current distribution, and so forth to see where those objects are. One can change locations through changing the values under Constants or, in some cases, the numbers directly in the objects.

Simulation Properties

This file allows the modification of antenna operating frequency, source amplitude, dimensions of the source and the Kirchhoff box by changing the associated variable values under the Constants.

Running the Simulation

After performing the above actions, continue as follows:

- Proceed to the Run Window by pressing the *Run* button in the left column of buttons.
- Check that you are using these run parameters:
  - *Time Step*: 1.3068153867636764e-11
  - *Number of Steps*: 2500
  - *Dump Periodicity*: 100
  - *Dump at Time Zero*: checked
- Click on the *Run* button in the upper left corner of the right pane.

You will see the output of the run in the right pane. The run has completed when you see the output, "Engine completed successfully". This result is shown in Fig. 2.22.

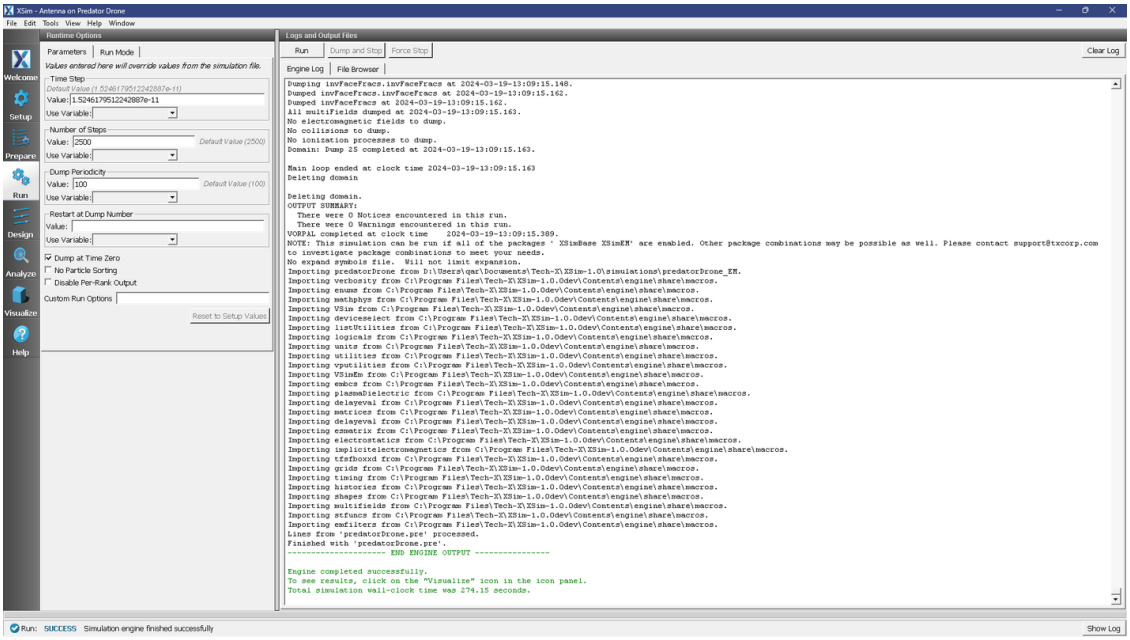

Fig. 2.22: The Run Window at the end of execution.

Analyzing the Results

After the run, one must analyze the Kirchhoff box data to get the far fields. This is done as follows:

- Proceed to the Analysis window by pressing the *Analyze* button in the left column of buttons.
- Click on *computeFarFieldFromKirchhoffBox.py*, then click on the *Open* button.

If you want, you can grab the dividing bar between the list of Analyzers in the *Analysis Controls* window and the *Analysis Results* window, and slide it left to cover up the *Analysis Controls* window, making more room for the *Analysis Results* window.

After performing the above actions continue as follows to compute the far field radiation pattern:

- In the resulting list, select *computeFarFieldFromKirchhoffBox* and press *Open*
- The analyzer fields should be filled as below:
  - *simulationName*: predatorDrone
  - *fieldLabel*: E
  - *farFieldRadius*: 1024.0
  - *backgroundEpsRel*: 1
  - *numPeriods*: 0.25
  - *numFarFieldTimes*: 2
  - *frequency*: 1.0e9
  - *numTheta*: 45
  - *numPhi*: 60
  - *zeroThetaDirection*: (0,0,1)
  - *zeroPhiDirection*: (1,0,0)
  - incidentWaveAmplitude - blank
  - incidentWaveDirection - (0,0,0)
  - varyingMeshMaxRadius - 1024.0
  - principalPlanesOnly - checked
- Click *Analyze* in the top right corner.
- The analysis is completed when you see the output shown in Fig. 2.23.

If you want the script to run faster, lower numTheta to 8 and numPhi to 16.

- Press the *Analyze* button in the top left of the window.

At completion, you will see Fig. 2.23. The far field data is written to vsh5 files in the simulation directory.

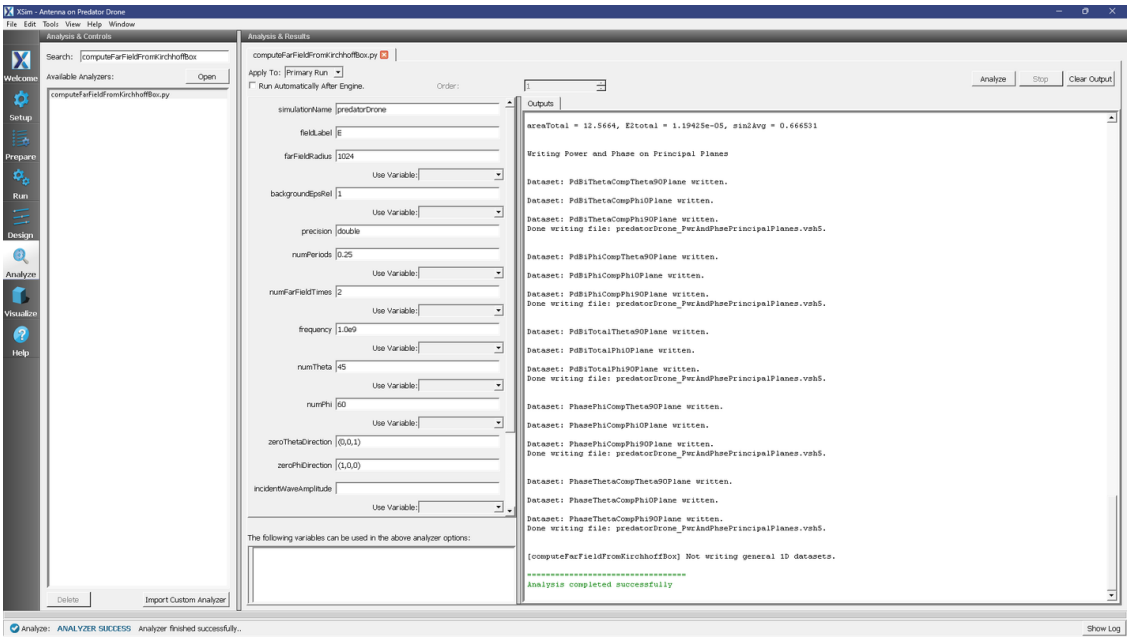

Fig. 2.23: The Analysis window at the end of execution.

Visualizing the Results

Proceed to the Visualize Window by pressing the Visualize button in the left column of buttons.

The radiation pattern in real space can be visualized by doing the following:

- Expand *Scalar Data*
- Expand *E*
- Select one of the scalar fields, such as E_x
- Check *Clip Plot*
- Check *Display Contours* and set the number of contours to 10
- Set minimum to -75 and maximum to 60
- Click on *Plane Controls*, change only Plane3 and set Z to -1.
- Expand Geometries
- Check *predatorDronePecShapesTriangles_surface*
- Move the Dump slider to dump 15 to see the same far field as Fig. 2.24.

An odd number of contours will result in a contour at zero field, which often leads to a less attractive plot with the zero contour filling up the space. Thus, in this case, an even number of contours is suggested.

The far field radiation pattern, which was computed in the section on *Analyzing the Results* can also be displayed. Remove the previous image. Then check the *PdBiTotal* box under Scalar Data, and move the dump slider to the beginning dump. You will see a 3D radiation surface, representing the Far Field radiation power level at each angle that was processed. Colors and radius are in units of dBi, decibels relative to isotropic. A notable peak in the radiation pattern is evident in the forward, upward, and downward directions, as seen in Fig. 2.25.

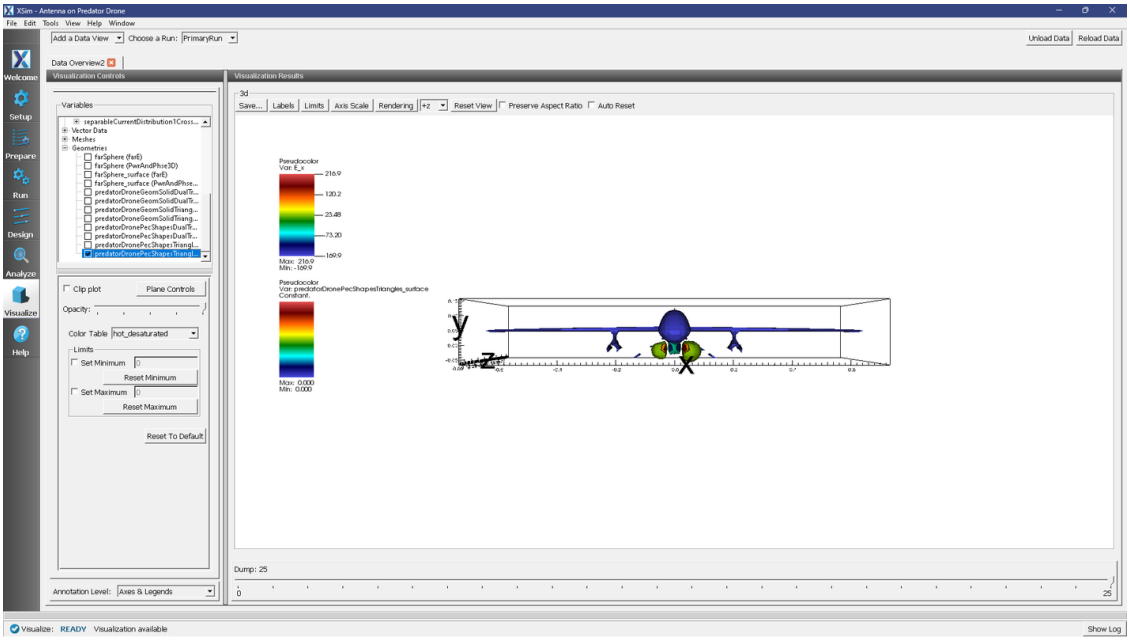

Fig. 2.24: The radiation pattern in real space

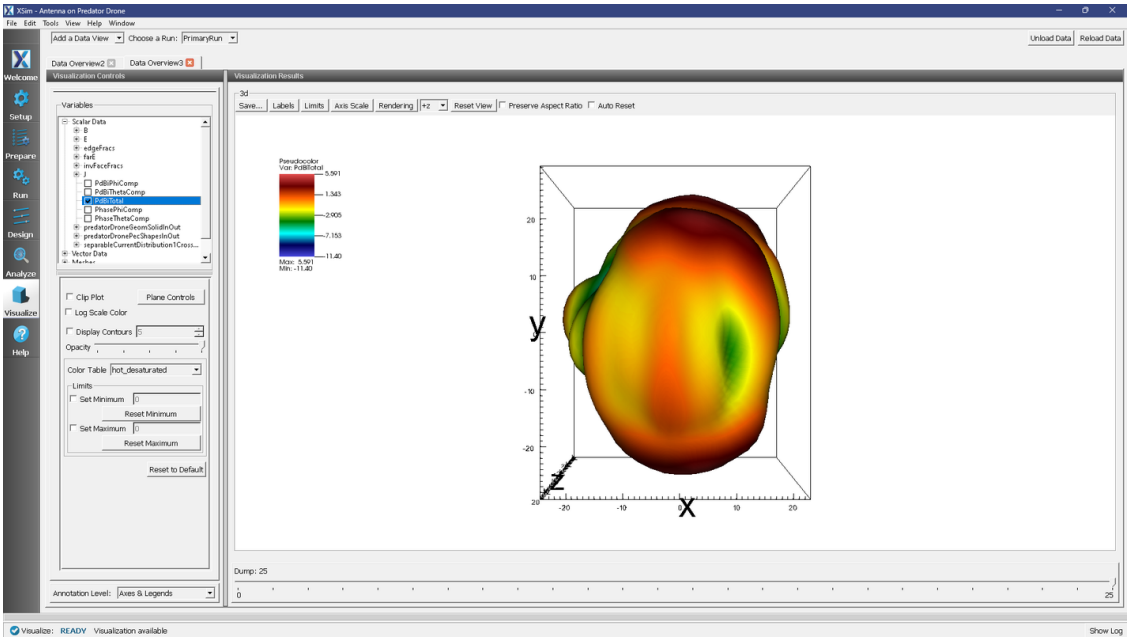

Fig. 2.25: The far field radiation pattern

Further Experiments

Upon close inspection you will note that the mesh size is slightly too large to fully resolve the thin wing structures of the tail section. You can experiment with smaller cell size to resolve these structures. Beware that more cells will increase the run time.

This example can be extended to meet any antenna placement problem with by addition of parameters to define the current distribution center. The vertical extent of the simulation box could be shrunk to reduce the simulation time, which would then allow greater resolution of the wavelength.

The main driver of simulation accuracy is the number of points per wavelength. Because of this lower frequencies will simulate in less time as they require fewer cells to achieve the same resolution in the wave.

2.1.5 Dipole Above Conducting Plane (dipoleOnConductingPlane.sdf)

Keywords:

`dipoleOnConductingPlane, far field, radiation`

Problem Description

This problem illustrates how to obtain far fields within XSim by simulating an infinitesimally short dipole mounted a variable height above a conducting plane. The conducting plane is simulated by using the method of images and utilizes an equal magnitude dipole with direction rotated azimuthally by PI, on the opposite side of the plane. This example is similar to the Oscillating Dipole Above Conducting Plane of XSimBase, but modified with functionality available as part of the XSimEM package to obtain the far field radiation pattern. The number of lobes in the far field will vary as a function of height above the conducting plane. There will be 2*HEIGHT/WAVELENGTH + 1 lobes.

Opening the Simulation

The Dipole Above Conducting Plane example is accessed from within XSimComposer by the following actions:

- Select the *New* → *From Example…* menu item in the *File* menu.
- In the resulting *Examples* window expand the *XSim for Electromagnetics* option.
- Expand the *Antennas* option.
- Select “Dipole Above Conducting Plane” and press the *Choose* button.
- In the resulting dialog, create a New Folder if desired, and press the *Save* button to create a copy of this example.

All of the properties and values that create the simulation are now available in the Setup Window as shown in Fig. 2.26. You can expand the tree elements and navigate through the various properties, making any changes you desire. The right pane shows a 3D view of the geometry, if any, as well as the grid, if actively shown. To show or hide the grid, expand the Grid element and select or deselect the box next to `Grid`.

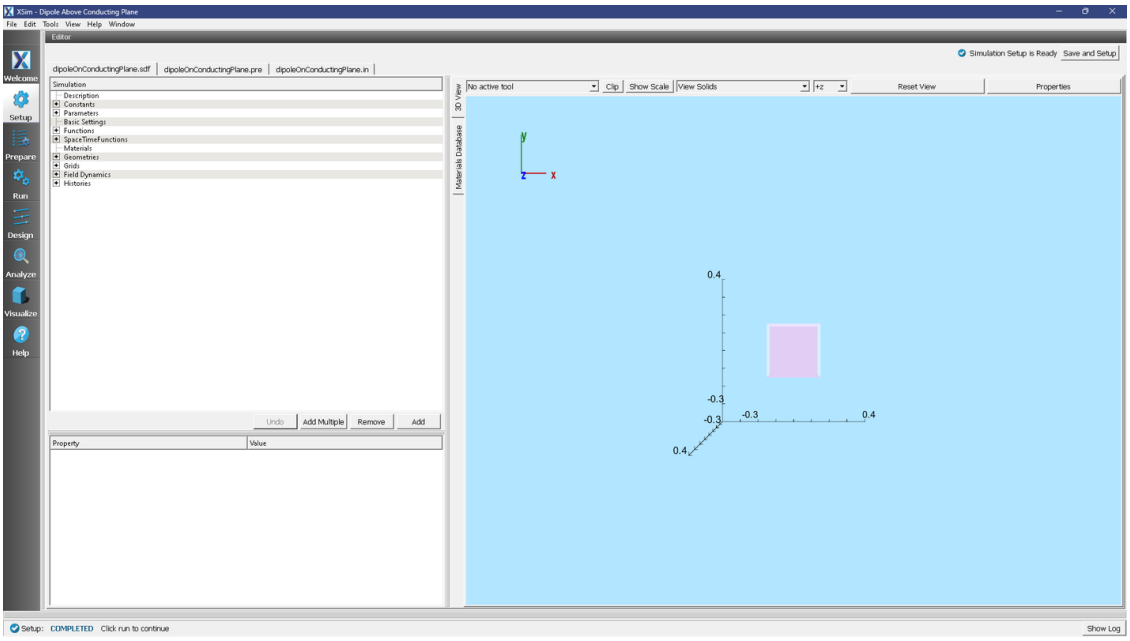

Fig. 2.26: Setup Window for the Dipole Above Conducting Plane example.

Simulation Properties

This setup includes several *Constants* and *Parameters* to help define the dipole signals, including the frequency and height of the antenna.

There are open boundary conditions on each side of the simulation domain.

The conducting plane is simulated by using the method of images and utilizes an equal magnitude dipole with direction rotated azimuthally by PI, on the opposite side of the plane.

Running the Simulation

After performing the above actions, continue as follows:

- Proceed to the run window by pressing the Run button in the left column of buttons.
- Check that you are using these run parameters:
  - *Time Step*: 1.7380644643956894e-11
  - *Number of Steps*: 500
  - *Dump Periodicity*: 50
  - *Dump at Time Zero*: Checked
- Click on the *Run* button in the upper left corner of the right pane.

You will see the output of the run in the right pane. The run has completed when you see the output, “Engine completed successfully”. This result is shown in Fig. 2.27.

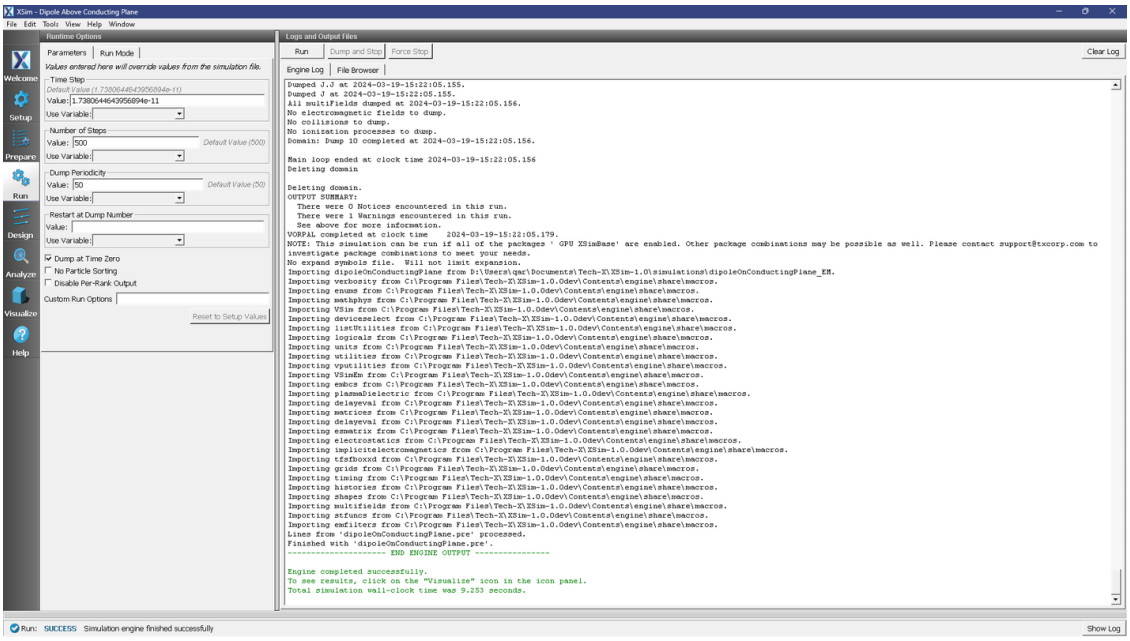

Fig. 2.27: The Run Window at the end of execution.

Analyzing the Results

- Proceed to the Analysis window by pressing the Analyze button in the left column of buttons.
- Select the Default *computeFarFieldFromKirchhoffBox.py* Analyzer
- Input values for the variables given on the left hand side of the screen. Check that these have the following values:
  - simulationName - dipoleOnConductingPlane (name of the input file)
  - fieldLabel - E
  - farFieldRadius - 1024.0
  - numPeriods - 0.25
  - numFarFieldTimes - 2
  - frequency - 3.0e9
  - numTheta - 45 (number of points in the theta direction)
  - numPhi - 90 (number of points in the phi direction)
  - zeroThetaDirection - (0,0,1)
  - zeroPhiDirection - (1,0,0)
  - incidentWaveAmplitude - blank
  - incidentWaveDirection - (0,0,0)
  - varyingMeshMaxRadius - 1024.0
  - principalPlanesOnly - checked
- Click the *Analyze* button near the top right of the window.

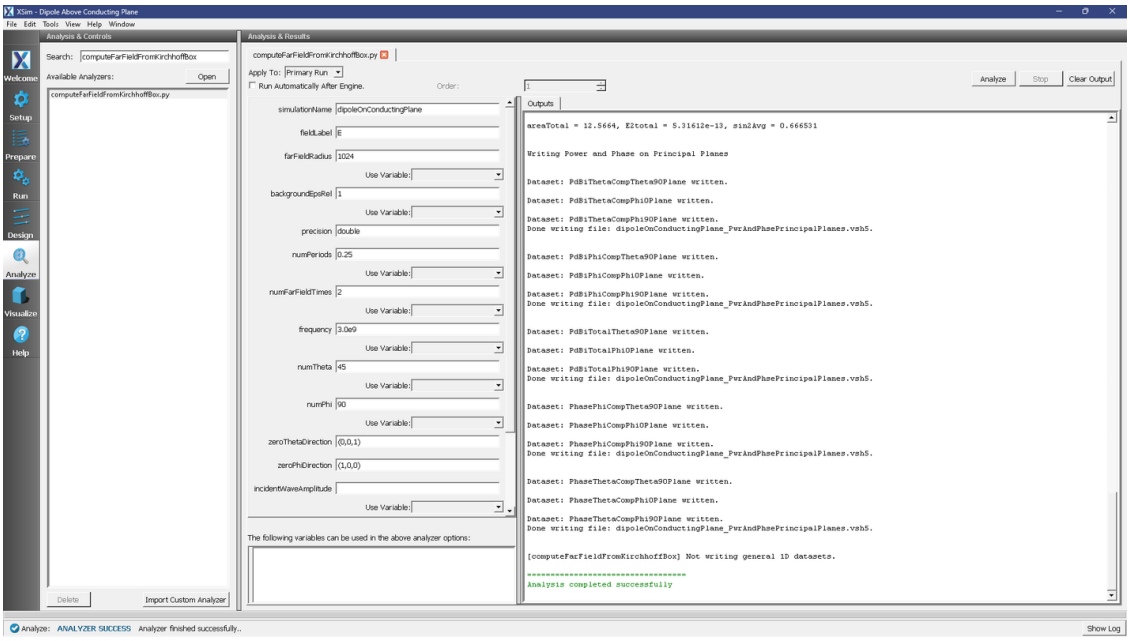

Fig. 2.28: The Analyze Window at the end of execution.

Visualizing the Results

The far field radiation pattern can be found in the Scalar Data variables of the data overview tab. Expand farE and then check the farE_magnitude box. You may need to rotate the view and check the *Clip Plot* box to hide the virtual far field pattern under the conducting plane.

Further Experiments

The number of lobes in the far field is dependent on Antenna Orientation and height. If vertically oriented there will be 2*Height/Wavelength +1 lobes. A horizontally oriented dipole will produce 2*Height/Wavelength lobes.

The resolution of the far field pattern can be changed by editing the number of theta and phi points in the analysis.

If the Simulation domain is made too small, the results will be distorted as the entire near field must be within the simulation domain in order to achieve a proper transformation to the far field.

Note that an infinite perfect electric conducting plane is simulated in the computational engine via image theory. An equal infinitesimal dipole is placed the same distance from the conducting "plane" in order to achieve the result of having an infinite electric conductor.

2.1.6 Dipole Antenna (dipoleAntenna.sdf)

Keywords:

`antenna, electromagnetics, current source`

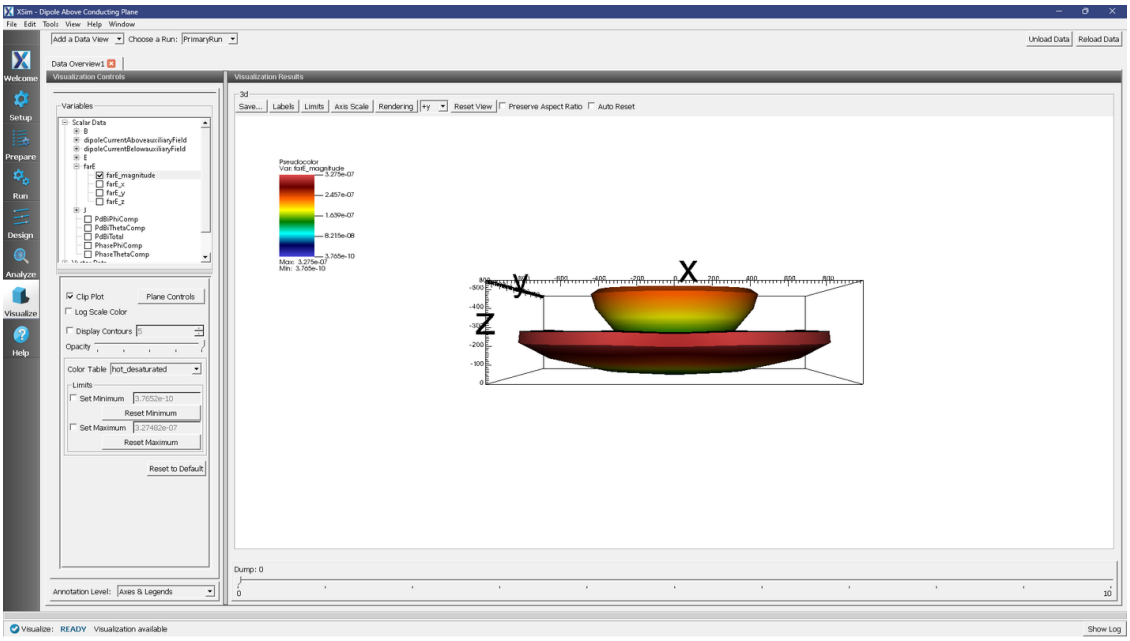

Fig. 2.29: The far field radiation pattern

Problem Description

Dipole antennas are the simplest and most widely used type of antenna. In the most basic setup, a dipole antenna is composed of an oscillating current/voltage source in between two electrodes. The frequency of the source will determine the wavelength of the electromagnetic radiation emitted from the antenna according to the dispersion relation

$$\lambda = \frac{c}{f}.$$

Most commonly, the electrodes will be 1/4 of the emitted wavelength. In this example, the antenna will be driven with a current oscillating with a frequency of 1 GHz. Therefore, the emitted wavelength will be roughly 30 cm, meaning we will make each of the electrodes 7.5 cm. This will make the total length of the antenna 15 cm, which is why dipole antennas are sometimes called half-wave antennas. It is easiest to drive the antenna when the electrodes are a quarter wavelength.

For more background information on dipole antennas, visit the Wikipedia page: https://en.wikipedia.org/wiki/Dipole_antenna

Opening the Simulation

The Dipole Antenna example is accessed from within XSimComposer by the following actions:

- Select the *New* → *From Example…* menu item in the *File* menu.
- In the resulting *Examples* window expand the *XSim for Electromagnetics* option.
- Expand the *Antennas* option.
- Select *Dipole Antenna* and press the *Choose* button.
- In the resulting dialog box, create a *New Folder* if desired, then press the *Save* button to create a copy of this example.

The resulting Setup Window is shown Fig. 2.30.

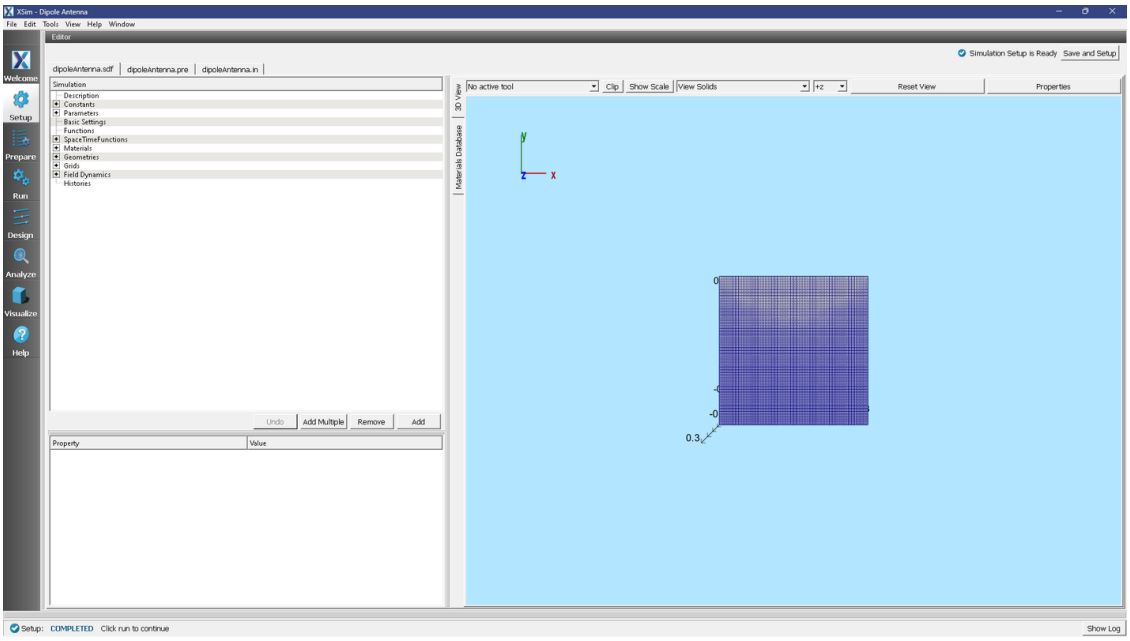

Fig. 2.30: Setup Window for the Dipole Antenna example.

Simulation Properties

In this simulation, we will excite the antenna and watch the dipole electromagnetic radiation emanate from the antenna. A *distributed current* source is used to apply the driving current. A volume for the current source and the functional form of the current is set under *Field Dynamics* → *CurrentDistriubtions* → *driveCurrent*. The user has the ability to set all three components of the current within the volume. In this example, we set the x-component of the current using the *driveCurrent* spacetime function. The *driveCurrent* function is a sine wave oscillating at 1 GHz to which a smooth turn on profile has been applied.

There are open boundaries on the walls of the simulation.

Running the Simulation

After performing the above actions, continue as follows:

- Proceed to the run window by pressing the Run button in the left column of buttons.
- Check that you are using these run parameters:
  - *Time Step*: 4.573853853672867e-12
  - *Number of Steps*: 1000
  - *Dump Periodicity*: 100
  - *Dump at Time Zero*: Checked
- Click on the *Run* button in the upper left corner of the right pane.

You will see the output of the run in the right pane. The run has completed when you see the output, “Engine completed successfully”. This result is shown in Fig. 2.31.

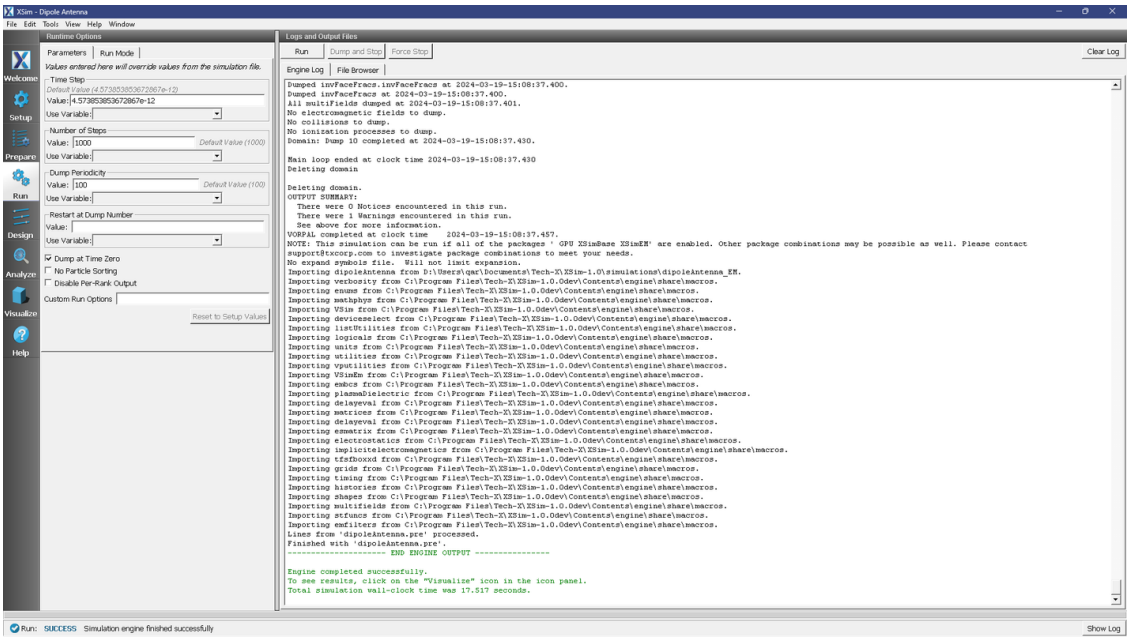

Fig. 2.31: The Run window at the end of execution.

Visualizing the Results

After performing the above actions, the results can be visualized as follows:

- Proceed to the Visualize Window by pressing the *Visualize* button in the navigation column.

1. Expand *Scalar Data*
2. Expand *E*
3. Select *E_y*
4. Check *Clip Plot*
5. Set *Set Minimum* to -0.1, and *Set Maximum* to 0.1.
6. Scroll through the dumps to see how the y-component of the electric field changes with time. The last dump is shown in Fig. 2.32.

Further Experiments

1. Add an RCS Box around the antenna to measure the far field radiation pattern at a location of your choosing.
2. Modify the driving frequency or the dimensions of the electrodes.

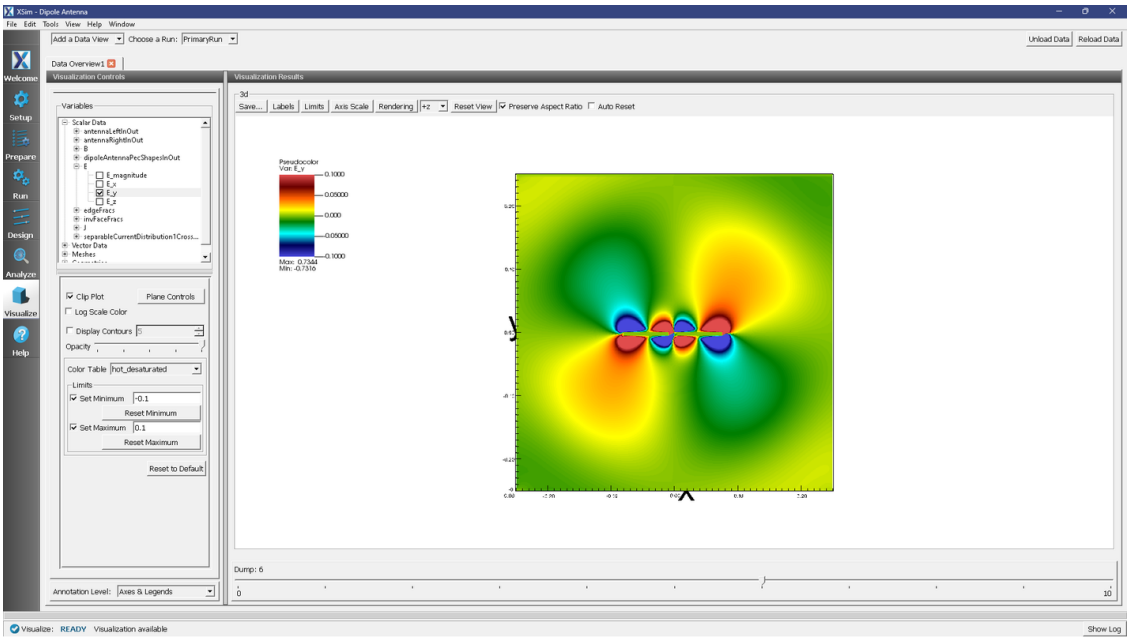

Fig. 2.32: The y-component of the Electric Field shows the expected 4-lobe pattern.

2.1.7 Dish Antenna (dishAntenna.sdf)

Keywords:

`electromagnetics, antennas`

Problem Description

The Dish Antenna simulation illustrates how to get the radiation pattern from a source in the presence of a complex shape.

Opening the Simulation

The Dish Antenna example is accessed from within XSimComposer by the following actions:

- Select the *New* → *From Example…* menu item in the *File* menu.
- In the resulting *Examples* window expand the *XSim for Electromagnetics* option.
- Expand the *Antennas* option.
- Select "Dish Antenna" and press the *Choose* button.
- In the resulting dialog, create a New Folder if desired, and press the *Save* button to create a copy of this example.

All of the properties and values that create the simulation are now available in the Setup Window as shown in Fig. 2.33. You can expand the tree elements and navigate through the various properties, making any changes you desire. The right pane shows a 3D view of the geometry, if any, as well as the grid, if actively shown. To show or hide the grid, expand the Grid element and select or deselect the box next to `Grid`.

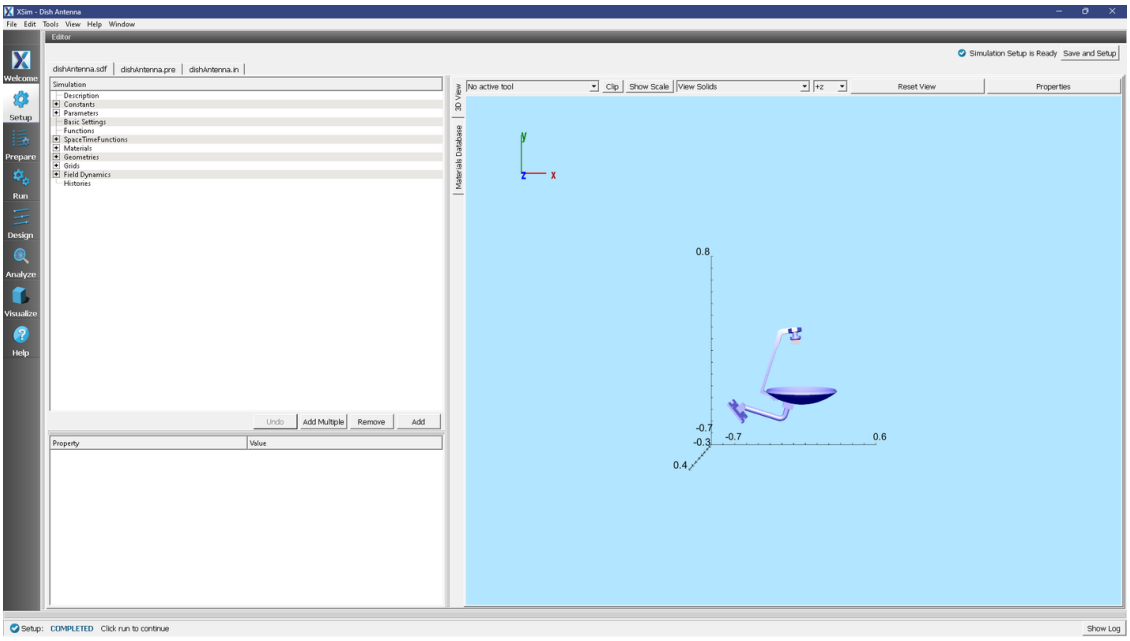

Fig. 2.33: Setup Window for the Dish Antenna example.

Simulation Properties

One can set the parameters of the grid and the source through the setup tree. The parameters are put under the Constants section.

Running the Simulation

After performing the above actions, continue as follows:

- Proceed to the run window by pressing the Run button in the left column of buttons.
- Check that you are using these run parameters:
  - *Time Step*: 9.147707707345734e-12
  - *Number of Steps*: 1000
  - *Dump Periodicity*: 50
  - *Dump at Time Zero*: Checked
- Click on the *Run* button in the upper left corner of the right pane.

You will see the output of the run in the right pane. The run has completed when you see the output, "Engine completed successfully". This result is shown in Fig. 2.34.

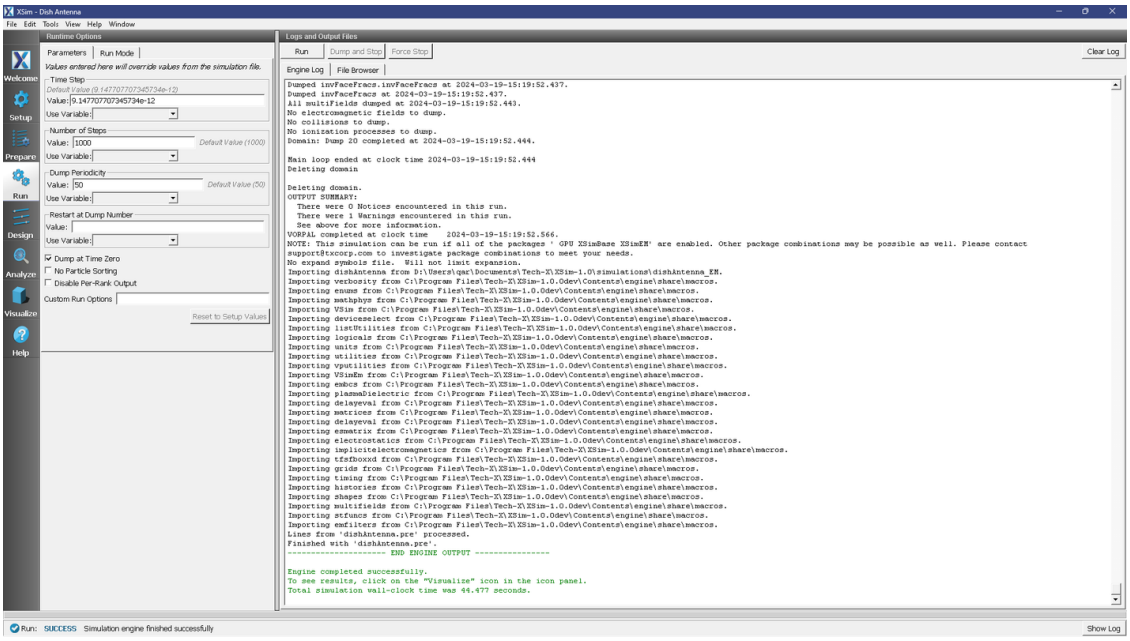

Fig. 2.34: The Run Window at the end of execution.

Visualizing the Results

After performing the above actions, continue as follows:

- Proceed to the Visualize Window by pressing the Visualize button in the left column of buttons.

To view the electric field reflected from the dish antenna as shown in Fig. 2.35, do the following:

- Expand *Scalar Data*
- Expand *E*
- Select *E_x*
- Check *Clip Plot*
- Expand *Geometries*
- Select *dishAntennaPecShapesTriangles*

It is easier to see the fields if you change the color scale minimum and maximum. To do so, check the *Set Minimum* and *Set Maximum* boxes, and set a fixed minimum of -2 and a fixed maximum of 2.

Move the slider at the bottom of the right pane to see the electric field at different times.

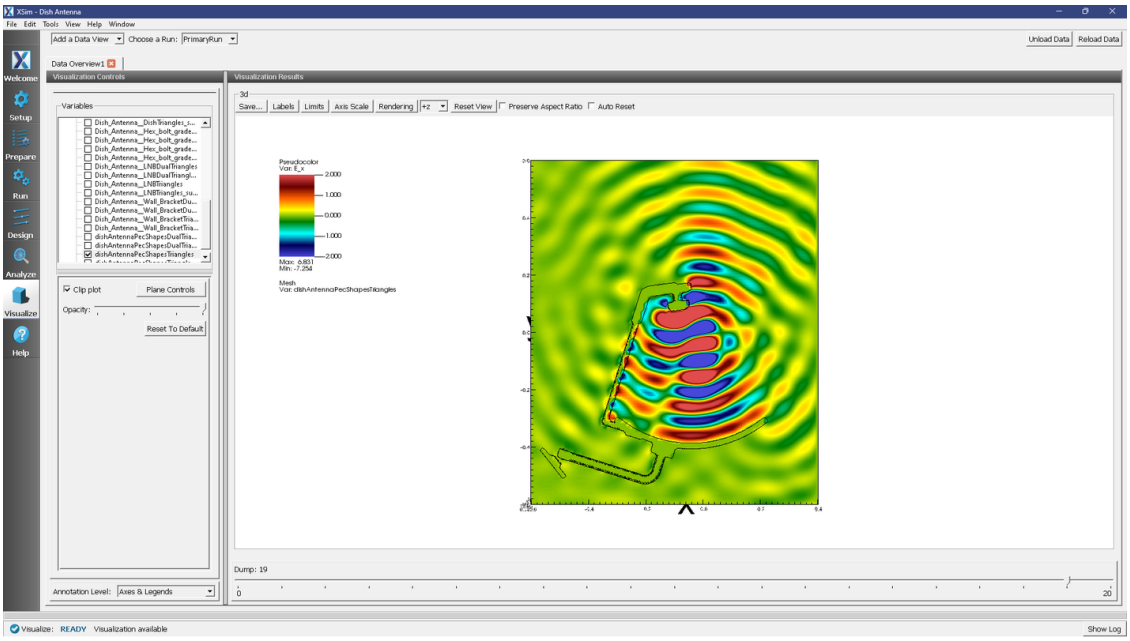

Fig. 2.35: Visualization of a slice of the electric field as a color contour plot at dump 19.

Further Experiments

Additional experiments worth investigating are:

- Change the resolution to see whether more resolution gives a different answer.
- Change the frequency of the source. Be careful, because at high frequencies with the chosen resolution, one will require a large amount of memory.

2.1.8 Half-Wave Dipole in Free Space (halfWaveDipoleAntenna.sdf)

Keywords:

`halfWaveDipoleAntenna, far field, radiation`

Problem Description

This problem illustrates how to obtain far field radiation patterns from XSim simulation data. The simulation itself consists of a half-wavelength long current source in free space.

Opening the Simulation

The Half Wave Dipole Antenna example is accessed from within XSimComposer by the following actions:

- Select the *New* → *From Example…* menu item in the *File* menu.
- In the resulting *Examples* window expand the *XSim for Electromagnetics* option.
- Expand the *Antennas* option.
- Select "Half-Wave Dipole in Free Space" and press the *Choose* button.
- In the resulting dialog, create a New Folder if desired, and press the *Save* button to create a copy of this example.

All of the properties and values that create the simulation are now available in the Setup Window as shown in Fig. 2.36. You can expand the tree elements and navigate through the various properties, making any changes you desire. The right pane shows a 3D view of the geometry, if any, as well as the grid, if actively shown. To show or hide the grid, expand the Grid element and select or deselect the box next to `Grid`.

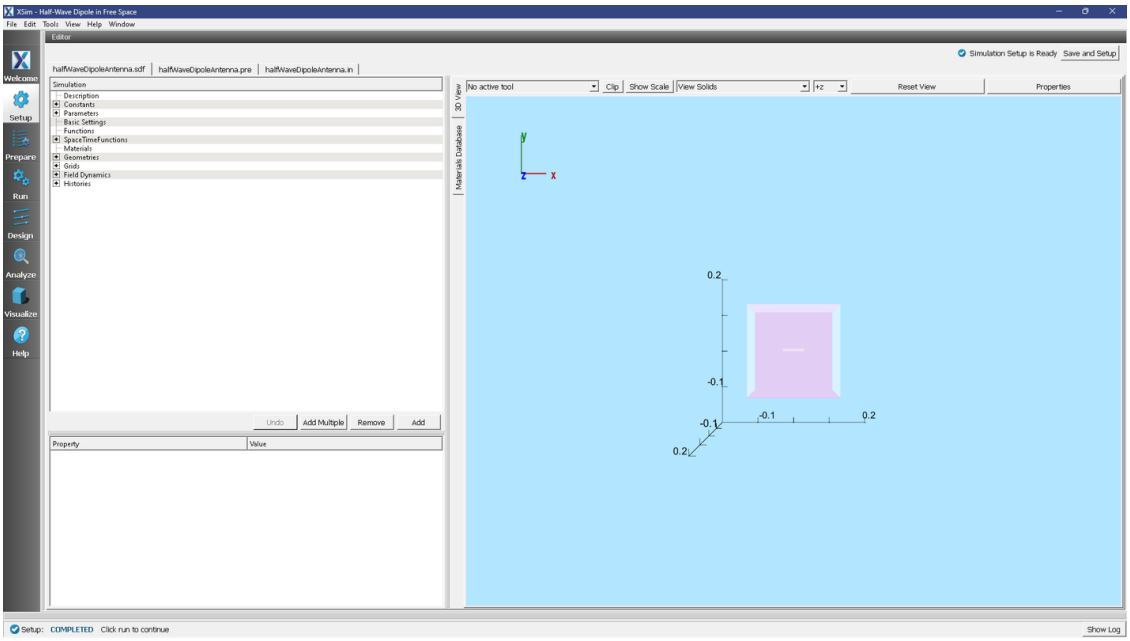

Fig. 2.36: Setup Window for the Half Wave Dipole Antenna example.

Simulation Properties

This example includes *Constants* for easy adjustment of simulation properties, Including:

- AMPLITUDE: The amplitude of the signal
- FREQUENCY: The frequency of the antenna

There are also *SpaceTimeFunctions* to define the current driver of the half wavelength source.

Other properties of the simulation include open boundaries on all sides. A Distributed Current source is used to set the current of the half wavelength antenna.

Running the Simulation

After performing the above actions, continue as follows:

- Proceed to the run window by pressing the Run button in the left column of buttons.
- Check that you are using these run parameters:
  - *Time Step*: 9.147707707345834e-12
  - *Number of Steps*: 4000
  - *Dump Periodicity*: 100
  - *Dump at Time Zero*: Checked
- Click on the *Run* button in the upper left corner of the right pane.

You will see the output of the run in the right pane. The run has completed when you see the output, "Engine completed successfully". This result is shown in Fig. 2.37.

- NOTE: the correct elements will not appear in the visualization step if the analysis step has not been performed first.

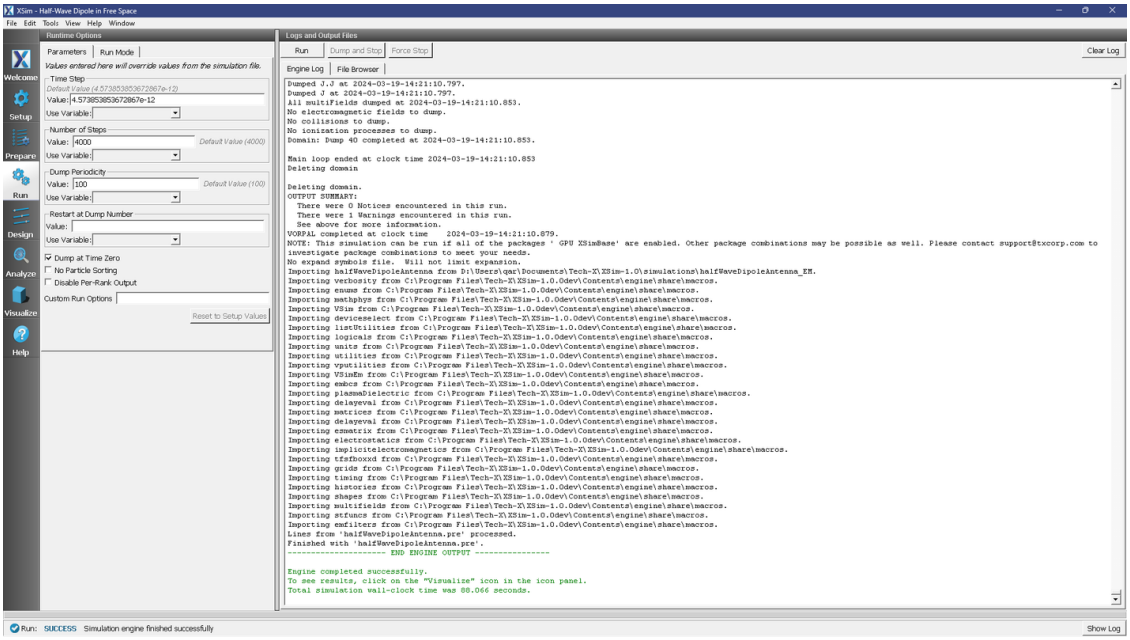

Fig. 2.37: The Run Window at the end of execution.

Analyzing the Results

After performing the above actions, continue as follows:

- Proceed to the Analysis window by pressing the Analyze button in the left column of buttons.
- Select *computeFarFieldFromKirchhoffBox.py (default)*. Then click *Open*.
- For this example, edit the following input parameters:
  - simulationName - halfWaveDipoleAntenna (name of the input file)
  - fieldLabel - E (name of the electromagnetic field)
  - farFieldRadius - 10.0 (radius of the far sphere, i.e., distance to the far zone)
  - numPeriods - 0.25
  - numFarFieldTimes - 2
  - frequency - 3.0e9
  - numTheta - 45 (number of points in the theta direction)
  - numPhi - 90 (number of points in the phi direction)
  - zeroThetaDirection - (0,0,1) (determines orientation of far field coordinate system)
  - zeroPhiDirection - (1,0,0) (determines orientation of far field coordinate system
  - incidentWaveDirection - (0,0,0)
  - incidentWaveAmplitude - blank
  - varyingMeshMaxRadius - 10.0
  - principalPlanesOnly - checked
- Click the *Analyze* button in the upper right corner.

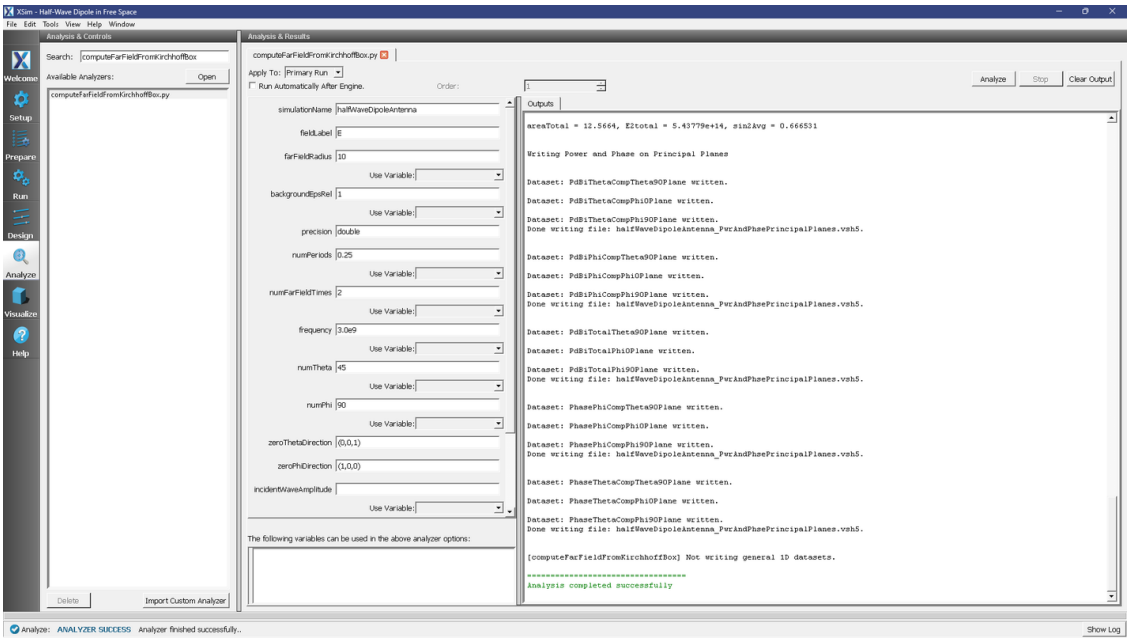

Fig. 2.38: The Analysis window at the end of execution.

Visualizing the Results

After performing the above actions, continue as follows:

- Proceed to the Visualize Window by pressing the Visualize button in the left column of buttons.

The far field radiation pattern can be found in the scalar data variables of the data overview tab:

- Expand *Scalar Data*
- Expand *farE*
- Select *farE_magnitude*
- Move the dump slider forward in time to see the evolution
- Click and drag to rotate the image

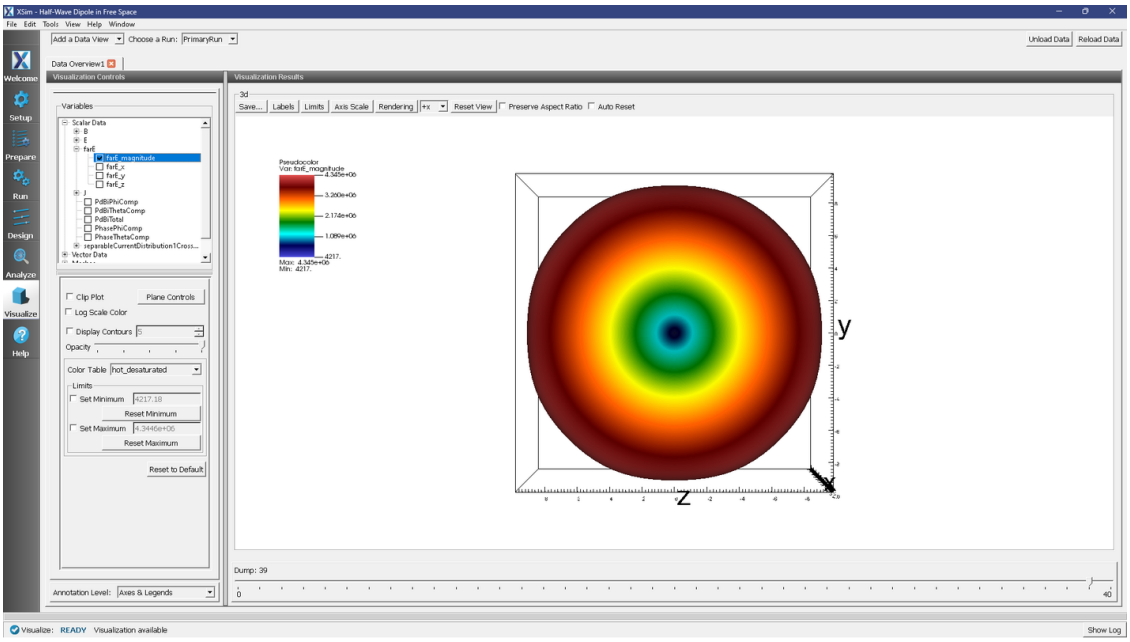

Fig. 2.39: The far field radiation pattern

Further Experiments

The resolution of the far field pattern can be changed by editing the number of theta, phi, and sphere points in the far field history.

Try implementing a conducting plane to see how it affects the far field.

If the Simulation domain is made too small, the results will be distorted as the entire near field must be within the simulation domain in order to achieve a proper transformation to the far field.

2.1.9 Horn Antenna (hornAntenna.sdf)

Keywords:

sectoral, horn antenna, far field, radiation

Problem description

This example illustrates how to obtain the far field radiation pattern of a sectoral horn antenna. A horn antenna consists of a flaring metal waveguide shaped like a horn that directs radio waves into a beam. Horns are widely used as antennas at UHF and microwave frequencies. A sectoral horn is only flared along one axis, the other horn axis has constant width and is equivalent to the width of the waveguide. Sectoral horns produce a fan shaped beam, wider in the plane of the narrow sides.

Opening the Simulation

The Horn Antenna example is accessed from within XSimComposer by the following actions:

- Select the *New* → *From Example…* menu item in the *File* menu.
- In the resulting *Examples* window expand the *XSim for Electromagnetics* option.
- Expand the *Antennas* option.
- Select "Horn Antenna" and press the *Choose* button.
- In the resulting dialog, create a New Folder if desired, and press the *Save* button to create a copy of this example.

The *Setup Window* is shown Fig. 2.40. One can click and unclick the grid, the farFieldBox0 in the histories, the current distribution, and so forth to see where those objects are. One can change locations through changing the values under Constants or, in some cases, the numbers directly in the objects.

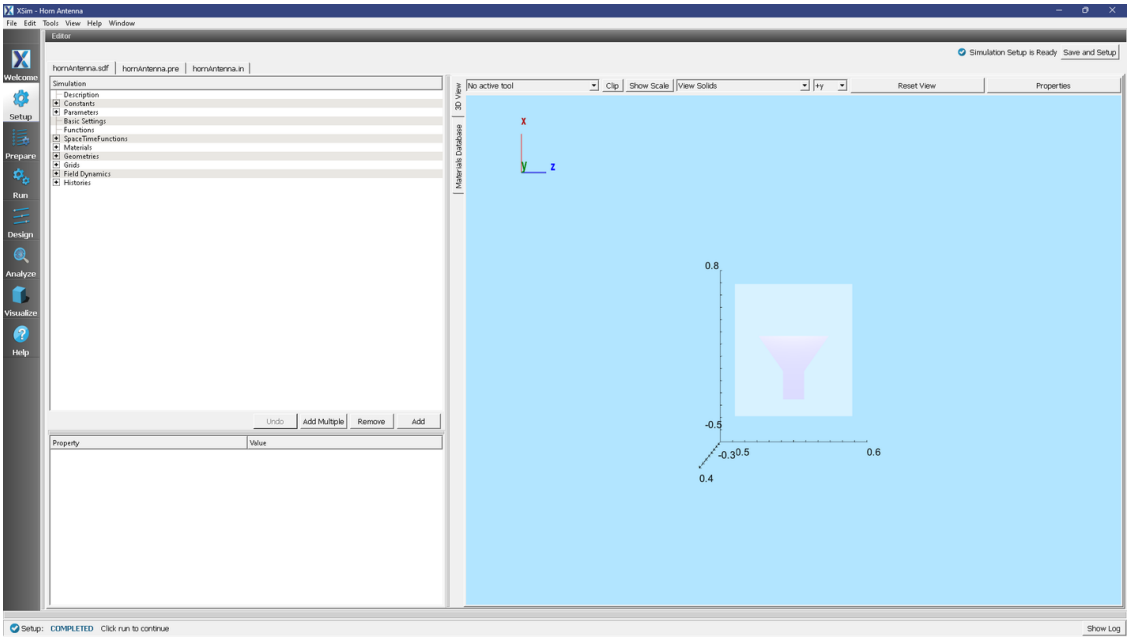

Fig. 2.40: Setup Window for the Horn Antenna example.

Simulation Properties

The antenna geometry in this example has been set up using CSG in the graphical setup interface. The dimensions of the antenna can be adjusted by tuning the sizes of the various wedges and cubes used in the antenna's construction. Under *Constants*, the wavelength may be modified, as well as the grid size and resolution. The polarization of the antenna may be altered by going into *CurrentDistributions* and changing the components of the driving current source.

There are two scales that we need to resolve in this simulation. One is the wavelength and one is the smallest geometric scale to resolve (i.e. in this simulation it is the wall width). So, NX, NY, and NZ have been set up to resolve both scales.

Running the Simulation

After performing the above actions, continue as follows:

- Proceed to the run window by pressing the Run button in the left column of buttons.
- Check that you are using these run parameters:
  - *Time Step*: 8.928466081792827e-12
  - *Number of Steps*: 1300
  - *Dump Periodicity*: 100
  - *Dump at Time Zero*: Checked
- Click on the *Run* button in the upper left corner of the right pane.

You will see the output of the run in the right pane. The run has completed when you see the output, "Engine completed successfully". This result is shown in Fig. 2.41.

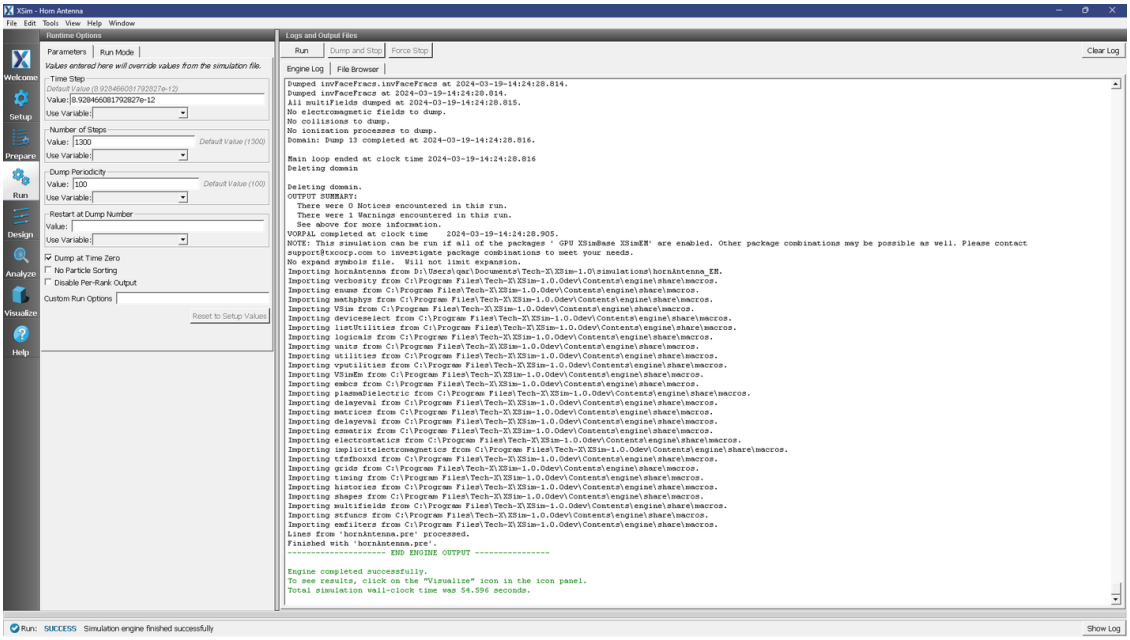

Fig. 2.41: The Run Window at the end of execution.

Analyzing the Results

After performing the above actions, continue as follows:

- Proceed to the Analysis window by pressing the Analyze button in the left column of buttons.
- In the resulting dialog, select computeFarFieldFromKirchhoffBox.py and press Open.
- Input values for the analyzer parameters. The analyzer may be run multiple times, allowing the user to experiment with different values.
  - simulationName - hornAntenna (name of the input file)
  - fieldLabel - E (name of the electric field)
  - farFieldRadius - 10.0 (distance to far field in m, 10.0 is a good value)
  - numPeriods - 0.25
  - numFarFieldTimes - 2
  - frequency - 2.0e9
  - numTheta - 45 (number of theta points in the far field, 18 for a quick calculation, 45 for finer resolution)
  - numPhi - 90 (number of phi points in the far field, 36 for a quick calculation, 90 for finer resolution)
  - zeroThetaDirection - (0,0,1) (determines orientation of far field coordinate system)
  - zeroPhiDirection - (1,0,0) (determines orientation of far field coordinate system
  - incidentWaveAmplitude - blank
  - incidentWaveDirection - (0,0,0)
  - varyingMeshMaxRadius - 10.0
  - principalPlanesOnly - checked
- Click Analyze
- The analysis is completed when you see "Analysis completed successfully" in the Outputs. Depending on the values of numTheta, numPhi, and timeStepStride, the script may need to run for several minutes or longer.

Visualizing the Results

Under *Scalar Data* plot $E_{magnitude}$. To slice inside the horn, select *Clip Plot* in the lower left hand corner. Click on *Plane Controls* and change the cut-plane normal to lie along Y instead of Z. Move the dump slider to view the electric field emanating from the horn. You can get a better look by adjusting the color scale. Select *Log Scale Color* in the lower left hand corner. Try adjusting the min and max until the signal is well resolved (see Fig. 2.43).

The far field radiation pattern can be found in the *Scalar Data* variables of the *Data Overview* tab. Open the farE tree element and check the *farE_magnitude* box. The far field mesh can also be plotted; it can be found under *Geometries*.

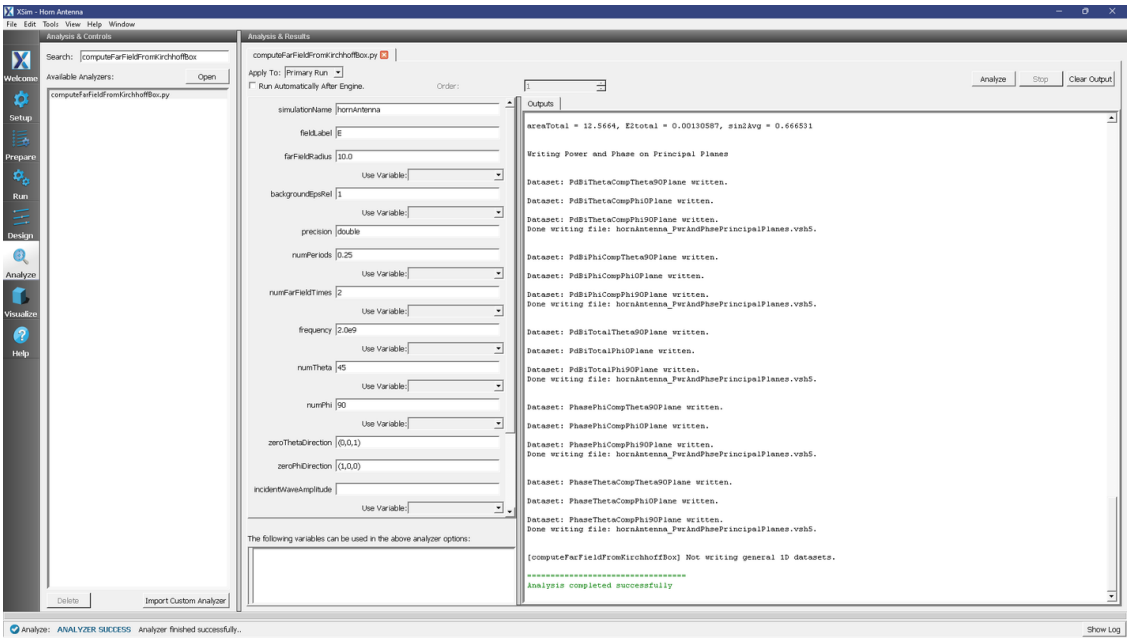

Fig. 2.42: The **Analyze** panel after running computeFarFieldFromKirhhoffBox.py.

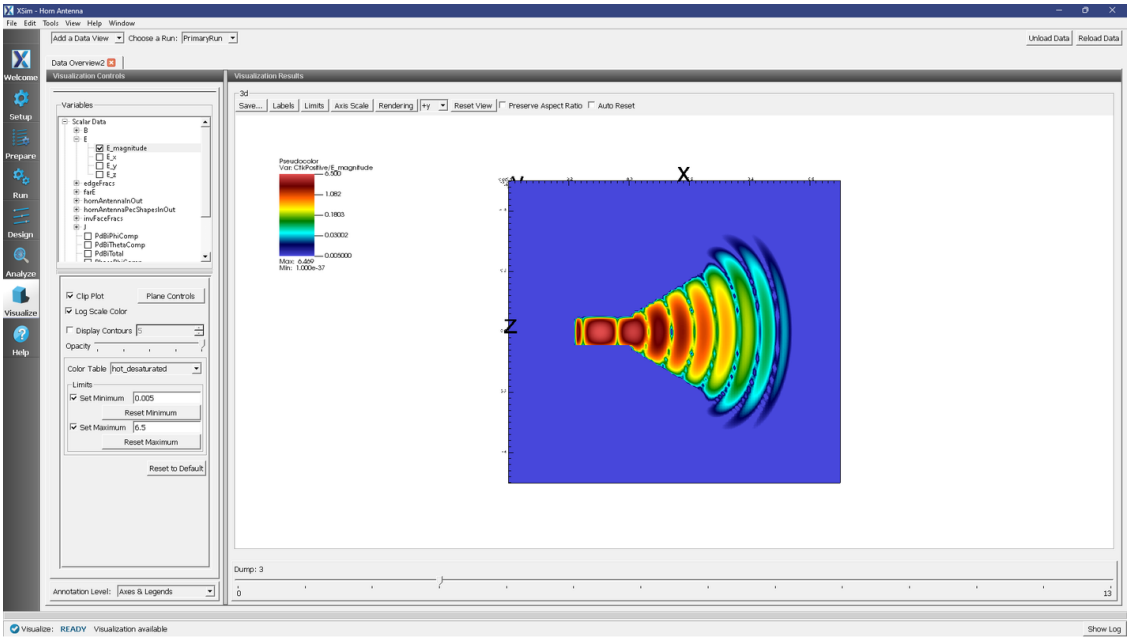

Fig. 2.43: The $E_{magnitude}$ field propagating out of the horn at dump 3. The color scale has been log scaled and the min and max have been fixed to 0.005 and 6.5, respectively. The optimal min/max values will depend on the dump selected. The view has been rotated to show the x-z plane.

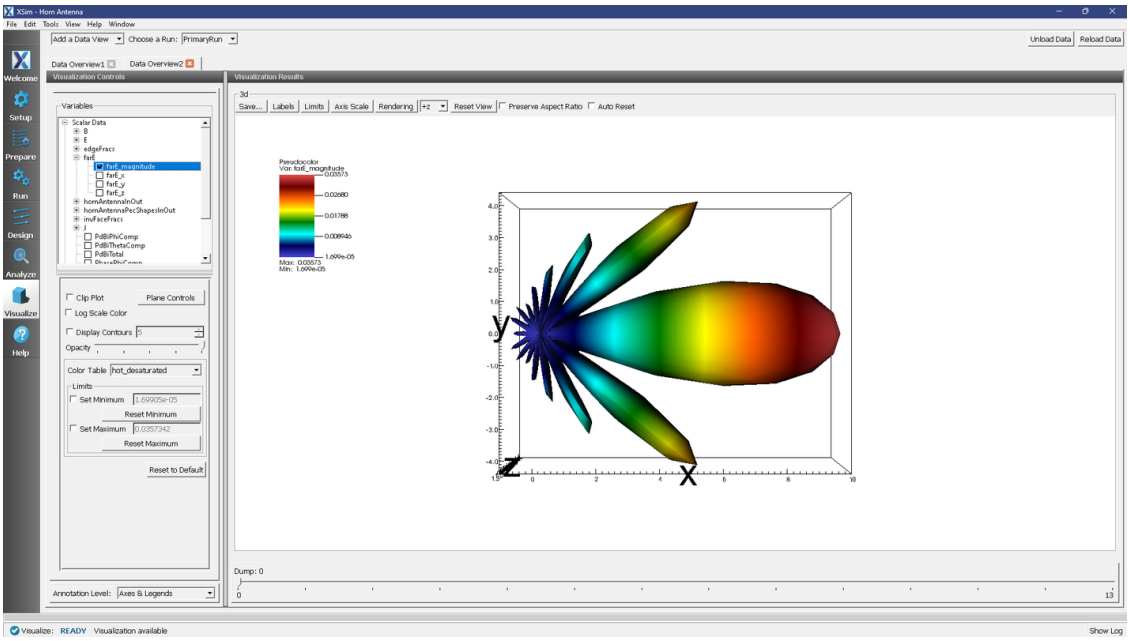

Fig. 2.44: The far field radiation pattern.

Further Experiments

The physical dimensions of the pyramidal horn can be modified in the GUI.

To turn the antenna into an E-plane sectoral horn, try changing the polarization to lie along the flared direction (z).

Try experimenting with different far field resolutions by changing the values of numTheta and numPhi during the *Analyze* step. You can also experiment with different far field distances by changing the value of farFieldRadius.

Try making the domain and the size of the Kirchhoff box larger or smaller (size of the Kirchhoff box is tied to the domain size by default). If the simulation domain is made too small, the results may appear distorted because the entire near field must be resolved within the simulation domain in order to achieve a proper transformation to the far field.

2.1.10 Loop Antenna from a Coaxial Cable (coaxialLoopAntenna.sdf)

Keywords:

`coaxial, coaxial waveguide, coaxial cable`

Problem description

This example illustrates how to use the coaxial cable Field Boundary Condition and Constructive Solid Geometry to create a coaxial loop antenna.

Opening the Simulation

The Coaxial Loop Antenna example is accessed from within XSimComposer by the following actions:

- Select the *New* → *From Example…* menu item in the *File* menu.
- In the resulting *Examples* window expand the *XSim for Electromagnetics* option.
- Expand the *Antennas* option.
- Select "Loop Antenna From a Coaxial Cable" and press the *Choose* button.
- In the resulting dialog, create a New Folder if desired, and press the *Save* button to create a copy of this example.

The *Setup Window* is shown Fig. 2.45.

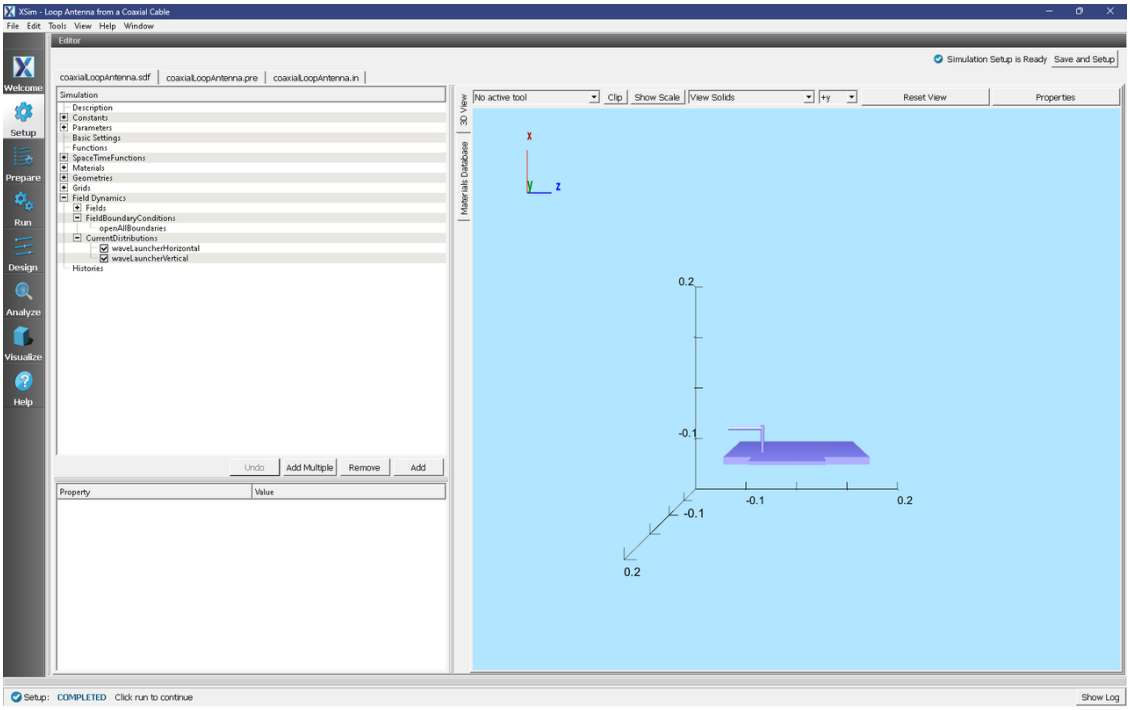

Fig. 2.45: Setup Window for the Coaxial Loop Antenna example.

Simulation Properties

This simulation makes use of the new coaxial waveguide Field Boundary Condition in XSim 8.1.

A coaxial waveguide is first constructed by creating a physical coaxial cable that enters the simulation domain. It is very important that this cable exists from at least 1 cell outside of the simulation boundary to 1 cell inside the simulation boundary. This is done by first creating a box primitive and setting it along the desired simulation boundary.

A cylinder corresponding to the outer diameter of the coaxial cable is then created, and subtracted from the plate.

A cylinder corresponding to the inner diameter of the coaxial cable is then created and extended into the simulation space.

It is then made into a loop antenna by adding a second, intersecting cylinder.

The wave itself is specified by a Field Boundary Condition.

Running the Simulation

After performing the above actions, continue as follows:

- Proceed to the run window by pressing the Run button in the left column of buttons.
- Check that you are using these run parameters:
  - *Time Step*: 1.8295415414691 47e-12
  - *Number of Steps*: 400
  - *Dump Periodicity*: 20
  - *Dump at Time Zero*: Checked
- Click on the *Run* button in the upper left corner. You will see the output of the run in the right pane.

The run has completed when you see the output, "Engine completed successfully."

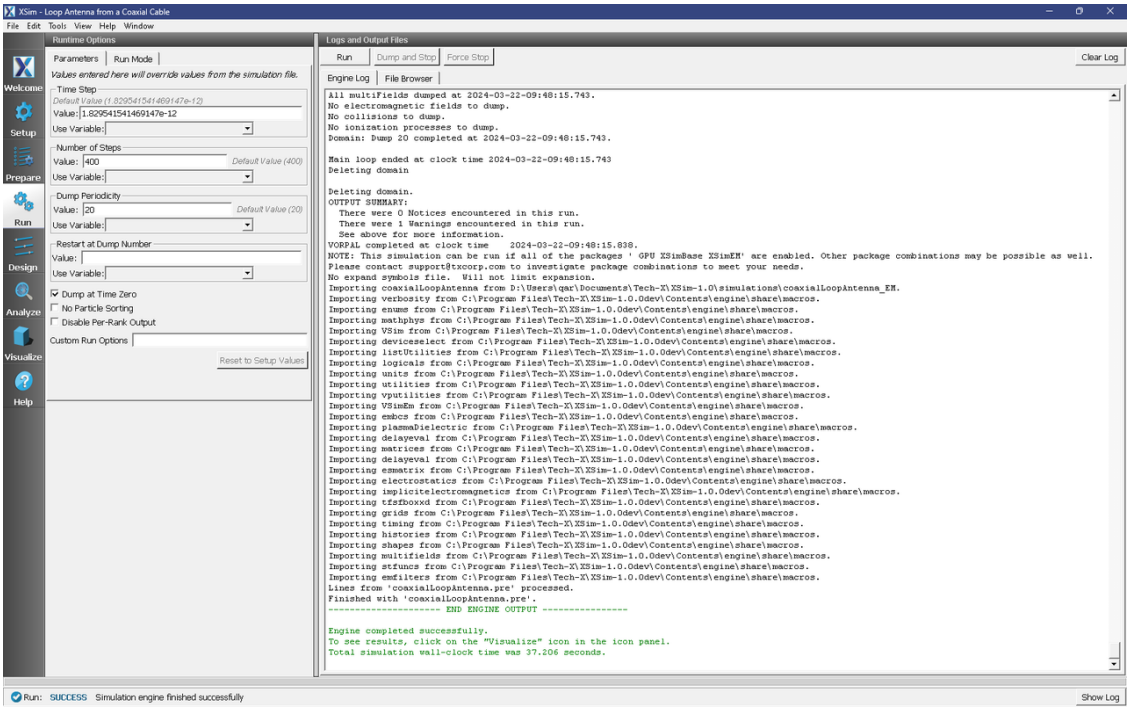

Fig. 2.46: The Run Window at the end of execution.

Visualizing the Results

After the run completes, the field may be visualized:

- Proceed to the Visualize Window by pressing the Visualize button in the left column of buttons.
- Expand *Scalar Data*
- Expand *E*,
- Select *E_z*.
- check *Clip Plot* in the lower left hand corner,
- Click on *Plane Controls* and change the clip plane normal to Y instead of Z.

- Adjust the origin of the normal vector from Y = 0 to Y = 0.05.
- Check the *Set Minimum* and *Set Maximum* boxes
- Set the minimum to -10 and the maximum to 10.
- Drag the *Dump* slider to the far right for dump 20.
- Finally, click and drag the visualization to rotate it so that you can see the field.

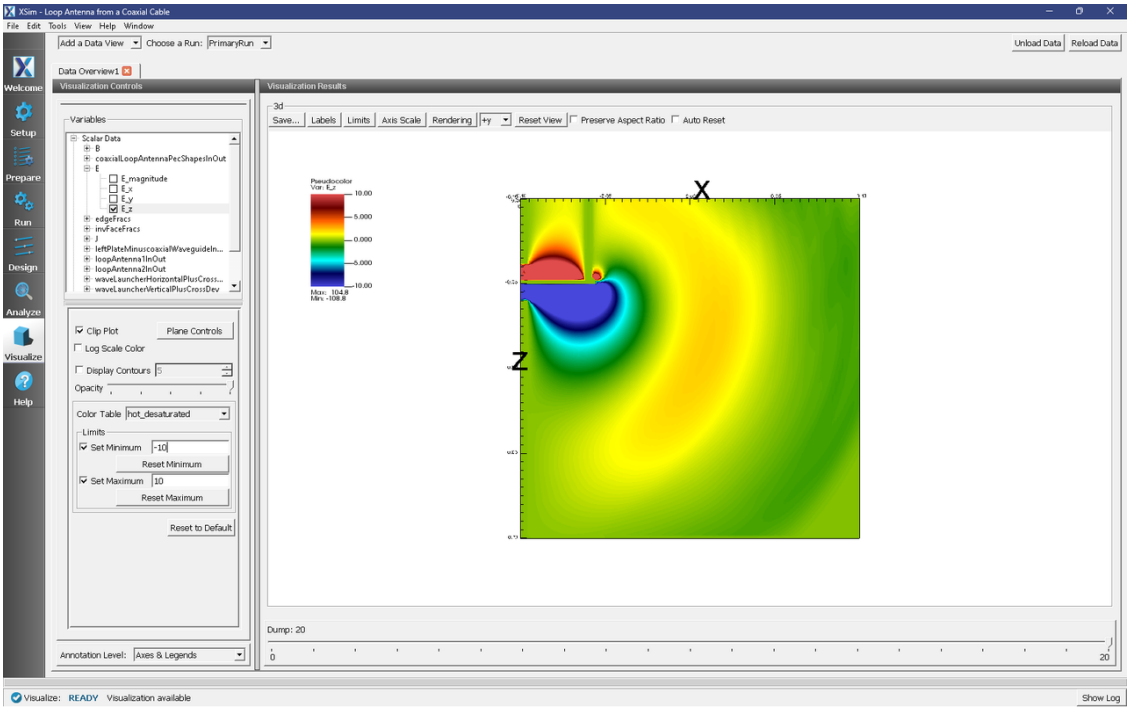

Fig. 2.47: The E_z field propagating off of the loop antenna.

2.1.11 Patch Antenna Far Field (patchAntennaFarField.sdf)

Keywords:

`patchAntenna, far field, radiation`

Problem Description

This problem takes the same patch antenna from the *Patch Antenna* example (currently text-based only, visual setup coming soon) and modifies it to calculate the far-field radiation pattern. It is fed with a 5.5GHz source on a microstrip feed line. The patch itself is mounted on a dielectric substrate made of alumina.

Opening the Simulation

The Patch Antenna example is accessed from within XSimComposer by the following actions:

- Select the *New* → *From Example…* menu item in the *File* menu.
- In the resulting *Examples* window expand the *XSim for Electromagnetics* option.
- Expand the *Antennas* option.
- Select "Patch Antenna with Far Fields" and press the *Choose* button.
- In the resulting dialog, create a new folder if desired, and press the *Save* button to create a copy of this example.

The **Setup** window is now shown with all the implemented physics and geometries. See Fig. 2.48.

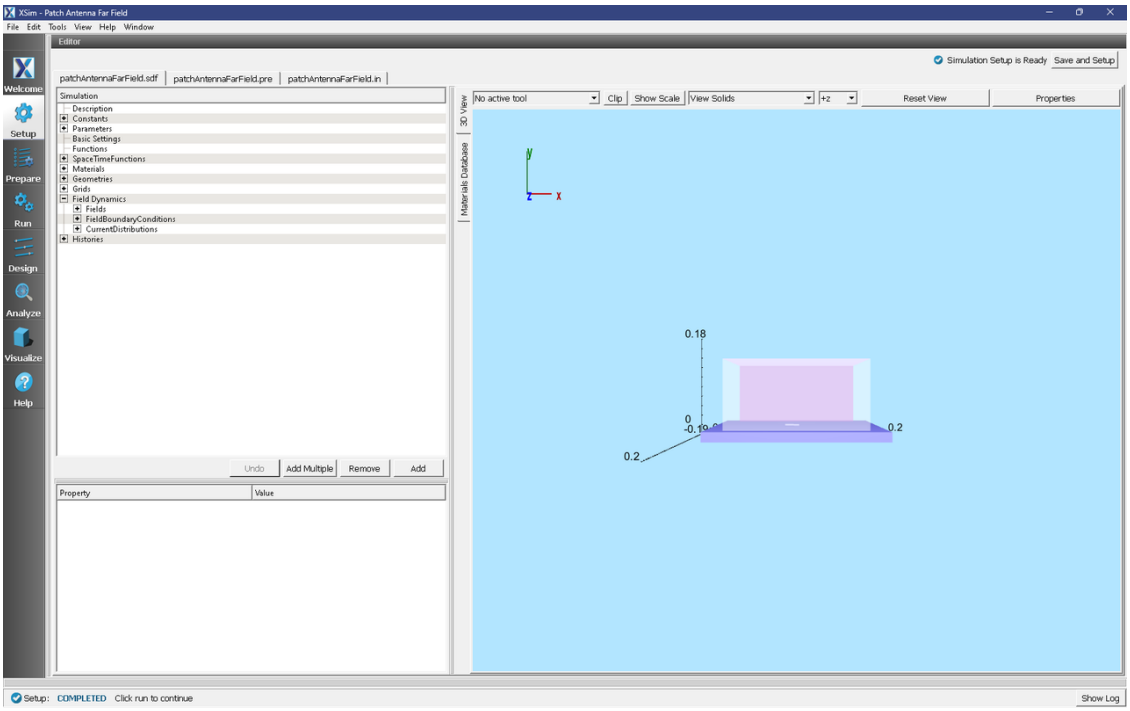

Fig. 2.48: Setup Window for the Patch Antenna example.

Simulation Properties

This file allows the modification of antenna operating frequency, dimensions, orientation, simulation domain size.

Running the Simulation

After performing the above actions, continue as follows:

- Proceed to the run window by pressing the Run button in the left column of buttons.
- Check that you are using these run parameters:
  - *Time Step*: 1.877687371507808e-12
  - *Number of Steps*: 2343
  - *Dump Periodicity*: 100

  - *Dump at Time Zero*: Checked

- Click on the *Run* button in the upper left corner of the right pane.

You will see the output of the run in the right pane. The run has completed when you see the output, "Engine completed successfully". This result is shown in Fig. 2.49.

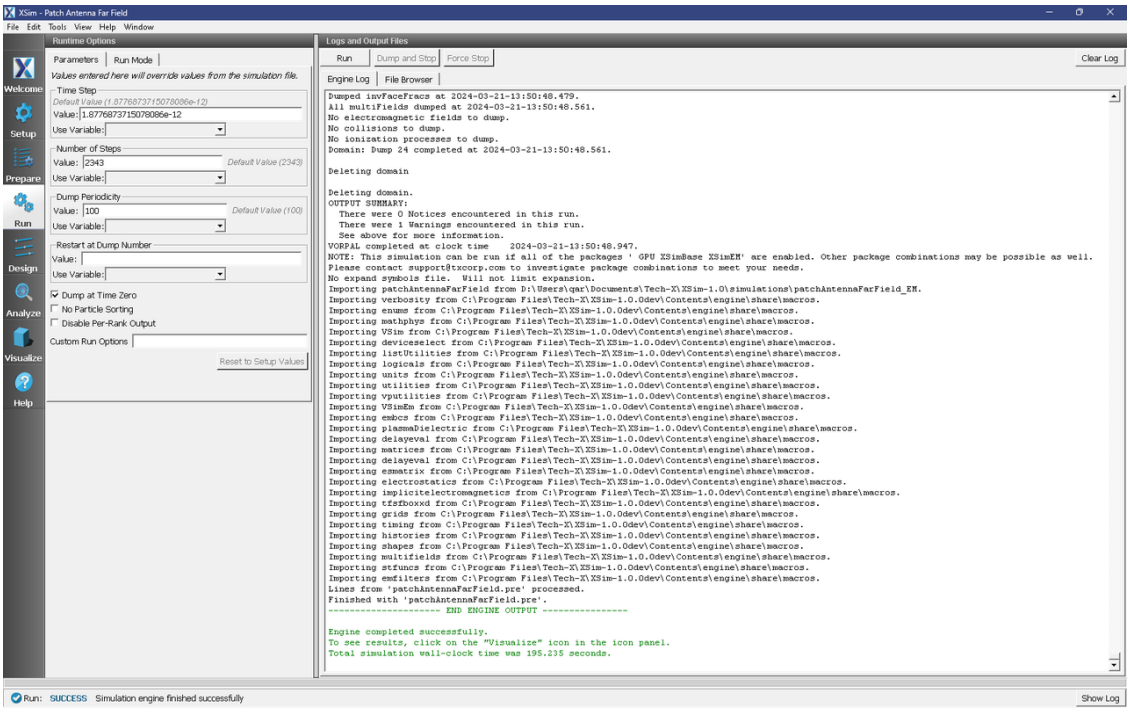

Fig. 2.49: The Run Window at the end of execution.

Analyzing the Results

After performing the above actions continue as follows to compute the far field radiation pattern:

- Proceed to the Analysis Window by pressing the *Analyze* button in the navigation column.
- In the resulting list, select *computeFarFieldFromKirchhoffBox* and press *Open*
- The analyzer fields should be filled as below:
  - *simulationName*: patchAntennaFarField
  - *fieldLabel*: E
  - *farFieldRadius*: 1024.0
  - *backgroundEpsRel*: 1
  - *precision*: double
  - *numPeriods*: 0.25
  - *numFarFieldTimes*: 2
  - *frequency*: 5.5e9
  - *numTheta*: 45

  - *numPhi*: 60
  - *zeroThetaDirection*: (0,0,1)
  - *zeroPhiDirection*: (1,0,0)
  - *incidentWaveAmplitude*: blank
  - *incidentWaveDirection*: (0,0,0)
  - *varyingMeshMaxRadius*: 1024.0
  - *principalPlanesOnly*: checked
- Click *Analyze* in the top right corner.
- The analysis is completed when you see the output shown in Fig. 2.50.

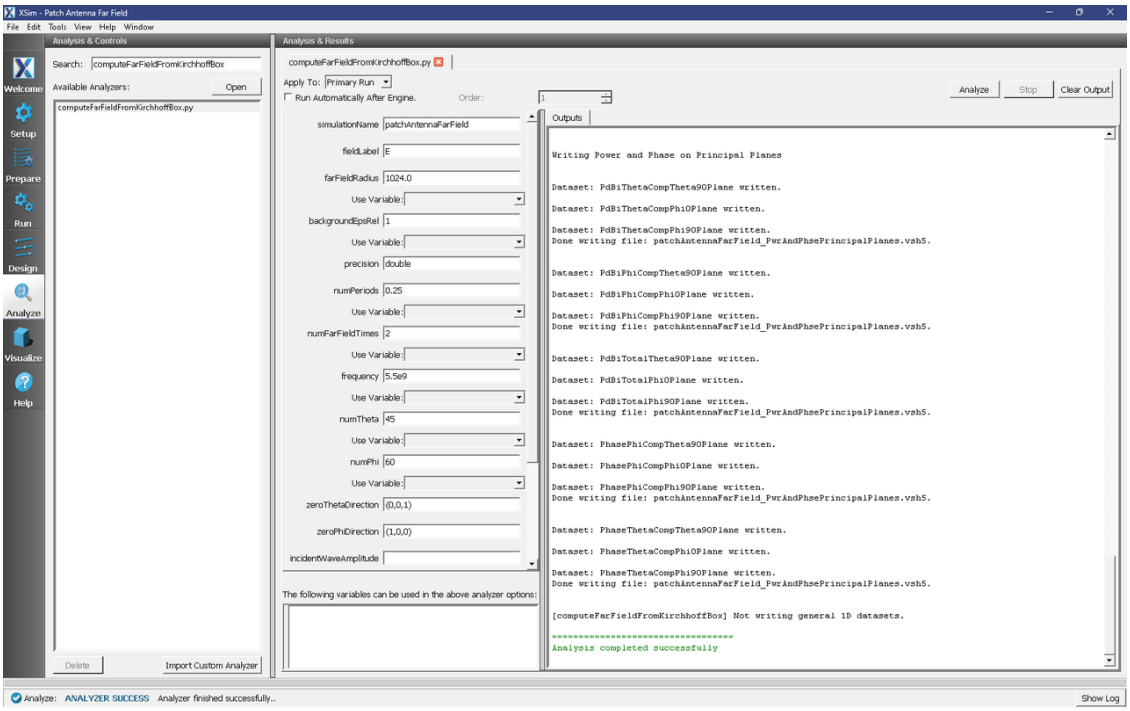

Fig. 2.50: Add the computeFarFieldFromKirhhoffBox.py script to your simulation.

Visualizing the Results

The far field radiation pattern can be found in the *Scalar Data* variables of the *Data Overview* tab. Check the *farE* box. The far field mesh can also be plotted; it can be found under *Geometries*.

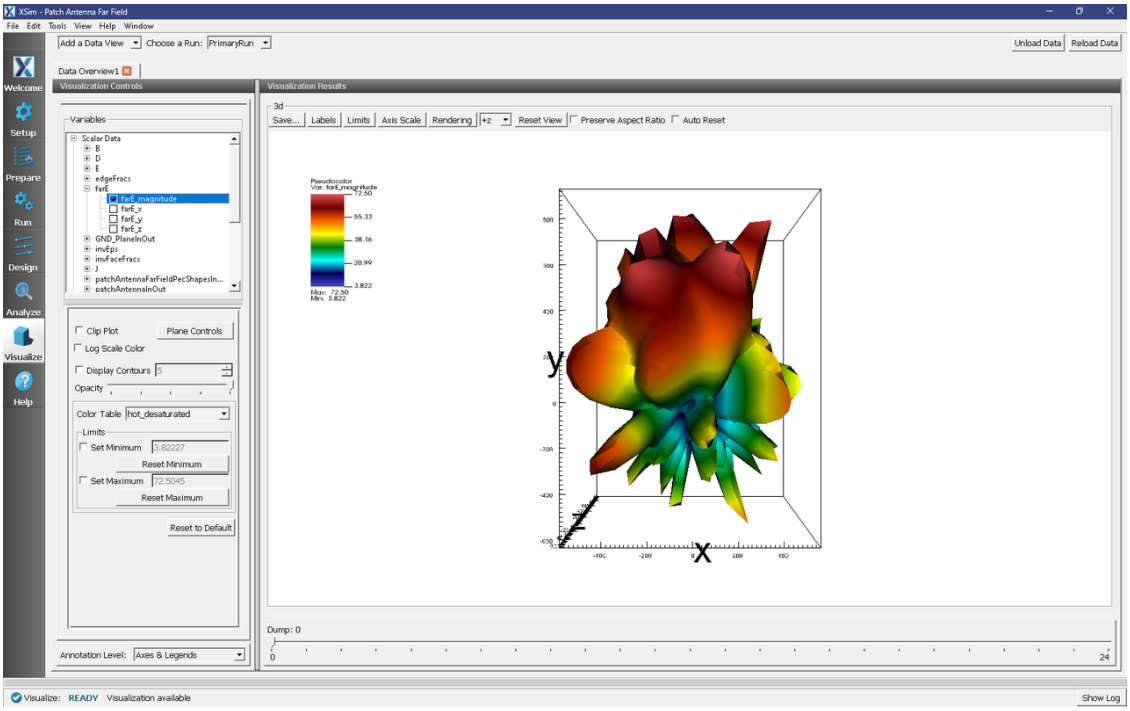

Fig. 2.51: The Far Field Radiation Pattern

Further Experiments

The physical dimensions of the patch can be modified to turn it into any rectangular patch. This model can in fact be used to simulate any form of patch antenna, simply modify the geometry in the Setup Window by expanding the *Parameters* tree node and adjusting the values of PATCH_WIDTH, PATCH_LENGTH, PATCH_THICKNESS, FEED_WIDTH, FEED_LENGTH, and FEED_OFFSET. The thickness of the alumina die may also be adjusted by modifying DIE_THICKNESS.

2.2 Cavities and Waveguides

2.2.1 Cylindrical Waveguide (cylindricalWaveguide.sdf)

Keywords:

`electromagnetics, waveguide, dispersion`

Problem Description

This XSimEM example illustrates how to find the modes of a cylindrical waveguide.

Simulation Properties

A section of cylindrical waveguide is simulated with the goal of extracting its propagating mode frequencies. The simulation is only two cells wide in X, but through the use of a phase-shifting periodic boundary condition, a much longer waveguide is simulated. The modes are extracted for longitudinal k-vectors, $\frac{2\pi n}{L_x}$. The maximum current is $I_0 = I\left(\tau/2\right)$. The waveguide is first excited with a transverse current that is off axis so as to excite modes of any symmetry. The temporal excitation is chosen to excite only a range of frequencies, from somewhat below the lowest cutoff up to the modes corresponding to $n = 1$. The Fourier transform of a history recording the electric field shows a clean output with a modest number of modes. Precise values for those frequencies can be obtained using the extractModes analyzer.

Opening the Simulation

The Cylindrical Waveguide example is accessed from within XSimComposer by the following actions:

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *XSim for Electromagnetics* option.
- Expand the *Cavities and Waveguides* option.
- Select *Cylindrical Waveguide* and press the *Choose* button.
- In the resulting dialog, create a New Folder if desired, and press the *Save* button to create a copy of this example.

All of the properties and values that create the simulation are now available in the Setup Window as shown in Fig. 2.52. You can expand the tree elements and navigate through the various properties, making any changes you desire. The right pane shows a 3D view of the geometry, if any, as well as the grid, if actively shown. (To show or hide the grid, expand the Grid element and select or deselect the box next to `Grid`.) For the current view, the setup has been rotated to be able to see down the waveguide, and the view of the grid has been turned off. The box inside the waveguide is the location of the current source that will drive the waveguide.

The sinc hat function is used to excite this example. This function has a Fourier spectrum that is fairly flat for $f_l < f < f_h$ and falls off rapidly over a frequency width of δ_f, so that it is nearly zero for $f < f_l - \delta_f$ or $f < f_h + \delta_f$. δ_f is automatically calculated by the sinc hat function based on the suppression factor and frequency gap factor. This excitation gives a range of modes to be analyzed.

Running the Simulation

After performing the above actions, continue as follows:

- Proceed to the run window by pressing the Run button in the left column of buttons.
- Check that you are using these run parameters:
  - *Time Step*: 5.168041416871939e-12
  - *Number of Steps*: 20000
  - *Dump Periodicity*: 2000
  - *Dump at Time Zero*: Checked
- Click on the *Run* button in the upper left corner of the right pane.

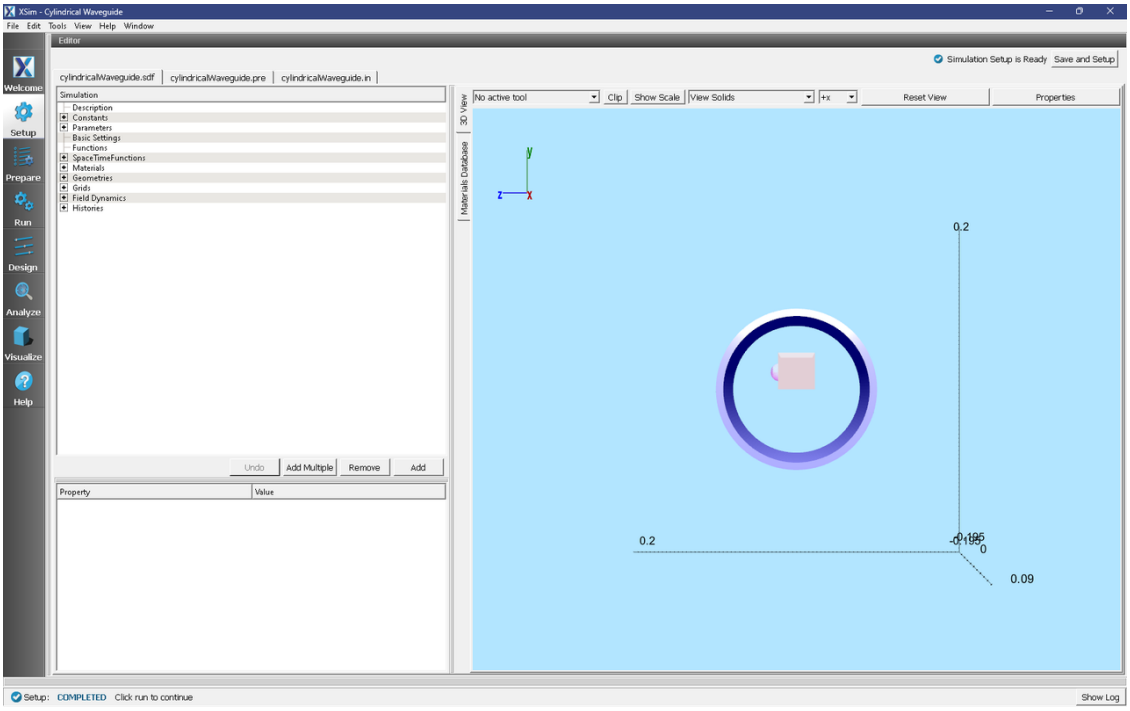

Fig. 2.52: Initial Setup Window for the Cylindrical Waveguide example.

You will see the output of the run in the right pane. The run has completed when you see the output, "Engine completed successfully". This result is shown in Fig. 2.53.

Visualizing the Spectrum

After performing the above actions, continue as follows:

- Proceed to the Visualize Window by pressing the Visualize icon in the left panel.
- Select History under Data View. The first History tab will have eMid_0 and eMid_1 plotted. Open a second History tab in order to plot eMid_2 as well.
- Then for each plot select the Fourier Amplitude (dB) checkbox
- In the upper right corner of each plot, select Limits and set X-Axis max to 2e9.
- The result should be that shown in Fig. 2.54.

One can see the TM mode in this spectrum. One can measure the mode frequency by projecting the spectrum down on the axis. With this simulation of 20,000 steps, for a total time of 103 ns, one expects the peak to have a width of roughly 1/103 ns or 0.01 GHz. This gives the error in the frequency from this method.

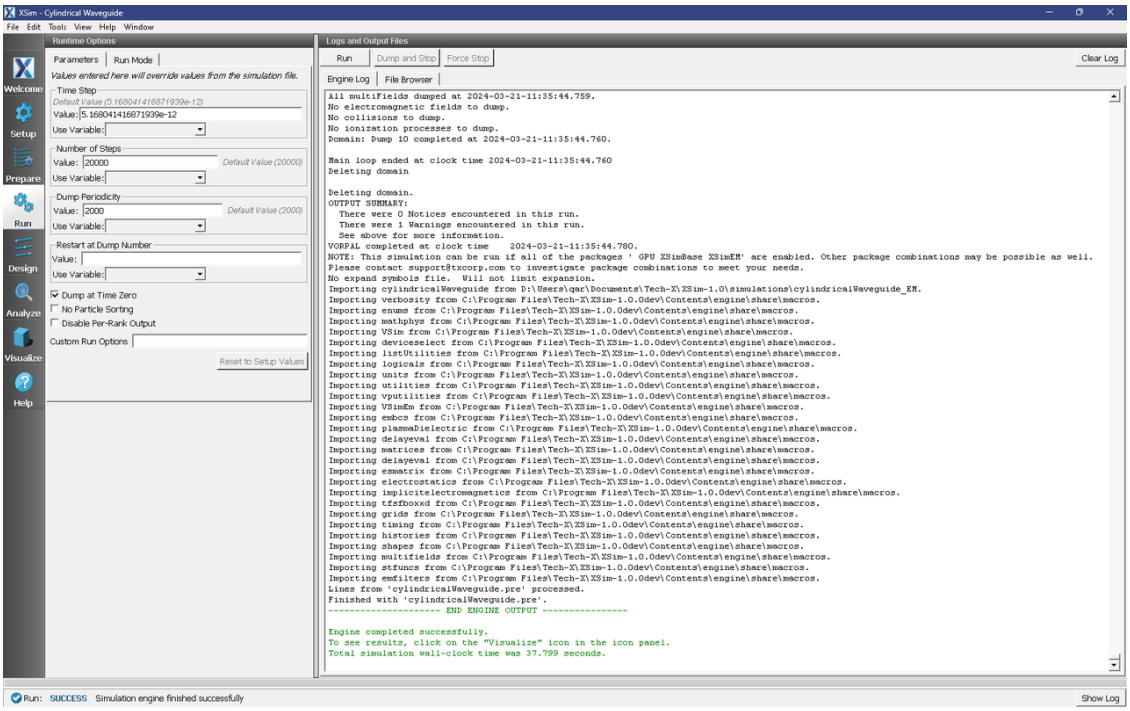

Fig. 2.53: Run Window for the Cylindrical Waveguide example after the initial run.

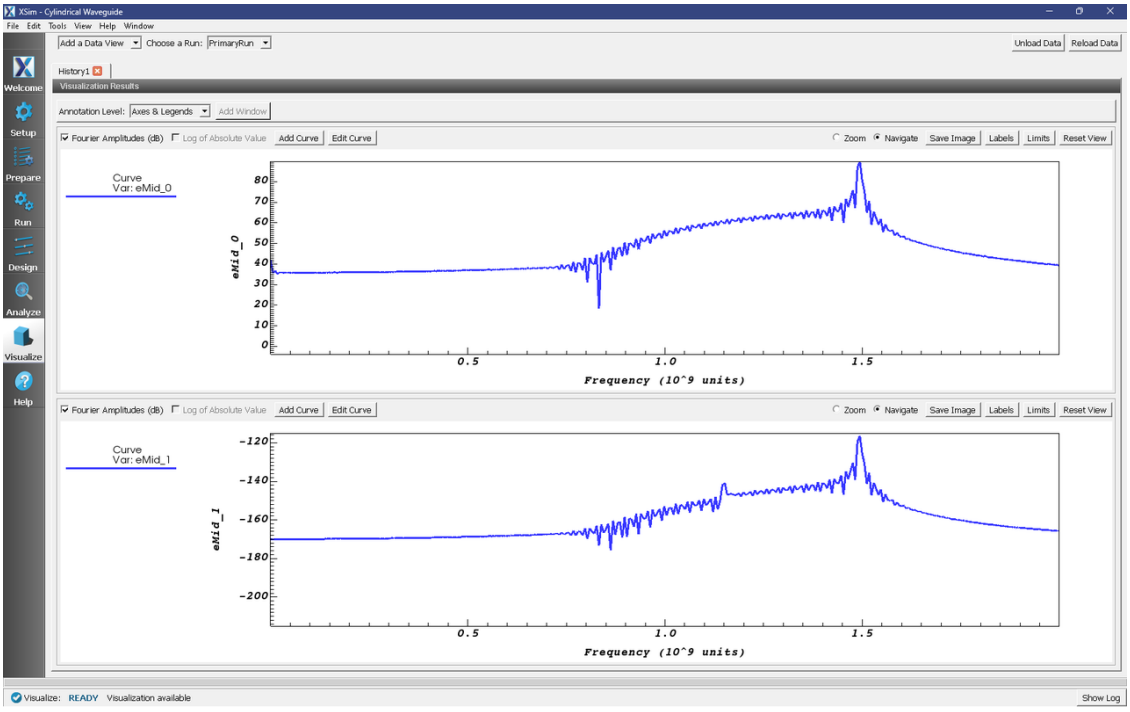

Fig. 2.54: Spectrum for the Cylindrical Waveguide example after the initial run.

Computing More Accurate Modes

We can obtain more accurate frequencies using the Filter Diagonalization Method. To do this, we need to take a bit more data. We need to have the number of dumps equal to three times the number of modes, so we run again, restarting from dump 10 for another 300 steps, dumping every 50 time steps. This will give us an additional 6 dumps. The Run Window for this part of the simulation is shown in Fig. 2.55.

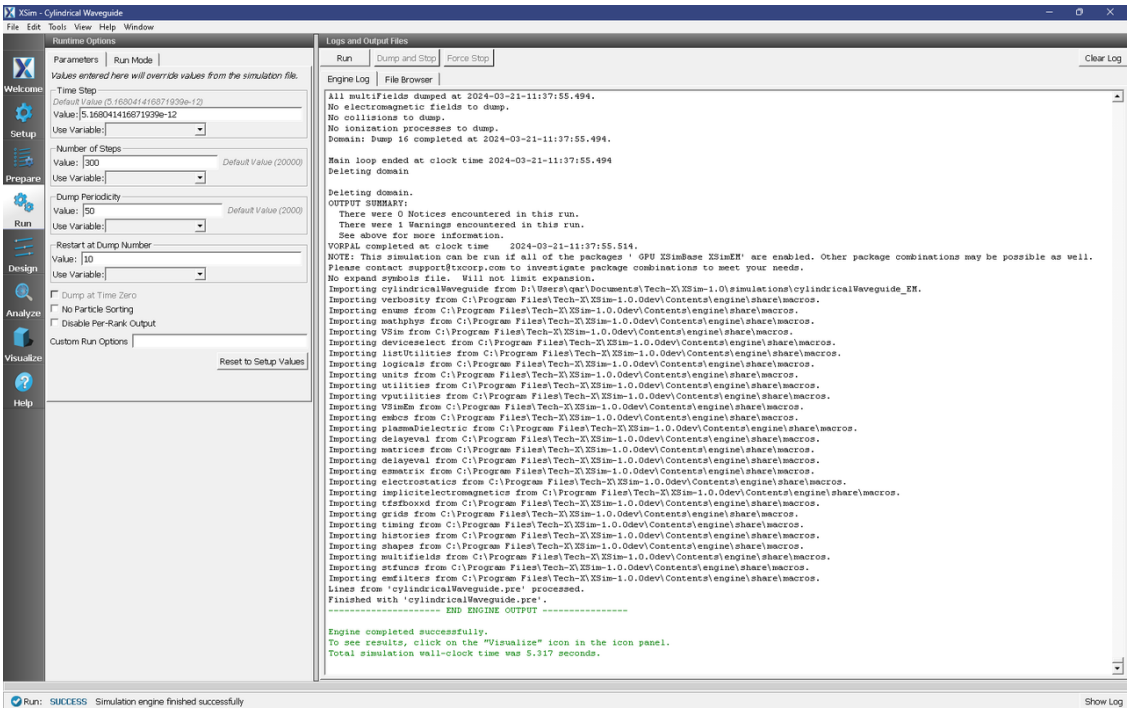

Fig. 2.55: Run Window for the Cylindrical Waveguide example for the second run.

We now move to the Analyze Window, open extractModes, and set the field to be E. Then set the number of modes to be 2, and the begin and end dumps to be 10 and 16, respectively. Also set sampleType to 1. Upon hitting the `Analyze` button in the upper right, one sees the analysis output as shown in Fig. 2.56.

The computed mode frequencies are shown along with the inverse-Q values. Since this system is not lossy, the values of invQ, when significant, indicate that the mode calculations are dubious. However, we see that the 2nd mode has been well obtained.

These modes will now show up in the visualize panel, where one can reload the data, and modes will show up as seen in Fig. 2.57. The well obtained mode occupies dumps 1-16.

2.2.2 Circular Metal Waveguide Dispersion (circMetalWaveguideDisp.sdf)

Keywords:

`Waveguide, Dispersion Relation, Fourier Transform, Phase Shifting Boundary Conditions`

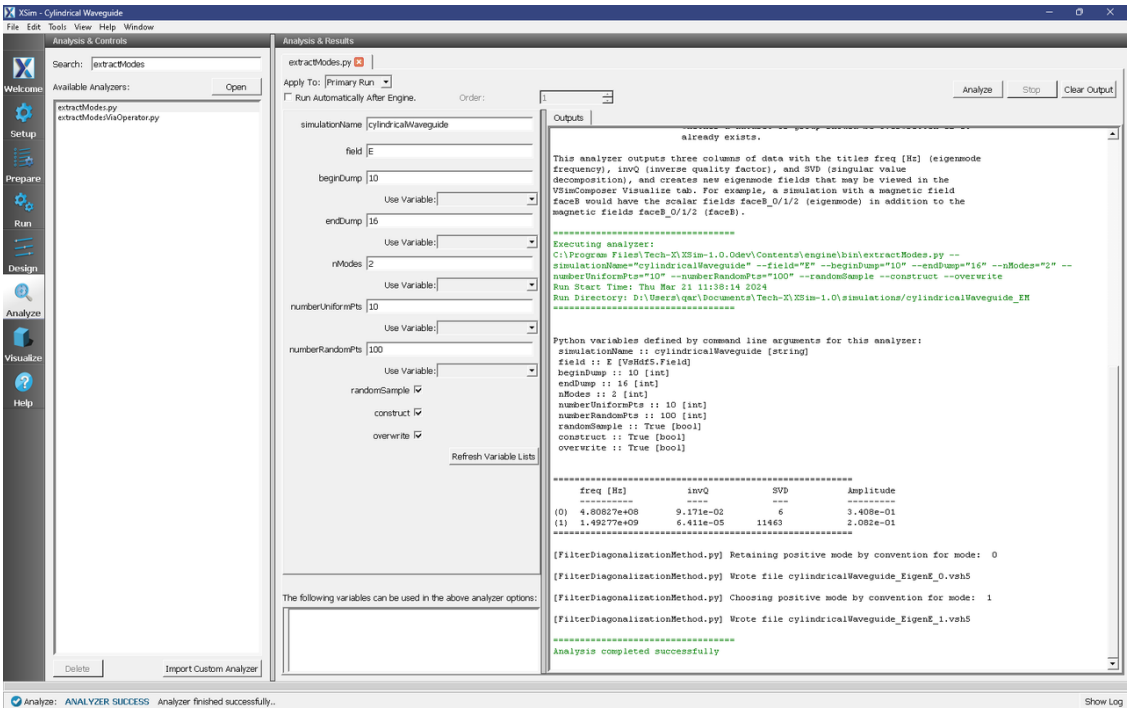

Fig. 2.56: Analysis window for the Cylindrical Waveguide example for mode extraction.

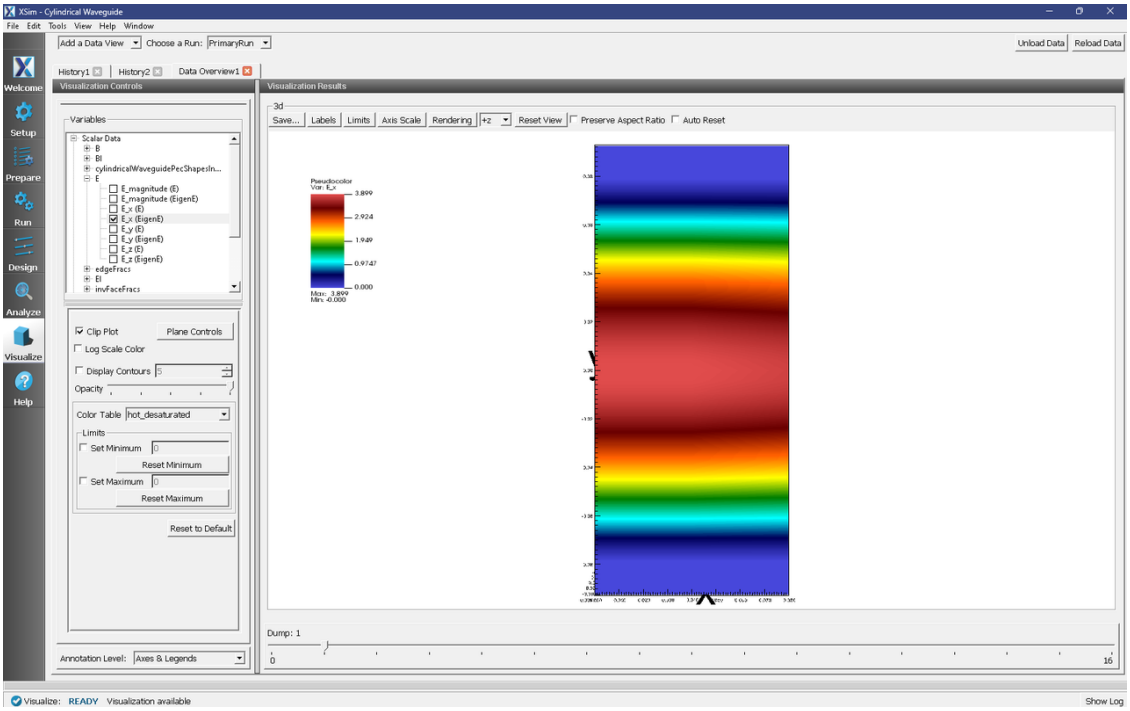

Fig. 2.57: Analysis window for the Cylindrical Waveguide example for mode extraction.

Problem description

This XSimEM example demonstrates several unique capabilities of XSim that can be used to efficiently model waveguides. One unique capability is the use of phase shifting periodic boundary conditions. These work by adding a phase to a wave at a boundary to mimic a much longer physical dimension. We also demonstrate how to use an analyzer called extractModesViaOperator.py to accurately compute the excited modes in the waveguide, even if some of the modes are much weaker than the dominant mode. In this example, we use extractModesViaOperator.py to compute the dispersion relation (ω vs k) and compare the numerically determined dispersion relation with the theoretical dispersion relation using standard waveguide theory.

This example involves five steps. In the first step, the waveguide is "pinged" with a short pulse in the current density that excites a range of modes. The Fourier transform then shows the range of frequencies of the modes. In the second step, the waveguide is excited using a sinc pulse function multiplied by a Gaussian envelope to excite a flat band of frequencies with sharp cutoffs at either end. The excitation current density is in the transverse directions (z and y) which excites an electric field primarily in the transverse direction. In the third step, the data is restarted from the end of the second run and saved at shorter time intervals in order to resolve the frequencies of interest. The output from the third step is used by the extractModesViaOperator.py analyzer to compute the eigenmodes in step 4. Finally, in step 5, the dispersion relation is computed by varying the wavelength which is resolved in the simulation.

Opening the Simulation

The circular metal waveguide dispersion example is accessed from within XSimComposer by the following actions:

- Go to *File* → *New* → *From Example...*
- In the resulting *Examples* window expand the *XSim for Electromagnetics* option.
- Expand the *Cavities and Waveguides* option.
- Select "Circular Metal Waveguide Dispersion" and press the *Choose* button.
- In the resulting dialog, create a New Folder if desired, and press the *Save* button to create a copy of this example.

The properties and values that create the simulation are accessible in the left pane when the Setup Window is selected as shown in Fig. 2.58. The right pane shows a 3D view of the selected geometry components, grids and current distributions.

The geometry can be visualized by expanding "Geometries" in the left pane. The hollow circular waveguide can be seen more clearly by de-selecting the "Grid" option. The length in the direction of propagation (x) is a small fraction of either transverse direction. This is possible because we are using "phase shifting periodic" boundary conditions in the *x* -direction. The phase shifting BCs are selected under "Basic Settings".

A sinc hat function is used to excite the waveguide.

Phase Shifting Boundary Conditions and Phase Shift

Before discussing phase-shifting periodic boundary conditions, we first review ordinary periodic BCs. In this discussion, periodic BC's are applied in the *x*-direction. With normal periodic BC's, there are two criteria in resolving a wave (or mode) of interest. (1) There must be enough grid points along the wavelength to resolve the spatial profile. Typically we sample at least 20 cells along a wavelength (λ_x), or $DX = \lambda_x/20$, and (2) The length of the simulation domain, represented by L_x, must be one wavelength long, $L_x = \lambda_x$. Periodic BC's in the x-direction means that $F(0) = F(L_x)$, where F is any field quantity. Now introduce a grid that extends from 0,1,...,nx and let $x = 0$ at grid point 0 and $x = L_x$ at grid point *nx*. Periodicity on the grid implies $F(0) = F(nx)$. Finally, let's assume that $F(x) \sim \sin(k_x x)$. Then, applying the condition for periodicity, $\sin(0) = \sin(k_x L_x)$, which is exactly met if $L_x = \lambda_x$.

With phase-shifting periodic BC's, we are no longer required to meet the second criteria discussed in the preceding paragraph. To see this, let's suppose that $L_x < \lambda_x$. Then the periodicity condition can still be met by setting $k_x L_x - \phi_0 = 0$,

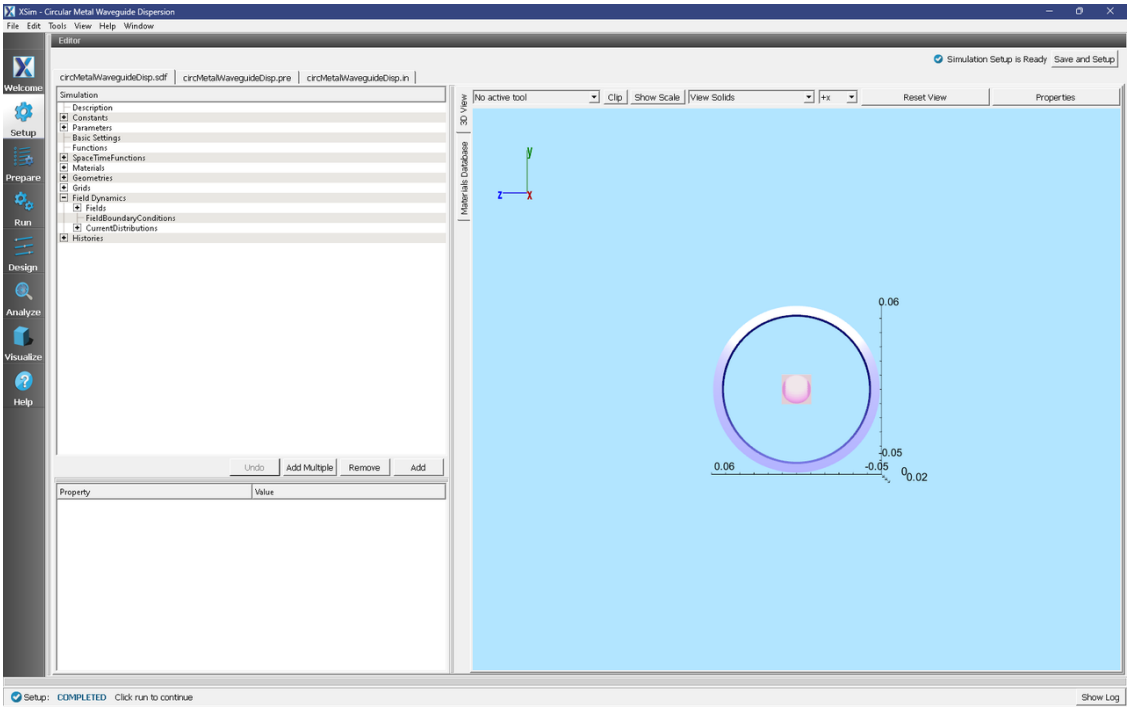

Fig. 2.58: Initial Setup Window for the Waveguide Dispersion example. The block in the middle is the region in which the current density is driven.

where ϕ_0 is the phase shift equal to $k_x L_x$, which is the exact phase shift we have chosen to apply in XSim for this example. The numerical implementation is more challenging than the conceptual picture we just discussed. To implement phase-shifting periodic BC's, we need to treat the fields as complex numbers and set $F(L_x) = \exp(i\phi_0)F(0)$. For the grid, we pick 2 cells such that $L_x = 2DX$.

By using phase-shifting periodic BC's, we can simulate different physical lengths without changing the simulation length L_x through the phase shift $\phi_0 = k_x L_x$. Because $k_x = 2\pi/L_x$, we can solve for $\omega(k_x)$ without requiring a simulation of length L_x.

Running the Simulation and Analyzing Results

We now walk through the five steps discussed in the Introduction. In the first step, we test to determine the minimum frequency that will propagate through the waveguide. For the circular waveguide in this example, we know the analytical result which is the lowest cutoff frequency. However, we still do this step to demonstrate how to determine the lowest frequency that will propagate for cases that are not analytic. In the second step, the current density "rings up" to a maximum value, then rings down to 0. Because we are exciting modes at resonant frequencies of the cavity, the *E*- and *B*-fields continue to oscillate after the excitation is turned off. We wish to determine the resonant modes using Eigen mode analysis using data saved in Step 3 when the externally driven source is turned off and the cavity is "ringing" at its natural frequencies. In step 4, extractModesViaOperator.py is used to compute the excitation frequencies in the cavity at a given mode. Finally, in step 5, you will change the value of "mode", which is defined under "Constants" to compute the dominant frequencies which are excited in the cavity at a given wavelength. We assume the maximum wavelength which can be resolved in a traditional periodic simulation is 0.2 m and denote this as *lambdaMax*. We then define the parameter $k_x = \frac{mode}{12}\frac{2\pi}{lambdaMax}$ and run 25 simulations with *mode* =0,1,2,...,24. Finally, we set the phase shift to $k_x \times L_x$ to ensure the correct phase shift is applied for the phase shifting periodic BC's. In this way, we are able to compare the simulation with waveguide theory without explicitly changing the length of the simulation domain in the x -direction.

Step 1: Determining the lowest cutoff frequency - The "Ping" Run

The Ping run is used to determine the lowest propagation frequency in the waveguide. In this simulation, we can analytically compute the lowest propagation frequency at a given k based on the allowable modes in a circular wave guide. We can then compare the computed lowest propagation frequency with theory. However, for arbitrarily-shaped waveguides, this step is necessary since no theory is available to compute the lowest propagation frequency. We have defined a PINGON constant (with values of 0 or 1) in order to easily transition from the "Ping" run to the "Excitation" run in Step 2. The default is PINGON=1 to do the "Ping" run first. For Step 1, perform the following steps

- Click on the Run Window.
- This simulation may be accelerated by changing the Run Mode to Parallel
- Click the *Run* button on the upper left corner of the *Logs and Output Files* pane

The run has completed when you see the output, "Engine completed successfully." A snapshot of the simulation run completion is shown in Fig. 2.59.

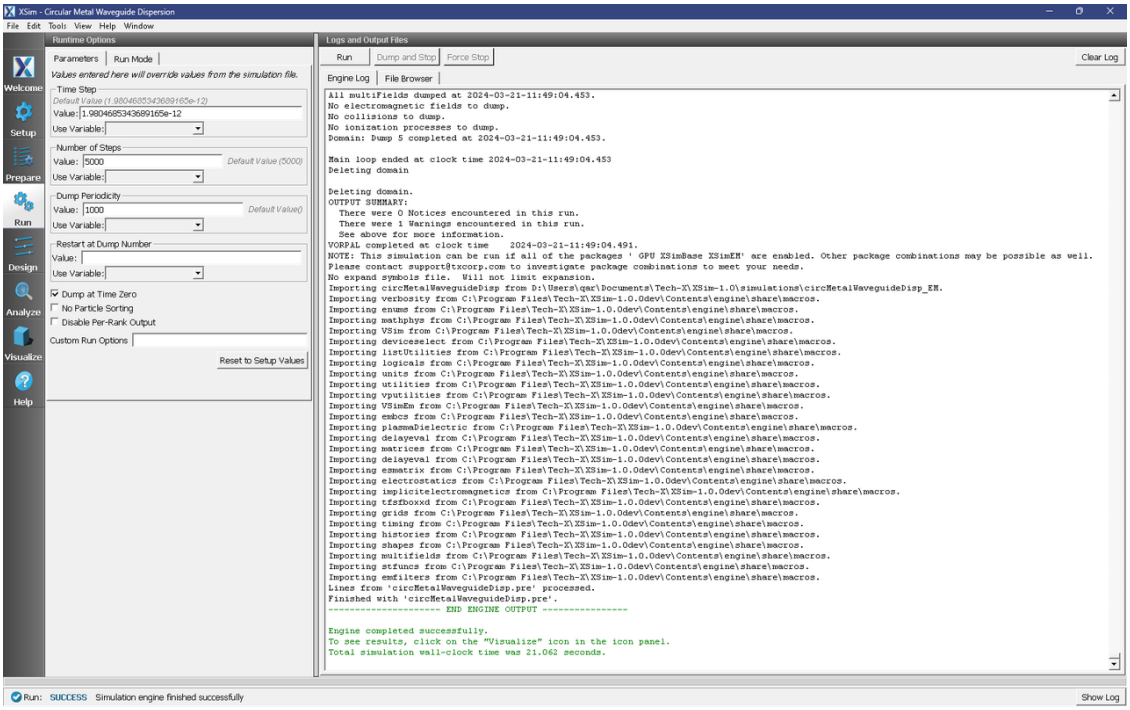

Fig. 2.59: The Run window at the end of a Step 1.

After the run has completed, click on the *Visualize* tab on the left side of the visualization window. After the data has loaded, click on *Add a Data View* and then select *History*. Perform the following steps to analyze the history:

- Graph 1 select *jMid_1*
- Graph 2 select *eMid_1*

In Graph 1, to see the data, click on the "Limits" option above the plot. Set the X-axis upper limit to 2e-10. You should see a step function in the current density beginning at t=0 and ending abruptly after about 20 time steps. This is the "ping" that we impose on the simulation. To determine the lowest frequency mode that we excite, click the "Fourier Amplitudes (dB)" box in Graph 2. Again, click on the "Limits" button above and to the right of the graph. Set the X-axis upper limit to 4e9. You will see that the first peak occurs at a frequency of about 1.75×10^9 Hz, which is the lowest cutoff frequency in the simulation and is what we find analytically using waveguide theory. Therefore, we have confirmed that XSim reproduces linear theory. You can confidently use XSim to numerically determine the

lowest cutoff frequency for non-analytically solvable shapes. The visualization window after the above steps have been performed is shown in Fig. 2.60

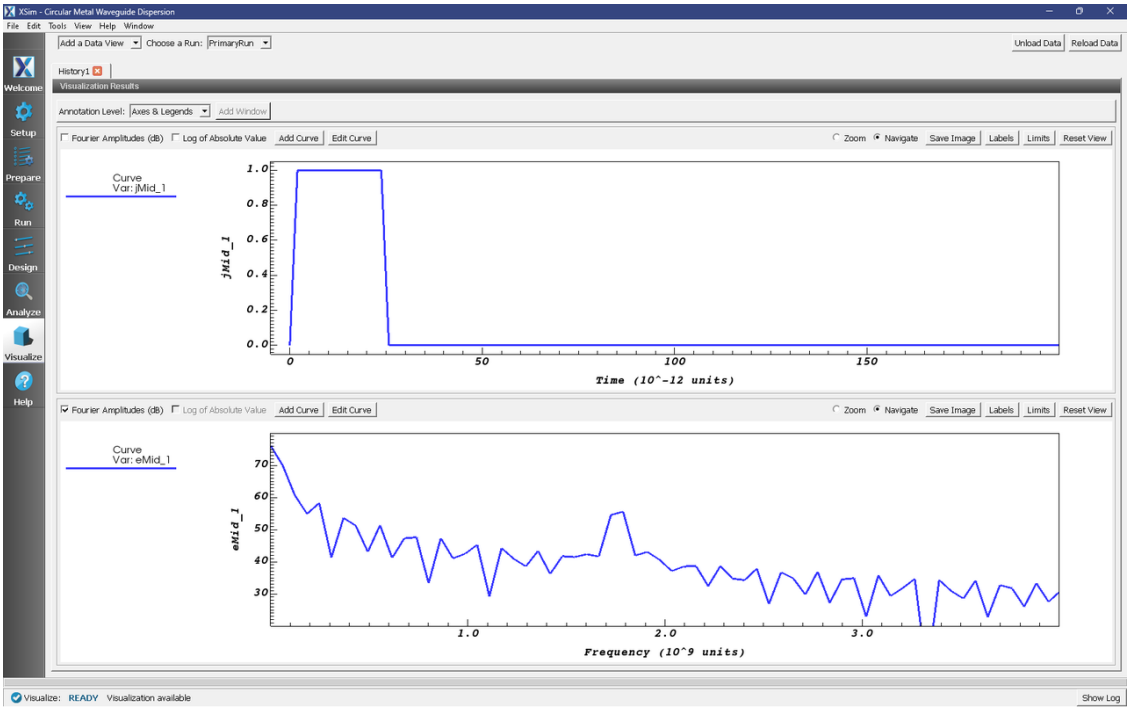

Fig. 2.60: History plots showing the lowest cutoff frequency at 1.75×10^9 Hz as the result of pinging the waveguide with an abruptly applied current density.

Step 2: Excitation

From Step 1, we know that the lowest propagation frequency is $\sim 1.75 \times 10^9$ Hz. We can now check that the lowest cutoff frequency (*frequency low* in **SincHat**) is correct. The parameter *FREQ_MIN* is passed into *frequency low* in the **SincHat** function and is indeed $\sim 1.75 \times 10^9$ Hz. We can now proceed to excite the waveguide with confidence knowing that lowest excitation frequency is correct. To excite the waveguide, perform the following steps:

- Go to the Setup Window and expand the *Constants* option
- Set *PINGON* to 0
- This will change the applied current to the **SincHat** function.
- Click on *Save and Setup* in the upper right corner of the visualization window
- Go to the Run Window by pressing the Run button in the left column of buttons.
- Set Number of Steps to 21000. This ensures that the simulation runs long enough for the current density to "ring up" then "ring down" to 0.
- Leave the "Dump Periodity" at 1000.
- This simulation may be accelerated by running on multiple MPI ranks. The parallel options are in the *Run Mode* tab
- To run the file, click on the *Run* button in the upper left corner of the *Logs and Output Files* pane. You will see the output of the run in this pane. The run has completed when you see the output, "Engine completed successfully." A snapshot of the simulation run completion is shown in Fig. 2.61.

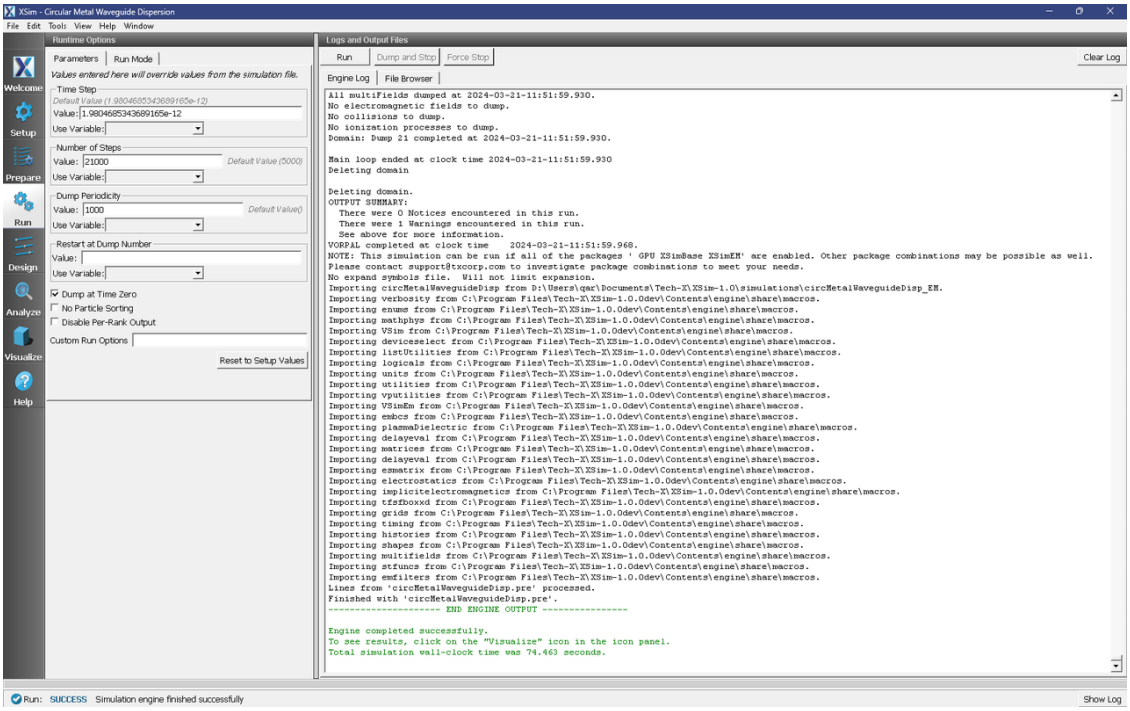

Fig. 2.61: The Run window at the end of a Step 2.

Step 3: Evolving the excited cavity

The purpose of Step 3 is to continue the simulation with only the *E*- and *B*- fields excited due to the externally imposed current density from Step 2. We will also save more data which we can analyze using Eigenmode analysis for Step 4.

- Go to the Setup Window and expand the Basic Settings option.
- Change “number of steps” to “10000”.
- Change “steps between dumps” to “100”.
- Change “dump in groups of” to “3”.
- Click on *Save and Setup* in the upper right corner of the visualization window.
- Go to the Run Window and click the “Reset to Setup Values” button in the lower right corner of the “Run Options” section.
- In the *Run Options* section, uncheck the *Dump at Time Zero* box and set the *Restart at Dump Number* to 21.
- NOTE: the “Dump Periodicity” must be blank for the “dump in groups of 3” setting to work. In this case, the simulation will revert to the values defined in “Basic Settings” which are to dump every 100 time steps in groups of 3 (21000,21001,21002, . . . , 21100,21101,21102,. . . , etc). If you fill in an integer into Dump Periodicity, the default gets overwritten and the data are not dumped in groups of 3, which is required for the extractModesViaOperator.py analyzer. So, there should be 300 new dumps making a total of 321 dumps.
- Click run. The run has completed when you see the output, “Engine completed successfully.” A snapshot of the simulation run completion is shown in Fig. 2.62. When this run is finished, the last step should be step 31000.

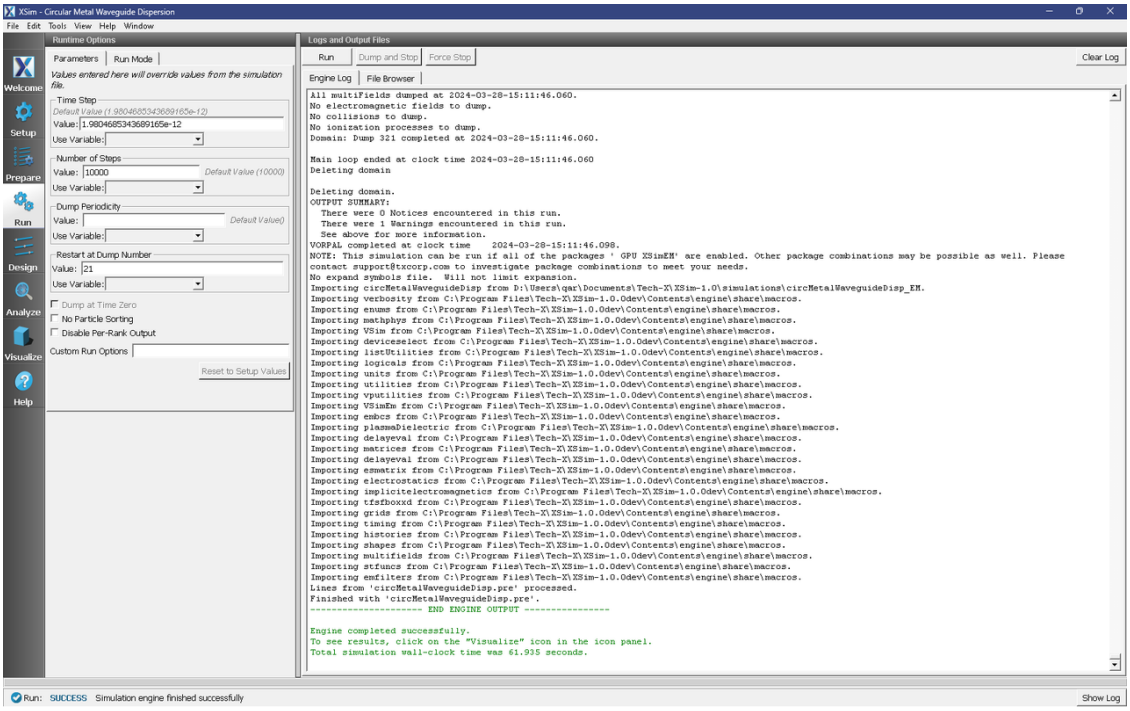

Fig. 2.62: The Run window at the end of Step 3.

Step 4: Computing the eigenmodes

- Go to the analyzer window by selecting *Analyze* in the left column.
- Select *extractModesViaOperator.py* from the list of available analyzers. Then click "Open" on the top right of the *Analysis Controls* pane.
- Compute the electric field eigenfunctions. After the analyzer loads, ensure the following parameters are entered:
  - **simulationName:** "circMetalWaveguideDisp"
  - **outputSimName:** leave blank
  - **realFields:** "E"
  - **imagFields:** leave blank
  - **secondFieldFactor:** leave blank
  - **operator:** "d2dt2"
  - **dumpRange:** "21:321"
  - **cellSamples:** ":,5:55:5,5:55:5"
  - **cutoff:** "1e-12"
  - **maxNumModes:** "-1"
  - **initialModeNumber:** "0"
  - **normalizeModes:** checked
  - **testing:** unchecked
  - **compMajorC:** unchecked

- **overwrite:** checked

The dump range runs only over the data saved in step 3 and we are only sampling every 5 cells in the z- and y- directions which is adequate to compute the eigen modes. Double-check your entries against what is shown in Fig. 2.63. After you run the analyzer, you will need to scroll up to find the computed frequencies. The screenshot shown in Fig. 2.63 is from a visualization window that is scrolled up so that the computed frequencies can be compared with what you ran. Figure Fig. 2.63 shows that there are 6 unique modes. One indication that you are using extractModesViaOperator.py correctly is that the imaginary part of the frequency (which represents attenuation of the wave) is nearly 0 or at least much smaller than the real part. The lowest excited mode is $\sim 1.76 \times 10^9$ Hz, which is discussed further below in the context of standard waveguide theory.

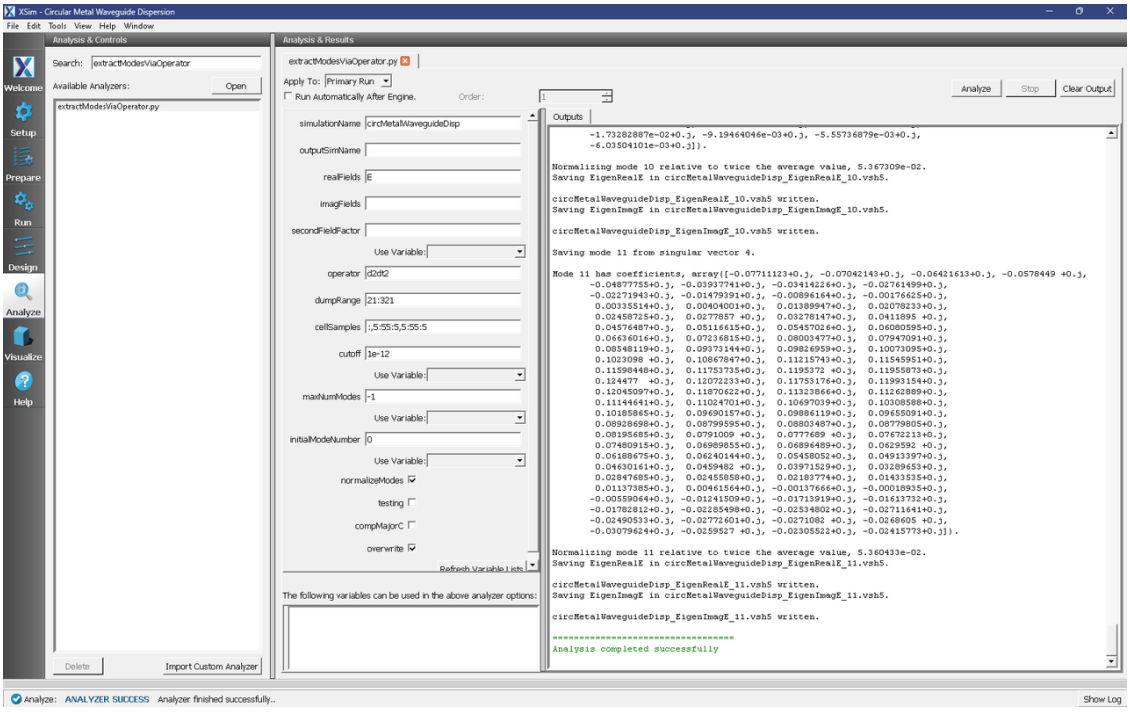

Fig. 2.63: Computing the electric field eigenfunctions and frequencies using the extractModesViaOperator.py analyzer.

Step 5: Computing the Dispersion Relation

Once the Eigenmode analysis is complete in Step 4, it is necessary to record the three lowest-order modes. The pre-loaded value of *mode* is 1, which means we are simulating $k_x = \frac{1}{24}\frac{2\pi}{\lambda}$, where $\lambda = 0.2$ m. Also, the waveguide dispersion relation is given by $\omega^2 = k_x^2 c^2 + \omega_{mn}^2$, where $\omega_{mn}^2 = \frac{c^2}{R^2}(x'_{mn})^2$, where m and n are integers and x'_{mn} is the nth root of $J'_m(x) = 0$ (the derivative of the cylindrical Bessel function of the first kind). The first three allowable modes are $x'_{11} \approx 1.841, x'_{21} \approx 3.054$ and $x'_{01} \approx 3.832$. Furthermore, the frequency range that is excited lies between $f_l = 1.75 \times 10^9$ Hz (which is the lowest allowable propagation frequency) and $f_h = 5.2 \times 10^9$ Hz. Given these parameters, we expect from waveguide theory for the three lowest modes to be $\sim 1.76 \times 10^9, 2.92 \times 10^9, 3.66 \times 10^9$ Hz, which is what is found from extractModesViaOperator.py. Repeating these steps for modes 1 to 24 yields the dispersion relation shown in Fig. 2.64. The overlap between the simulation results and theoretical values is evident in Fig. 2.64.

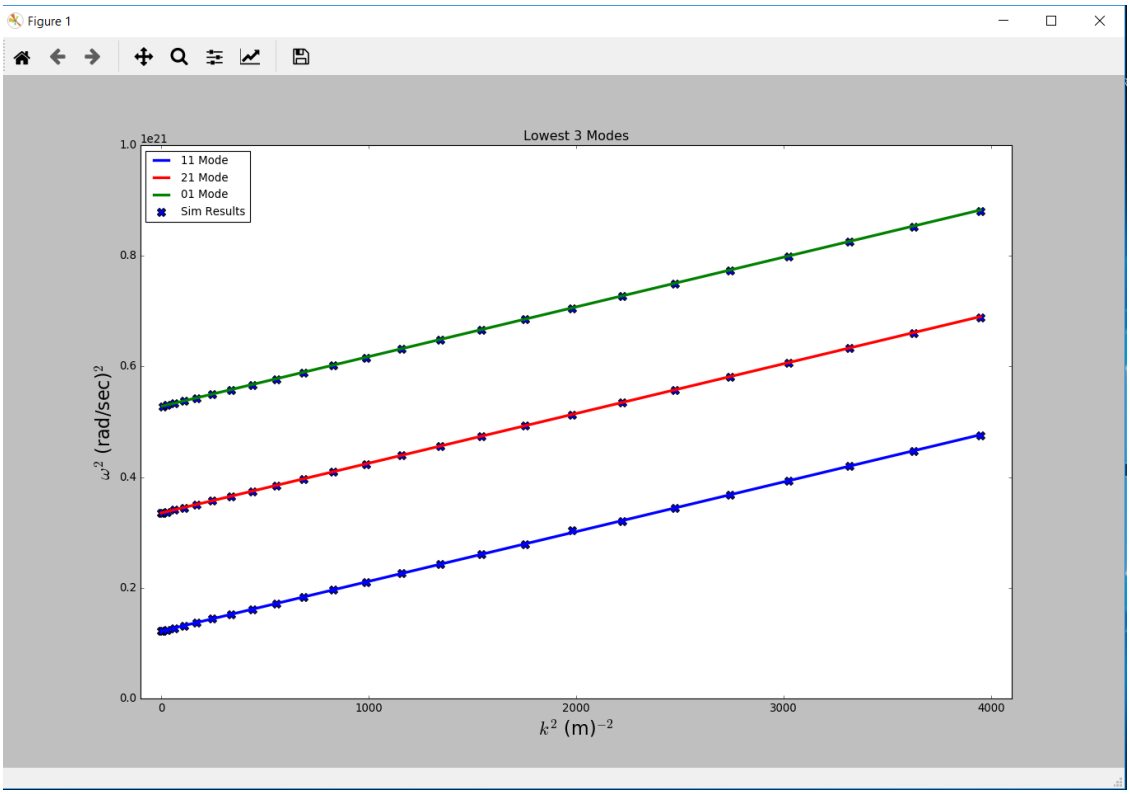

Fig. 2.64: Dispersion relation found by changing *mode* in XSim. The solid line is theoretical dispersion relation $\omega^2 = k_x^2 c^2 + \omega_{mn}^2$. The 'X's represent data from the simulation results.

Convergence Study

To demonstrate that the simulation results converge to the theoretical value, we have performed a series of simulations in which $DX = DY = DZ$ are varied. The values chosen are *DX* = 0.25, 0.33, 0.5 and 0.733 cm. We then computed the lowest propagation frequency using *extractModesViaOperator.py* and plotted this frequency versus DX^2. Using Richardson extrapolation, we then compute the lowest propagation frequency for $DX^2 = 0$, which provides a better comparison with the theoretical frequency than a simulation with finite DX^2. The field calculation error scales approximately with DX^2 so a plot of frequency versus DX^2 should be a line. The linear correlation between frequency and DX^2 is shown in Fig. 2.65. The extrapolated frequency at $DX^2 = 0$ is $\sim 1.75686 \times 10^9$ Hz. The theoretical value is $\sim 1.75681 \times 10^9$. Therefore, we see from Fig. 2.65 that the computation of the lowest propagation frequency for the DC mode converges with order DX^2, which is expected since the field solve is 2nd order accurate. Furthermore, using Richardson extrapolation, we see that the difference between theory and simulation, with $DX^2 = 0$, is 0.003 %.

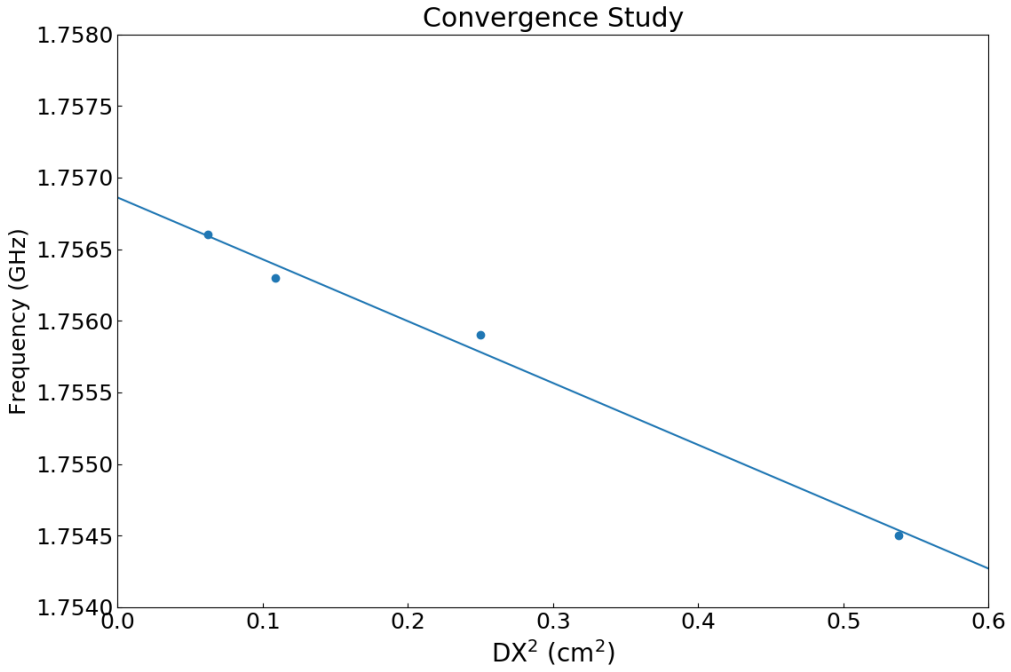

Fig. 2.65: Plot demonstrating convergence of frequency calculation.

Visualizing the results

After performing the above actions, continue as follows:

- Proceed to the Visualize Window by pressing the Visualize button in the left column of buttons.
- Click on *Reload Data*

The first step will be to ensure that we are driving the current density as expected and that the *E*- and *B*- fields are "ringing" as a result of driving the current density. Follow these steps to perform this check:

- Click on the *Add a Data View* pull-down menu at the top of the visualization window
- Click on *History*. This will open a new tab.
- Under Graph 1 plot *jMid_1*. This is the *y*- component of the current density

- Under Graph 2 plot *eMid_1*.

Your plots should be similar to Fig. 2.66. The current density is driven for a short time period. However, because we are driving resonant modes, the current density excites the transverse electric field and parallel magnetic field. You can also plot *eMid_0*, *bMid_1*, and *bMid_2* to compare with Fig. 2.66. Since *eMid_0* is ~ 0, we are driving a TE waveguide.

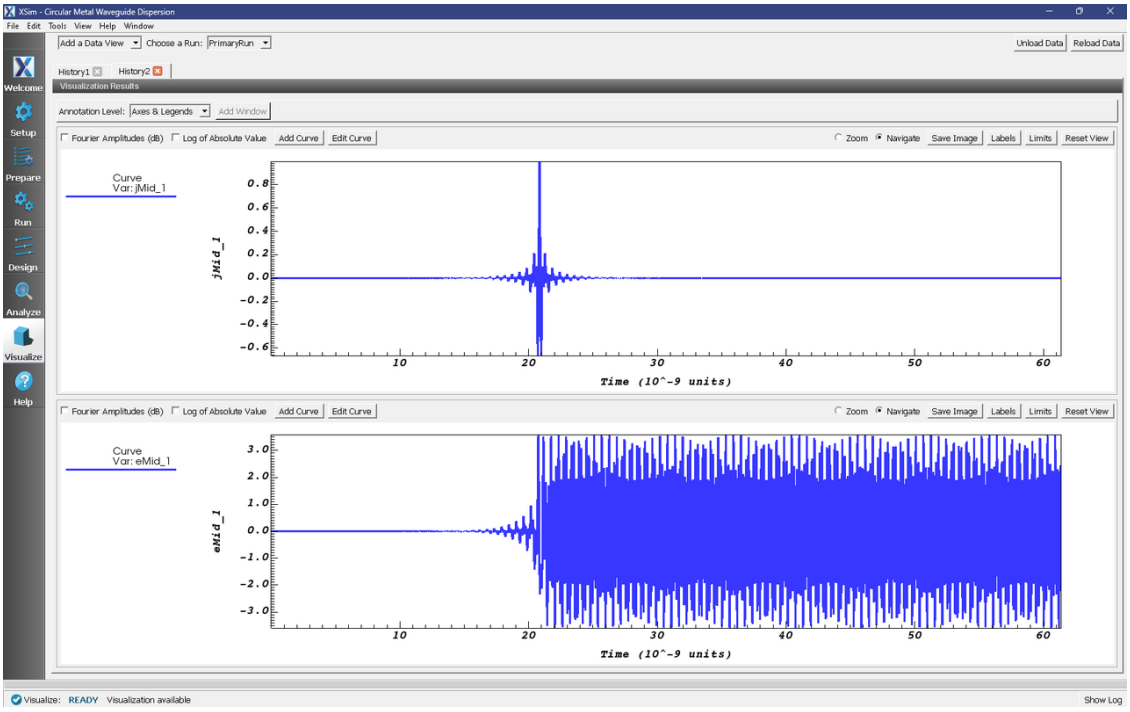

Fig. 2.66: History plots showing the modes driven in this simulation

We next wish to examine the spectral characteristics of the current density, which is driven between *FREQ_MIN* and *FREQ_MAX*. Therefore, the Fourier Transform of J_y or J_z should be greatest in this frequency range. Returning to the History visualization window, under Graphs 2, 3, and 4, change the plotted quantity to *none*, so that only Graph 1 shows a plot. Now check the "Fourier Amplitudes (dB)" box in the upper left corner. Finally, check the "Zoom" box on the right side of the visualization window and highlight from 0 to 10^{10} Hz which will expand the lower frequency part of the plot. After you have zoomed in on the plot, re-check the "Navigate" box. The result of these steps should lead to a plot that looks similar to Fig. 2.67. The driving frequencies lie between *FREQ_MIN* and *FREQ_MAX* as expected. The rate at which the Fourier signal dampens below *FREQ_MIN* and above *FREQ_MAX* depends on the Gaussian envelope and the parameter called *OMEGA_SIGMA*.

Further Experiments

The result of varying *mode* from 0 to 24 is shown in Fig. 2.64. One experiment you can perform is to reproduce Fig. 2.64 by running 25 simulations with *mode* varying from 0 to 24. Each simulation takes just a few minutes so the 25 simulations takes about one hour. For each simulation record the three lowest frequencies that are excited using extractModesViaOperator.py.

Another experiment would be to change the dimensions of the waveguide. The constant *InnerRadius* is used to determine the lowest frequency that will propagate in the waveguide. Therefore, changing *InnerRadius* will automatically compute *FREQ_CUTOFF_TE*. You will then need to change *BGNY*, *ENDY*, *BGNZ*, and *ENDZ* so that the primitive fits within the simulation domain. The primitive geometry called "pipe0" scales with the constant *InnerRadius*.

A third experiment would be to modify *FREQ_MIN* and *FREQ_MAX* and compute the dispersion curve for these new values. No mode will propagate below *FREQ_CUTOFF_TE* so do not set *FREQ_MIN* below *FREQ_CUTOFF_TE*.

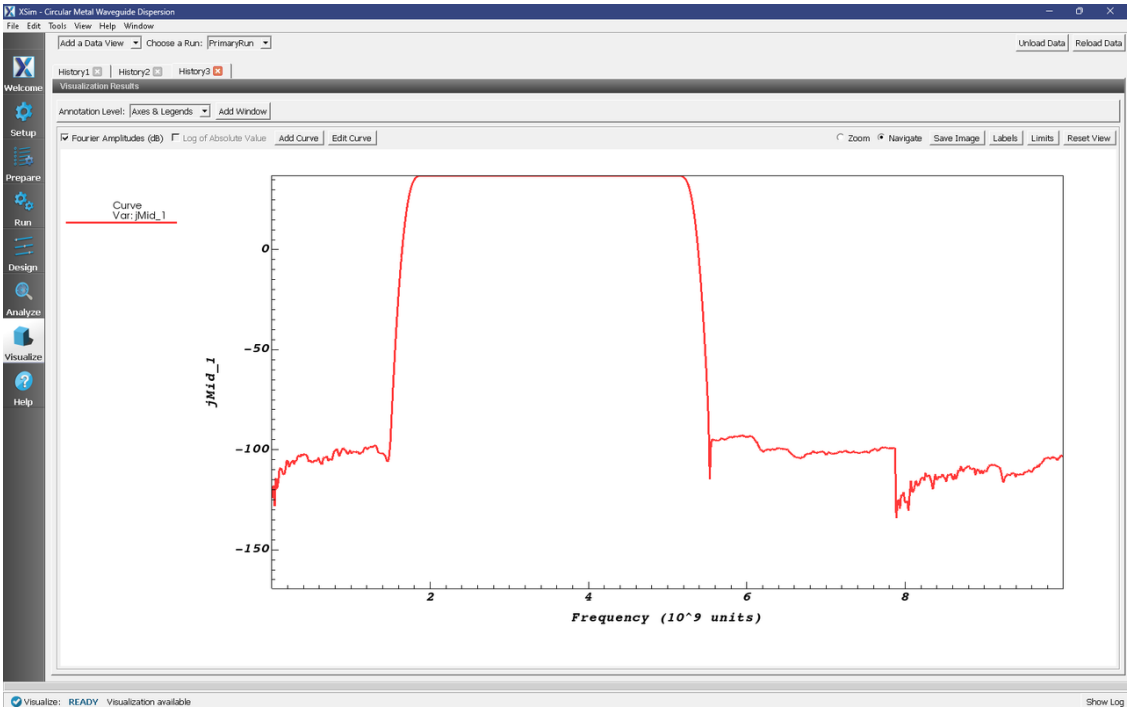

Fig. 2.67: Fourier Transform of the current density showing that the current density is mainly driven between *FREQ_MIN* and *FREQ_MAX* as expected.

Lastly, you can try to propagate a TM mode instead of a TE mode.

2.2.3 Pillbox Cavity (pillboxCavity.sdf)

Keywords:

`Pillbox cavity, Figures of merit, Transit time factor, Geometry factor`

Problem description

This XSimEM example demonstrates the usage of XSim in computing the eigenmodes and figures of merit of two simple cavities. One may select either the closed pillbox cavity for which the analytic solution is well known, or a cavity based on the closed pillbox, but having outlets leading to the periodic domain boundaries. Like other examples utilizing the extractModes.py analyzer, the simulation run is done in two steps. In the first step, the cavity is excited by a sinc pulse current source and output is dumped only at the end of this excitation run. Then in the second step, output is dumped at intervals which are sufficiently short compared to the frequencies of interest. The output from the second run is used by the extractModes.py analyzer to compute the eigenmodes. Then, the computeTransitTimeFactor.py and computeCavityG analyzers are used to compute the transit time factors and geometry factors of the eigenmodes.

Opening the Simulation

The pillbox cavity example is accessed from within XSimComposer by the following actions:

- Go to *File* → *New* → *From Example…*
- In the resulting *Examples* window expand the *XSim for Electromagnetics* option.
- Expand the *Cavities and Waveguides* option.
- Select "Pillbox Cavity" and press the *Choose* button.
- In the resulting dialog, create a New Folder if desired, and press the *Save* button to create a copy of this example.

The properties and values that create the simulation are accessible in the left pane when the Setup Window is selected. The right pane shows a 3D view of the selected geometry components, grids and current distributions.

The geometry of the closed pillbox cavity is called *pillboxCavityAnalytical* and the geometry of the periodic cavity with outlets on either end is called *pillboxCavityWithTube*. These can be visualized individually by expanding *Geometries*, de-selecting and then expanding *CSG*, and then selecting either *pillboxCavityAnalytical* or *pillboxCavityWithTube*.

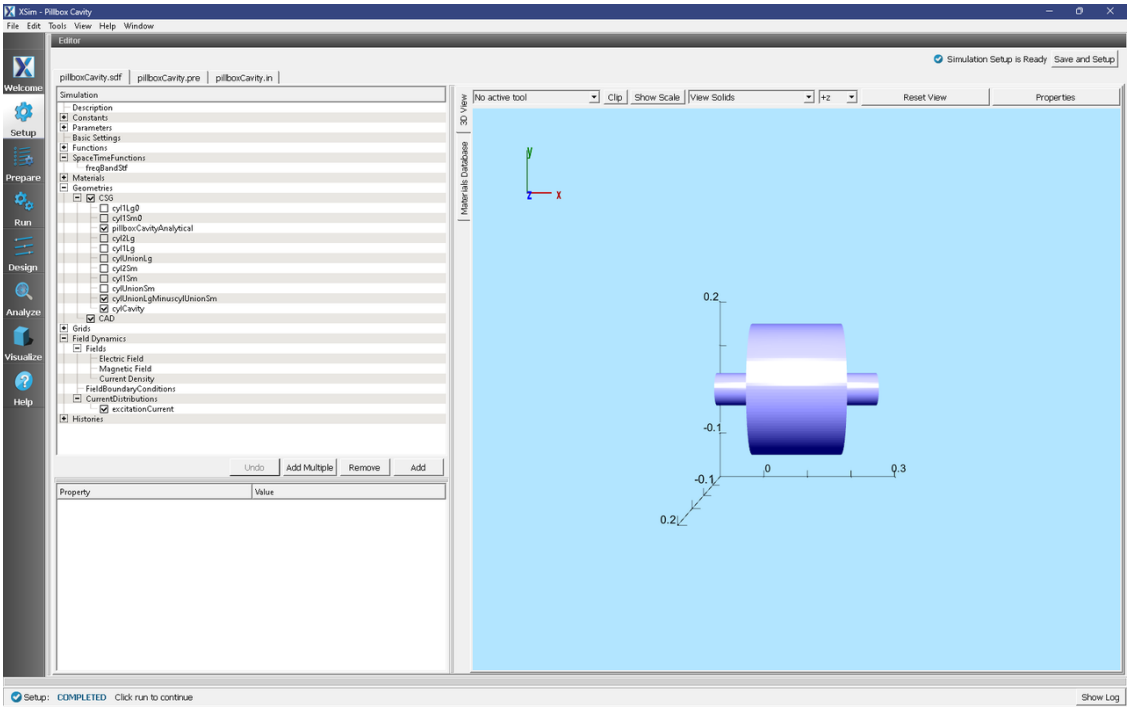

Fig. 2.68: Visualizing the periodic cavity geometry in the Setup Window.

Running the Simulation and Analyzing Results

Step 1: Cavity selection

- If you want to model the closed cavity, skip **Step 1** and go to **Step 2**. The closed cavity is set by default.
- To model the periodic cavity, go to the Setup Window.
- Go to *Geometries* → *CSG*.
- Click on *pillboxCavityAnalytical* under *CSG*.

- The bottom left pane will show properties of the selected geometry. At this time, the material should be set to *PEC* (perfect electric conductor). Double click on *PEC* and select the blank line.
- Now click on *pillboxCavityWithTube* under *CSG*.
- Select *PEC* as the material for *pillboxCavityWithTube*.

Step 2: Excitation

- Go to the Run Window by pressing the Run button in the left column of buttons.
- This simulation may be accelerated by running on multiple MPI ranks. The parallel options are in the *Run Mode* tab
- To run the file, click on the *Run* button in the upper left corner of the *Logs and Output Files* pane. You will see the output of the run in this pane. The simulation will run for 30000 time steps and dump output once at the end. The run has completed when you see the output, "Engine completed successfully." A snapshot of the simulation run completion is shown in Fig. 2.69.

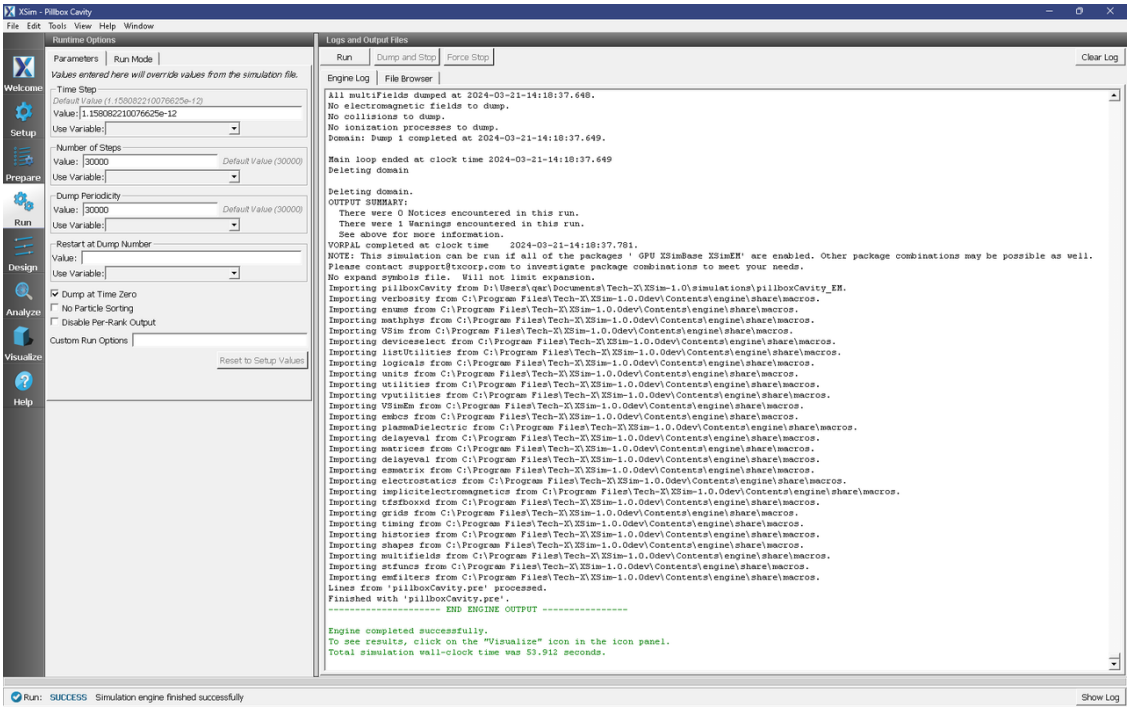

Fig. 2.69: The Run window at the end of a successful execution.

Step 3: Evolving the excited cavity

- After the first step is complete, change *Number of Steps* to 2000, change *Dump Periodicity* to 100.
- In the *Additional Run Options* Box, make sure that the *Dump at Time Zero* box is unchecked and that *Restart at Dump Number* is set to 1.
- Click run. The run has completed when you see the output, "Engine completed successfully." A snapshot of the simulation run completion is shown in Fig. 2.70. When this run is finished, the last step should be step 32000.

Note: The simulation must be run in two steps because there must be no driving currents flowing in the simulation while dumping data used to extract the eigenmodes. So, while the drive is ringing the cavity, there is no need to dump data. We switch the dump periodicity after the driving current has shut off in order to resolve the frequency of the eignemodes of interest.

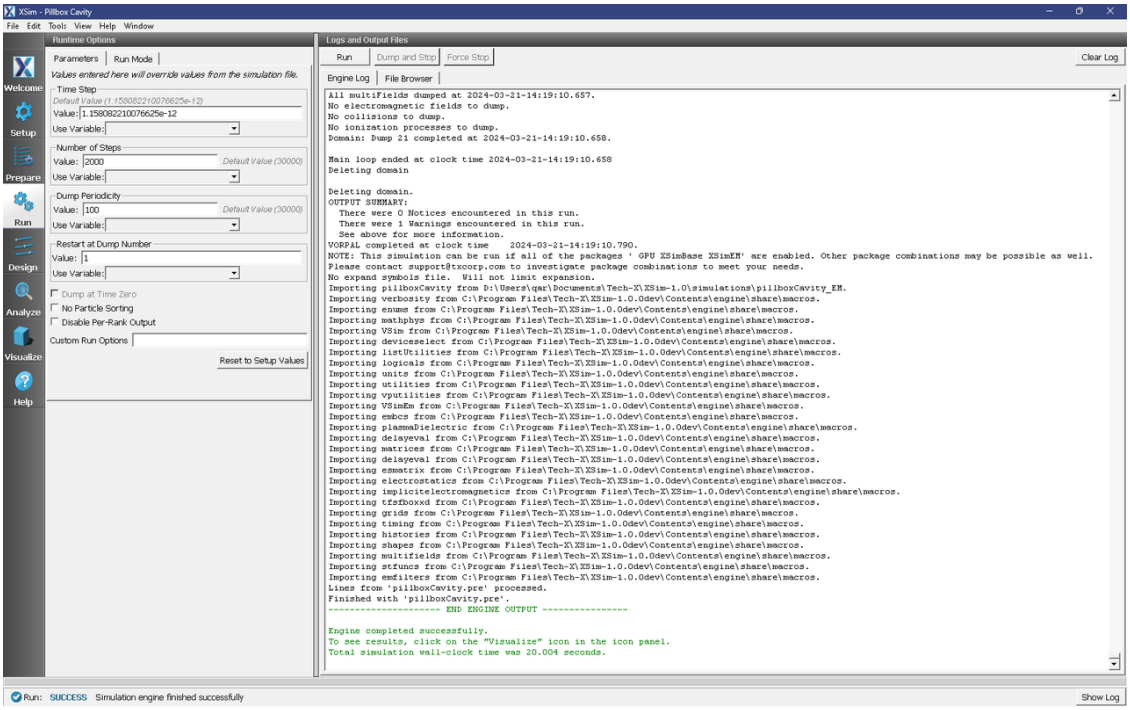

Fig. 2.70: The Run window at the end of a successful execution.

Step 4: Computing the eigenmodes

- Go to the analyzer window by selecting *Analyze* in the left column.
- Select *extractModes.py* from the list of available analyzers. Then click "Open" on the top right of the *Analysis Controls* pane.
- Compute the electric field eigenfunctions. After the analyzer loads, ensure the following parameters are entered:
  - **simulationName:** "pillboxCavity"
  - **field:** "E"
  - **beginDump:** "2"
  - **endDump:** "21"
  - **nModes:** "5"
  - **numberUniformPoints:** "20"
  - **numberRandomPoints:** "100"
  - **construct:** "checked"
  - **overwrite existing files:** "checked"

Double-check your entries against what is shown in Fig. 2.71.

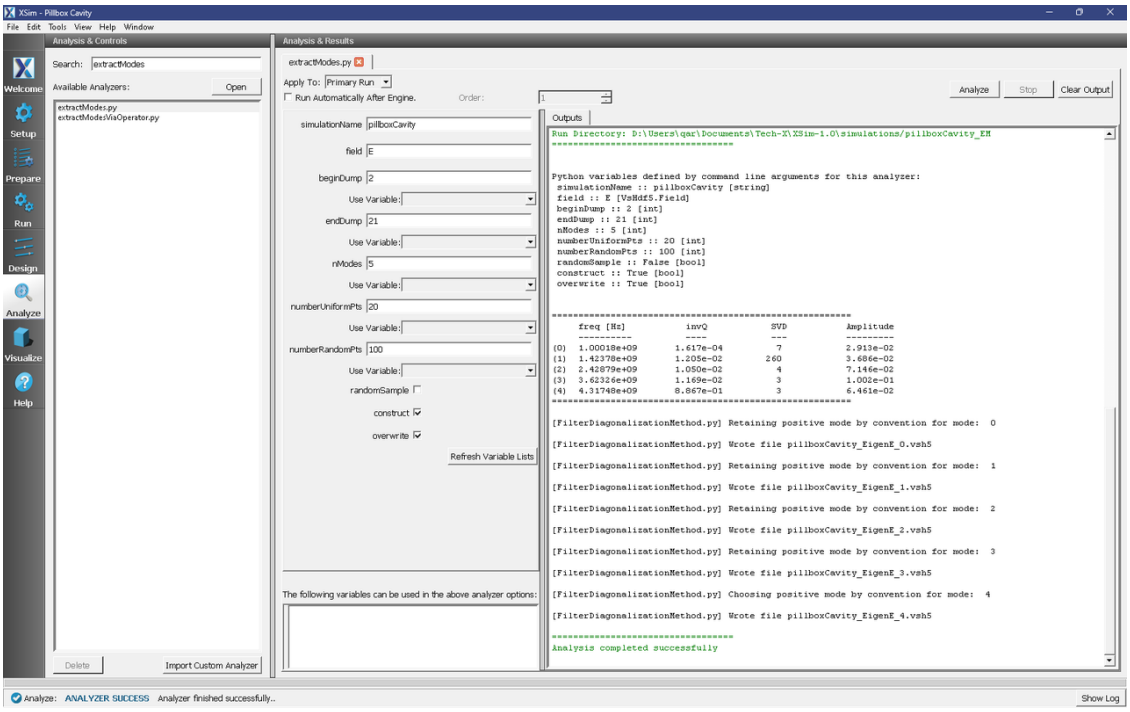

Fig. 2.71: Computing the electric field eigenfunctions and frequencies using the extractModes.py analyzer.

- Press the *Analyze* button which is located in the upper right corner.
- Compute the magnetic field eigenfunctions with the following parameter.
  - **field:** "B"

  After the analysis is finished, and scrolling down in the *Outputs* log pane you should see what is shown in Fig. 2.72.
- Note that *extractModes.py* outputs the frequencies of the computed modes in the *Run Output* pane. The first mode, mode 0, should have a frequency of approximately 1 GHz.

Step 5: Computing the transit time factor

- Select *computeTransitTimeFactor.py* from the available analyzers and press "Open" on the top right of the *Analysis Controls* pane.
- After the analyzer loads, ensure the following parameters are entered:
  - **simulationName:** "pillboxCavity"
  - **beginDump:** "0"
  - **endDump:** "0"
  - **beta:** "1"
  - **axis:** "0"
  - **offsetx0:** "0"
  - **offsetx1:** "0"

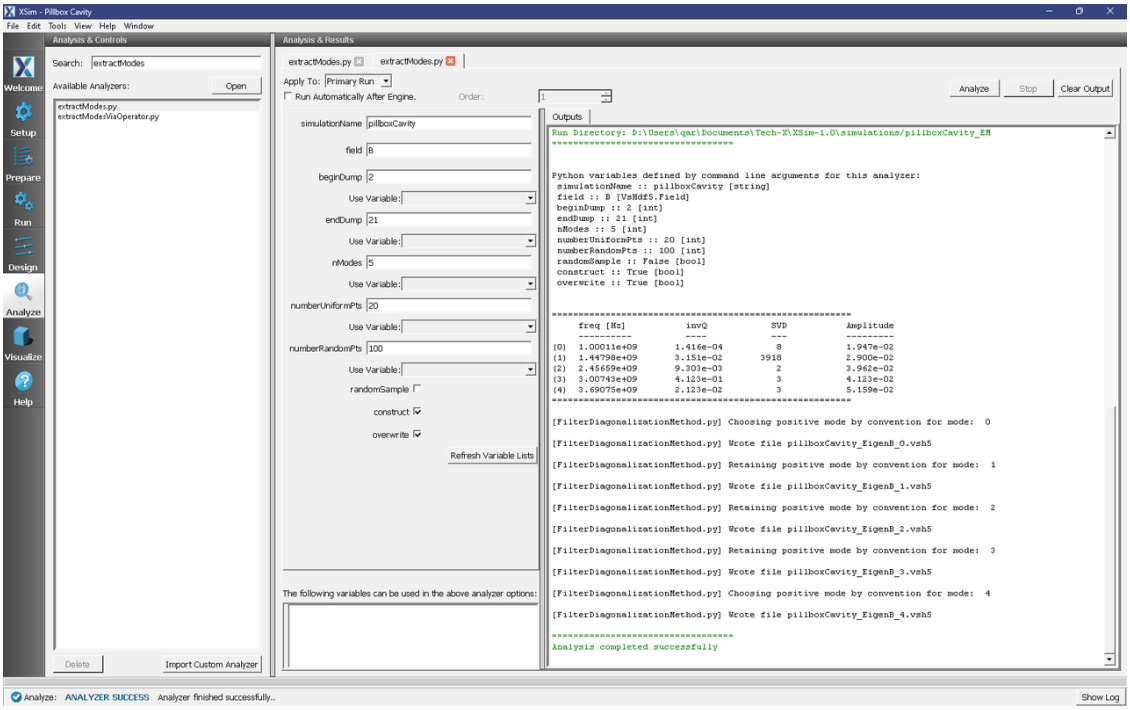

Fig. 2.72: The *Outputs* pane after Analyzing to determine the eigenmodes of the magnetic field.

And compare against what is shown in Fig. 2.73

- Press *Analyze*.
- If you have selected the closed cavity, the transit time factor (the value following "Transit time factor, T=Vacc/V0 =") should be very close the analytic value of $2/\pi$.

Step 6: Computing the geometry factor

- Select *computeCavityG.py* and click "Open".
- If you have selected the closed cavity, then enter "pillboxCavityAnalytical" for cavityGeometryName. Otherwise, enter "pillboxCavityWithTube" for cavityGeometryName.
- Select begin dump to 0 and end dump to 2.
- Press the *Analyze* button which is located in the upper right corner.
- If you have selected the closed cavity, the geometry factor at the mode frequency of 999.97 MHz should be very close the analytic value of 257.

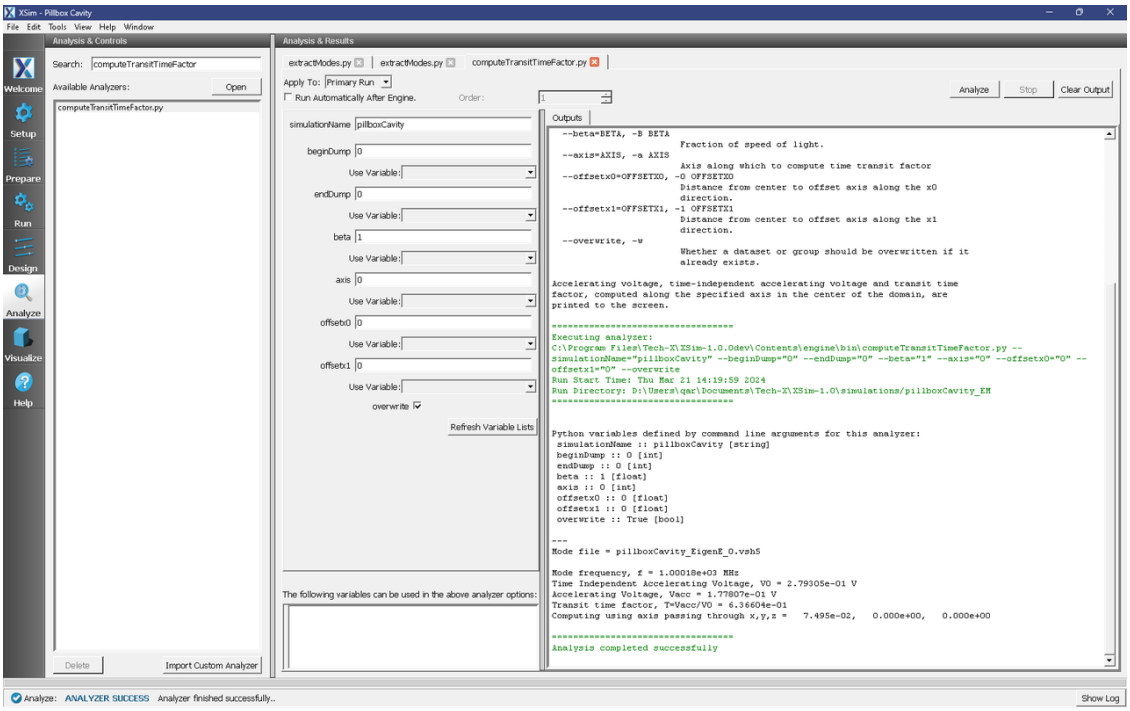

Fig. 2.73: Computing the transit time factor for the eigenmode of interest.

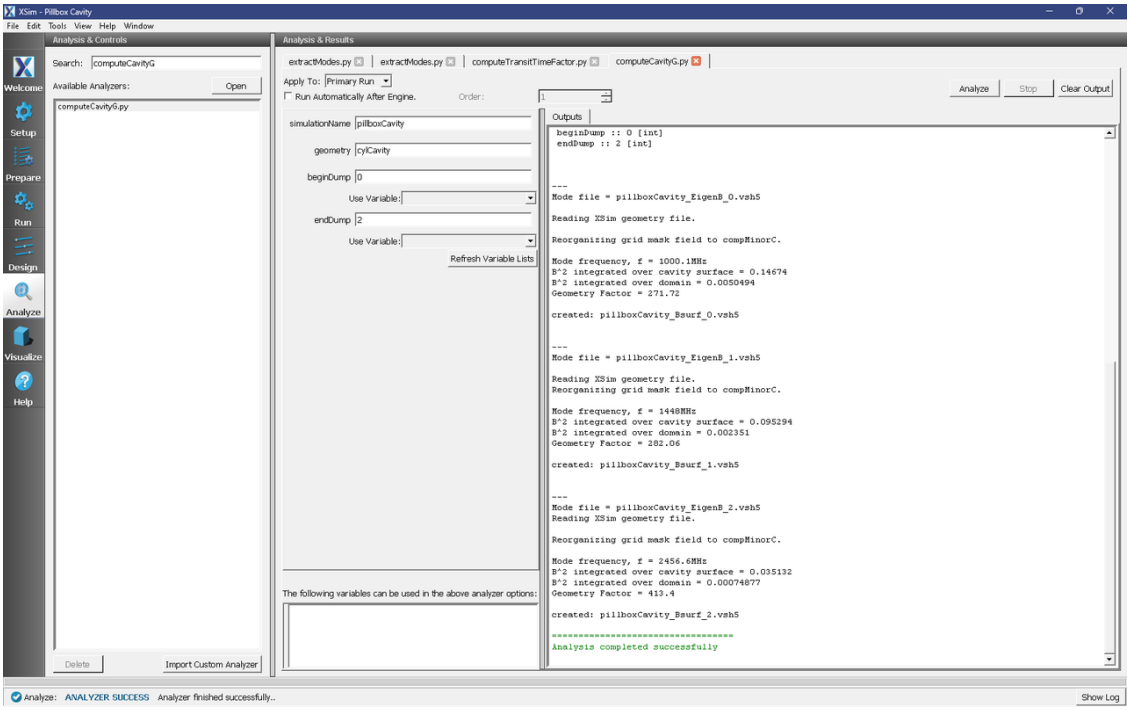

Fig. 2.74: Computing the geometry factor for the eigenmode of interest.

Visualizing the results

After performing the above actions, continue as follows:

- Proceed to the Visualize Window by pressing the Visualize button in the left column of buttons.

To see the projection of the magnetic field of the fundamental mode onto the cavity walls, do the following:

- Expand *Scalar Data*
- Expand *Bsurf*
- Check *Bsurf_magnitude*
- Click the *Plane Controls* button at the bottom of the *Visualization Controls* pane on the left of the Composer window.
- Select *X* as the "Clip Plane Normal" and ".05" as the "Origin of Normal Vector" for "X". Leave the "Origin of Normal Vector" for "Y" and "Z" as 0.
- Rotate the visualization by left clicking and dragging with your mouse.
- You should see a visualization of the magnitude of the magnetic field of the fundamental mode projected onto the wall of the cavity as in Fig. 2.75

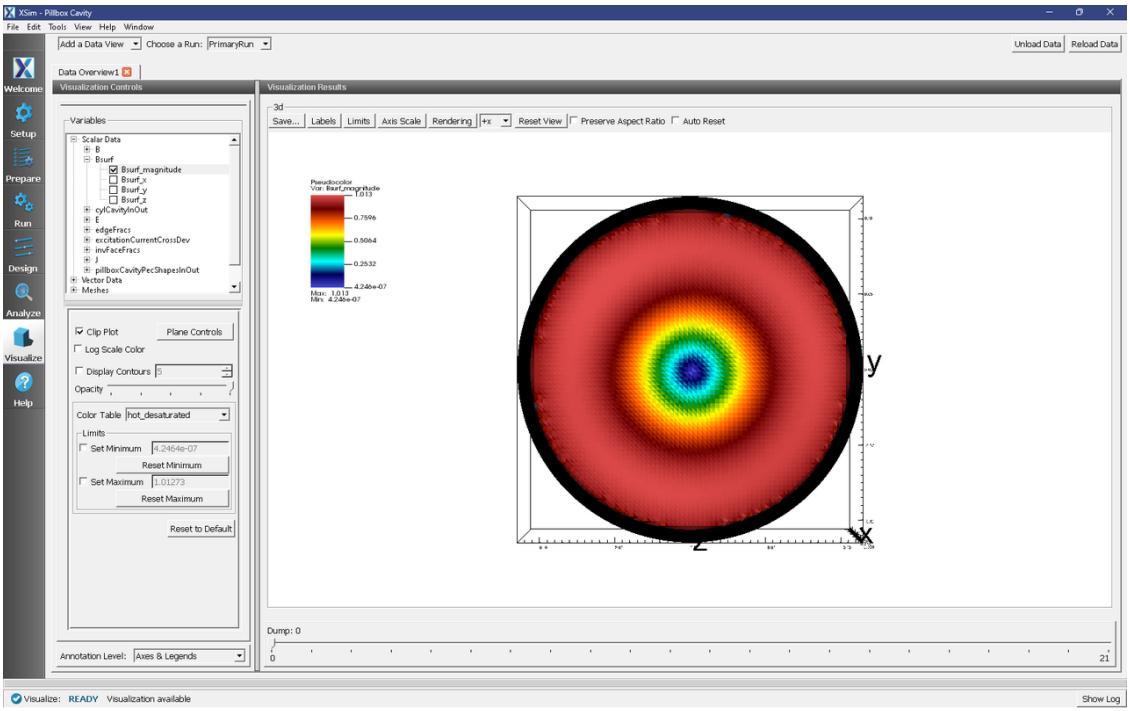

Fig. 2.75: The magnitude of the magnetic field on the wall of the cavity

To see a more quantitative visualization of the eigenmode fields, as shown in Fig. 2.76, do the following:

- Add a *Field Analysis Data View*
- Select the *E_x (EigenE)* as a field
- Under the *Layout* drop-down menu, select *Side-by-side 2d/1d*

The Bessel function dependence of the x-component of the electric field will be clearly plotted on the right.

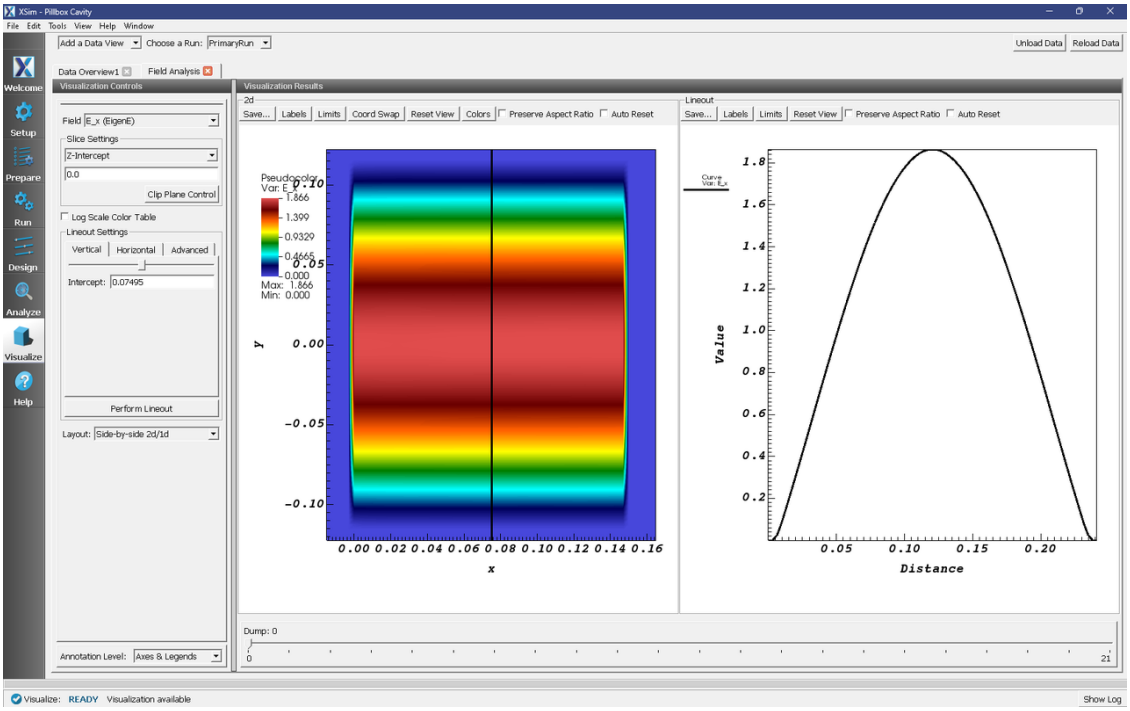

Fig. 2.76: Axial component of the electric field in the $z = 0$ plane (left) and plot of the axial electric field along $z = 0, x = 0.07495$ (right).

2.2.4 Rectangular Waveguide (rectangularWaveguide.sdf)

Keywords:

`Field Boundary Condition, rectangularWaveguide, Rectangular Waveguide`

Problem description

This example illustrates how to create a rectangular waveguide using the Rectangular Waveguide Field Boundary Condition and Constructive Solid Geometry.

Three waveguides are demonstrated in this example .

Opening the Simulation

The Rectangular Waveguide example is accessed from within XSimComposer by the following actions:

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *XSim for Electromagnetics* option.
- Expand the *Cavities and Waveguides* option.
- Select *Rectangular Waveguide* and press the *Choose* button.
- In the resulting dialog, create a New Folder if desired, and press the *Save* button to create a copy of this example.

The Setup Window is shown in Fig. 2.77.

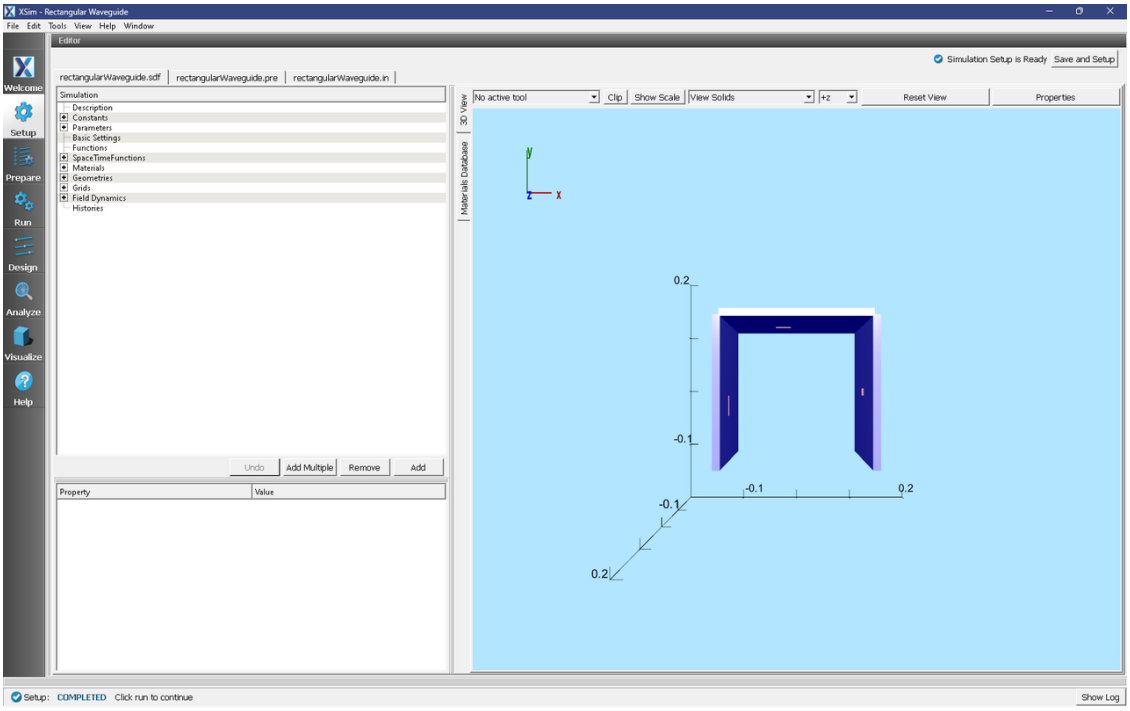

Fig. 2.77: Setup Window for the Rectangular Waveguide example.

Simulation Properties

This simulation demonstrates how to create a rectangular waveguide. There are three rectangular waveguides in this simulation. Each has been constructed by creating a physical waveguide on the simulation boundary, and defining the wave that is carried into the simulation. First a metal plate from a box primitive has been placed on the simulation boundary. It is important that this plate extends from at least one cell outside of the simulation boundary to at least one cell inside of the simulation. Next a box primitive corresponding to the size and orientation of the actual waveguide has been created. This is then subtracted from the previously created metal plate. It is important to note here that the polarization parameter will always be parallel to the width. The wave carried in this waveguide is then created by adding a FieldBoundaryCondtion of Rectangular Waveguide. The waveguide surface must be specified to match the intended simulation boundary and on the right location to match the physically constructed waveguide.

Several standard waveguide sizes are available, or User-Defined may be selected to specify a custom size. If no "Turn On Time" is specified, it will be set to a time of 2.5 periods of the carried signal, and a warning will be provided after running the simulation.

Running the Simulation

After performing the above actions, continue as follows:

- Proceed to the run window by pressing the Run button in the left column of buttons.
- Check that you are using these run parameters:
  - *Time Step*: 1.829541541469147e-12
  - *Number of Steps*: 400
  - *Dump Periodicity*: 20
  - *Dump at Time Zero*: Checked

- Click on the *Run* button in the upper left corner. You will see the output of the run in the right pane.

The run has completed when you see the output, "Engine completed successfully."

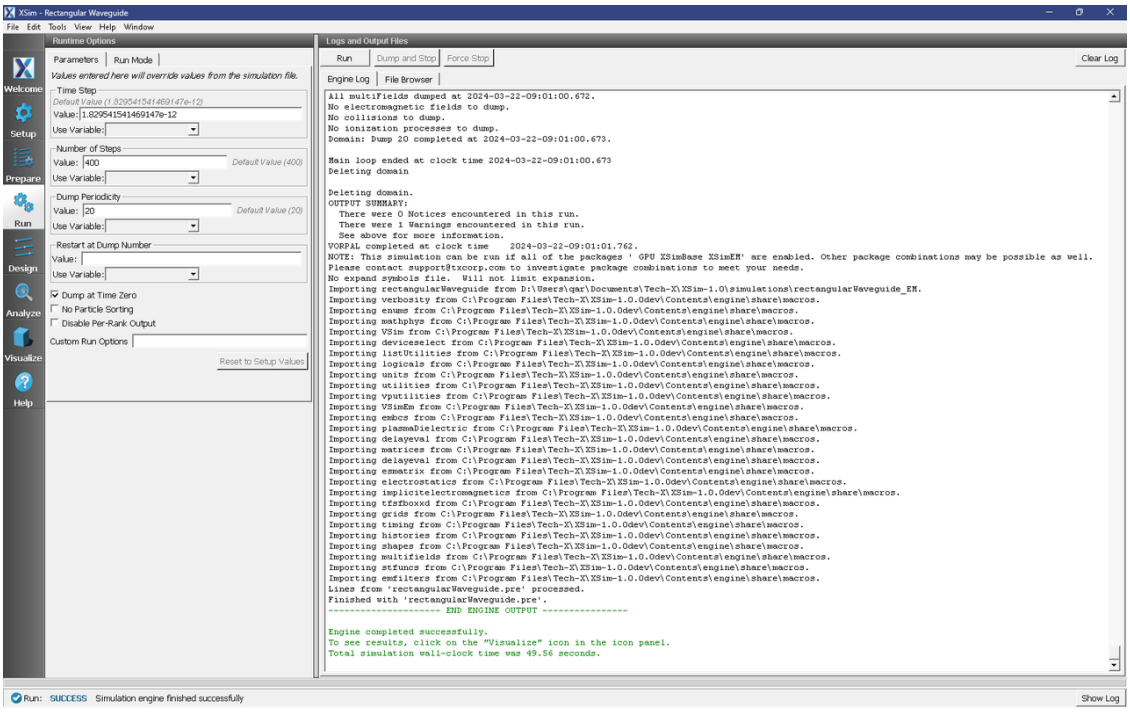

Fig. 2.78: The Run Window at the end of execution.

Visualizing the Results

After a successful run, go to the Visualize Window by pressing *Visualize* in the left column.

Expand *Scalar Data*, E, and select E_y. To slice inside the field, check the *Clip Plot* box. Now step through time using the Dump slider on the bottom of the right pane. This is shown below.

Further Experiments

Waveguides can be added or subtracted to this simulation.

2.2.5 Rectangular Metal Waveguide Dispersion (rectMetalWaveguideDisp.sdf)

Keywords:

`Waveguide, Dispersion Relation, Fourier Transform, Phase Shifting Boundary Conditions`

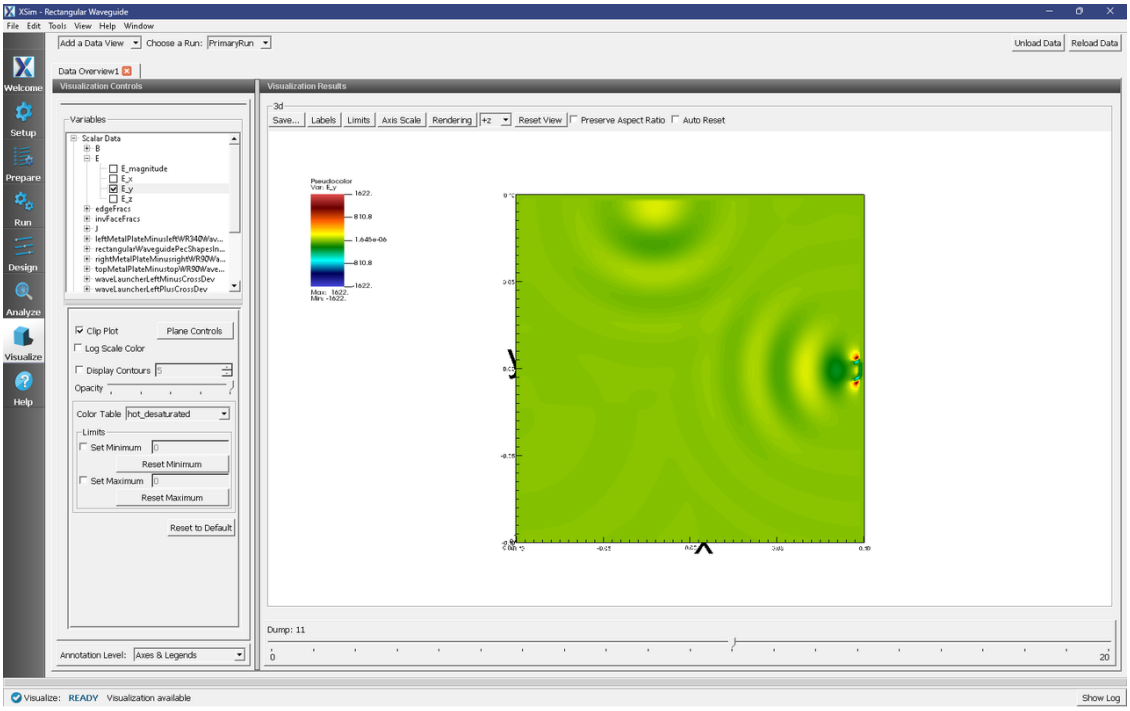

Fig. 2.79: The E_y field propagating out of the two waveguides centered on the z axis.
The effects of the third waveguide can be viewed by adjusting the "Origin of Normal Vector" parameter under the Plane Controls button.

Problem description

This XSimEM example demonstrates several unique capabilities of XSim that can be used to efficiently model waveguides. One unique capability is the use of phase shifting periodic boundary conditions. These work by adding a phase to a wave at a boundary to mimic a much longer physical dimension. We also demonstrate how to use an analyzer called extractModesViaOperator.py to accurately compute the excited modes in the waveguide, even if some of the modes are much weaker than the dominant mode. In this example, we use extractModesViaOperator.py to compute the dispersion relation (ω vs k) and compare the numerically determined dispersion relation with the theoretical dispersion relation using standard waveguide theory.

This example involves five steps. In the first step, the waveguide is "pinged" with a short pulse in the current density that excites a range of modes. The Fourier transform then shows the range of frequencies of the modes. In the second step, the waveguide is excited using a sinc pulse function multiplied by a Gaussian envelope to excite a flat band of frequencies with sharp cutoffs at either end. The excitation current density is in the transverse directions (z and y) which excites an electric field primarily in the transverse direction. In the third step, the data is restarted from the end of the second run and saved at shorter time intervals in order to resolve the frequencies of interest. The output from the third step is used by the extractModesViaOperator.py analyzer to compute the eigenmodes in step 4. Finally, in step 5, the dispersion relation is computed by varying the wavelength which is resolved in the simulation.

Opening the Simulation

The rectMetalWaveguideDisp example is accessed from within XSimComposer by the following actions:

- Go to *File* → *New* → *From Example...*
- In the resulting *Examples* window expand the *XSim for Electromagnetics* option.
- Expand the *Cavities and Waveguides* option.
- Select "Rectangular Metal Waveguide Dispersion" and press the *Choose* button.
- In the resulting dialog, create a New Folder if desired, and press the *Save* button to create a copy of this example.

The properties and values that create the simulation are accessible in the left pane when the Setup Window is selected as shown in Fig. 2.80. The right pane shows a 3D view of the selected geometry components, grids and current distributions.

The geometry can be visualized by expanding "Geometries" in the left pane. The hollow rectangular waveguide can be seen more clearly by de-selecting the "Grid" option. The length in the direction of propagation (x) is a small fraction of either transverse direction. This is possible because we are using "phase shifting periodic" boundary conditions in the *x* -direction. The phase shifting BCs are selected under "Basic Settings".

The simulation is excited with a *sinc hat* function, which has a formula of

$$(t, f_l, f_h, \delta_f, t_{off}) = H(t_{off} - t)\exp(-0.5\delta_f^2(t - 0.5t_{off})^2)\times \frac{\sin(2\pi f_h(t - 0.5t_{off})) - \sin(2\pi f_l(t - 0.5t_{off}))}{(2\pi f_h - 2\pi f_l)(t - 0.5t_{off})}$$

δ_f is calculated according to. frequencyGap = (frequency high - frequency low)*frequency gap factor

numSigma = sqrt(-2.0*log(suppression factor))

sigmaT = (TWOPI*frequencyGap)/numSigma

As mentioned in the introduction, this function has a Fourier spectrum that is fairly flat over the desired range, $f_l < f < f_h$, of frequencies and falls off rapidly over a frequency width of δ_f, so that it is nearly zero for $f < f_l - \delta_f$ or $f > f_h + \delta_f$.

Phase Shifting Boundary Conditions and Phase Shift

Before discussing phase-shifting periodic boundary conditions, we first review ordinary periodic BCs. In this discussion, periodic BC's are applied in the *x*-direction. With normal periodic BC's, there are two criteria in resolving a wave (or mode) of interest. (1) There must be enough grid points along the wavelength to resolve the spatial profile. Typically we sample at least 20 cells along a wavelength (λ_x), or $DX = \lambda_x/20$, and (2) The length of the simulation domain, represented by L_x, must be one wavelength long, $L_x = \lambda_x$. Periodic BC's in the x-direction means that $F(0) = F(L_x)$, where F is any field quantity. Now introduce a grid that extends from 0,1,...,nx and let $x = 0$ at grid point 0 and $x = L_x$ at grid point *nx*. Periodicity on the grid implies $F(0) = F(nx)$. Finally, let's assume that $F(x) \sim \sin(k_x x)$. Then, applying the condition for periodicity, $\sin(0) = \sin(k_x L_x)$, which is exactly met if $L_x = \lambda_x$.

With phase-shifting periodic BC's, we are no longer required to meet the second criteria discussed in the preceding paragraph. To see this, assume $L_x < \lambda_x$. Then the periodicity condition can still be met by setting $k_x L_x - \phi_0 = 0$, where ϕ_0 is the phase shift equal to $k_x L_x$, which is the exact phase shift we have chosen to apply in XSim for this example. The numerical implementation is more challenging than the conceptual picture we just discussed. To implement phase-shifting periodic BC's, we need to treat the fields as complex numbers and set $F(L_x) = \exp(i\phi_0)F(0)$. For the grid, we pick 2 cells such that $L_x = 2DX$.

By using phase-shifting periodic BC's, we can simulate different physical lengths without changing the simulation length L_x through the phase shift $\phi_0 = k_x L_x$. Because $k_x = 2\pi/L_x$, we can solve for $\omega(k_x)$ without requiring a simulation of length L_x.

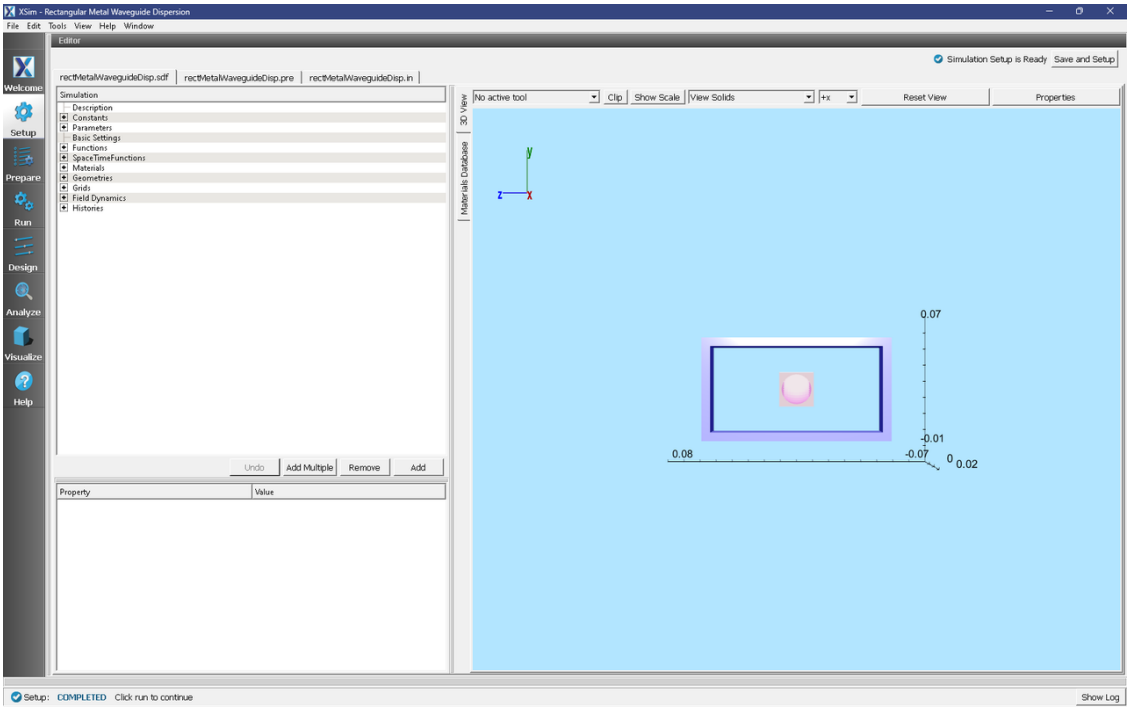

Fig. 2.80: Initial Setup Window for the Waveguide Dispersion example. The block in the middle is the region in which the current density is driven.

Running the Simulation and Analyzing Results

We now walk through the five steps discussed in the Introduction. In the first step, we test to determine the minimum frequency that will propagate through the waveguide. For the rectangular waveguide in this example, we know the analytical result which is the lowest cutoff frequency. However, we still do this step to demonstrate how to determine the lowest frequency that will propagate for cases that are not analytic. In the second step, the current density "rings up" to a maximum value, then rings down to 0. Because we are exciting modes at resonant frequencies of the cavity, the E- and B-fields continue to oscillate after the excitation is turned off. We wish to determine the resonant modes using Eigen mode analysis using data saved in Step 3 when the externally driven source is turned off and the cavity is "ringing" at its natural frequencies. In step 4, extractModesViaOperator.py is used to compute the excitation frequencies in the cavity at a given mode. Finally, in step 5, you will change the value of "mode", which is defined under "Constants" to compute the dominant frequencies which are excited in the cavity at a given wavelength. We assume the maximum wavelength which can be resolved in a traditional periodic simulation is 0.2 m and denote this as *lambdaMax*. We then define the parameter $k_x = \frac{mode}{12}\frac{2\pi}{lambdaMax}$ and run 25 simulations with *mode* =0,1,2,…,24. Finally, we set the phase shift to $k_x \times L_x$ to ensure the correct phase shift is applied for the phase shifting periodic BC's. In this way, we are able to compare the simulation with waveguide theory without explicitly changing the length of the simulation domain in the x -direction.

Step 1: Determining the lowest propagation frequency - The "Ping" Run

The Ping run is used to determine the lowest propagation frequency in the waveguide. In this simulation, we can analytically compute the lowest propagation frequency at a given k based on the allowable modes in a square wave guide. We can then compare the computed lowest propagation frequency with theory. However, for arbitrarily-shaped waveguides, this step is necessary since no theory is available to compute the lowest propagation frequency. We have defined a PINGON constant (with values of 0 or 1) in order to easily transition from the "Ping" run to the "Excitation" run in Step 2. The default is PINGON=1 to do the "Ping" run first. For Step 1, perform the following steps

- Click on the Run Window.
- This simulation may be accelerated by changing the Run Mode to Parallel.
- Click the *Run* button on the upper left corner of the *Logs and Output Files* pane

The run has completed when you see the output, "Engine completed successfully." A snapshot of the simulation run completion is shown in Fig. 2.81.

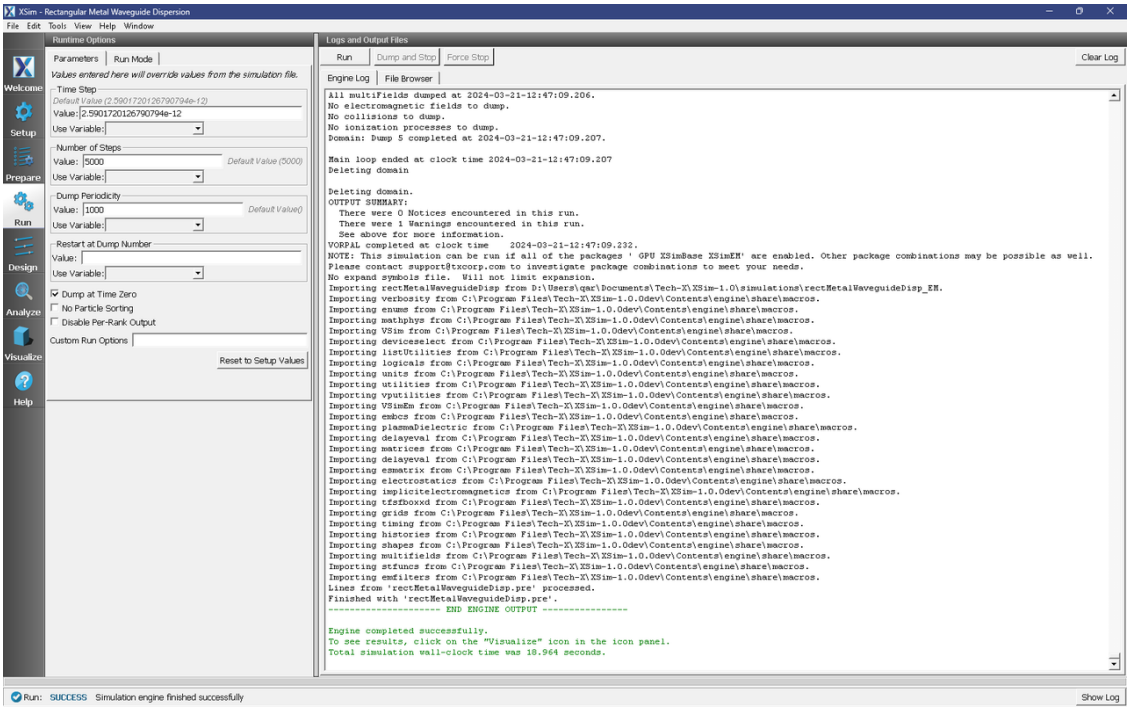

Fig. 2.81: The Run window at the end of a Step 1.

After the run has completed, click on the *Visualize* tab on the left side of the visualization window. After the data has loaded, click on *Add a Data View* and then select *History*. Perform the following steps to analyze the history:

- Graph 1 select *jMid_1*
- Graph 2 select *eMid_1*

In Graph 1, to see the data, click on the "Zoom" option above the plot. Left click with your mouse on the upper left corner of the graph and drag the "zoom" box to about 0.2×10^{-9} s (hold the left mouse button when you drag). You should see a step function in the current density beginning at t=0 and ending abruptly after about 20 time steps. This is the "ping" that we impose on the simulation. To determine the lowest frequency mode that we excite, click the "Fourier Amplitudes (dB)" box in Graph 2. Again, zoom in on the left most portion of the plot. You will see that the first peak occurs at a frequency of about 1.5×10^{9} Hz, which is the lowest cutoff frequency in the simulation and is what we find analytically using waveguide theory. Therefore, we have confirmed that XSim reproduces linear theory. You can

confidently use XSim to numerically determine the lowest cutoff frequency for non-analytically solvable shapes. The visualization window after the above steps have been performed is shown in Fig. 2.82

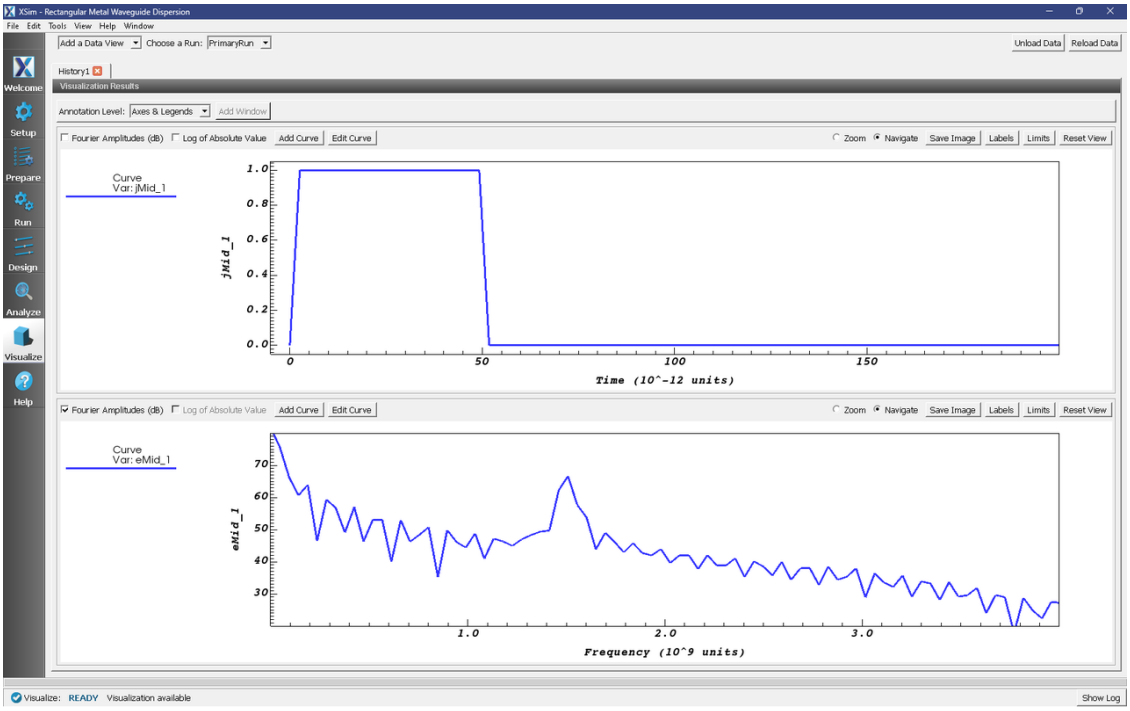

Fig. 2.82: History plots showing the lowest cutoff frequency at 1.5×10^9 Hz as the result of pinging the waveguide with an abruptly applied current density.

Step 2: Excitation

From Step 1, we know that the lowest propagation frequency is $\sim 1.5 \times 10^9$ Hz. We can now check that the lowest cutoff frequency (*frequency low* in *SincHat*) is correct. The parameter *FREQ_MIN* is passed in to *frequency low* and is indeed $\sim 1.5 \times 10^9$ Hz. We can now proceed to excite the waveguide with confidence knowing that lowest excitation frequency is correct. To excite the waveguide, perform the following steps:

- Go to the Setup Window and expand the *Constants* option
- Set *PINGON* to 0
- Click on *Save and Setup* in the upper right corner of the visualization window
- Go to the Run Window by pressing the Run button in the left column of buttons.
- Set Number of Steps to 21000. This ensures that the simulation runs long enough for the current density to "ring up" then "ring down" to 0.
- Leave the "Dump Periodity" at 1000.
- This simulation may be accelerated by running on multiple MPI ranks. The parallel options are in the *Run Mode* tab
- To run the file, click on the *Run* button in the upper left corner of the *Logs and Output Files* pane. You will see the output of the run in this pane. The run has completed when you see the output, "Engine completed successfully." A snapshot of the simulation run completion is shown in Fig. 2.83.

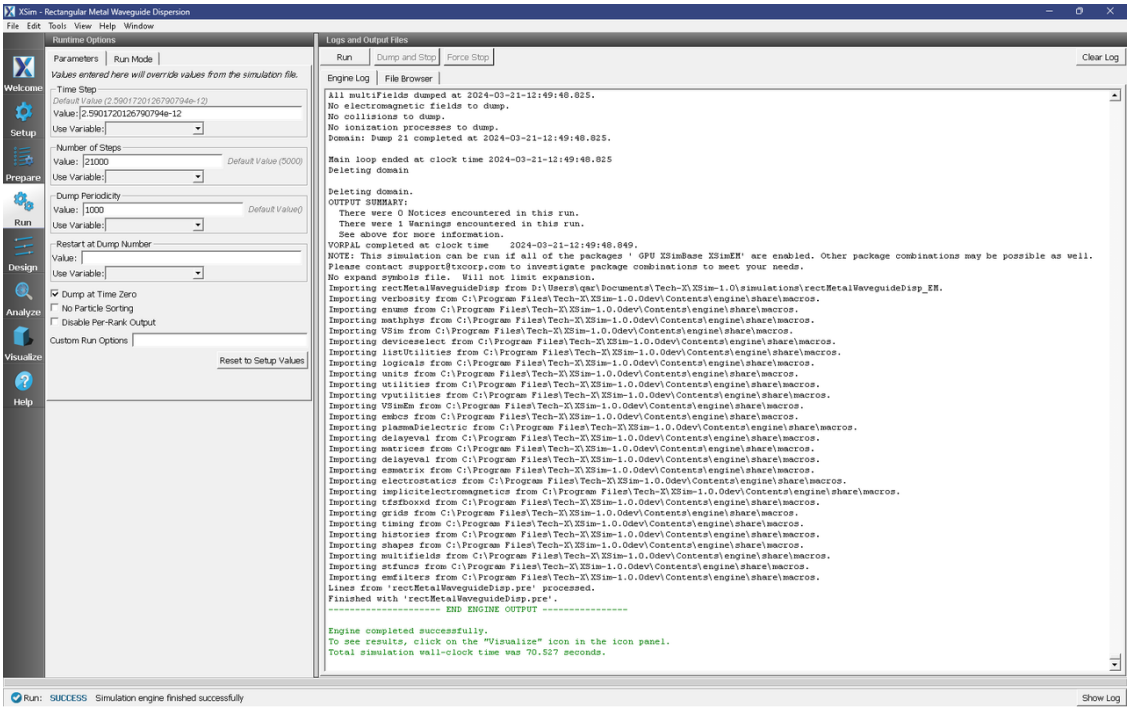

Fig. 2.83: The Run window at the end of a Step 2.

Step 3: Evolving the excited cavity

The purpose of Step 3 is to continue the simulation with only the *E*- and *B*- fields excited due to the externally imposed current density from Step 2. We will also save more data which we can analyze using Eigenmode analysis for Step 4.

- Go to the Setup Window and expand the *Basic Settings* option.
- Change “number of steps” to “10000”.
- Change “steps between dumps” to “100”.
- Change “dump in groups of” to “3”.
- Click on *Save and Setup* in the upper right corner of the visualization window.
- Go to the Run Window and click the “Reset to Setup Values” button in the lower right corner of the “Run Options” section.
- In the *Run Options* section, uncheck the *Dump at Time Zero* box and set the *Restart at Dump Number* to 21.
- NOTE: the “Dump Periodicity” must be blank for the “dump in groups of 3” setting to work. In this case, the simulation will revert to the values defined in “Basic Settings” which are to dump every 100 time steps in groups of 3 (21000,21001,21002, . . . , 21100,21101,21102,. . . , etc). If you fill in an integer into Dump Periodicity, the default gets overwritten and the data are not dumped in groups of 3, which is required for the extractModesViaOperator.py analyzer. So, there should be 300 new dumps making a total of 321 dumps.
- Click run. The run has completed when you see the output, “Engine completed successfully.” A snapshot of the simulation run completion is shown in Fig. 2.84. When this run is finished, the last step should be step 31000.

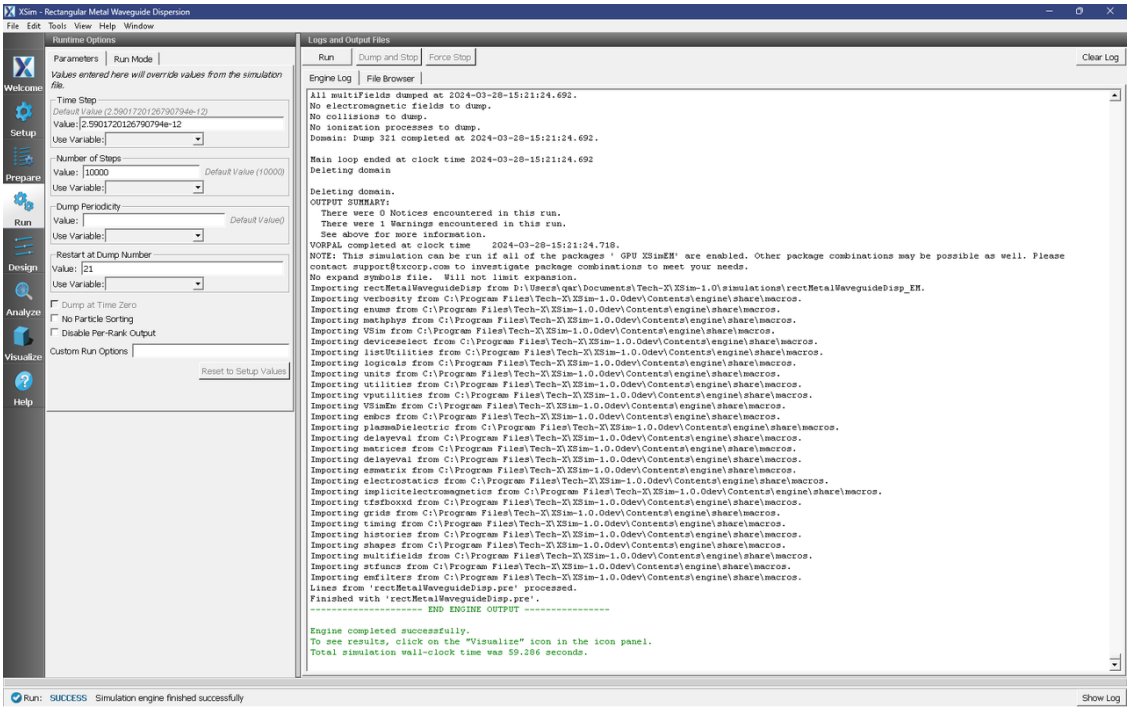

Fig. 2.84: The Run window at the end of Step 3.

Step 4: Computing the eigenmodes

- Go to the analyzer window by selecting *Analyze* in the left column.
- Select *extractModesViaOperator.py* from the list of available analyzers. Then click "Open" on the top right of the *Analysis Controls* pane.
- Compute the electric field eigenfunctions. After the analyzer loads, ensure the following parameters are entered:
  - **simulationName:** "rectMetalWaveguideDisp"
  - **outputsimName:** leave blank
  - **realFields:** "E"
  - **imagFields:** leave blank
  - **secondFieldFactor:** leave blank
  - **operator:** "d2dt2"
  - **dumpRange:** "21:321"
  - **cellSamples:** ":,5:25:5,5:45:5"
  - **cutoff:** "1e-12"
  - **maxNumModes:** "-1"
  - **initialModeNumber:** "0"
  - **normalizeModes:** checked
  - **testing:** leave blank
  - **overwrite:** checked

Also, check the "Overwrite Existing Files" box. The dump range runs only over the data saved in step 3 and we are only sampling every 5 cells in the z- and y- directions which is adequate to compute the eigen modes. Double-check your entries against what is shown in Fig. 2.85. After you run the analyzer, you will need to scroll up to find the computed frequencies. The screenshot shown in Fig. 2.85 is from a visualization window that is scrolled up so that the computed frequencies can be compared with what you ran. Figure Fig. 2.85 shows that there are 6 modes, but two are degenerate resulting in 5 unique modes. One indication that you are using extractModesViaOperator.py correctly is that the imaginary part of the frequency (which represents attenuation of the wave) is nearly 0 or at least much smaller than the real part. The lowest excited mode is $\sim 1.5 \times 10^9$ Hz, which is discussed further below in the context of standard waveguide theory.

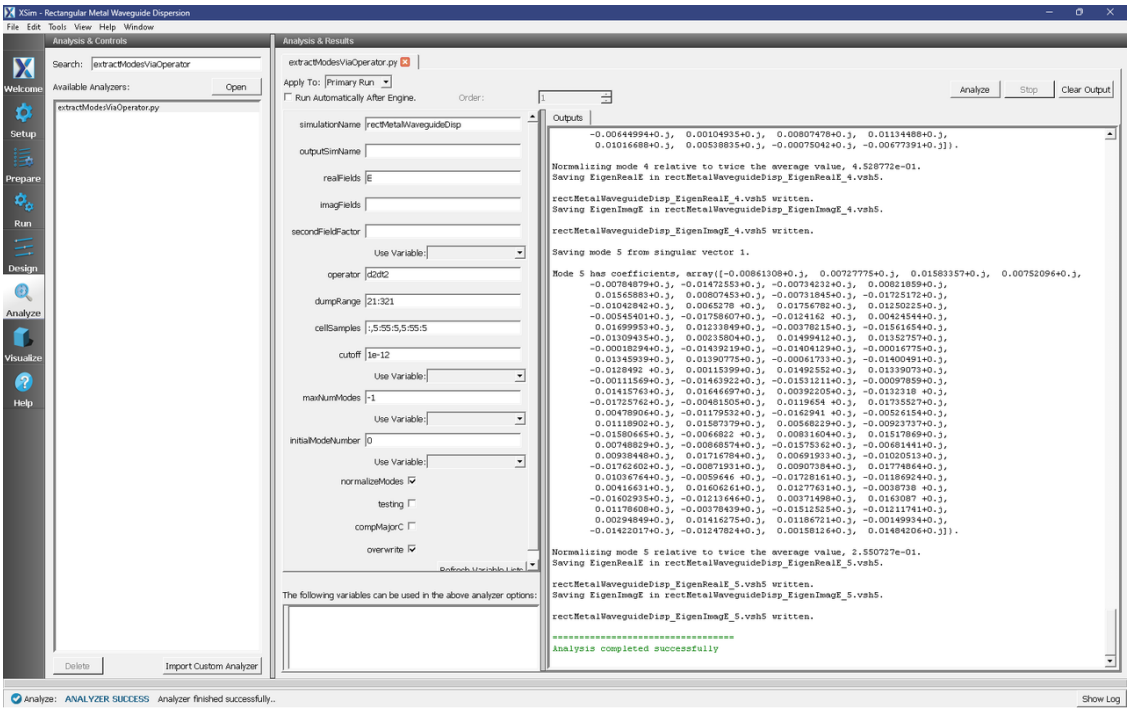

Fig. 2.85: Computing the electric field eigenfunctions and frequencies using the extractModesViaOperator.py analyzer.

Step 5: Computing the Dispersion Relation

Once the Eigenmode analysis is complete in Step 4, it is necessary to record the three lowest-order modes. The pre-loaded value of *mode* is 0, which means we are simulating $k_x = 0$. The waveguide dispersion relation is given by $\omega^2 = k_x^2 c^2 + \omega_{mn}^2$, where $\omega_{mn}^2 = c^2\pi^2\left((m/a)^2 + (n/b)^2\right)$, and where a and b are the y and z dimensions, respectively and m and n are integers. In this simulation, we analyzed the (m, n) = (1,0),(0,1) and (1,1) modes. Furthermore, the frequency range that is excited lies between $f_l = 1.5 \times 10^9$ Hz (which is the lowest allowable propagation frequency) and $f_h = 4.5 \times 10^9$ Hz. Given these parameters, we expect from waveguide theory for the three lowest cutoffs to be $\sim 1.5 \times 10^9, 3 \times 10^9, 3.35 \times 10^9$ Hz, which is what is found from extractModesViaOperator.py. Repeating these steps for modes 1 to 24 yields the dispersion relation shown in Fig. 2.86. The overlap between the simulation results and theoretical values is evident in Fig. 2.86.

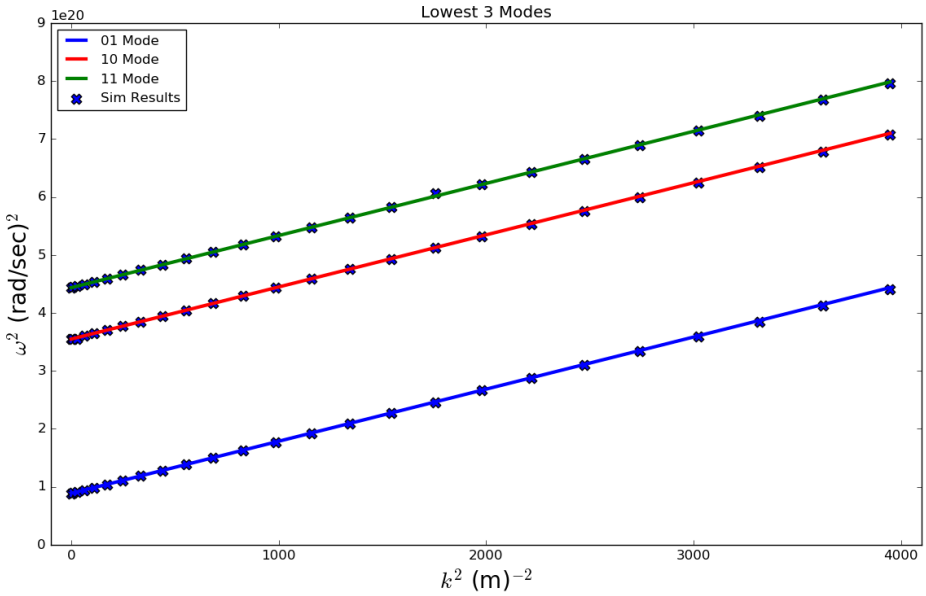

Fig. 2.86: Dispersion relation found by changing *mode* in XSim. The solid line is the theoretical dispersion relation $\omega^2 = k_x^2 c^2 + \omega_{mn}^2$. The 'X's represent data from the simulation results.

Convergence Study

To demonstrate that the simulation results converge to the theoretical value, we have performed a series of simulations in which $DX = DY = DZ$ are varied. The values chosen are *DX* = 0.205, 0.41, 0.50 and 0.615 cm. We then computed the lowest propagation frequency using *extractModesViaOperator.py* and plotted this frequency versus DX^2. Using Richardson extrapolation, we then compute the lowest propagation frequency for $DX^2 = 0$, which provides a better comparison with the theoretical frequency than a simulation with finite DX^2. The field calculation error scales approximately with DX^2 so a plot of frequency versus DX^2 should be a line. The linear correlation between frequency and DX^2 is shown in Fig. 2.87. The extrapolated frequency at $DX^2 = 0$ is $\sim 1.4992 \times 10^9$ Hz. The theoretical value is $\sim 1.4990 \times 10^9$. We see from Fig. 2.87 that the computation of the lowest propagation frequency for the DC mode converges with order DX^2, which is expected since the field solve is 2nd order accurate. Furthermore, using Richardson extrapolation, we see that the difference between theory and simulation, with $DX^2 = 0$, is 0.02 %.

Visualizing the results

After performing the above actions, continue as follows:

- Proceed to the Visualize Window by pressing the Visualize button in the left column of buttons.

The first step will be to ensure that we are driving the current density as expected and that the *E*- and *B*- fields are "ringing" as a result of driving the current density. Follow these steps to perform this check:

- Click on the *Add a Data View* pull-down menu at the top of the visualization window
- Click on *History*. This will open a new tab.
- Click on *Reload Data*.
- Under Graph 1 plot *jMid_1*. This is the *y*- component of the current density
- Under Graph 2 plot *eMid_1*.

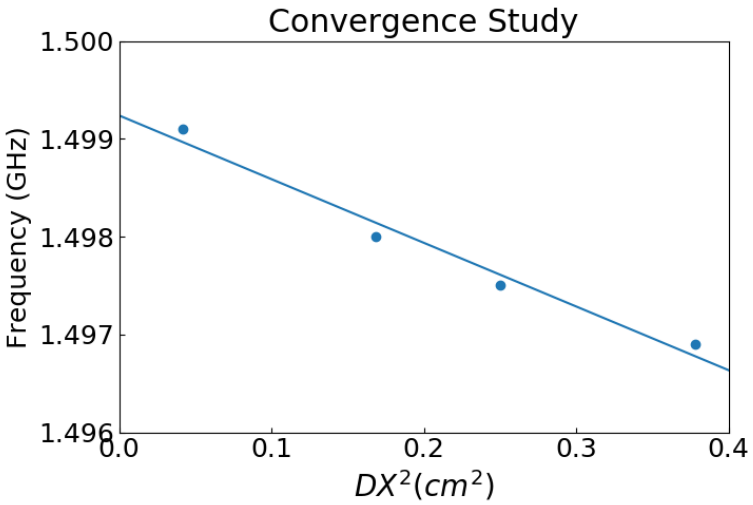

Fig. 2.87: Plot demonstrating convergence of frequency calculation.

- Under Graph 2 click the reset button to put the axis scaling back from the first plots make in Step 1 above.

Your plots should be similar to Fig. 2.88. The current density is driven for a short time period. However, because we are driving resonant modes, the current density excites the transverse electric field and parallel magnetic field. You can also plot *eMid_0*, *bMid_1*, and *bMid_2* to compare with Fig. 2.88. We have driven a TE mode since *eMid_0* is nearly 0.

We next wish to examine the spectral characteristics of the current density, which is driven between *FREQ_MIN* and *FREQ_MAX*. Therefore, the Fourier Transform of J_y or J_z should be greatest in this frequency range. Returning to the History visualization window, under Graphs 2, 3, and 4, change the plotted quantity to *none*, so that only Graph 1 shows a plot. Now check the "Fourier Amplitudes (dB)" box in the upper left corner. Finally, check the "Zoom" box on the right side of the visualization window and highlight from 0 to 10^{10} which will expand the lower frequency part of the plot. After you have zoomed in on the plot, re-check the "Navigate" box. The result of these steps should lead to a plot that looks similar to Fig. 2.89. The driving frequencies lie between *FREQ_MIN* and *FREQ_MAX* as expected. The rate at which the Fourier signal dampens below *FREQ_MIN* and above *FREQ_MAX* depends on the Gaussian envelope and the parameter called *OMEGA_SIGMA*.

Further Experiments

The result of varying *mode* from 0 to 24 is shown in Fig. 2.86. One experiment you can perform is to reproduce Fig. 2.86 by running 25 simulations with *mode* varying from 0 to 24. Each simulation takes just a few minutes so the 25 simulations take about one hour. For each simulation record the three lowest frequencies that are excited using extractModesViaOperator.py.

Another experiment would be to change the dimensions of the waveguide. The Constants *a* and *b* are used to determine the lowest frequency that will propagate in the waveguide. Therefore, changing *a* and *b* will automatically compute *FREQ_CUTOFF_TE*. You will then need to change *BGNY*, *ENDY*, *BGNZ*, and *ENDZ* accordingly. Finally, you will need to modify the dimensions and position of the primitives constructed under *Geometries* which are used to construct the waveguide.

A third experiment would be to modify *FREQ_MIN* and *FREQ_MAX* and compute the dispersion curve for these new values. No mode will propagate below *FREQ_CUTOFF_TE* so do not set *FREQ_MIN* below *FREQ_CUTOFF_TE*.

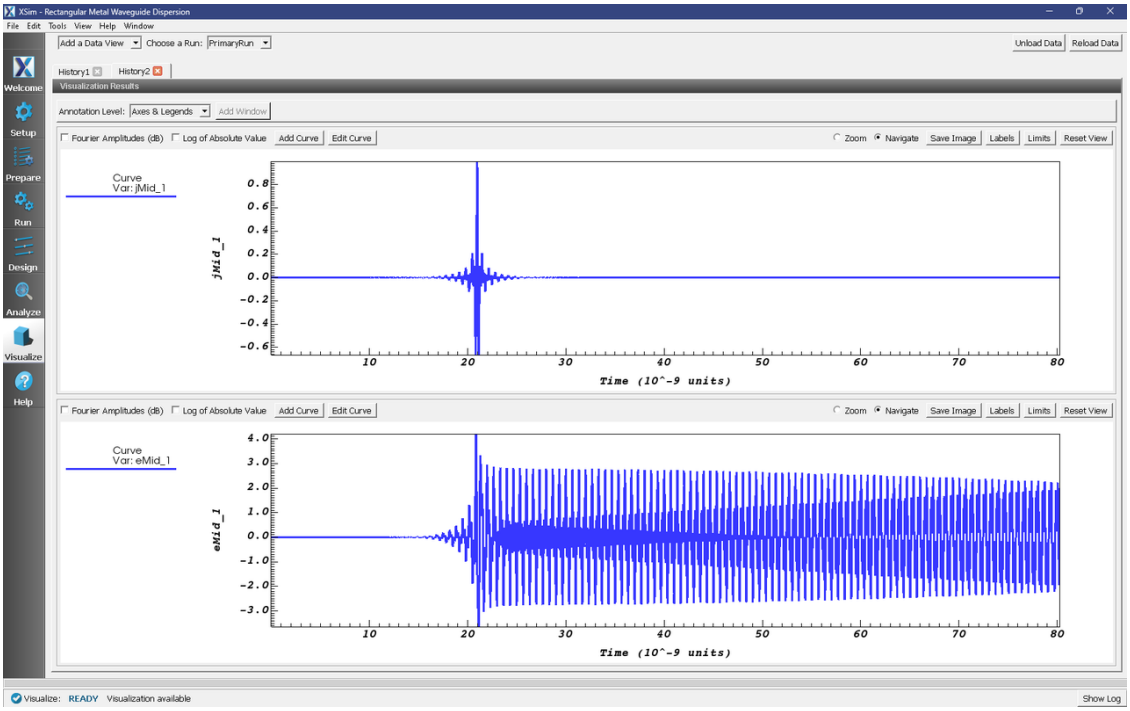

Fig. 2.88: History plots showing the modes driven in this simulation

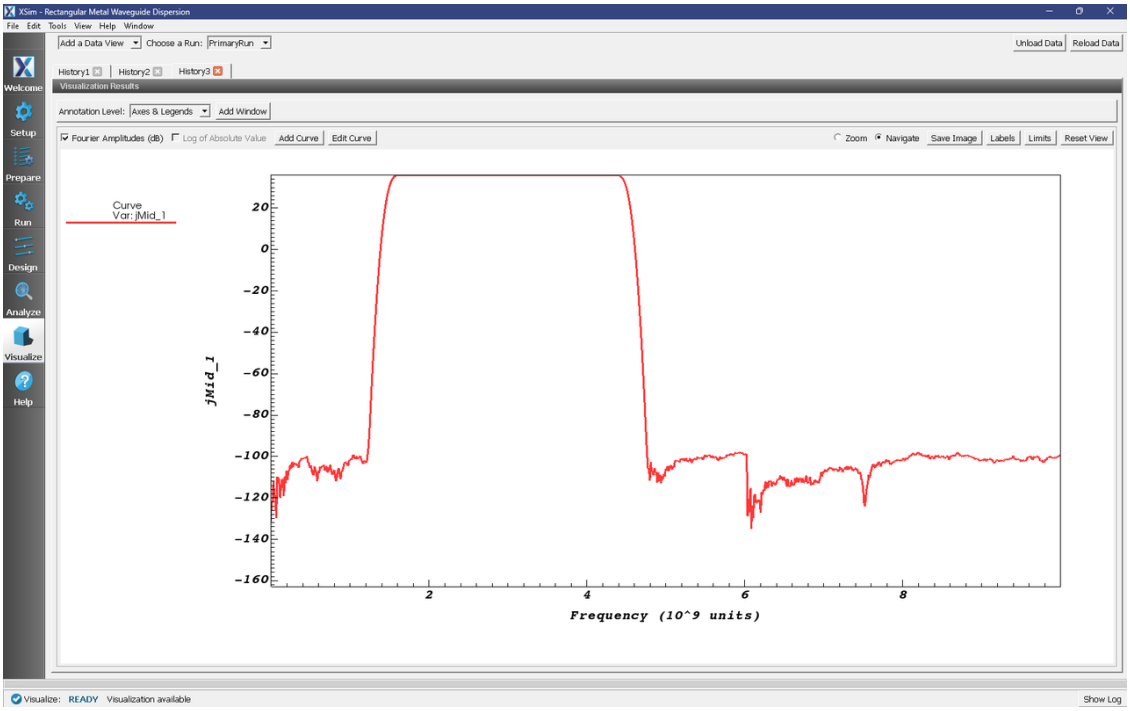

Fig. 2.89: Fourier Transform of the current density showing that the current density is mainly driven between *FREQ_MIN* and *FREQ_MAX* as expected.

Lastly, you can try to excite a TM mode instead of a TE mode.

2.2.6 S-Matrix of Box Cavity (sMatrix.sdf)

Keywords:

electromagnetics, sMatrix

Problem description

A common measurement made on a 2-port RF device is reflection and transmission of an RF signal, for either a single frequency, or for a range of frequencies. This measurement results in the Scattering-Matrix, or S-Matrix, whose elements S11 and S21 are the reflected and transmitted signal for unit input at Port 1. Xsim provides the capability to simulate these S-Matrix parameters for arbitrarily complex devices connected to waveguides propagating TE, TM, and TEM modes. To demonstrate this capability, we show in this example how to measure S11 and S21 in a dual-mode cavity filter, connected to a WR-90 waveguide, with the narrow-band band-pass tuned to pass frequencies between 9.95 and 10.05 GHz.

The Dual Mode Cavity Filter operates by coupling the TE01 waveguide mode into the two nearly degenerate TE102 and TE201 modes of the cavity, since the length of the cavity is very close to its width. The differences in these values, along with the symmetry breaking along the waveguide axis, determine the frequency separation of the two modes. This separation is what gives the filter finite-bandwidth since frequencies between these modes are passed, and frequencies above or below the modes are rejected. A pole in the transmitted signal just below the band contributes to sharpness of the band's lower edge, but this pole moves easily to the upper frequency edge with small adjustments to the cavity dimension parameters, and the user is encouraged to experiment in finding optimal placement of this pole. Some relevant parameters are shown in Fig. 2.90.

Opening the Simulation

The Scattering Matrix example is accessed from within XsimComposer by the following actions:

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *XSim for Electromagnetics* option.
- Expand the *Cavities and Waveguides* option.
- Select "S-Matrix of Box Cavity" and press the *Choose* button.
- In the resulting dialog, create a New Folder if desired, and press the *Save* button to create a copy of this example.

The *Setup Window* is now shown with the waveguide in the *3D View*. Fig. 2.91.

Simulation Properties

The simulation geometry consists of a standard WR-90 rectangular waveguide with the filter cavity (also referred to as the Device-Under-Test (DUT) in this writeup) in the center. A planar antenna in the waveguide, near the DUT, launches the incident wave while allowing reflected signals to pass through into the waveguide behind it. The antenna is constructed of two planar current sources with each current source one cell thick in the x-direction and directly adjacent to each other along x. The amplitudes and phases of the current sources are tuned so that the electric and magnetic fields which launch in the -x direction cancel whereas in the +x direction, the fields add to a non-trivial value. The waveguide ends are terminated in gradual absorbing layers with negligible reflection, and the reflected and transmitted signals are measured just in front of these absorbers.

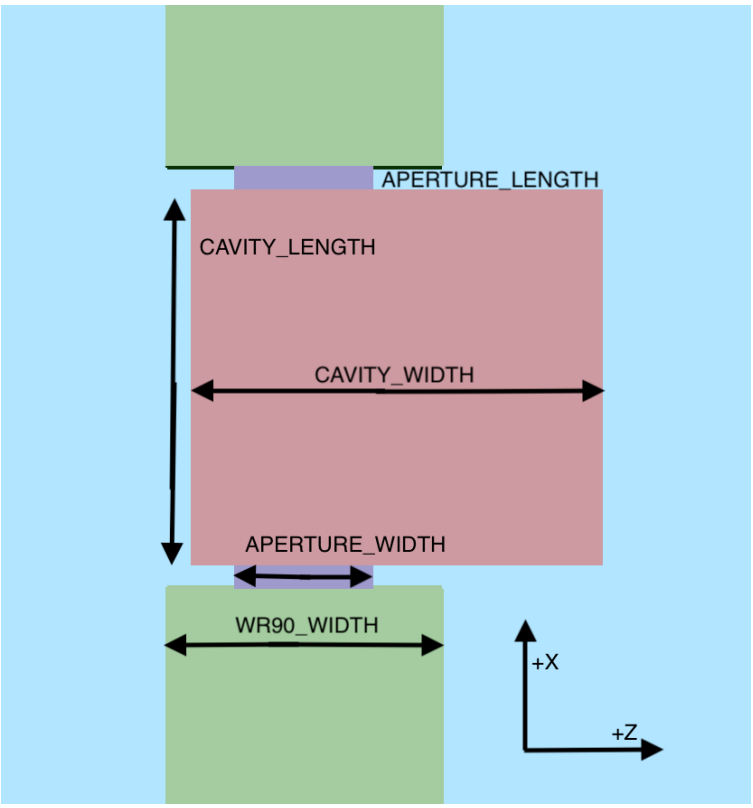

Fig. 2.90: Some relevant parameters for the S-Matrix Box Cavity example.

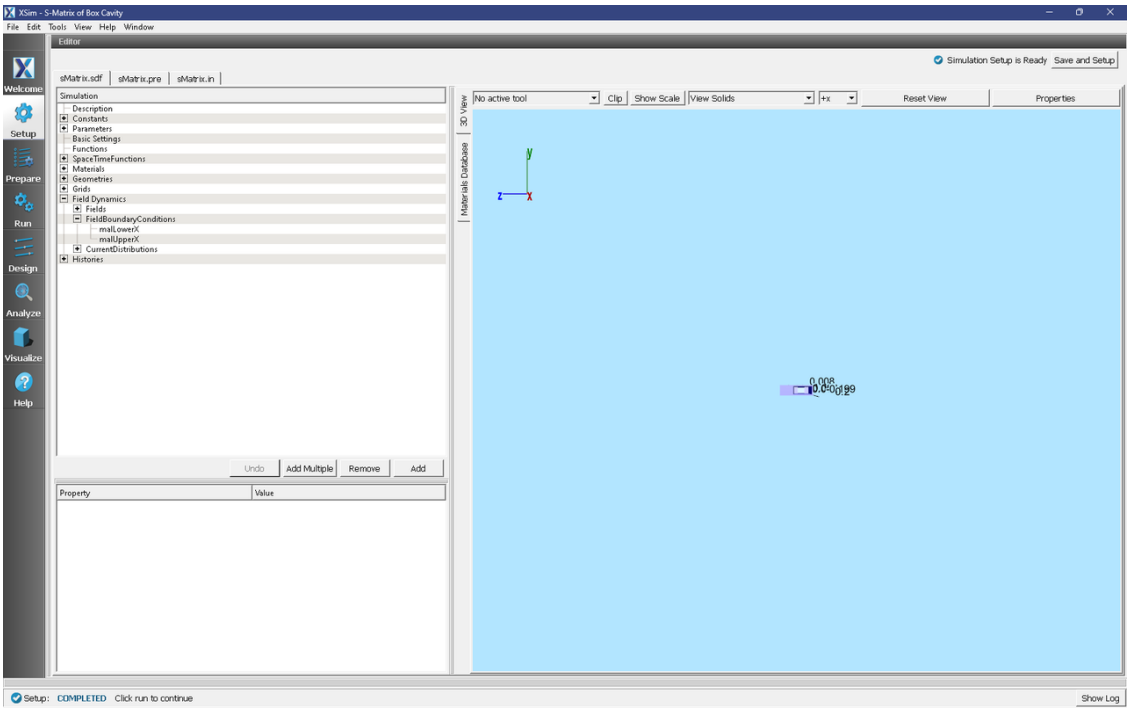

Fig. 2.91: Setup Window for the Scattering Matrix example.

This example is parameterized in the waveguide and DUT geometry specification, allowing for easy modification to either. Thus this example is effectively a template for an S-Matrix simulation of any device. The time histories of voltage signals used to measure S11 and S21 are also built in and automated for easy substitution. Furthermore, these signals are easily turned into S11 and S21 frequency variation curves using the standard "Fourier Amplitudes" capabilities in XsimComposer, or if single frequency, then the S11 and S21 values are just the amplitudes of the signals.

The x axis is the axial direction of the waveguide, with the parameters WAVEGUIDE_LENGTH, APERTURE_LENGTH, and CAVITY_LENGTH controlling the lengths of each component. These parameters are also used to control the position of each component, allowing a change to one of them to properly adjust the component positions.

The height (Y axis) of each component is standardized to the parameter WR90_HEIGHT, and centered around the Y axis.

The waveguide and aperture widths are centered around the Z axis, while the cavity is not.

The excited mode is the standard lowest mode, TE01, and in particular note that for this mode, the Ez component of the field is zero. The spatial profile of the current source is consistent with the TE01 mode. Since we are launching a TE01 mode, the perpendicular component of the electric field is 0 at the boundaries. Because the larger waveguide dimension is along z the lowest order mode coincides with the y-component of the electric field being excited. Therefore the function "waveguideEyProfile" is the spatial function used in the current source in order to launch the desired waveguide mode.

Running the simulation

After performing the above actions, continue as follows:

- Proceed to the run window by pressing the Run button in the left column of buttons.
- Check that you are using these run parameters:
  - *Time Step*: 1.2250750302237241e-12
  - *Number of Steps*: 17899
  - *Dump Periodicity*: 182
  - *Dump at Time Zero*: Checked
- Click on the *Run* button in the upper left corner of the right pane.

You will see the output of the run in the right pane. The run has completed when you see the output, "Engine completed successfully". This result is shown in Fig. 2.92.

This example is more sophisticated than some of the others, in that successful determination of S-Matrix parameters is not the result of a single run, but rather a result of a procedure involving several runs. This includes at least one Calibration Run, and at least one Data Run to determine S11 and S21. Below we discuss in detail some of the features of this example.

Frequency Band vs. Single Frequency

The user may choose whether to compute a single-frequency value of the S-Matrix parameters, or to compute the variation of the parameters as a function of frequency across a user specified frequency band. The constant, FREQCENTER, specifies either the single frequency or the center frequency of the band. The constant, FREQBANDWIDTH, provides the bandwidth or is set to 0 if a single-frequency simulation is desired.

With a single frequency simulation, the constant, NUMBEROFCYCLESTODRIVE, should be large enough to ensure that the S11 and S21 histories reach a steady amplitude. The *History* data view can be used to obtain the S-Matrix value, which is just the amplitude of the signal.

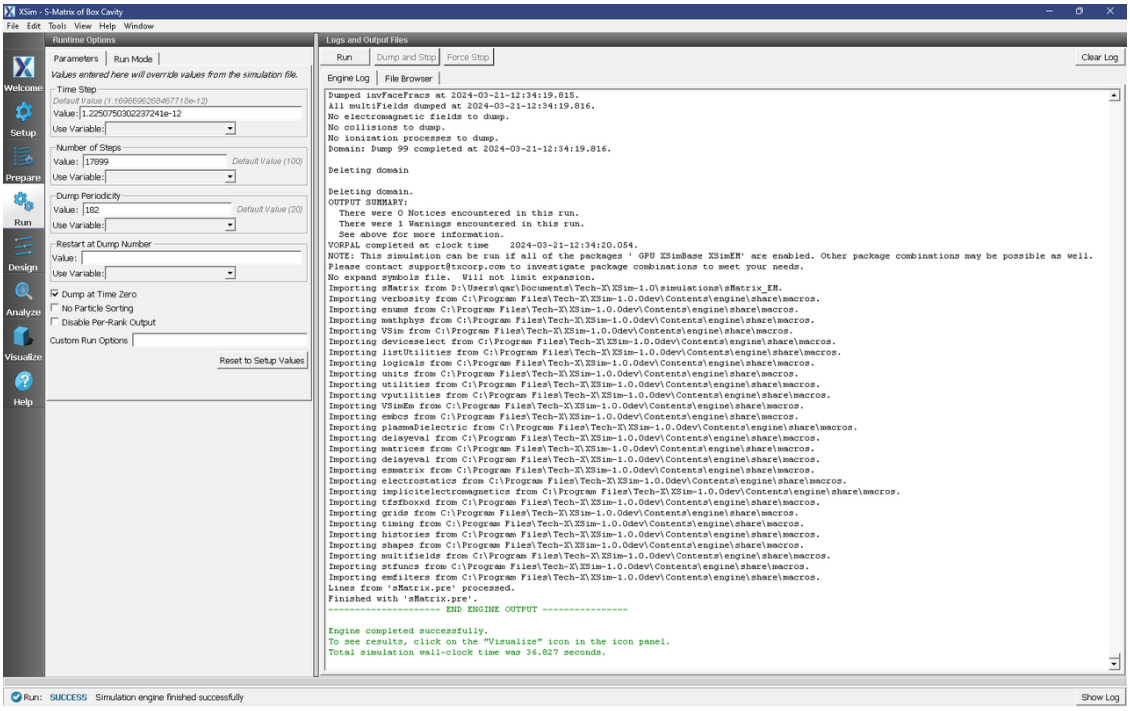

Fig. 2.92: The Run Window at the end of execution.

With a finite frequency band, the same constant, NUMBEROFCYCLESTODRIVE, can be adjusted upward to increase the detail and sharpness of the S-Matrix variation with frequency. In particular, increasing NUMBEROFCYCLESTODRIVE causes the FFT of the S11 and S21 histories to attain a sharper cutoff at FREQHI and FREQLO. The constant, NUMBEROFCYCLESTOCOAST, may also need to be adjusted upward if the DUT contains internal mode oscillation of large Q (quality) factor. This variable needs to be large enough so that the signal histories have decreased to a negligible value (10^{-4}, relative to maximum) by the end of the simulation. The *FFT* button in the *History* data view is then used to give the S-Matrix variation with frequency, with the plot's Y-axis units being dB. Be aware that it is usually necessary to zoom in significantly on this plot in order to see the frequency band of interest.

Finally, in both these cases, only the amplitude of the complex-valued S-Matrix parameters can be obtained with XsimComposer. More sophisticated post-processing (not covered in this example) is needed in order to get the phase information.

Calibration Run

The Calibration Run is done first, and the user must ensure that in the geometries, only the material of the object *myCalibrationWaveguide* is set to PEC, i.e., ensuring the material of the object *myWaveguideAndDUT* is set to empty. In the Calibration Run, the DUT is automatically omitted and replaced with a continuation of the waveguide, so that this is a near trivial simulation of a straight length of waveguide that should have effectively 100% transmission. The calibration run serves two very important purposes:

To ensure that there is negligible (below 1% amplitude, -40 dB) reflected voltage (S11). If the reflection is too high it indicates that either the absorbing boundaries (MAL's) are not working well enough, or that the waveguide's "modeProfile" description is not accurate enough, and/or that there is not enough grid resolution. To decrease reflections at the MAL boundaries, you can decrease the "damping factor" in the MAL boundary condition. If you do this, you may need to increase the length of the MAL. Another potential issue is the value of TRISE, which is the length of time over which the current source is turned in the single frequency runs. Increasing TRISE could lead to a better measure of S11 and S21.

To adjust the DRIVENORMALIZATION constant, which runs in proportion to observed transmitted voltage (S21), so that the next time the calibration run is done, the transmitted voltage (S21) will be exactly unit amplitude (single frequency) or zero dB (across frequency band). For example, if the first Calibration Run shows an amplitude of 0.667 for S21, change DRIVENORMALIZATION to 1.5 times its present value for the next Calibration Run, since 1/0.667 = 1.5.

Changing center frequency, or any waveguide parameter, or even the nominal cell size, will require re-calibration. If not sure, always recalibrate, when changing a parameter.

Data Run

Once the Calibration Run is successful at achieving unit transmission with negligible reflection, the Data Run is then done. The user should ensure that only the material of the object *myWaveguideAndDUT* is set to PEC, i.e., ensuring the material of the object *metalMinuscalibrationWaveguide* is set to empty.

Visualizing the results

After performing the above actions, continue as follows:

Proceed to the Visualize Window by pressing the Visualize button in the left column of buttons.

The S-Matrix results are shown by adding a *History* data view. Once the history graphs are displayed check *Fourier Amplitudes* to get an FFT of the data. To look at the results from 8 to 12 GHz, click *Limits* in the upper right corner of each graph and set the X-axis lower limit to 6e9, and upper limit to 14e9.

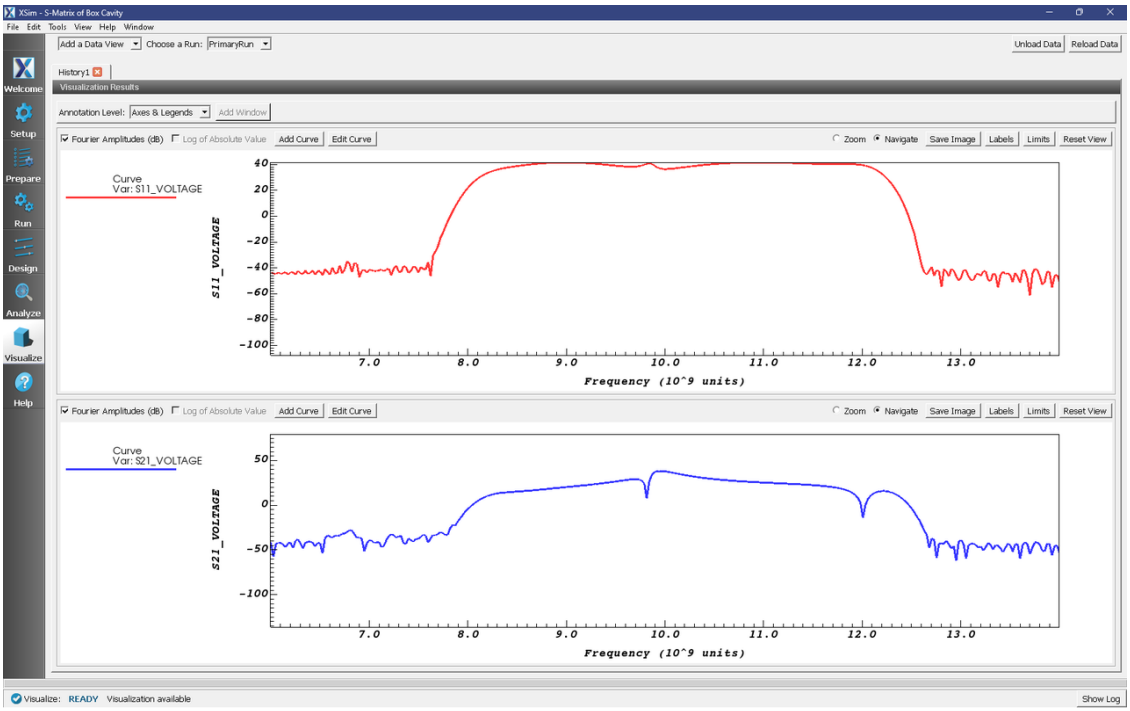

Fig. 2.93: Fourier transforms of the histories S11_Voltage and S21_Voltage as a function of frequency (in GHz).

Further Experiments

Experiment with finding optimal placement of the pole in the transmitted signal.

2.3 Photonics

2.3.1 Dielectric Waveguide Mode Calculation (dielectricWaveguideModeCalc.sdf)

Keywords:

Mode Extraction, Photonic Waveguide, Guided Mode, Semiconductor

Problem Description

This example demonstrates the process for extracting the effective index and fields of a guided mode by directly solving an eigenvalue equation. The use of permittivity averaging enables second order accuracy in our solution. The waveguide axis runs parallel to the x-axis, and is surrounded by a background cladding with a greater permittivity. We will run the simulation for 1 step and then use the dielectricWaveguideModeCalc_invEps_0.h5 file to solve for the guided modes using the computeDielectricModes.py analyzer. This analyzer will find the fundamental mode of this waveguide and output this mode into a separate .vsh5 file. This mode file is then used to launch the extracted mode into the simulation using the Wave Launcher.

Eigenmodes in such a simulation have the form:

$$\mathbf{E}(\mathbf{x}, t) = \mathbf{E}(y, z)e^{i(kx-\omega t)}$$

The effective index of refraction of a waveguide mode is given by $\bar{n} = k/k_0$ where $k_0 = \omega/c$. If the waveguide has index of refraction n_w and the cladding $n_c < n_w$, then a *guided* mode will have a modal index in the range, $n_c < \bar{n} < n_w$.

This simulation can be performed with a XSimEM license.

Opening the Simulation

To open this example open an instance of XSimComposer and follow the steps below:

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *XSim for Electromagnetics* option.
- Expand the *Photonics* option.
- Select *Dielectric Waveguide Mode Calculation* and press the *Choose* button.
- In the resulting dialog, create a *New Folder* if desired, and press the *Save* button to create a copy of this example.

Simulation Variables

This example contains a number of variables defined to make the simulation easily modifiable.

- EPS_SILICON and EPS_SILICA: Relative permittivities of silicon and silica. These constants are used in multiple parameters and in the accompanying Python file for solving the waveguide mode.
- WAVELENGTH_VAC: Wavelength of the input signal. This wavelength is also used for the calculation of the fundamental guided mode of the device.

The *Materials* section contains just silicon and silica. The *Geometries* section includes the CSG waveguide and its defining parameters. In *Field Dynamics*, there are *FieldBoundaryConditions* and *CurrentDistributions* to be aware of. In photonics simulations, Matched Absorbing Layers (MALs) are the most stable boundary conditions for preventing reflections. This simulation makes use of XSim's Wave Launcher to launch a unidirectional wave down the waveguide with a mode defined by the file generated by the computeWaveguideModes.py analyzer.

Setting up the Pre-run Simulation

As delivered, the system is set up to generate the data needed to run the computeDielectricModes.py analyzer. To ensure that your simulation has second order accuracy, under *Basic Settings*, verify that the *dielectric solver* field is set to *permittivity averaging*. This algorithm is a powerful XSim feature. This setting is shown in Fig. 2.94. It is also necessary to make sure the wave launcher is deactivated before running for the first simulation since the mode file has not been generated until after running the computeWaveguideModes.py analyzer. To do it, expand *Field Dynamics*, expand *CurrentDistributions* and verify that *waveLauncher1* is *{inactive}*.

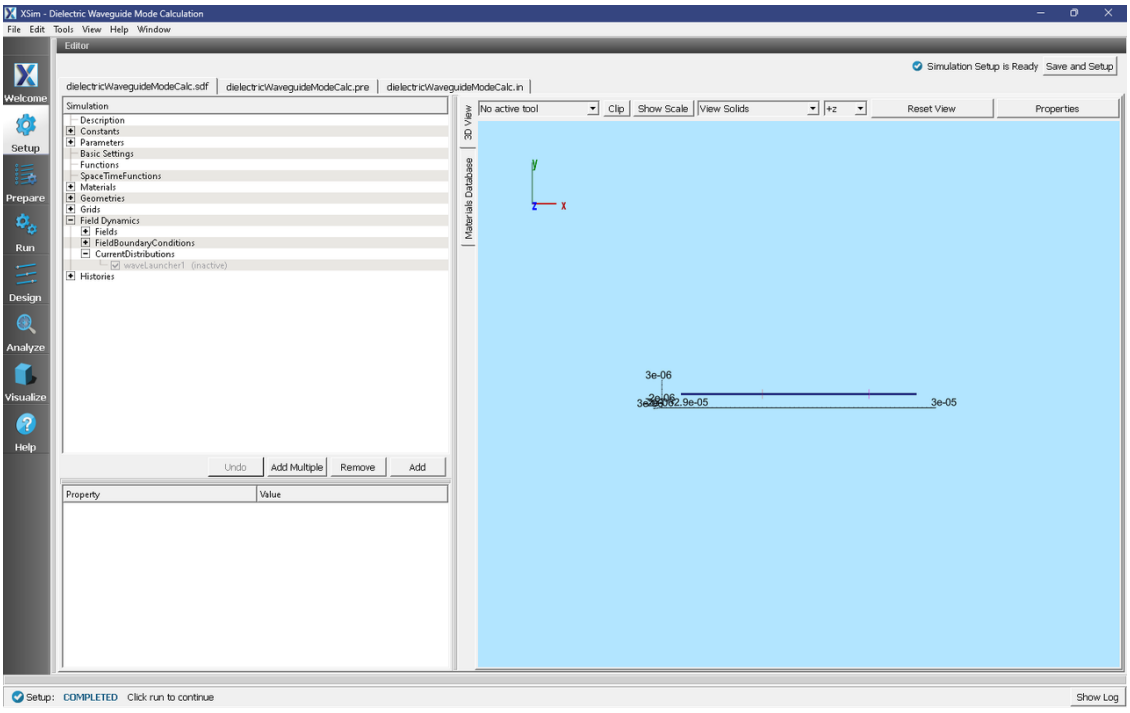

Fig. 2.94: Choosing the second order accurate, *permittivity averaging* for the *dielectric solver* field under *Basic Settings* and wave launcher *{inactive}*.

Running the Pre-run Simulation

After performing the above actions, continue as follows:

- Proceed to the Run Window by pressing the *Run* button in the left column of buttons. If you changed something in the setup, you will be asked to Save. Click *Save* upon the request to save.
- In the left pane :
  - Set *Number of Steps* to 1
  - Set *Dump Periodicity* to 1.
  - Check *Dump at Time Zero*.
- Click on the *Run* button in the upper left corner of the right pane.

You will see the output of the run in the right pane. The run has completed when you see the output, "Engine completed successfully". This result is shown in Fig. 2.95.

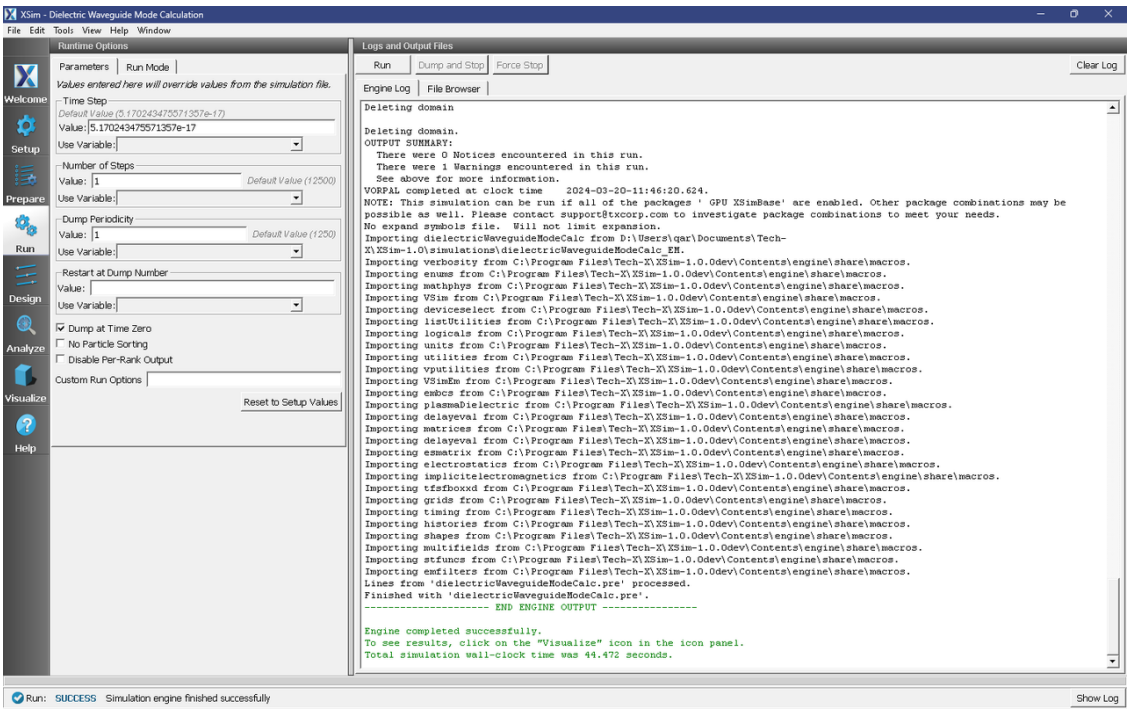

Fig. 2.95: Running the simulation for one step to get the permittivity data for the analyzer.

Solving for the Eigenmodes

After performing the above actions, continue as follows:

- Proceed to the Analyze Window by clicking the *Analyze* button on the left.
- Select *computeWaveguideModes.py* and click *Open* under the list.
- Update the analyzer fields accordingly by choosing the corresponding variables under *Use Variable* :
  - *simulationName*: dielectricWaveguideModeCalc
  - *datasetName*: invEps

  - *transverseSliceX*: PORT_X_INPUT
  - *transverseSliceLY*: PORT_YBGN
  - *transverseSliceUY*: PORT_YEND
  - *transverseSliceLZ*: PORT_ZBGN
  - *transverseSliceUZ*: PORT_ZEND
  - *vacWavelength*: 1.55e-6
  - *nModes*: 1
  - *writeFieldProfile*: D
  - *modeFileName*: port1
  - *Normalize*: checked
  - *compMajorC*: unchecked
  - *Overwrite*: checked
- Click *Analyze* in the top right corner.

The analyzer will only find guided modes. The results should resemble Fig. 2.96. We see that the analyzer found the fundamental mode and gives the effective index of the mode under *Neff*. This parameter is necessary for using the *Wave Launcher* to launch a unidirectional wave. This value is saved as a parameter as *NEFF_FROM_ANALYZER*.

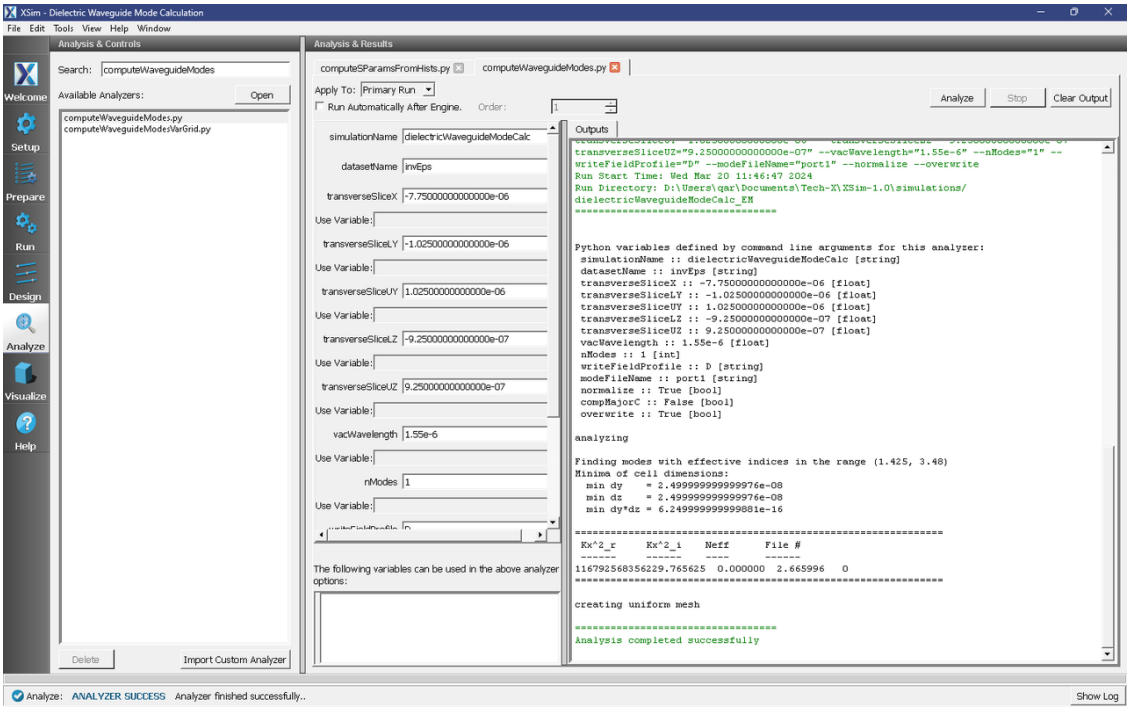

Fig. 2.96: The analyzer window after a successful run of computeWaveguideModes.py.

Setting up the Main Simulation

After performing the above actions, continue as follows:

- Proceed to the Setup Window by pressing the *Setup* button in the left column of buttons.
- Activate the wave launcher
  - Expand *Field Dynamics*
  - Expand *CurrentDistributions*
  - Right Click on *waveLauncher1*
  - Left click on *Activate*
- Click on *Save and setup* in the top right corner.

This setting is shown in Fig. 2.97.

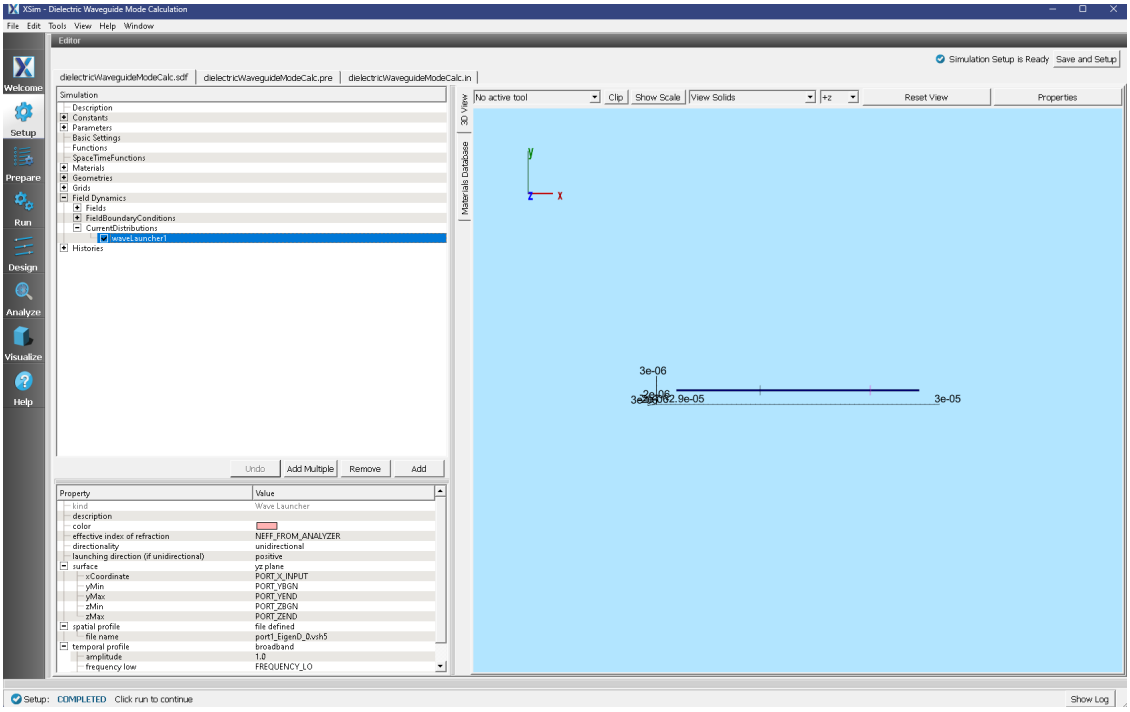

Fig. 2.97: Choosing the second order accurate, *permittivity averaging* for the *dielectric solver* field under *Basic Settings*.

Running the Main Simulation

After performing the above actions, continue as follows:

- Proceed to the Run Window by pressing the *Run* button in the left column of buttons. You will be asked to Save. Click *Save* upon the request to save.
- In the left pane, click on *Reset to Setup Values* to run the full simulation which launches this wave down the waveguide.
- Check *Dump at Time Zero*.
- Click on the *Run* button in the upper left corner of the right pane.

You will see the output of the run in the right pane. The run has completed when you see the output, "Engine completed successfully." This result is shown in Fig. 2.98.

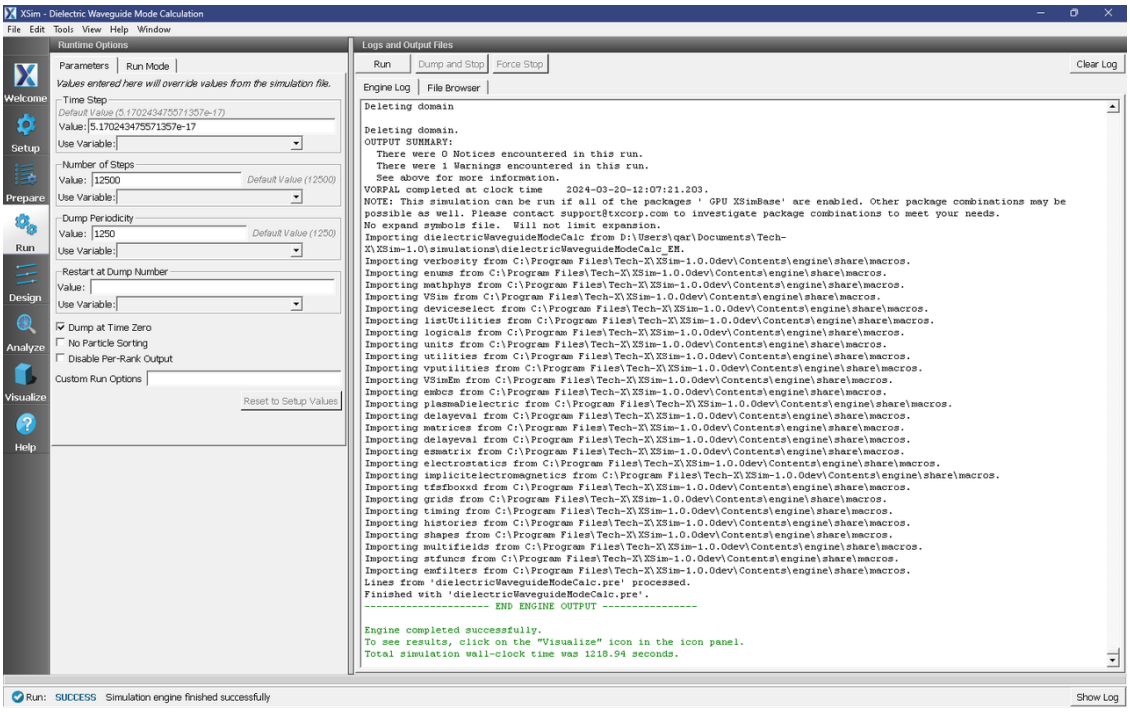

Fig. 2.98: Running the full simulation which launches this wave down the waveguide.

Visualizing the Results

After performing the above actions, continue as follows:

- Proceed to the Visualize Window by pressing the *Visualize* button in the left column of buttons.

You may need to *Reload Data* (bottom left). Visualize an eigenmode by following these steps:

- From the *Add a Data View* dropdown, select *Data Overview*.
- Expand *Scalar Data*, expand *EigenD*, and select *EigenD_magnitude*.

The resulting visualization pane should look like Fig. 2.99.

Analyzing the Results

When designing waveguides, one might be interested in calculating the S parameters and spectral power of the wave traveling down the waveguide. XSim features an analyzer that will compute both of these quantities.

- After visualizing the data and making sure the simulation is behaving as expected, proceed to the *Analyze* Window.
- Click on the *computeSParamsFromHists.py* analyzer to open it and fill in the necessary parameters as follows:
  - *simulationName* dielectricWaveguideModeCalc
  - *firstStep*: 0
  - *lastStep*: 12500

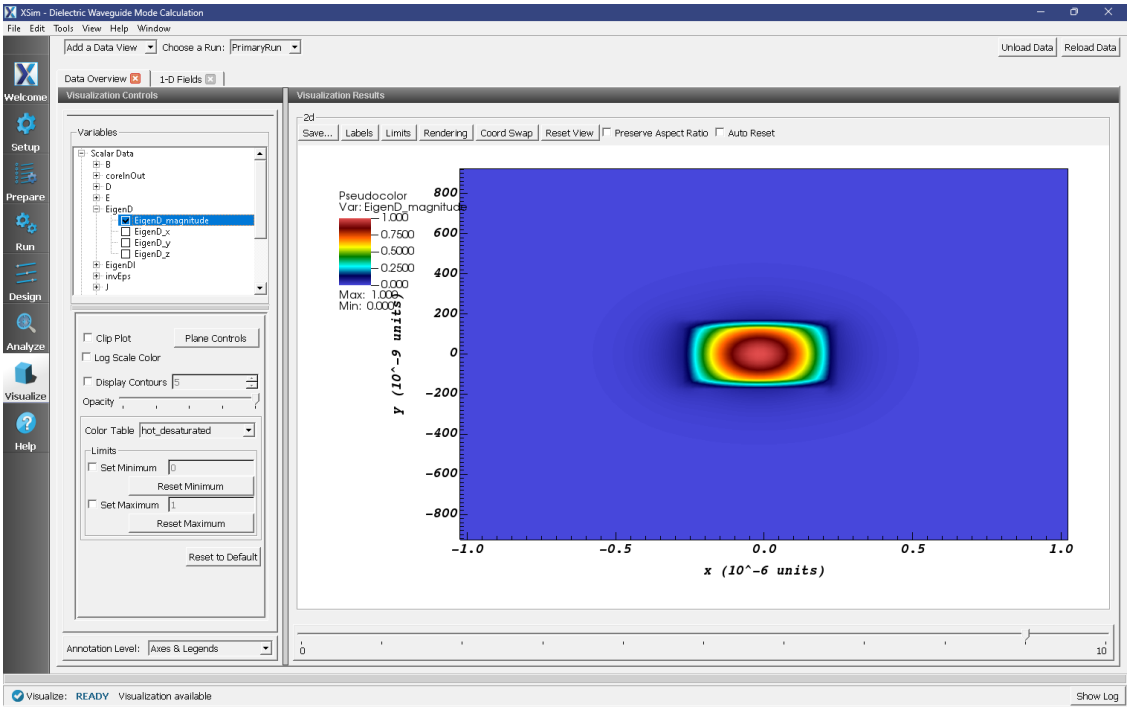

Fig. 2.99: The visualization pane showing the magnitude of the D field of the fundamental mode.

- *stepOffset*: 0
- *maxWavelength*: WAVELENGTH_MAX
- *minWavelength*: WAVELENGTH_MIN
- *inDirection*: 0
- *inSlabE*: slab0_E
- *inSlabB*: slab0_B
- *inSign*: 1
- *outDirection*: 0
- *outSlabE*: slab1_E
- *outSlabB*: slab1_B
- *outSign*: 1
- *outputFileName* : blank
- *outputSuffix*: blank
- *overwrite*: checked

- Click *Analyze* in the top right corner.

The results should resemble Fig. 2.100

Upon successful completion of the analyzer, proceed to the *Visualize* tab.

- From here, click on the drop down menu called *Add a Data View* and select *1-D Fields*.
- To visualize the S-parameters and the input power:

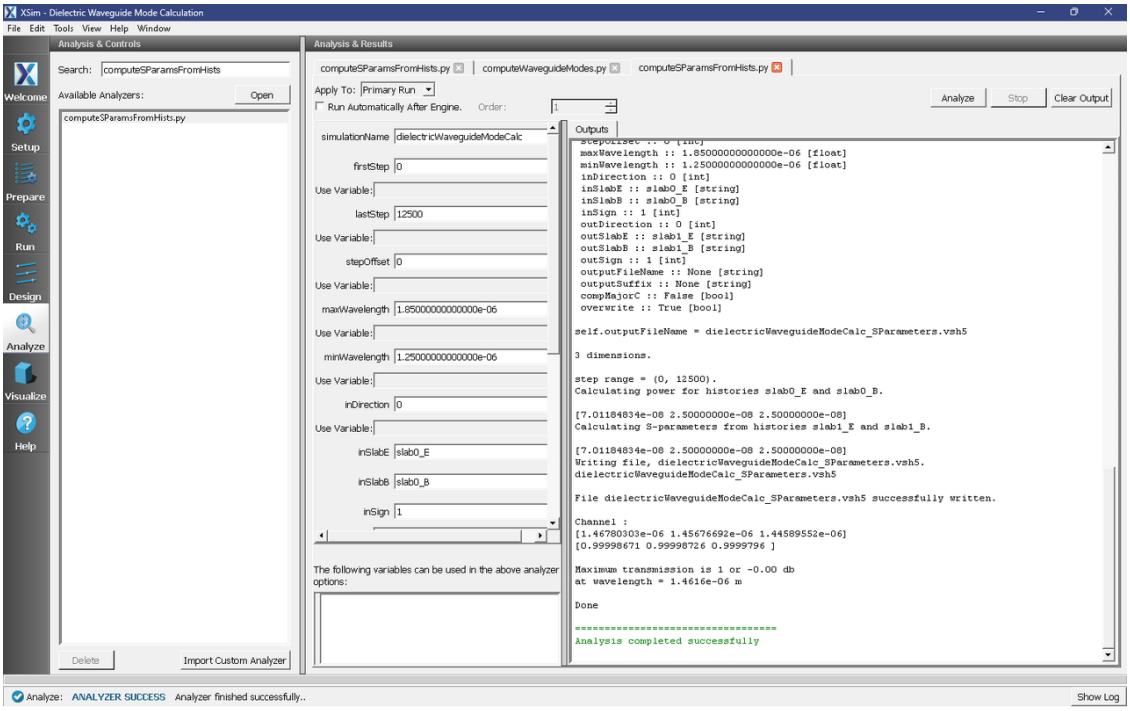

Fig. 2.100: The analyzer window after a successful run of computeSParamsFromHists.py.

- Click *Add Curve*
- Select *S_slab1_EAndslab1_B*
- Click *Apply*, followed by *Ok*.
- Click on the *Add Window* button and repeat the same process as before but this time select *slab0_EAndslab0_BPower* from the drop down menu.

The resulting plots will have the S-parameters per wavelength on top and the spectal input power on the bottom, as in Fig. 2.101

Further Experiments

Change the geometry on the Setup window and rerun the simulation and analyzer to see the effects on the modes.

Once you have your desired mode, launch it down the waveguide using the procedure laid out above.

One can run a full convergence study of eigenmode effective indices by changing the resolution in the Setup window and re-running the simulation and mode extraction script. A plot of the effective index as a function of transverse cell area is shown in Fig. 2.102. The linear relationship shows the second order accuracy of our dielectric algorithms.

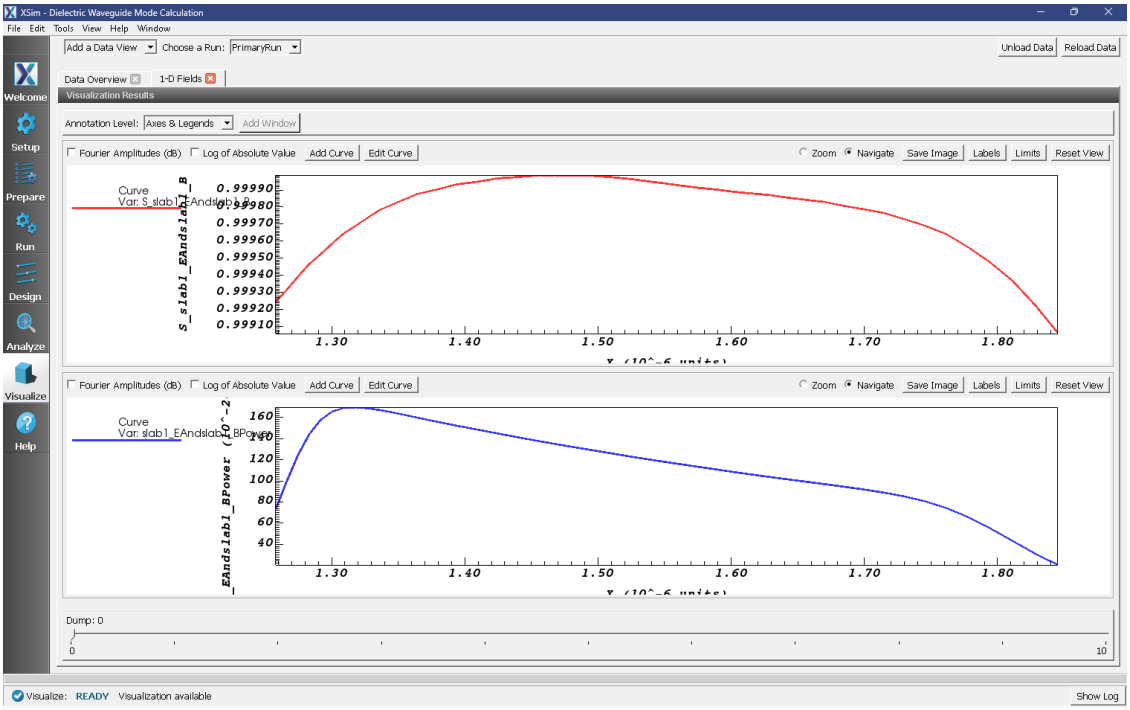

Fig. 2.101: The visualization window after adding in the plots of the S-parameters and the spectral input power.

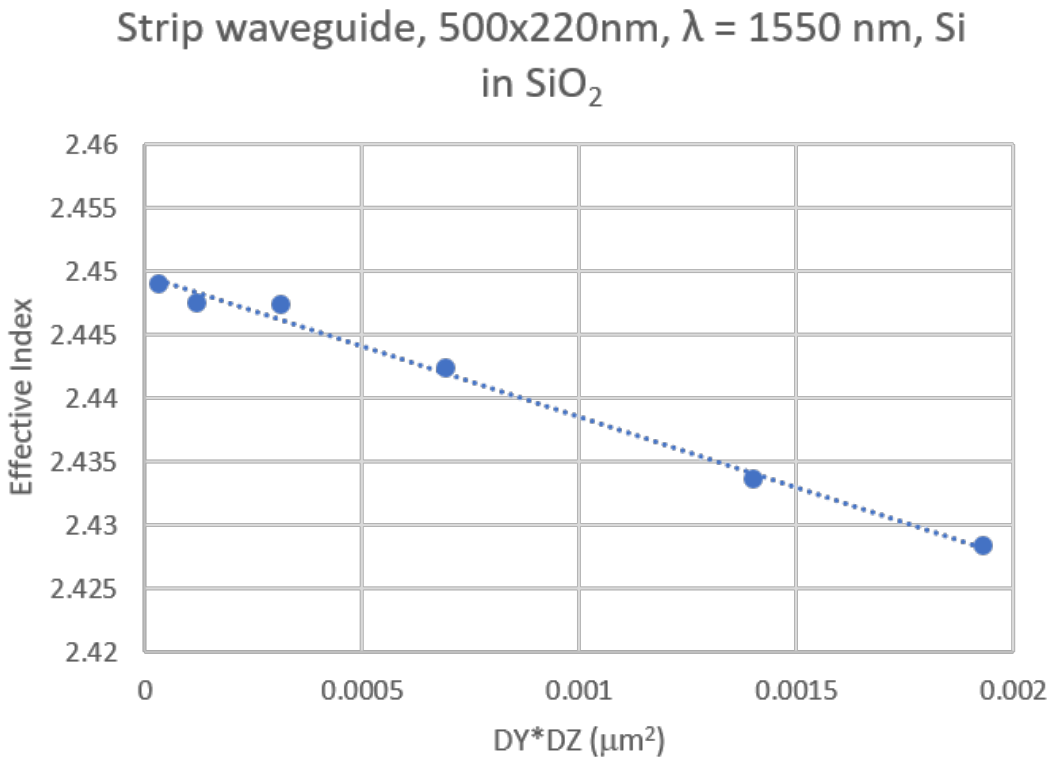

Fig. 2.102: The effective index as a function of transverse cell area for an eigenmode.

2.3.2 Microring Resonator with Gaussian Launcher (ringResonatorGaussianLaunch.sdf)

Keywords:

Ring Resonator, Mode Launcher, MAL, Guided Mode, Photonic Device, Semiconductor

Problem Description

The Ring Resonator consists of two straight Silicon waveguides and a Silicon waveguide ring that sits between the straight waveguides. All three waveguides rest on top of a Silicon Dioxide slab. The rest of the simulation domain is set to vacuum. Matched Absorbing Layers (MALs) are used to dampen the E, B and D fields near the boundary of the simulation.

The approximate fundamental guided mode profile is launched as a wide band pulse in the input waveguide. This mode is a simple 2D Gaussian distribution centered at the center of the waveguide. This approximate mode is accurate enough for this simulation. The exact mode can be calculated using the computeWaveguideModes analyzer. To go through the mode solve process check out the Multimode Fiber Mode Calculation example. We will use the computeSParamsViaOverlapIntegral.py analyzer to determine the transmission coefficients at the thru-port and drop port.

This simulation can be performed with a XSimEM license.

Opening the Simulation

The Ring Resonator example is accessed from within XSimComposer by the following actions:

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *XSim for Electromagnetics* option.
- Expand the *Photonics* option.
- Select *Ring Resonator* and press the *Choose* button.
- In the resulting dialog, create a New Folder if desired, and press the *Save* button to create a copy of this example.

All of the properties and values that create the simulation are available in the Setup Window as shown in Fig. 2.103. You can expand the tree elements and navigate through the various properties. The right pane shows a 3D view of the geometry, as well as the grid, if actively shown. To show or hide the grid, expand the Grid element and select or deselect the box next to *Grid*.

Simulation Properties

This example contains a number of *Constants* defined to make the simulation easily modifiable.

General Simulation Constants:

- WAVEL_MIN = the smallest wavelength in the frequency band
- WAVEL_MAX = the largest wavelength in the frequency band
- RADIUS_RING = radius of the ring
- WIDTH_WAVEGUIDE = Width of waveguides
- HEIGHT_WAVEGUIDE = Height of waveguides
- WIDTH_GAP = Width of the gap between ring and waveguides

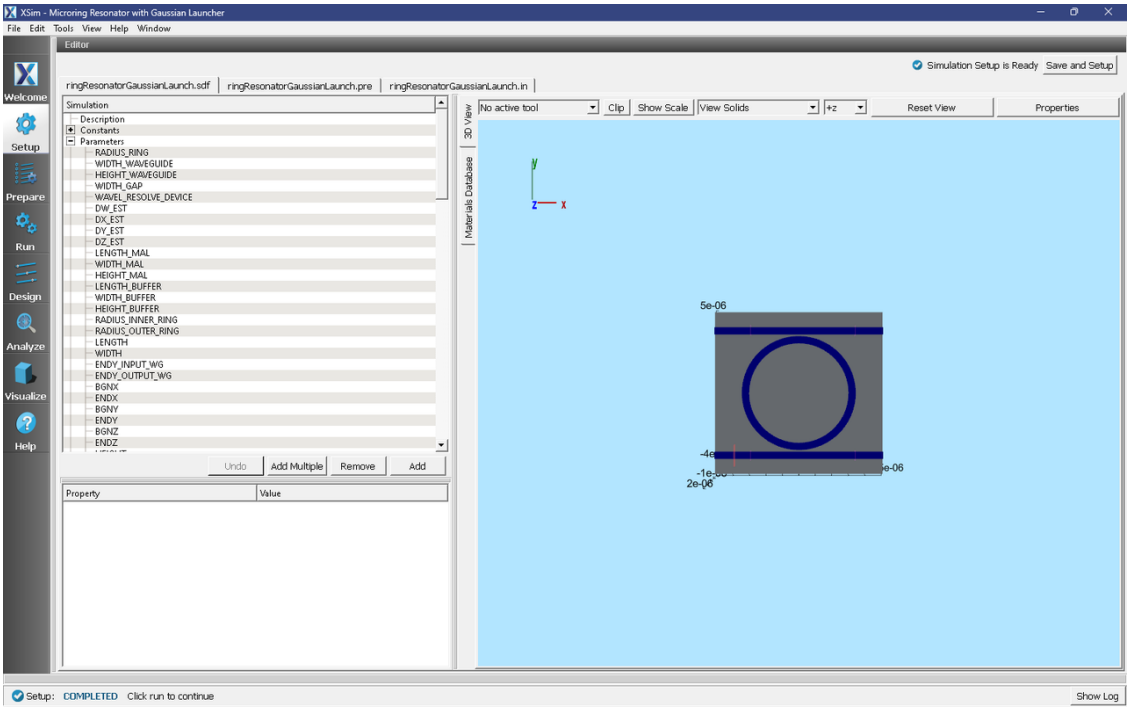

Fig. 2.103: The Setup window for the ring resonator example showing the external launching field.

This simulation applies a wide frequency band signal to the approximate fundamental spatial mode profile. The signal is defined under *SpaceTimeFunctions* and then assigned under *Field Dynamics*, *CurrentDistributions*, *separableDistributionCurrent* as shown in Fig. 2.103.

The *Materials* section contains just Silicon and Silica. This section is where one can add or edit materials that get attached to CSG objects. These *Materials* contain the relative permittivity.

In *Field Dynamics* there are *FieldBoundaryConditions* which set the boundary conditions of the simulation. In photonics simulations, Matched Absorbing Layers (MALs), are the most stable boundary conditions for preventing reflections.

Under *Basic Settings* you can see that the *dielectric solver* is set to *permittivity averaging*. This feature enables second order accuracy for simulations using dielectrics.

Running the Simulation

When you have saved the setup, continue as follows:

- Proceed to the run window by pressing the Run button in the left column of buttons.
- Check that you are using these run parameters:
  - *Time Step*: 7.343490630124558e-17
  - *Number of Steps*: 60000
  - *Dump Periodicity*: 6000
  - *Dump at Time Zero*: Checked
- Click on the *Run* button in the upper left corner of the right pane.

You will see the output of the run in the right pane. The run has completed when you see the output, “Engine completed successfully”. As seen in Fig. 2.104.

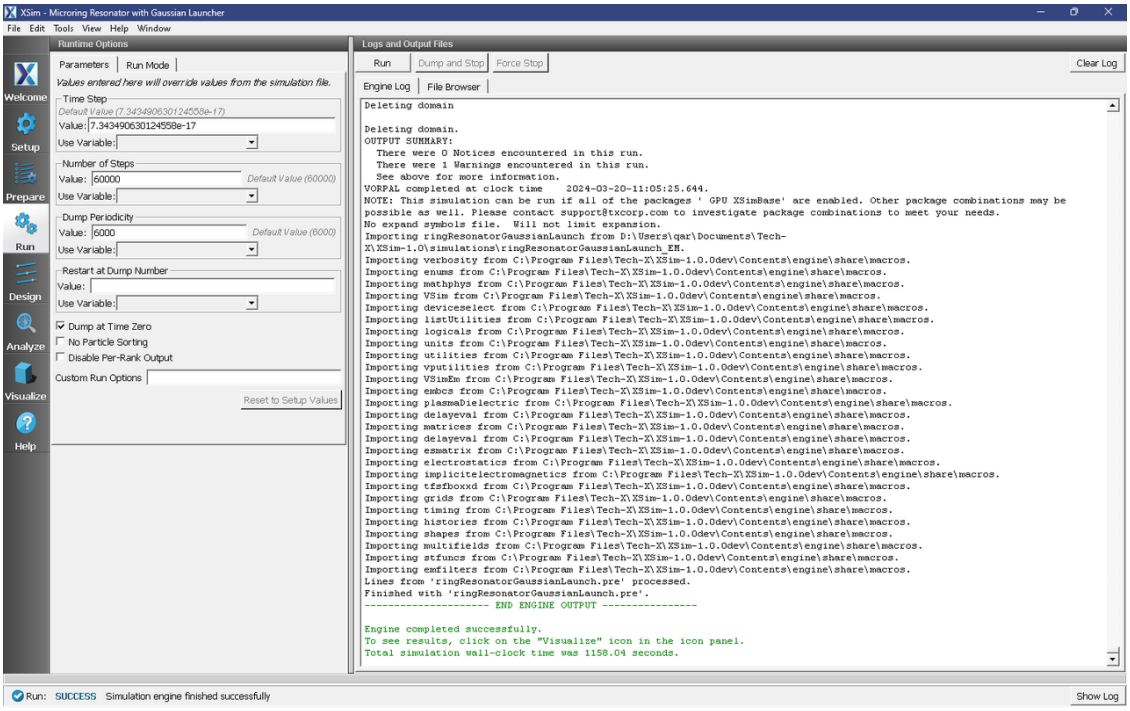

Fig. 2.104: Run window at completion.

Analyzing the Results

Using post analysis scripts, one can extract the transmission coefficients. This simulation uses 4 “EM Field on Plane” histories (slab0-slab4) and this data is used to calculate S parameters between these 4 locations using the analysis script computeSParamsViaOverlapIntegral.py.

Follow these steps:

- Proceed to the Analyze Window by clicking the *Analyze* button on the left.
- Select *computeSParamsViaOverlapIntegral.py* and click *Open* under the list.

Now update the analyzer fields accordingly.

- *maxWavelength*: 1.7e-06 (or use the corresponding parameter from the setup)
- *minWavelength*: 1.4e-06 (or use the corresponding parameter from the setup)
- *inPlane*: slab0
- *outPlane*: slab1 (or slab3)
- *outputFileSuffix*: S01 (or S03)

The remaining parameter default values will work. Hit the *Analyze* button in the top right corner. After a successful run the window should resemble Fig. 2.105. Continue by analyzing the drop-port. To do this change the *outSlabE* and *outSlabB* fields to slab3_E and slab3_B, respectively.

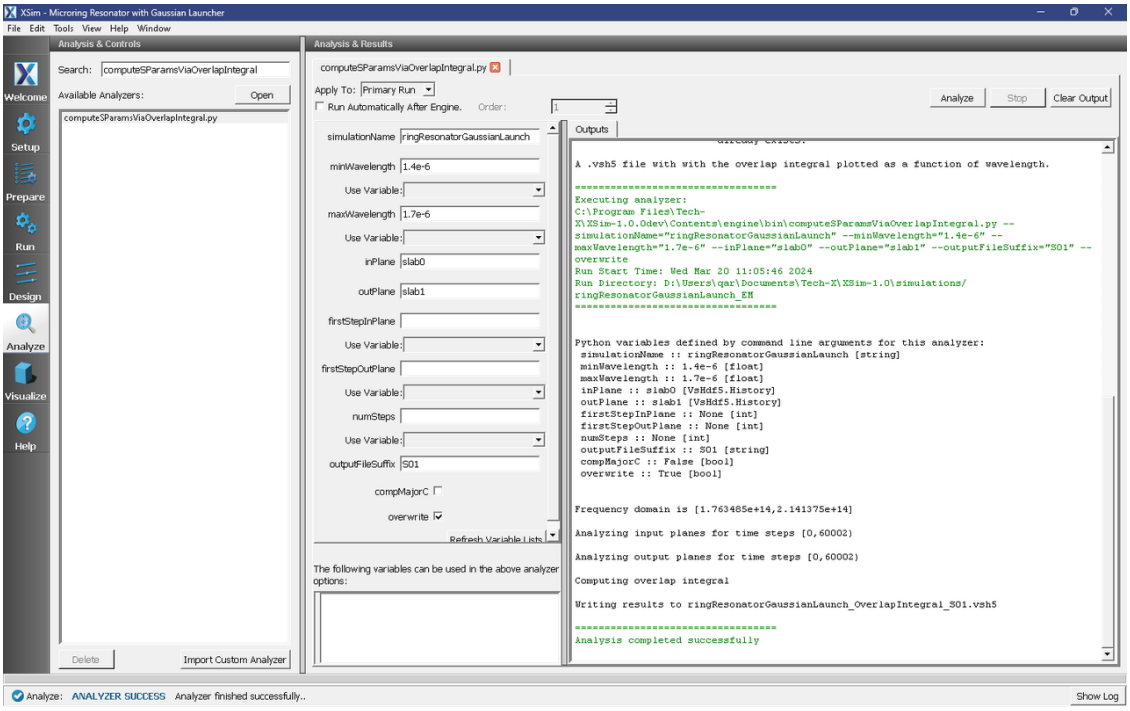

Fig. 2.105: Analyze window with output from computeSParamsViaOverlapIntegral.py.

Visualizing the results

After performing the above actions proceed to the Visualize Window by pressing the *Visualize* button in the left column of buttons.

One can visualize the magnetic field by performing the following:

- Near the top left corner of the window, select *Data Overview* from the *Add a Data View* drop-down.
- Expand *Scalar Data*, then *B*, then select *B_y*
- In the bottom left, select *Clip Plot*, and set z=0 in *Plane Controls*
- Finally, move the dump slider on the bottom of the window to watch the light propagate.

The results are shown in Fig. 2.106.

One can visualize the transmission coefficients by performing the following:

- Near the top left corner of the window, select *1-D Fields* from the *Add a Data View* drop-down.
- In the Plot Control panel select the Slab1 and Slab3 SParams data for Graphs 1 and 2, respectively.
- Set the other Graphs to *None*

The results are shown in Fig. 2.107. As expected, we see coupling of resonant wavelengths with the ring.

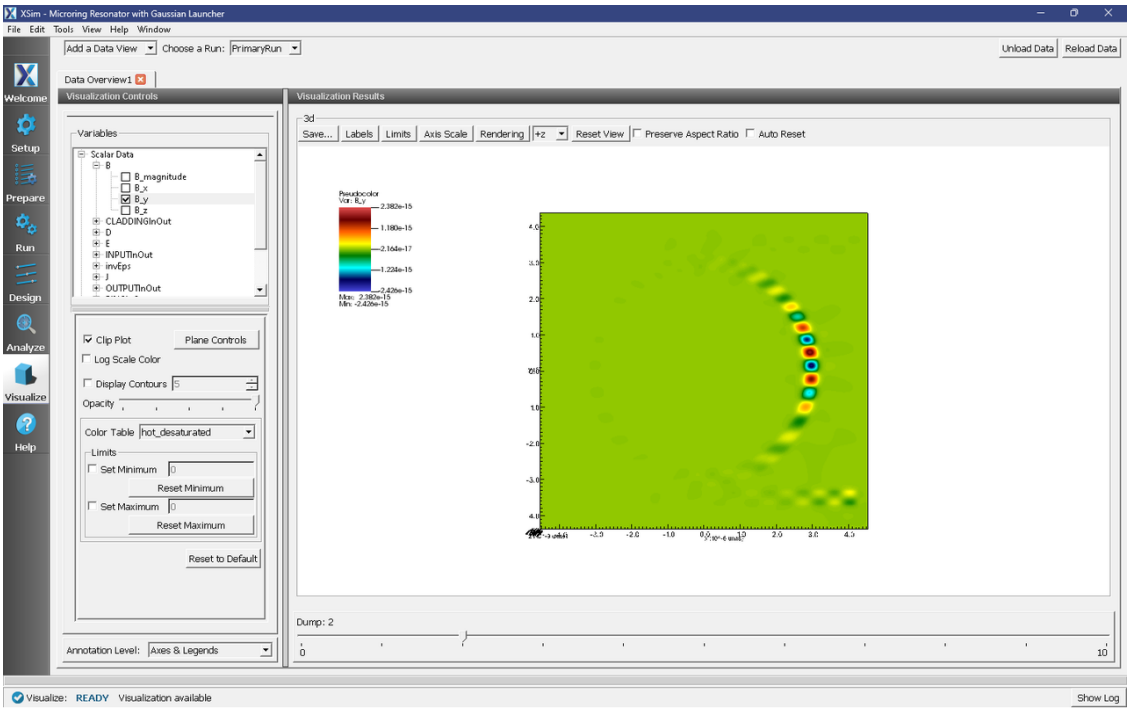

Fig. 2.106: Visualization of magnetic field.

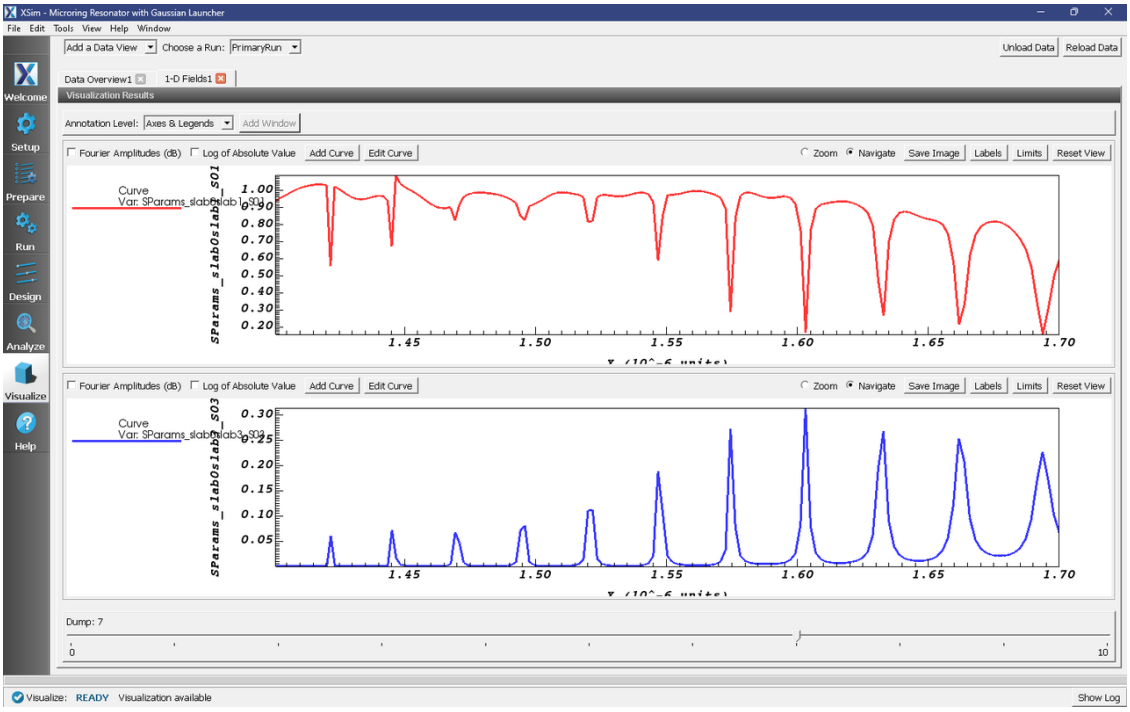

Fig. 2.107: Visualization of the s-parameters.

2.3.3 Dielectric Waveguide With Roughness (dielectricWaveguideModeCalcRough.sdf)

Keywords:

Photonic Waveguide, Guided Mode

Problem Description

This example demonstrates how to add roughness to a waveguide and how to implement a variable grid to resolve roughness that would otherwise be too computationaly expensive to resolve. Since roughness features of a waveguide are typically on the order of nanometers, a very high resolution (and therefore small cell size) is required. By using a variable grid, a user is able to increase the resolution where needed and decrease the resolution elsewhere to maintain the computational efficiency of a uniform grid.

Opening the Simulation

To open this example open an instance of XSimComposer and follow the steps below:

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *XSim for Electromagnetics* option.
- Expand the *Photonics* option.
- Select *Dielectric Waveguide With Roughness* and press the *Choose* button.
- In the resulting dialog, create a *New Folder* if desired, and press the *Save* button to create a copy of this example.

Simulation Variables

This example contains a number of variables defined to make the simulation easily modifiable.

- EPS_SILICON and EPS_SILICA: Relative permittivities of silicon and silica. These constants are used in multiple parameters and in the accompanying Python file for solving the waveguide mode.
- WAVELENGTH_CENTER: Center wavelength of the input signal. This wavelength is also used for the calculation of the fundamental guided mode of the device.
- DY_REFINED: The cell size desired within the refinement region.
- USE_DY_REFINED: A switch that when set to 1 will replace DY with DY_REFINED in calculations such as the CFL time step. By default this is set to 0.

The *Materials* section contains just silicon and silica. The *Geometries* section includes the GDS waveguide. In *Field Dynamics*, there are *FieldBoundaryConditions* and *CurrentDistributions* to be aware of. In photonics simulations, Matched Absorbing Layers (MALs) are the most stable boundary conditions for preventing reflections. This simulation makes use of XSim's Wave Launcher to launch a unidirectional wave down the waveguide with a mode defined by the file generated by the computeWaveguideModes.py analyzer.

Setting up the Roughness

As delivered, this example comes equipped with a geometry imported from a GDS file. For more on importing a geometry from a GDS file, see the Y Splitter example documentation. At the moment roughness is only supported on geometries made from importing a GDS. To add roughness to the waveguide, proceed as follows:

- Expand the geometry object in the *Geometries* section by clicking on the + symbol next to *waveguideGeom*.
- Select the *Boundary0* object under *waveguideGeom*.
- **Switch the *roughness type* option from *roughnessDisabled* to *stochasticODE*. Fill in the following parameters under *stochasticODE* as follows:**
  - RMS amplitude: 50e-9
  - normal tolerance: 45
  - Ramp-up distance: 0.0
  - Ramp-down distance: 0.0
  - ODE Correlation: 1000e-9
  - ODE step size: 140e-9
  - Random seed: Leave as is
  - export to CSV:

The geometry in the 3D visualization view should look like:

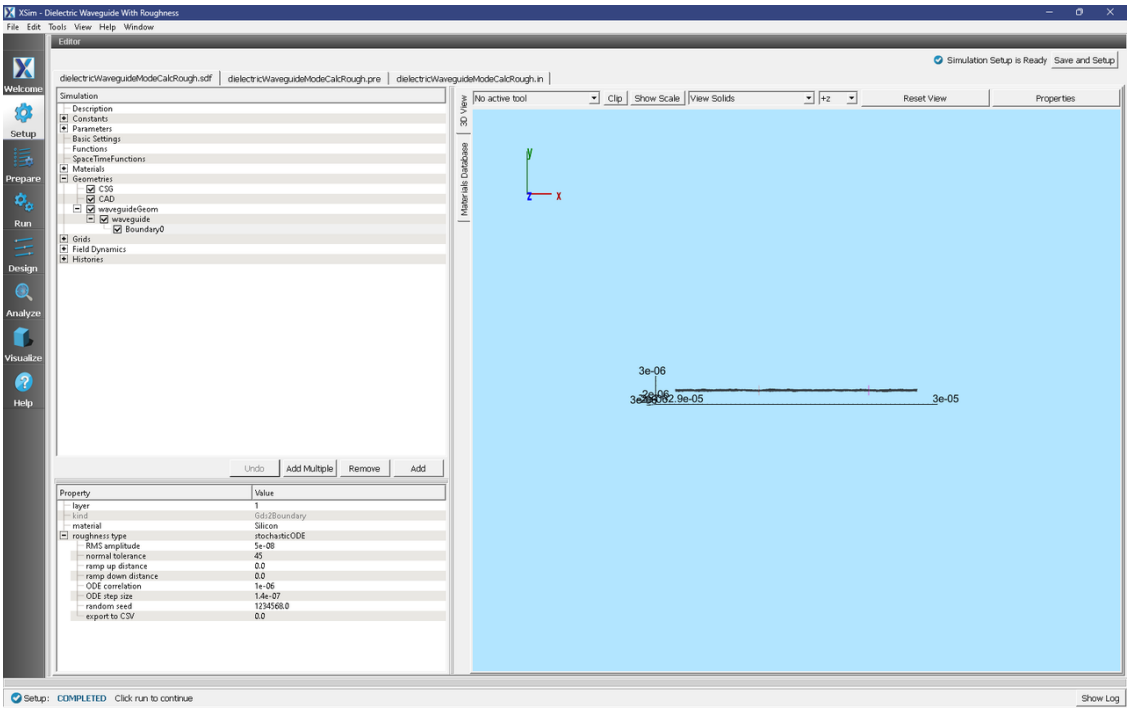

Fig. 2.108: The waveguide with roughness added to the upper and lower walls in the y direction.

Running the Uniform Grid Simulation

After performing the above actions, continue as follows:

- Proceed to the Run Window by pressing the *Run* button in the left column of buttons. If you changed something in the setup, you will be asked to Save. Click *Save* upon the request to save.
- In the left pane : * Set *Number of Steps* to 12500. * Set *Dump Periodicity* to 1250. * Check *Dump at Time Zero*.
- Click on the *Run* button in the upper left corner of the right pane.

You will see the output of the run in the right pane. The run has completed when you see the output, "Engine completed successfully". This result is shown in Fig. 2.109.

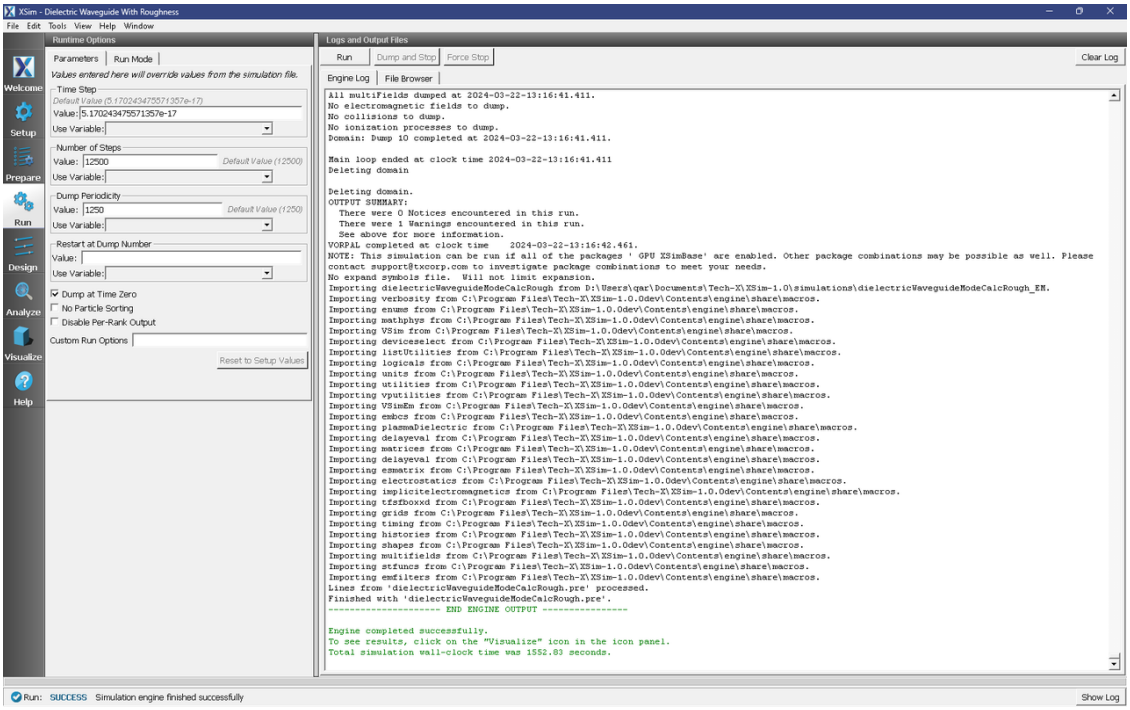

Fig. 2.109: Running the full simulation.

Visualizing the Uniform Grid Results

After performing the above actions, continue as follows:

- Proceed to the Visualize Window by pressing the *Visualize* button in the left column of buttons.

You may need to *Reload Data* (bottom left). Visualize the electric field and a cross section of the geometry by following these steps:

- From the *Add a Data View* dropdown, select *Data Overview*.
- Expand *Scalar Data*, expand *E*, and select *E_y*.
- Expand *Geometries* and select *waveguideDualTriangles*.

The resulting visualization pane should look like Fig. 2.110.

Slide the dump selector at the bottom of the window to the right to watch the wave travel down the waveguide. It is clear that with this level of roughness that the transmission of the wave will not be great.

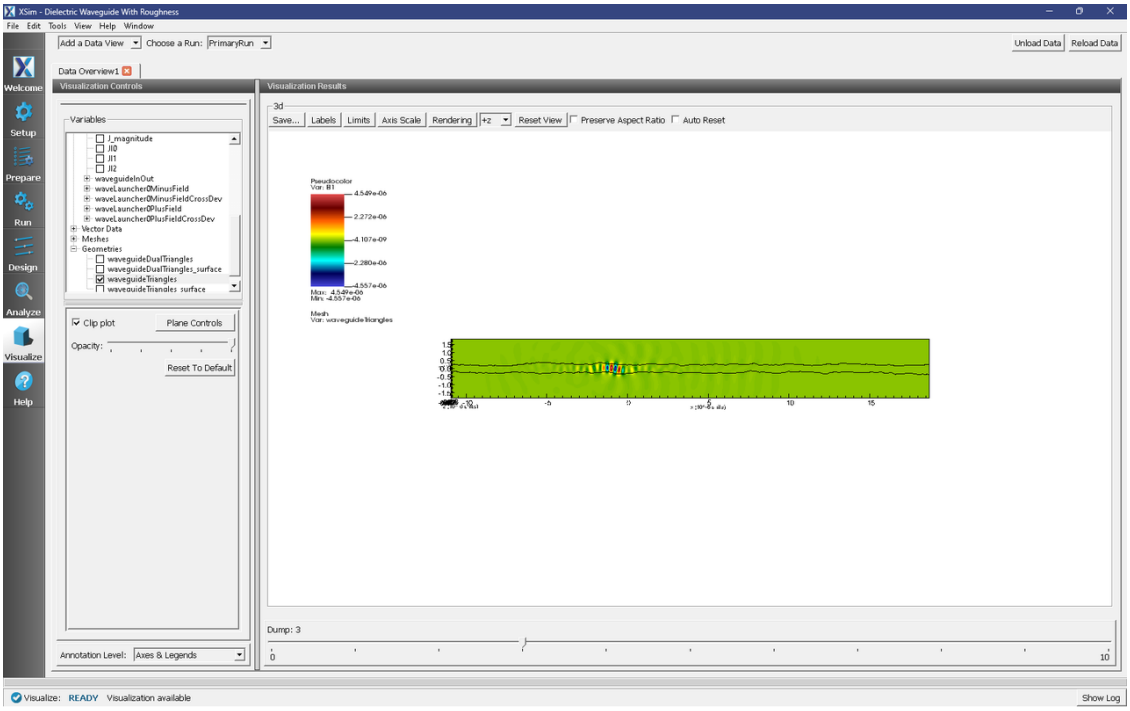

Fig. 2.110: The visualization pane showing the y component of the E field with the geometry on top.

Setting Up the Variable Grid

In XSim, a variable grid is implemented into the simulation by adding in a *Refinement Region*. This is a 3D box in which the cell sizes inside can be changed to increase the resolution in one or multiple directions. The refinement region will then take the total number of cells in each direction and will automatically adjust the size of the cells outside of the refinement region to smoothly fit in the rest of the cells into the domain. In this simulation, the resolution in y is increased. In this example, a refinement region has already been created and is inactive. To activate this refinement region, proceed as follows:

- Expand the *Grids* and *Grid* sections.
- Right click on *refinementRegion0* and click *Activate*.

This refinement region was created by performing the follwing actions:

- Expand the *Grids* section and right click on *Grid*.
- Select *Create Refinement Region*.

A new option will appear under *Grid* called *refinementRegion0*. To make the box surround the waveguide and increase the resolution in y, fill in the following options in the refinement region as follows:

- xMin: GRID_X_BGN
- xMax: GRID_X_END
- cellSizeX: DX
- yMin: -3.5e-7
- yMax: 3.5e-7
- cellSizeY: DY
- zMin: -2e-7

- zMax: 2e-7
- cellSizeZ: DZ

Now to change the resolution in the y direction, expand the *Parameters* section and select *USE_DY_REFINED*. Set this parameter equal to 1. This parameter acts as a switch that will replace DY with a smaller value specified by DY_REFINED in all calculations. Since the grid has changed, a new mode file needs to be calculated. Expand the *Field Dynamics* section and under *CurrentDistributions*, deactivate *waveLauncher0* by right clicking on it and select *Deactivate*. Next, expand the *Geometries* tab and expand the *waveguideGeom* geometry. Select *Boundary0* and set *roughness type* to "roughnessDisabled". Save the simulation and continue to the run tab. Now to generate the necessary data to recalculate the fundamental mode of the waveguide, proceed as follows:

- Click on the *Reset to Setup Values* button.
- Set the *Number of Steps* and *Dump Periodicity* to 1.
- Make sure that *Dump at Time Zero* is selected.
- Click *Run*.

Upon successful completion of the engine, proceed to the next section.

Solving for the Eigenmode

After performing the above actions, continue as follows:

- Proceed to the Analyze Window by clicking the *Analyze* button on the left.
- Select *computeWaveguideModes.py* and click *Open* under the list.
- Update the analyzer fields accordingly by choosing the corresponding variables under *Use Variable* :
  - *simulationName*: dielectricWaveguideModeCalcRough
  - *datasetName*: invEps
  - *transverseSliceX*: PORT_X_INPUT
  - *transverseSliceLY*: PORT_YBGN
  - *transverseSliceUY*: PORT_YEND
  - *transverseSliceLZ*: PORT_ZBGN
  - *transverseSliceUZ*: PORT_ZEND
  - *vacWavelength*: 1.55e-6
  - *nModes*: 1
  - *writeFieldProfile*: D
  - *modeFileName*: port1
  - *Normalize*: checked
  - *compMajorC*: unchecked
  - *Overwrite*: checked
- Click *Analyze* in the top right corner.

The analyzer will only find guided modes. The results should resemble Fig. 2.111. We see that the analyzer found the fundamental mode and gives the effective index of the mode under *Neff*. This parameter is necessary for using the *Wave Launcher* to launch a unidirectional wave. This value is saved as a constant as *NEFF_FROM_ANALYZER*.

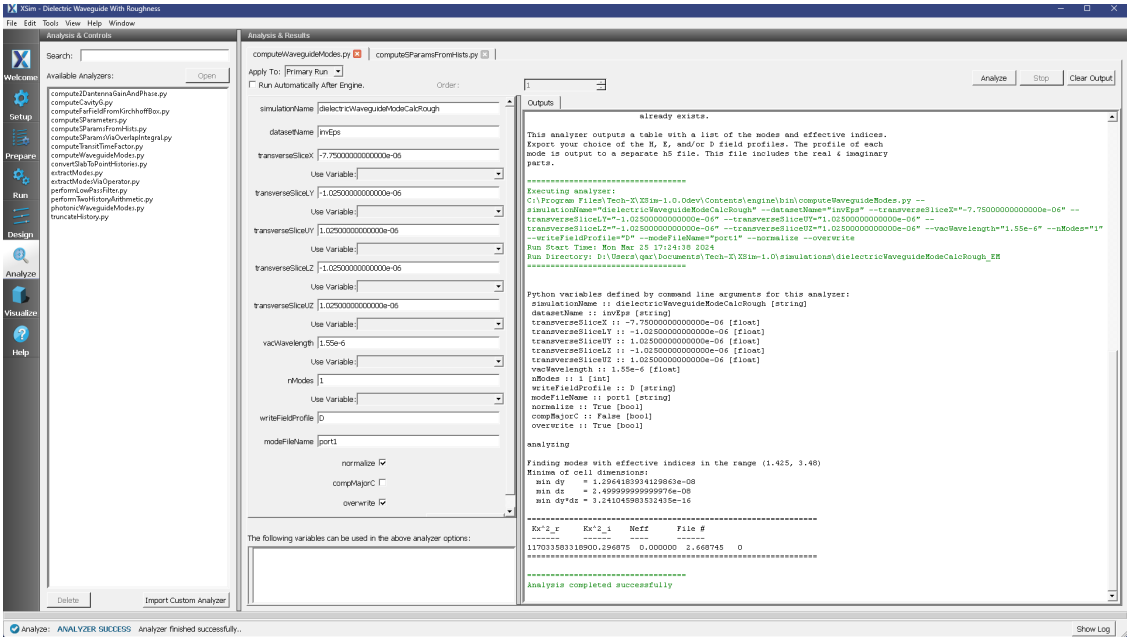

Fig. 2.111: The analyzer window after a successful run of computeWaveguideModes.py.

Setting up the Variable Grid Simulation

After performing the above actions, continue as follows:

- Proceed to the Setup Window by pressing the *Setup* button in the left column of buttons.
- Activate the wave launcher.
  - Expand *Field Dynamics*.
  - Expand *CurrentDistributions*.
  - Right Click on *waveLauncher0*.
  - Left click on *Activate*.
- Click on *Save and setup* in the top right corner.

From the setup tab we will now add in roughness that can be resolved with the variable grid. Add in roughness the same way as before but change the RMS amplitude to 10 nm.

Running the Variable Grid Simulation

To run the variable grid simulation, proceed as follows:

- Proceed to the Run Window by pressing the *Run* button in the left column of buttons. You will be asked to Save. Click *Save* upon the request to save.
- In the left pane, click on *Reset to Setup Values* to run the full simulation which launches this wave down the waveguide.
- Check *Dump at Time Zero*.
- Make sure that *Number of Steps* is set to 24000.
- Make sure that *Dump Periodicity* is set to 2400.

- Click on the *Run* button in the upper left corner of the right pane.

You will see the output of the run in the right pane. The run has completed when you see the output, "Engine completed successfully."

Analyzing the Results

When designing a waveguide and simulating the roughness, one might be interested in calculating the S parameters and spectral power of the wave traveling down the waveguide. XSim features an analyzer that will compute both of these quantities.

- After visualizing the data and making sure the simulation is behaving as expected, proceed to the *Analyze* Window.
- Click on the *computeSParamsFromHists.py* analyzer to open it and fill in the necessary parameters as follows:
  - *simulationName* dielectricWaveguideModeCalcRough
  - *firstStep*: 0
  - *lastStep*: 12500 (or 24000 for variable grid simulation)
  - *stepOffset*: 0
  - *maxWavelength*: WAVELENGTH_MAX
  - *minWavelength*: WAVELENGTH_MIN
  - *inDirection*: 0
  - *inSlabE*: slab0_E
  - *inSlabB*: slab0_B
  - *inSign*: 1
  - *outDirection*: 0
  - *outSlabE*: slab1_E
  - *outSlabB*: slab1_B
  - *outSign*: 1
  - *outputFileName* : blank
  - *outputSuffix*: blank
  - *overwrite*: checked
- Click *Analyze* in the top right corner.

The results should resemble Fig. 2.112

Upon successful completion of the analyzer, proceed to the *Visualize* tab.

- From here, click on the drop down menu called *Add a Data View* and select *1-D Fields*.
- To visualize the S-parameters and the input power:
  - Click *Add Curve*
  - Select *S_slab1_EAndslab1_B*
  - Click *Apply*, followed by *Ok*.
  - Click on the *Add Window* button and repeat the same process as before but this time select *slab0_EAndslab0_BPower* from the drop down menu.

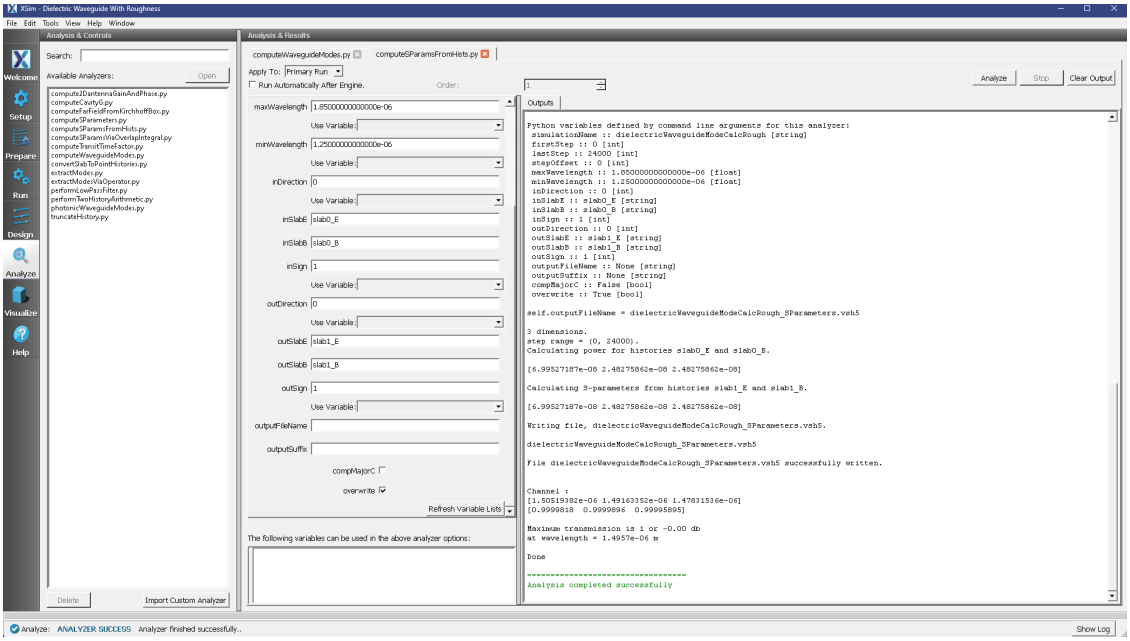

Fig. 2.112: The analyzer window after a successful run of computeSParamsFromHists.py.

The resulting plots will have the S-parameters per wavelength on top and the spectal input power on the bottom, as in Fig. 2.113

Further Experiments

Change the *DY_REFINED* parameter to analyze different roughness amplitudes. Change the variable grid to increase the resolution in the propagation direction and then increase the roughness in this dimension as well.

2.3.4 Microring Resonator with Mode Launcher (ringResonatorModeLaunch.sdf)

Keywords:

`Ring Resonator, Mode Launcher, MAL, Guided Mode, Photonic Device, Semiconductor`

Problem Description

The Ring Resonator consist of two straight Silicon waveguides and a Silicon waveguide ring that sits between the straight waveguides. All three waveguides rest on top of a Silicon Dioxide slab. The rest of the simulation domain is set to vacuum. Matched Absorbing Layers (MALs) are used to dampen the E, B and D fields near the boundary of the simulation.

The fundamental guided mode profile is launched as a wide band pulse in the input waveguide. This mode is imported from the file save_EigenD_0.vsh5, produced by the computeWaveguideModes.py analyzer. To go through the mode solve process check out the Multimode Fiber Mode Calculation example. We will use the computeSParamsViaOverlapIntegral.py analyzer to determine the transmission coefficients at the thru-port and drop port.

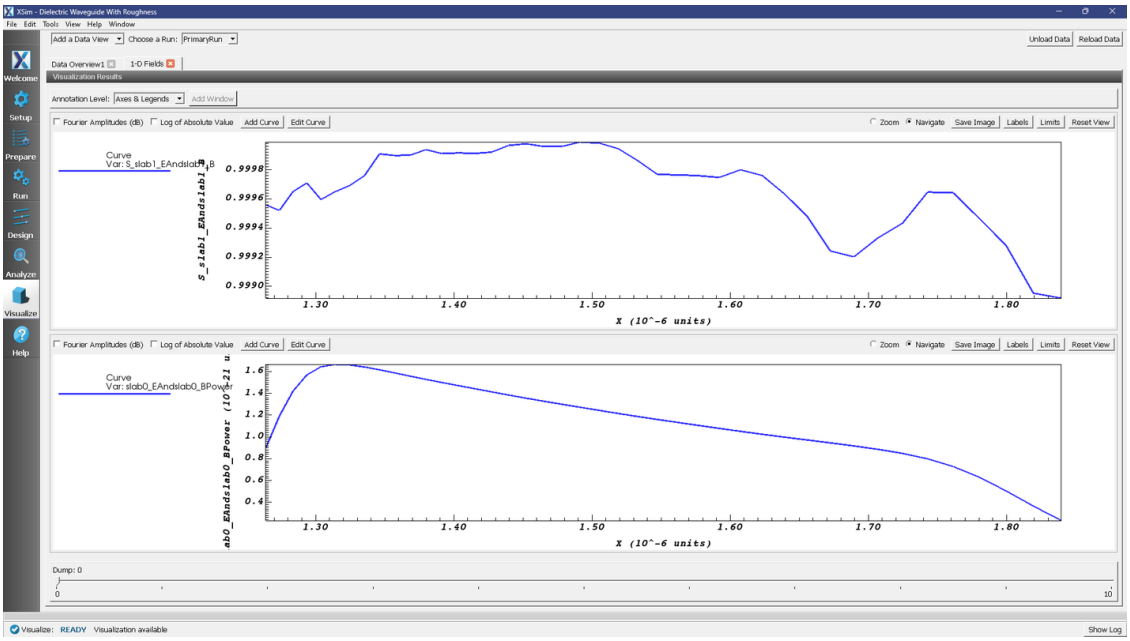

Fig. 2.113: The visualization window after adding in the plots of the S-parameters and the spectral input power.

Opening the Simulation

The Ring Resonator example is accessed from within XSimComposer by the following actions:

- Select the *New* → *From Example…* menu item in the *File* menu.
- In the resulting *Examples* window expand the *XSim for Electromagnetics* option.
- Expand the *Photonics* option.
- Select *Ring Resonator* and press the *Choose* button.
- In the resulting dialog, create a New Folder if desired, and press the *Save* button to create a copy of this example.

All of the properties and values that create the simulation are available in the Setup Window as shown in Fig. 2.114. You can expand the tree elements and navigate through the various properties. The right pane shows a 3D view of the geometry, as well as the grid, if actively shown. To show or hide the grid, expand the Grid element and select or deselect the box next to *Grid*.

Simulation Properties

This example contains a number of *Constants* defined to make the simulation easily modifiable.

General Simulation Constants:

- RESOLUTION_XY = the inverse of the number of cells per wavelength
- RESOLUTION_Z = the inverse of the number of cells per waveguide height in the z dimension
- WAVELENGTH_CENTER = the central wavelength used in the excitation
- WIDTH_EXCITATION = the width in wavelength space of the wide band signal
- RADIUS_RING = radius of the ring
- WIDTH_WAVEGUIDE = Width of waveguides

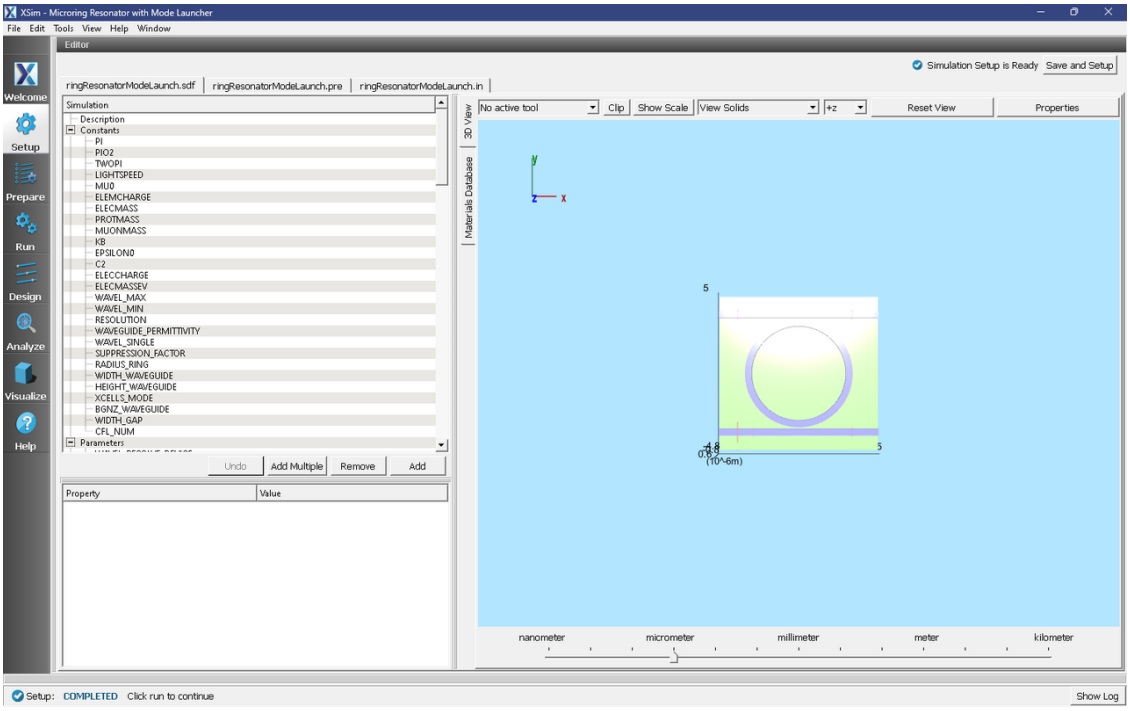

Fig. 2.114: The Setup window for the ring resonator example showing the external mode launching field.

- HEIGHT_WAVEGUIDE = Height of waveguides
- WIDTH_GAP = Width of the gap between ring and waveguides

This simulation applies a wide frequency band signal to the fundamental spatial mode profile. The signal is defined under *SpaceTimeFunctions* and then assigned under *Field Dynamics*, *Fields*, *externalModeLaunchingField1* as shown in Fig. 2.114.

The *Materials* section contains just Silicon and Silica. This section is where one can add or edit materials that get attached to CSG objects. These *Materials* contain the relative permittivity.

In *Field Dynamics* there are *FieldBoundaryConditions* which set the boundary conditions of the simulation. In photonics simulations, Matched Absorbing Layers (MALs), are the most stable boundary conditions for preventing reflections.

Under *Basic Settings* you can see that the *dielectric solver* is set to *permittivity averaging*. This feature enables second order accuracy for simulations using dielectrics.

Running the Simulation

When you have saved the setup, continue as follows:

- Proceed to the run window by pressing the Run button in the left column of buttons.
- Check that you are using these run parameters:
  - *Time Step*: 7.343490630124558e-17
  - *Number of Steps*: 60000
  - *Dump Periodicity*: 6000
  - *Dump at Time Zero*: Checked
- Click on the *Run* button in the upper left corner of the right pane.

You will see the output of the run in the right pane. The run has completed when you see the output, "Engine completed successfully". As seen in Fig. 2.115.

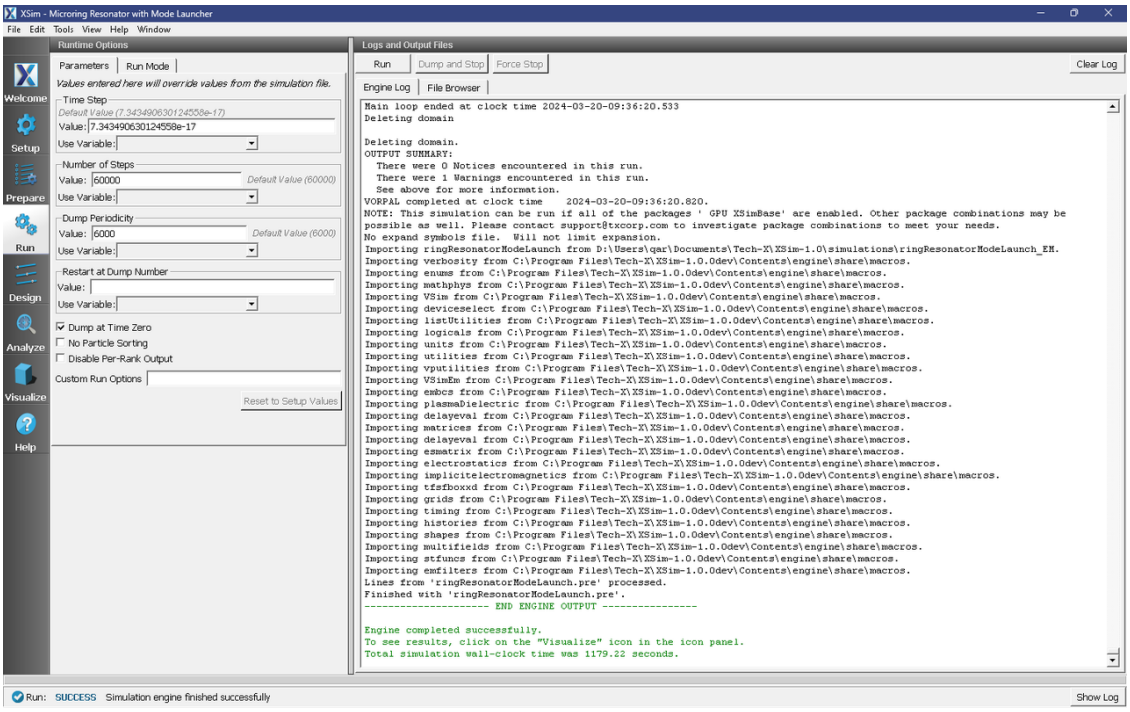

Fig. 2.115: Run window at completion.

Analyzing the Results

Using post analysis scripts, one can extract the transmission coefficients. This simulation uses 4 "EM Field on Plane" histories (slab0-slab4) and this data is used to calculate S parameters between these 4 locations using the analysis script computeSParamsViaOverlapIntegral.py.

Follow these steps:

- Proceed to the Analyze Window by clicking the *Analyze* button on the left.
- Select *computeSParamsViaOverlapIntegral.py* and click *Open* under the list.

Now update the analyzer fields accordingly.

- *maxWavelength*: 1.7e-6 (or use the corresponding parameter from the setup)
- *minWavelength*: 1.4e-6 (or use the corresponding parameter from the setup)
- *inPlane*: slab0
- *outPlane*: slab1 (or slab3)
- *outputFileSuffix*: S01 (or S03)

The remaining parameter default values will work. Hit the *Analyze* button in the top tight corner. After a successful run the window should resemble Fig. 2.116.

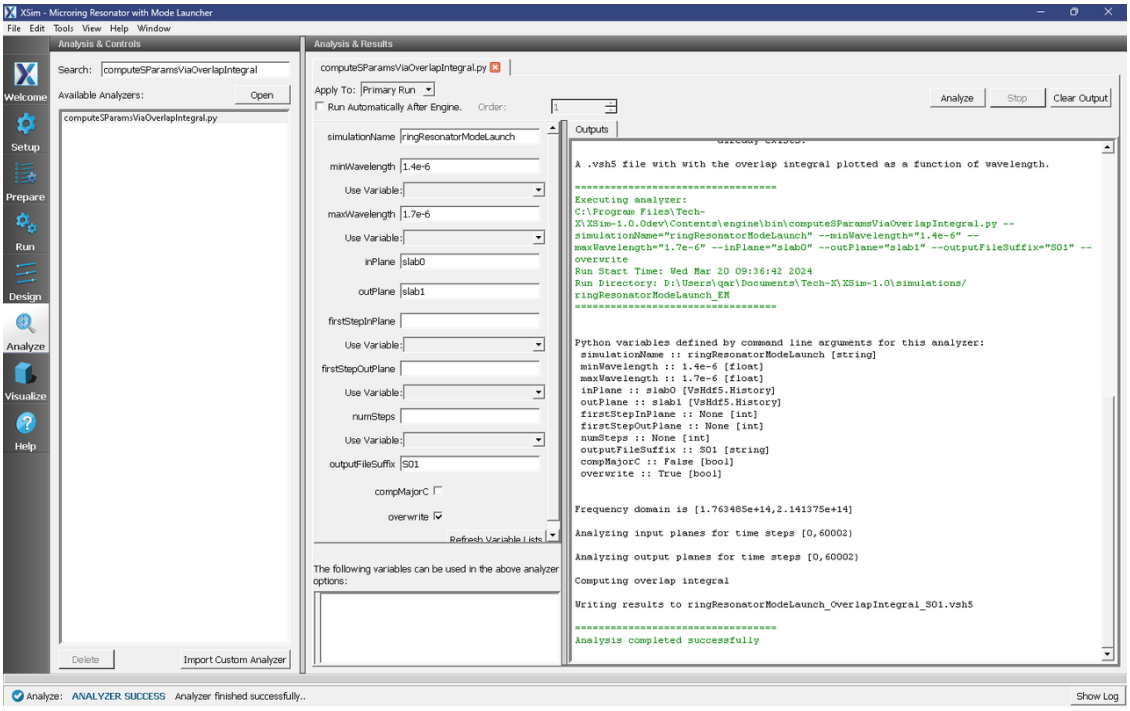

Fig. 2.116: Analyze window with output from computeSParamsViaOverlapIntegral.py.

Visualizing the results

After performing the above actions proceed to the Visualize Window by pressing the *Visualize* button in the left column of buttons.

One can visualize the magnetic field by performing the following:

- Near the top left corner of the window, select *Data Overview* from the *Add a Data View* drop-down.
- Expand *Scalar Data*, then *B*, then select *B_magnitude*
- In the bottom left, select *Clip All Plots*, and set z=0 in *Plane Controls*
- Finally, move the dump slider on the bottom of the window to watch the light propagate

The results are shown in Fig. 2.117.

One can visualize the transmission coefficients by performing the following:

- Near the top left corner of the window, select *1-D Fields* from the *Add a Data View* drop-down.
- In the Plot Control panel select the Slab1 and Slab3 SParams data for Graphs 1 and 2, respectively.
- Set the other Graphs to *None*

The results are shown in Fig. 2.118. As expected, we see coupling of resonant wavelengths with the ring.

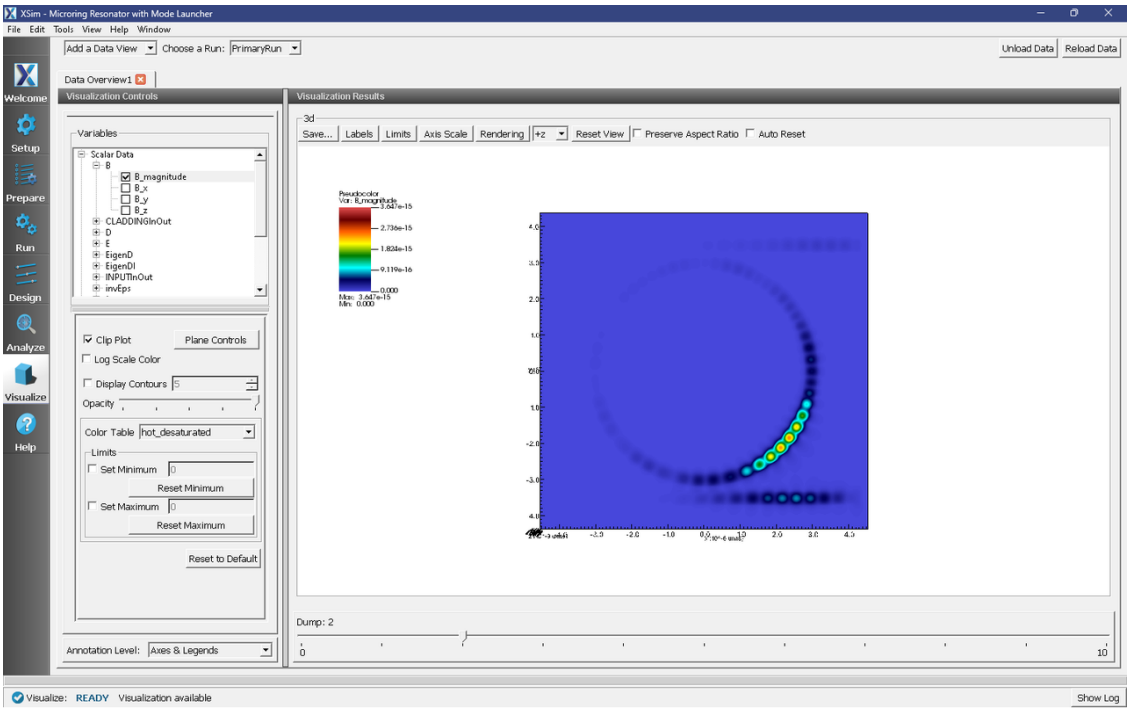

Fig. 2.117: Visualization of magnetic field.

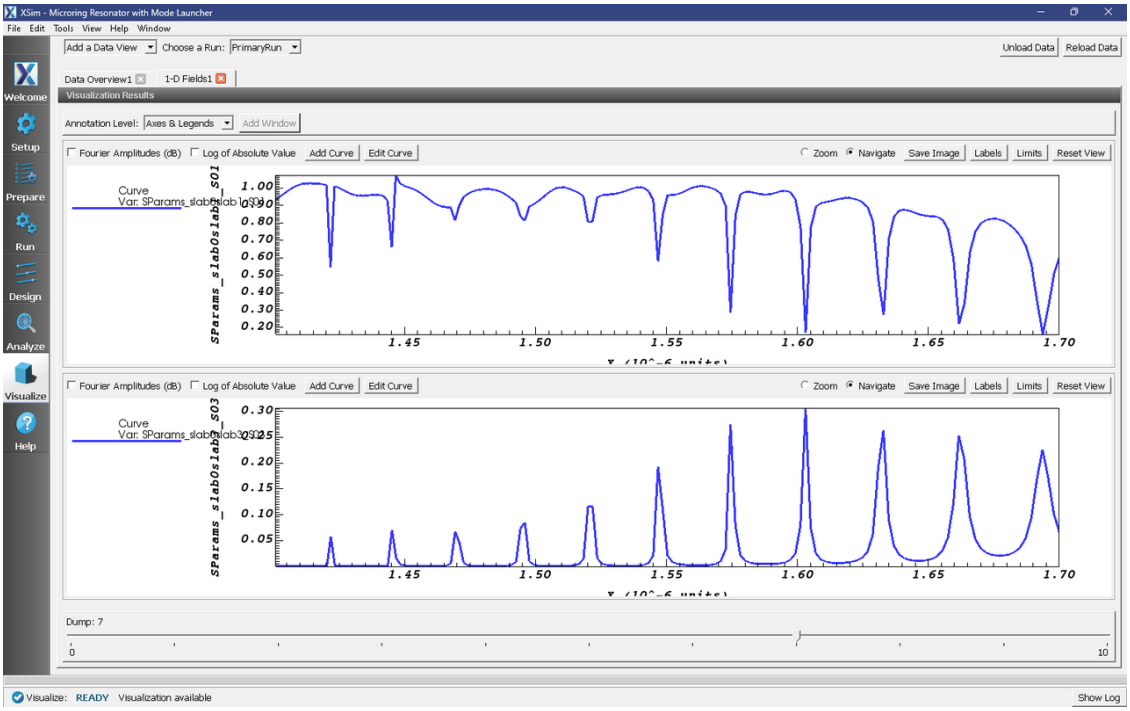

Fig. 2.118: Visualization of the s-parameters.

2.3.5 Y Splitter (ySplitter.sdf)

Keywords:

S Parameter, Mode Launcher, MAL, Guided Mode, Photonic Device, Semiconductor

Problem Description

The Y Splitter makes use of a gds file of a passive photonic device, intended to split the power of a single incoming wave into two separate branches.

The device itself is silicon, surrounded by silicon dioxide.

This examples makes use of a "Photonics simulation template" intended to allow for rapid setup of any passive photonic device, provided the ports are aligned with the X-Axis. Matched Absorbing Layers (MALs) are used to dampen the E, B and D fields near the boundary of the simulation.

The fundamental guided mode profile is launched as a wide band pulse in the input waveguide. This mode is imported from the file port1_EigenD_0.vsh5, produced by the computeWaveguideModes.py analyzer. We will use the computeSParamsViaOverlapIntegral.py analyzer to determine the transmission coefficients at the two exit ports of the device.

This simulation out of the box is set to a minimal resolution, allowing for a faster simulation time, with 8 cells per wavelength (in Silicon) used. For typical use cases a value of 12-18 cells per wavelength is recommended.

This simulation can be performed with a XSimEM license.

Opening the Simulation

The Y Splitter example is accessed from within XSimComposer by the following actions:

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *XSim for Electromagnetics* option.
- Expand the *Photonics* option.
- Select *Y Splitter* and press the *Choose* button.
- In the resulting dialog, create a New Folder if desired, and press the *Save* button to create a copy of this example.

All of the properties and values that create the simulation are available in the Setup Window as shown in Fig. 2.119. You can expand the tree elements and navigate through the various properties. The right pane shows a 3D view of the geometry, as well as the grid, if actively shown. To show or hide the grid, expand the Grid element and select or deselect the box next to *Grid*.

Simulation Properties

This example contains a number of *Constants* defined to make the simulation easily modifiable. These are also intended to be reset for a different gds file and will redefine the simulation size and parameters

General Simulation Constants:

- WAVELENGTH_SINGLE = Used for a single wavelength simulation and calculation of simulation information based on wavelength.
- WAVELENGTH_MIN = Minimum wavelength for a multi-wavelength simulation.
- WAVELENGTH_MAX = Maximum wavelength for a multi-wavelength simulation.

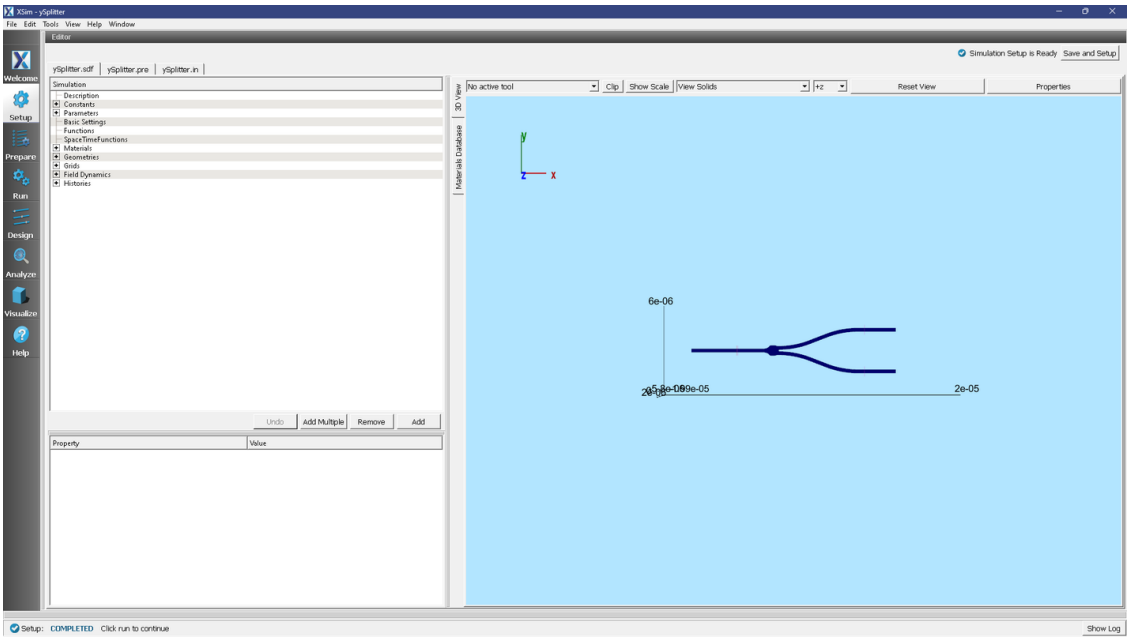

Fig. 2.119: The Setup window for the y splitter example showing the imported gds file.

- WAVELENGTH_XMAL = Number of wavelengths of MAL in the X-direction.
- WAVELENGTH_YMAL = Number of wavelengths of MAL in the Y-direction.
- WAVELENGTH_ZMAL = Number of wavelengths of MAL in the Z-direction.
- PORT1_X = Location of port 1 (excitation port) on the X-axis.
- PORT1_PHYS_YBGN = Beginning of Port 1 on the Y-axis.
- PORT1_PHYS_YEND = End of Port 1 on the Y-axis.
- PORT_PHYS_ZBGN = Beginning of all ports on the Z-axis
- PORT_PHYS_ZEND = End of all ports on the Z-axis
- PORT2_X = Location of port 2 on the X-axis.
- PORT2_PHYS_YBGN = Beginning of Port 2 on the Y-axis.
- PORT2_PHYS_YEND = End of Port 2 on the Y-axis.
- PORT3_X = Location of port 3 on the X-axis.
- PORT3_PHYS_YBGN = Beginning of Port 3 on the Y-axis.
- PORT3_PHYS_YEND = End of Port 3 on the Y-axis.
- PORT4_X = Location of port 4 on the X-axis.
- PORT4_PHYS_YBGN = Beginning of Port 4 on the Y-axis.
- PORT4_PHYS_YEND = End of Port 4 on the Y-axis.
- PORT5_X = Location of port 5 on the X-axis.
- PORT5_PHYS_YBGN = Beginning of Port 5 on the Y-axis.
- PORT5_PHYS_YEND = End of Port 5 on the Y-axis.

- PORT1_LEFT_SIDE = Set to 1 if on the left side of the simulation, -1 if on the right.
- PORT2_LEFT_SIDE = Set to 1 if on the left side of the simulation, -1 if on the right.
- PORT3_LEFT_SIDE = Set to 1 if on the left side of the simulation, -1 if on the right.
- PORT4_LEFT_SIDE = Set to 1 if on the left side of the simulation, -1 if on the right.
- PORT5_LEFT_SIDE = Set to 1 if on the left side of the simulation, -1 if on the right.
- MIN_Y_COORDINATE = Minimum Y coordinate of the component, used if less than any of the ports.
- MAX_Y_COORDINATE = Maximum Y coordinate of the component, used if greater than any of the ports.
- CELLS_PER_WAVELENGTH_AXIAL = Number of cells per wavelength (in Silicon) along the X axis.
- CELLS_PER_WAVELENGTH_CROSSSECTION = Number of cells per wavelength (in Silicon) along the Y and Z axis.

This simulation applies a wide frequency band signal to the fundamental spatial mode profile. The signal is defined under *SpaceTimeFunctions* and then assigned under *Field Dynamics*, *Fields*, *port1*

The *Materials* section contains just Silicon and Silica. This section is where one can add or edit materials that get attached to CSG objects. These *Materials* contain the relative permittivity.

In *Field Dynamics* there are *FieldBoundaryConditions* which set the boundary conditions of the simulation. In photonics simulations, Matched Absorbing Layers (MALs), are the most stable boundary conditions for preventing reflections.

Under *Basic Settings* you can see that the *dielectric solver* is set to *permittivity averaging*. This feature enables second order accuracy for simulations using dielectrics.

GDSII File Import

The GDS File has already been imported, however this example serves as a good demonstration of how to do so

It may be imported like any other CAD file, however a second dialog box will appear, shown below.

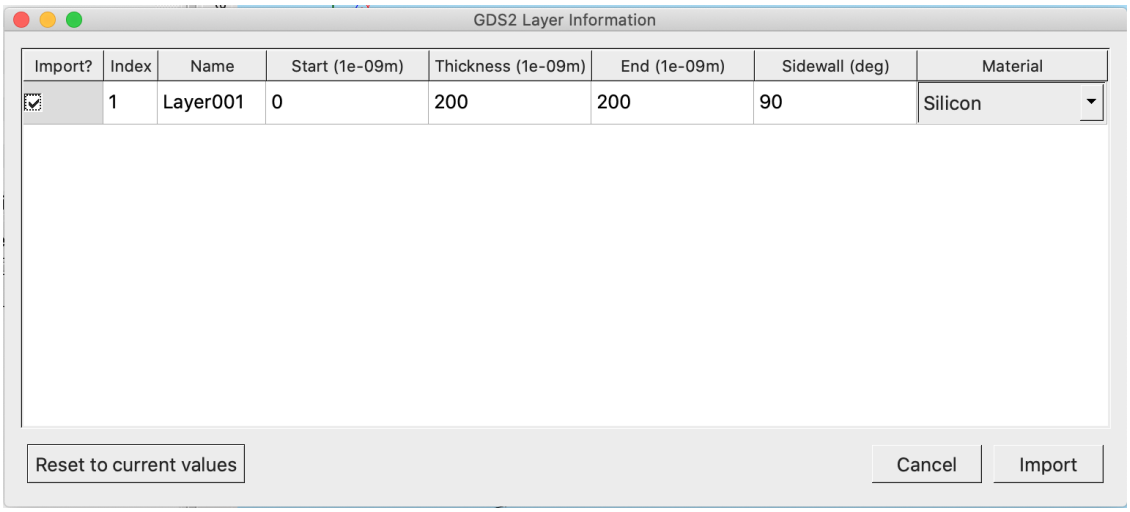

Fig. 2.120: GDS layer selection.

This allows for selection if the layer should be imported, its starting position on the Z axis, and thickness. Sidewall angles can also be set however this is a feature that is still under active development at this time and is recommended to be kept at 90 degrees.

Running the Simulation

When you have saved the setup, continue as follows:

- Proceed to the run window by pressing the Run button in the left column of buttons.
- Check that you are using these run parameters:
  - *Time Step*: 1.5124169044445238e-16
  - *Number of Steps*: 3000
  - *Dump Periodicity*: 500
  - *Dump at Time Zero*: Checked
- Click on the *Run* button in the upper left corner of the right pane.

You will see the output of the run in the right pane. The run has completed when you see the output, "Engine completed successfully". As seen in Fig. 2.121.

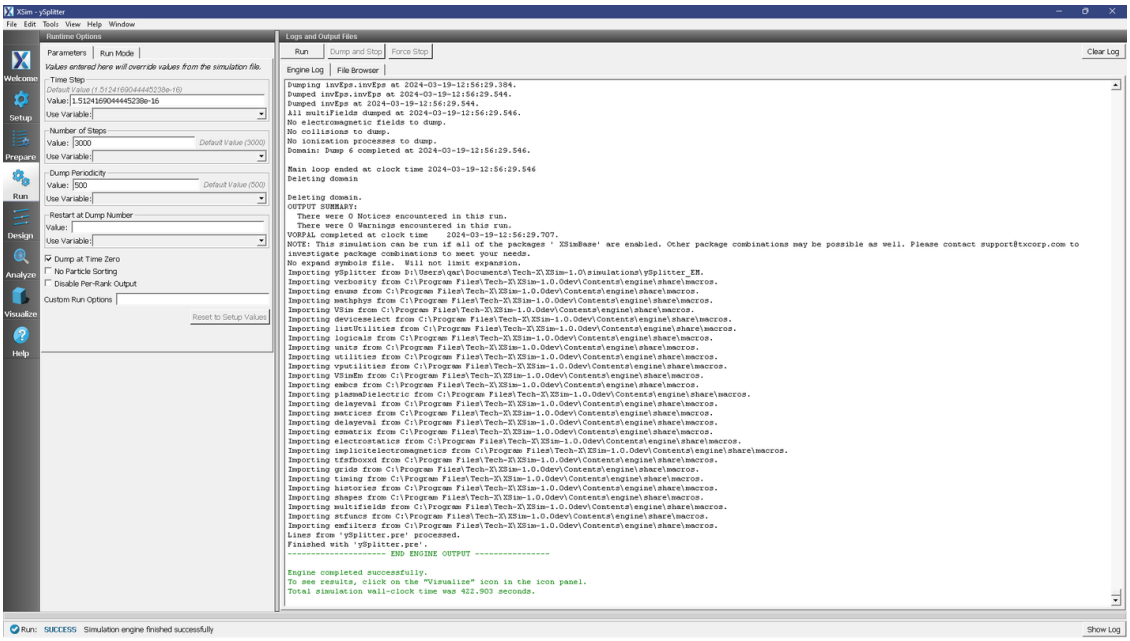

Fig. 2.121: Run window at completion.

Analyzing the Results

Using post analysis scripts, one can extract the transmission coefficients. This simulation uses 3 "EM Field on Plane" histories (Port1, Port2, Port3) and this data is used to calculate S parameters between these 3 locations using the analysis script computeSParamsViaOverlapIntegral.py.

Follow these steps:

- Proceed to the Analyze Window by clicking the *Analyze* button on the left.
- Select *computeSParamsViaOverlapIntegral.py* and click *Open* under the list.

Now update the analyzer fields accordingly.

- *maxWavelength*: 1.7e-6 (or use the corresponding parameter from the setup)

- *minWavelength*: 1.4e-6 (or use the corresponding parameter from the setup)
- *inPlane*: Port1
- *outPlane*: Port2 (or Port3)

The remaining parameter default values will work. Hit the *Analyze* button in the top right corner. After a successful run the window should resemble Fig. 2.122.

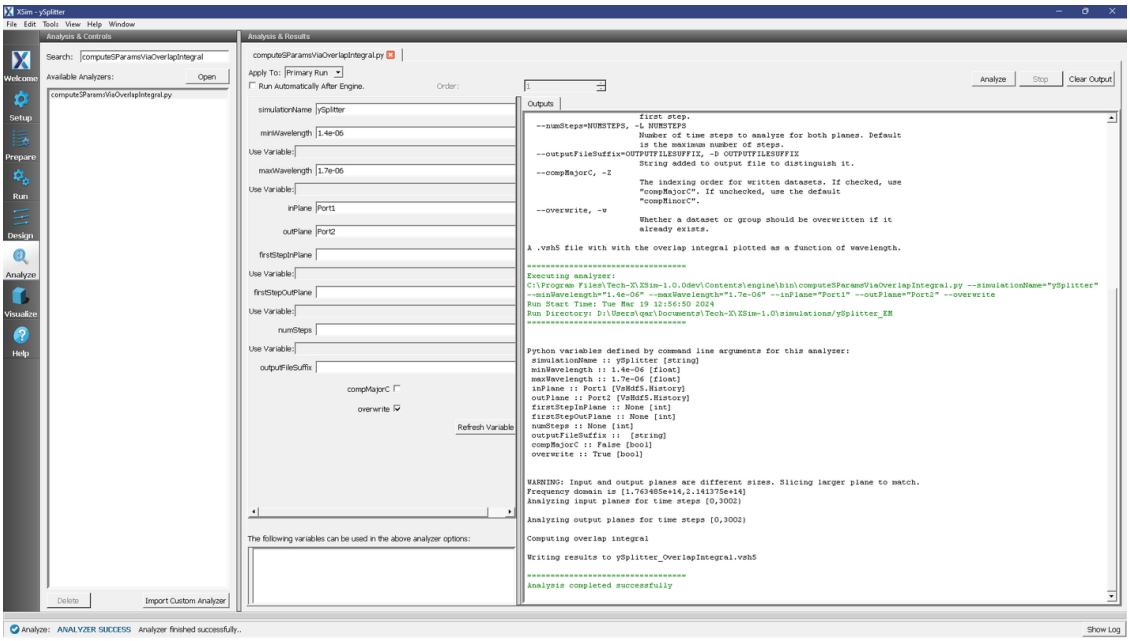

Fig. 2.122: Analyze window with output from computeSParamsViaOverlapIntegral.py.

Visualizing the results

After performing the above actions proceed to the Visualize Window by pressing the *Visualize* button in the left column of buttons.

- Near the top left corner of the window, select *1-D Fields* from the *Add a Data View* drop-down.
- In the Plot Control panel select the Port2 and Port3 SParams data for graph 1.
- Set the other Graphs to *None*

The results are shown in Fig. 2.123. As expected, we see equivalent power split between the two ports across the wavelength range

Further Experiments

This example in its current form uses a lower resolution than one might typically use when designing a waveguide. Try increasing the resolution, recalculating the mode and the two s-parameter curves should grow closer in agreement.

This example can be easily adapted for different GDS files.

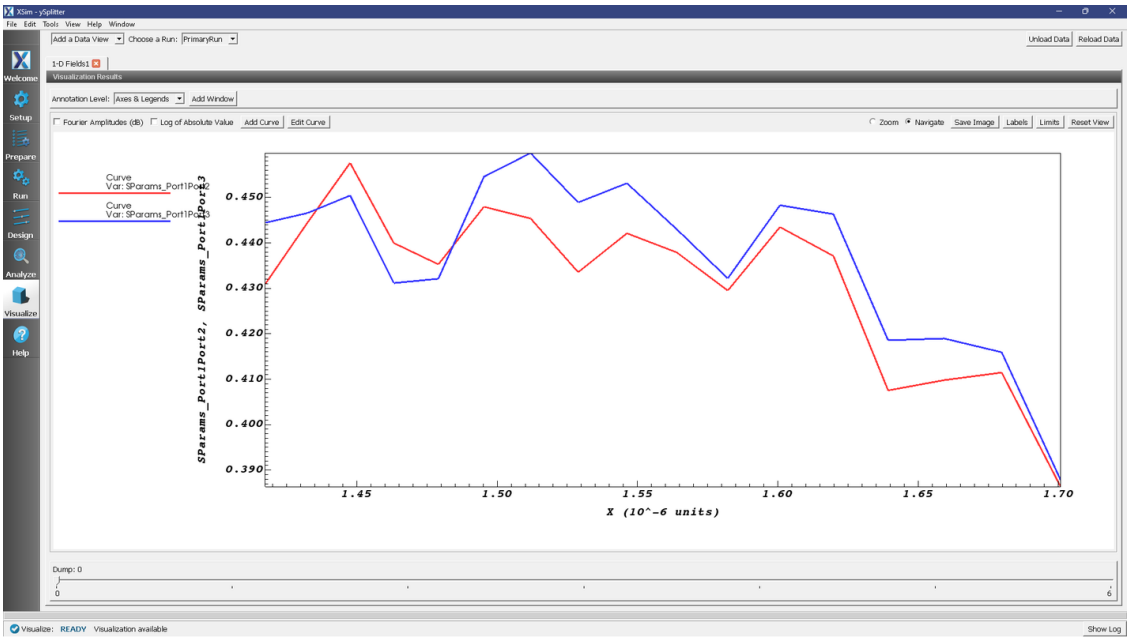

Fig. 2.123: Visualization of the s-parameters.

2.3.6 Three Material Waveguide (threeMaterialWaveguide.sdf)

Keywords:

Photonic Waveguide, Guided Mode, Semiconductor

Problem Description

This example demonstrates how to calculate and launch the fundamental mode of a rectangular silicon waveguide using a unidirectional wave launcher and a broadband signal. In this example, the waveguide itself is sitting on top of a silica slab and is surrounded on the other 3 walls by quartz.

This simulation can be performed with a XSimEM license.

Opening the Simulation

To open this example open an instance of XSimComposer and follow the steps below:

- Select the *New* → *From Example…* menu item in the *File* menu.
- In the resulting *Examples* window expand the *XSim for Electromagnetics* option.
- Expand the *Photonics* option.
- Select *Three Material Waveguide* and press the *Choose* button.
- In the resulting dialog, create a *New Folder* if desired, and press the *Save* button to create a copy of this example.

Simulation Variables

This example contains a number of variables defined to make the simulation easily modifiable.

- CPWX and CPWYZ: CPWX controls the number of cells per vacuum wavelength in the x direction and CPWYZ controls the number of cells per vacuum wavelength in both the y and z directions. Increasing one or both of these values will increase the resolution of the simulation.
- WAVELENGTH: Center wavelength of the input signal. This wavelength is also used for the calculation of the fundamental guided mode of the device.
- WAVEGUIDE_WIDTH and WAVEGUIDE_HEIGHT: These two parameters control the cross sectional area of the waveguide.

The *Materials* section contains silicon, silica, and quartz. The *Geometries* section includes the CSG waveguide, the silica slab, and their defining parameters. In this example, filling the rest of the empty space with quartz is done by going to the *Basic Settings* section and setting the *background permittivity* variable to 4.5. In *Field Dynamics*, there are *FieldBoundaryConditions* and *CurrentDistributions* to be aware of. In photonics simulations, Matched Absorbing Layers (MALs) are the most stable boundary conditions for preventing reflections. This simulation makes use of XSim's Wave Launcher to launch a unidirectional wave down the waveguide with a mode defined by the file generated by the computeWaveguideModes.py analyzer.

Solving for the Eigenmodes

To calculate the fundamental mode of the silicon waveguide, first make sure that in the *Setup* tab that *wavelauncher0* is set to *inactive*. Save the simulation and proceed to the *Run* tab. Set *Number of Steps* and *Dump Periodocity* to 1 and make sure that *Dump at Time Zero* is checked. Click on the *Run* button. Upon successful completion of the engine, proceed as follows:

- Proceed to the Analyze Window by clicking the *Analyze* button on the left.
- Select *computeWaveguideModes.py* and click *Open* under the list.
- Update the analyzer fields accordingly by choosing the corresponding variables under *Use Variable* :
  - *simulationName*: threeMaterialWaveguide
  - *datasetName*: invEps
  - *transverseSliceX*: 0
  - *transverseSliceLY*: PORT_BGNYZ
  - *transverseSliceUY*: PORT_ENDYZ
  - *transverseSliceLZ*: PORT_BGNYZ
  - *transverseSliceUZ*: PORT_ENDYZ
  - *vacWavelength*: 1.55e-6
  - *nModes*: 1
  - *writeFieldProfile*: D
  - *modeFileName*: port1
  - *Normalize*: checked
  - *compMajorC*: unchecked
  - *Overwrite*: checked
- Click *Analyze* in the top right corner.

The analyzer will only find guided modes. The results should resemble Fig. 2.124. We see that the analyzer found the fundamental mode and gives the effective index of the mode under *Neff*. This parameter is necessary for using the *Wave Launcher* to launch a unidirectional wave. This value is saved as a parameter as *NEFF*.

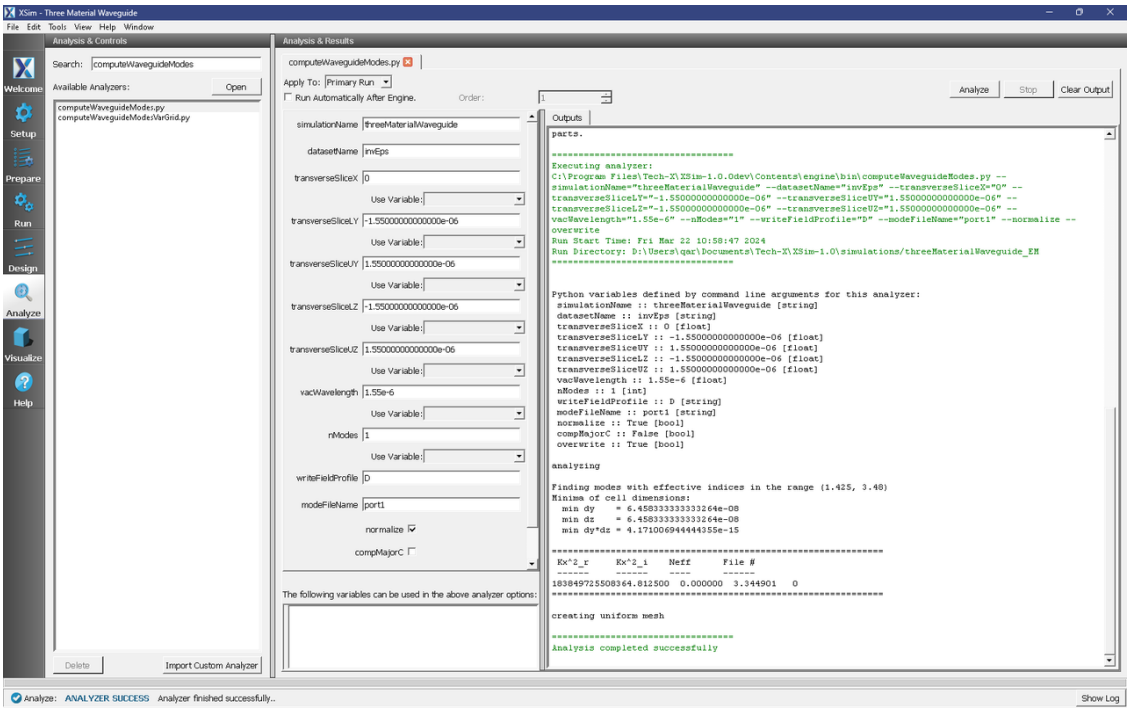

Fig. 2.124: The analyzer window after a successful run of computeWaveguideModes.py.

Setting up the Simulation

After performing the above actions, continue as follows:

- Proceed to the Setup Window by pressing the *Setup* button in the left column of buttons.
- Activate the wave launcher
  - Expand *Field Dynamics*
  - Expand *CurrentDistributions*
  - Right Click on *waveLauncher0*
  - Left click on *Activate*
- Click on *Save and setup* in the top right corner.

This setting is shown in Fig. 2.125.

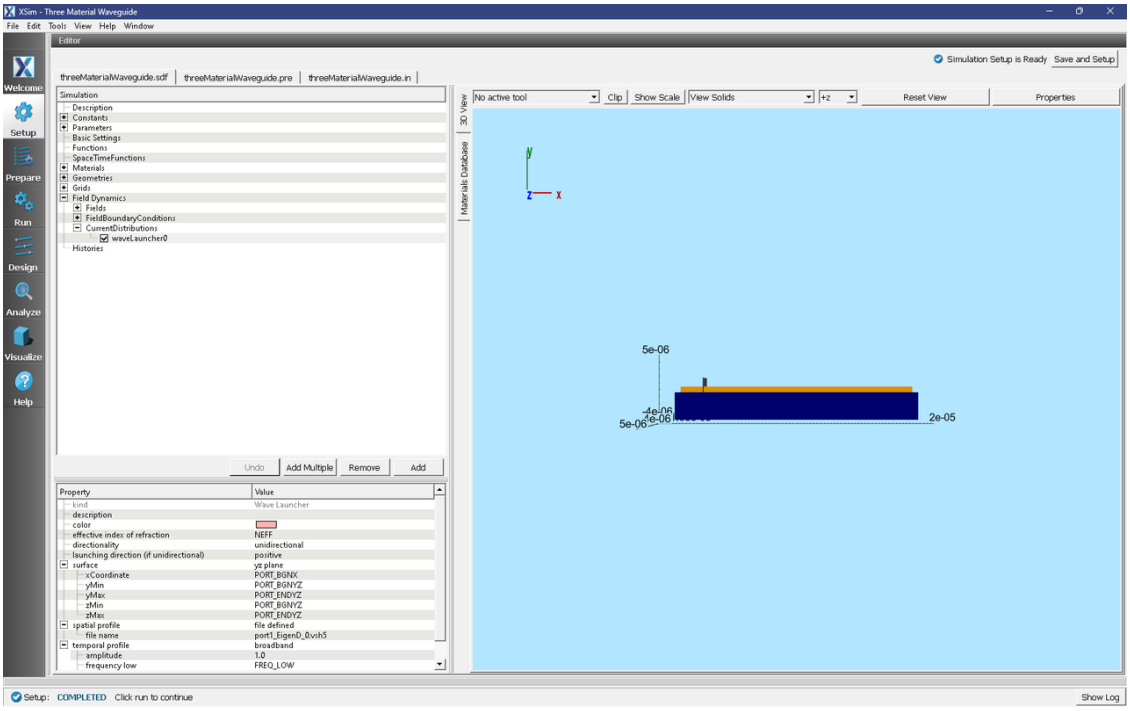

Fig. 2.125: Location of wavelauncher0 within the setup tree.

Running the Simulation

After performing the above actions, continue as follows:

- Proceed to the Run Window by pressing the *Run* button in the left column of buttons. You will be asked to Save. Click *Save* upon the request to save.
- In the left pane, click on *Reset to Setup Values* to run the full simulation which launches this wave down the waveguide.
- Check *Dump at Time Zero*.
- Click on the *Run* button in the upper left corner of the right pane.

You will see the output of the run in the right pane. The run has completed when you see the output, "Engine completed successfully." This result is shown in Fig. 2.126.

Visualizing the Results

After performing the above actions, continue as follows:

- Proceed to the Visualize Window by pressing the *Visualize* button in the left column of buttons.

You may need to *Reload Data* (bottom left). Visualize an eigenmode by following these steps:

- From the *Add a Data View* dropdown, select *Data Overview*.
- Expand *Scalar Data*, expand *EigenD*, and select *EigenD_magnitude*.

The resulting visualization pane should look like Fig. 2.127.

To visualize the E field as it travels down the waveguide, proceed as follows:

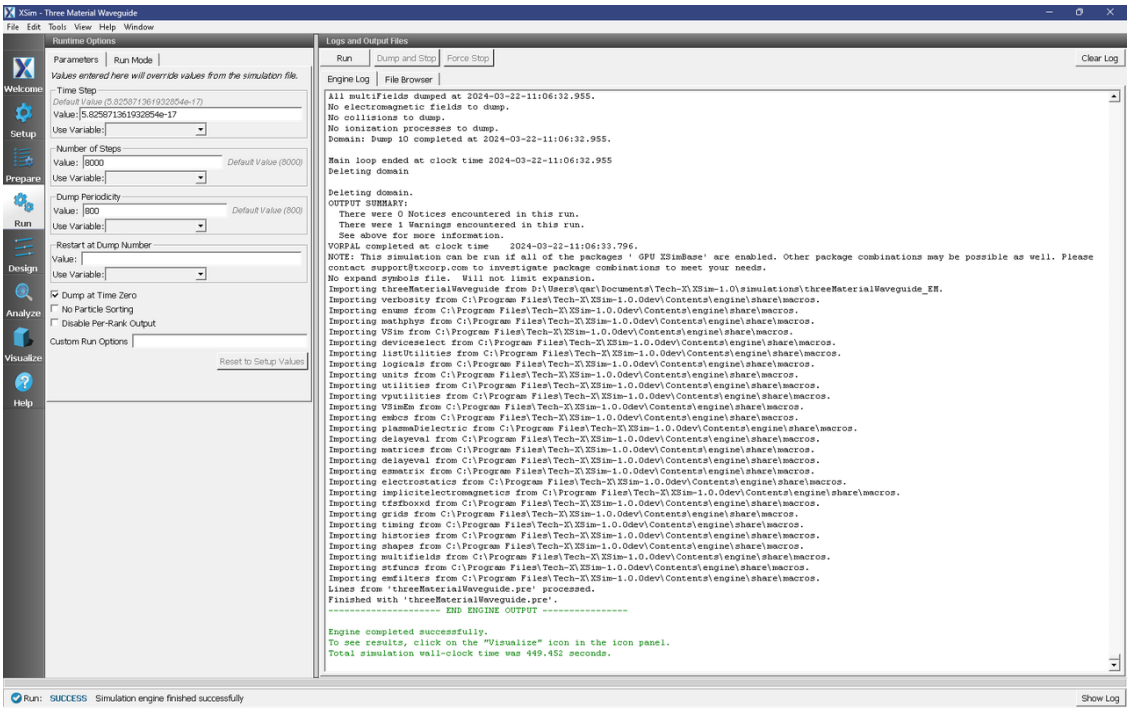

Fig. 2.126: Running the full simulation which launches the wave down the waveguide.

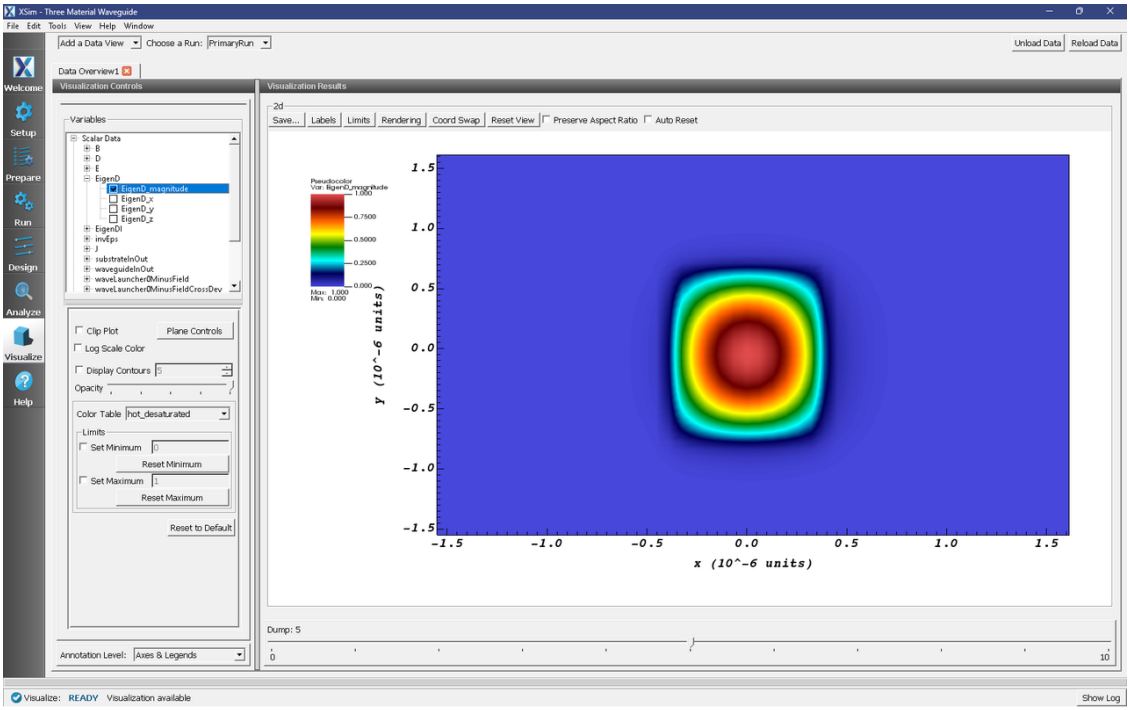

Fig. 2.127: The visualization pane showing the magnitude of the D field of the fundamental mode.

- From *Data Overview*, expand *Scalar Data*, expand *E*, then select *E_z*.
- Slide the dump number selector over to the right to watch the wave travel down the waveguide.

The resulting visualization pane should look like Fig. 2.128.

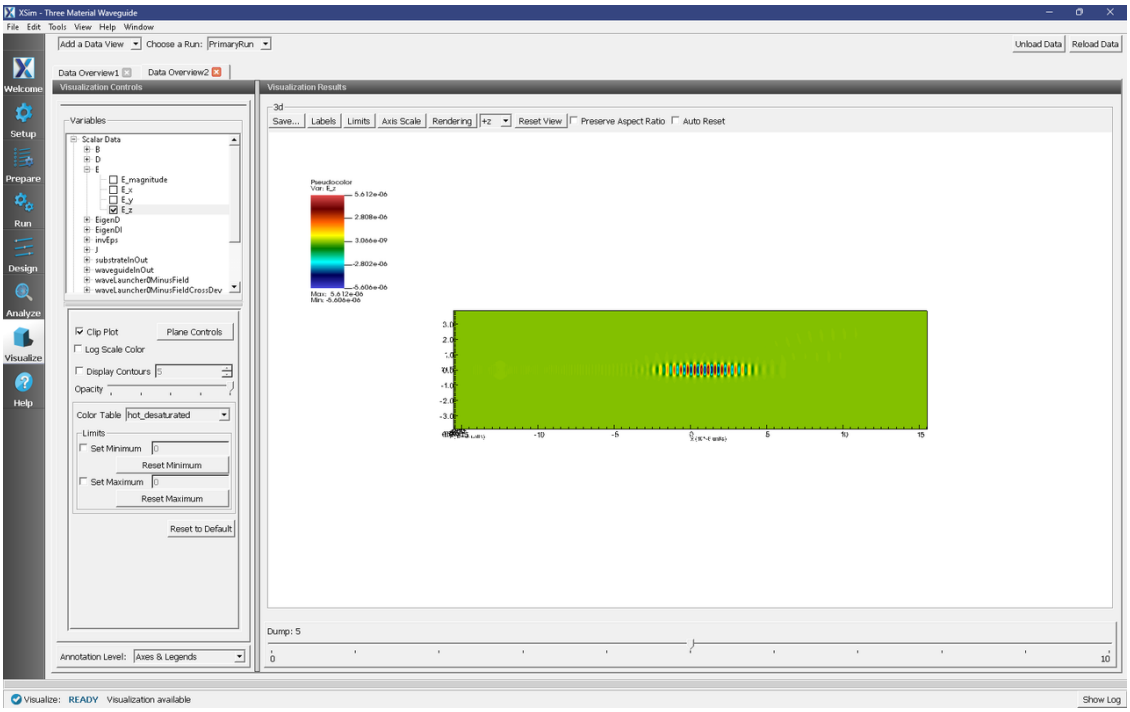

Fig. 2.128: The visualization pane showing the z component of the E field.

If instead you wish to view a cross section of the waveguide and E field, proceed as follows:

- From *Data Overview*, expand *Scalar Data*, expand *E*, then select *E_z*.
- In the visualization controls section, click on the *Plane Controls* button. Deselect Plane3 and select Plane1. Click the *Apply* button. Click *Ok*.
- In the visualization results view, to the left of the *Reset View* button is a drop down menu. Change this to *+x*.
- To make the field outside of the waveguide, change the minimum and maximum values under *Limits* to -5e-6 and 5e-6.

The resulting visualization pane should look like Fig. 2.129.

Further Experiments

Change the geometry on the Setup window and rerun the simulation and analyzer to see the effects on the modes. Change the materials used and see how the field inside and outside the waveguide changes.

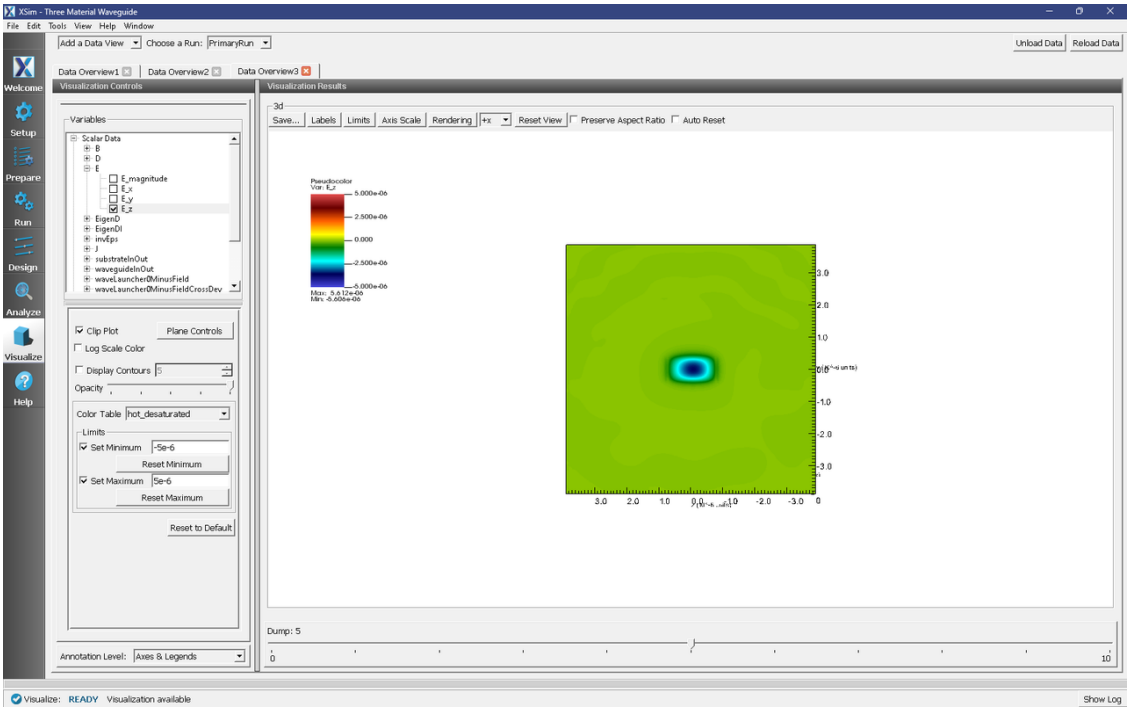

Fig. 2.129: The visualization pane showing the z component of the E field in a cross section of the domain.

2.4 Scattering

2.4.1 Scattering off a Metal Sphere (metalSphere.sdf)

Keywords:

`Mie Scattering Off Metal Sphere`

Problem description

The example describes the scattering of electromagnetic waves off a metal sphere. This phenomenon is often referred to as Mie scattering, where the radius of the sphere is comparable to the wavelength of the incident radiation.

An incident plane wave is launched toward the sphere. XSim computes the resulting fields in the vicinity of the sphere, within a computational domain by applying the proper boundary conditions around the surface of the sphere. The waves that exit the computational domain are absorbed into MAL layers.

The fields for points far away from the sphere, and beyond the computational domain are computed with the help of an analyzer that is part of the XSim distribution. The histories of the electric and magnetic fields are recorded along a closed surface known as a Kirchhoff box that lies within the MAL layers. This field information is then used to compute fields far away from the sphere center by applying the Kirchhoff integral theorem.

In this example, the radius of the metal sphere is set to equal 0.3367 times the wavelength of the incident radiation, which is 1m in length.

The wave is launched from the positive z direction, and the incident wave electric field is polarized along the x direction.

This example is set up as a cube, entirely parameterized off the NUM_WAVELENGTHS, WAVELENGTH and CELLS_PER_WAVELENGTH Constant. It is designed so that it may be easily adapted to take the radar cross section

of any geometry, at any wavelength.

Two wavelengths on all sides are devoted to the MAL absorbing boundary conditions. The resolution of the grid corresponds to 16 cells per wavelength, near the middle of the typically used 10-20 cells. The time step (Parameter DT) is chosen to be very close to the Courant condition limit, calculated using the DX, CFL_NUM and DMFRAC Parameters.

The recording time for the Kirchhoff box is calculated using the parameters TBGNKBOX and TENDKBOX. TBGNKBOX is set by adding the turn on time of the excitation source, and 2* the time it would take to cross the diagonal of the RCS Box. TENDKBOX is set by taking TBGNKBOX, and then adding the amount of time to cross the entire RCS box, +2.5 periods. This allows for the collection of 2.5 periods of data.

The number of timesteps (Parameter NUM_STEPS) for the simulation corresponds to TENDKBOX/DT. Note that the number of steps in the simulation must be set by hand in **Basic Settings**, and verify that the value used in the Run Panel corresponds to this parameter.

Opening the Simulation

The Mie Scattering, Metal Sphere example is accessed from within XSimComposer by the following actions:

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *XSim for Electromagnetics* option.
- Expand the *Scattering* option.
- Select "Mie Scattering - Metal Sphere" and press the *Choose* button.
- In the resulting dialog, create a New Folder if desired, and press the *Save* button to create a copy of this example.

All of the properties and values that create the simulation are now available in the Setup Window as shown in Fig. 2.130. You can expand the tree elements and navigate through the various properties, making any changes you desire. The right pane shows a 3D view of any geometry, as well as the grid, if actively shown. To show or hide the grid, expand the Grid element and select or deselect the box next to `Grid`.

The Setup Window shows the Kirchhoff box, with the metal sphere at the center. To see the grid, click on the drop down menu of "Grids" on the left side panel. Check the box beside "Grid". In order to view or change the wave frequency, click on the drop down menu "Parameters" and then click on "Frequency". To view or change the direction of the incident wave or polarization, you can click on the drop down menu "Field Dynamics", then the drop down menu "RCSBox". Following this, check the box beside "rcsBox0". The lower left panel displays a table with the wave property and its corresponding value.

Running the simulation

After performing the above actions, continue as follows:

- Proceed to the run window by pressing the Run button in the left column of buttons.
- Check that you are using these run parameters:
  - *Time Step*: 5.71337236814229e-11
  - *Number of Steps*: 2015
  - *Dump Periodicity*: 300
  - *Dump at Time Zero*: Checked
- Click on the *Run* button in the upper left corner of the right pane.

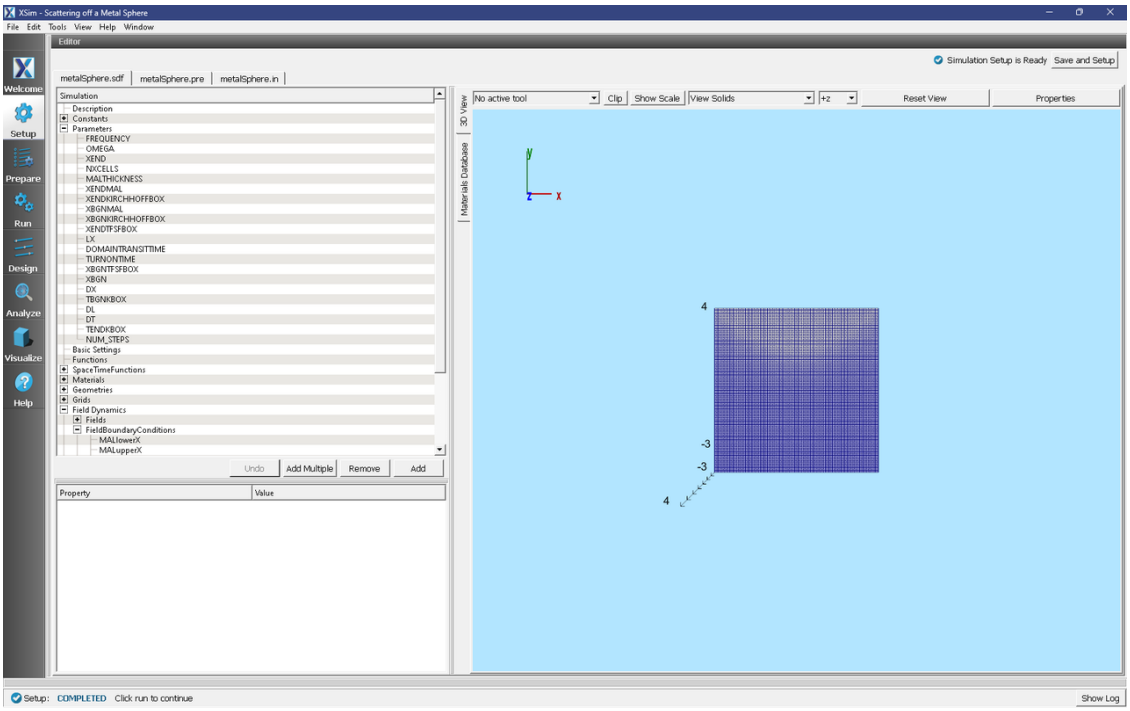

Fig. 2.130: Setup Window for the Mie Scattering example.

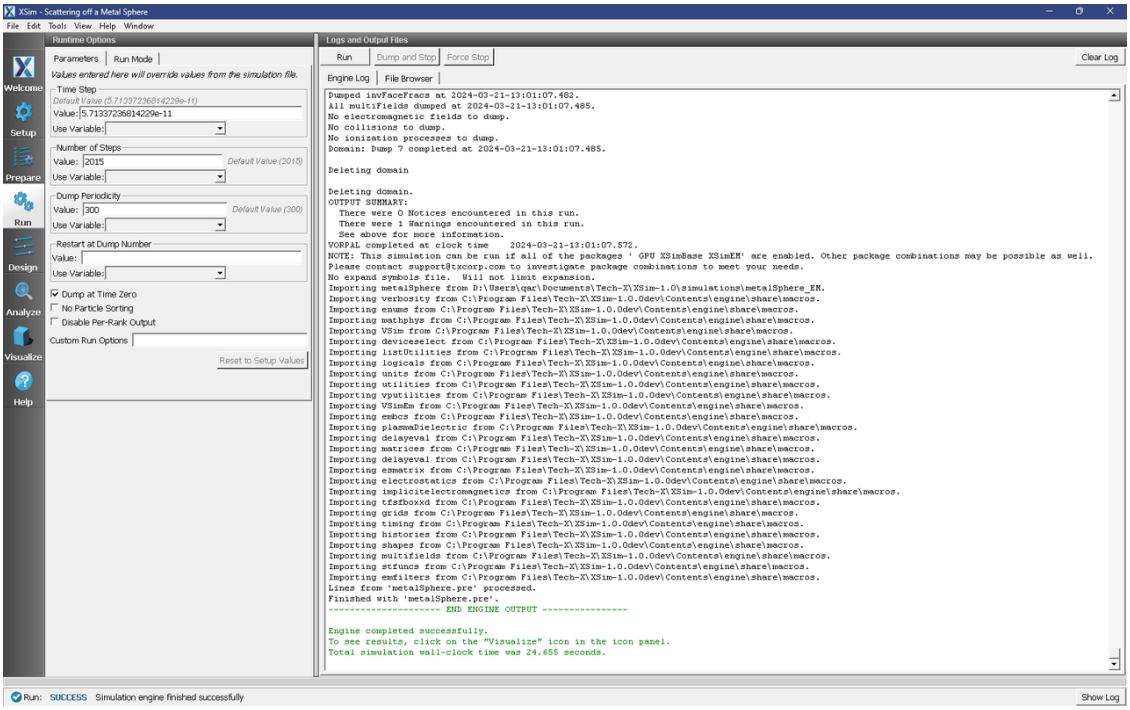

Fig. 2.131: The Run Window at the end of simulation.

You will see the output of the run in the right pane. The run has completed when you see the output, "Engine completed successfully". This result is shown in Fig. 2.131.

Running in 3D, this simulation uses 2000 time steps. The run takes about 5 minutes on a 4-core 2.3 GHz processor.

Running the Analyzer

- Proceed to the Analysis window by pressing the Analyze button in the left column of buttons.
- Select the Default *computeFarFieldFromKirchhoffBox.py* Analyzer
- Input values for the variables given on the left hand side of the screen. Check that these have the following values:
  - simulationName - metalSphere
  - fieldLabel - E
  - farFieldRadius - 1024.0
  - numPeriods - 0.25
  - numFarFieldTimes - 2
  - frequency - 0.3e9
  - numTheta - 45 (number of points in the theta direction)
  - numPhi - 90 (number of points in the phi direction)
  - zeroThetaDirection - (0,0,1)
  - zeroPhiDirection - (1,0,0)
  - incidentWaveAmplitude - 1.0
  - incidentWaveDirection - (0,0,1)
  - varyingMeshMaxRadius - 1.0
  - principalPlanesOnly - checked
- Click the *Analyze* button near the top right of the window.

Visualizing the results

After performing the above actions, click on *Visualize* in the column of buttons at the left. The program will load the data and provide you with certain options.

One of the quantities that is of interest in such scattering phenomena is radar cross section (RCS) measured in dBsm. The RCS, sometimes designated as σ, having units of m^2, is given as

$$RCS = 4\pi R_s^2 \frac{P_r}{P_i}$$

where R_s is the radial distance from scatterer, P_r is the power flux received at the point of interest, and P_i is the incident power flux. In MKS units the power flux is measured in W/m^2. RCS in dBsm is given as

$$10 \log_{10}(RCS)$$

To obtain plots of this quantity, click on the drop down menu, *Add a Data View*. In this menu, choose 1-D Fields. There is a large list of options to choose from. Figure Fig. 2.133.

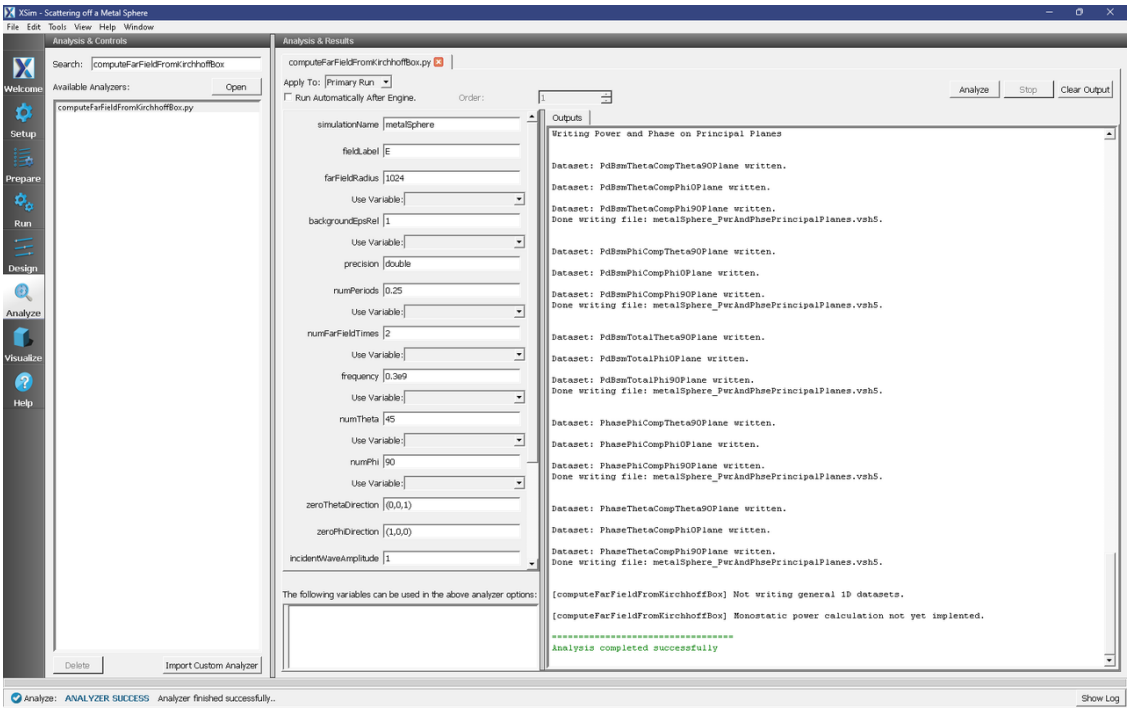

Fig. 2.132: The Analyze Window at the end of analyzer execution.

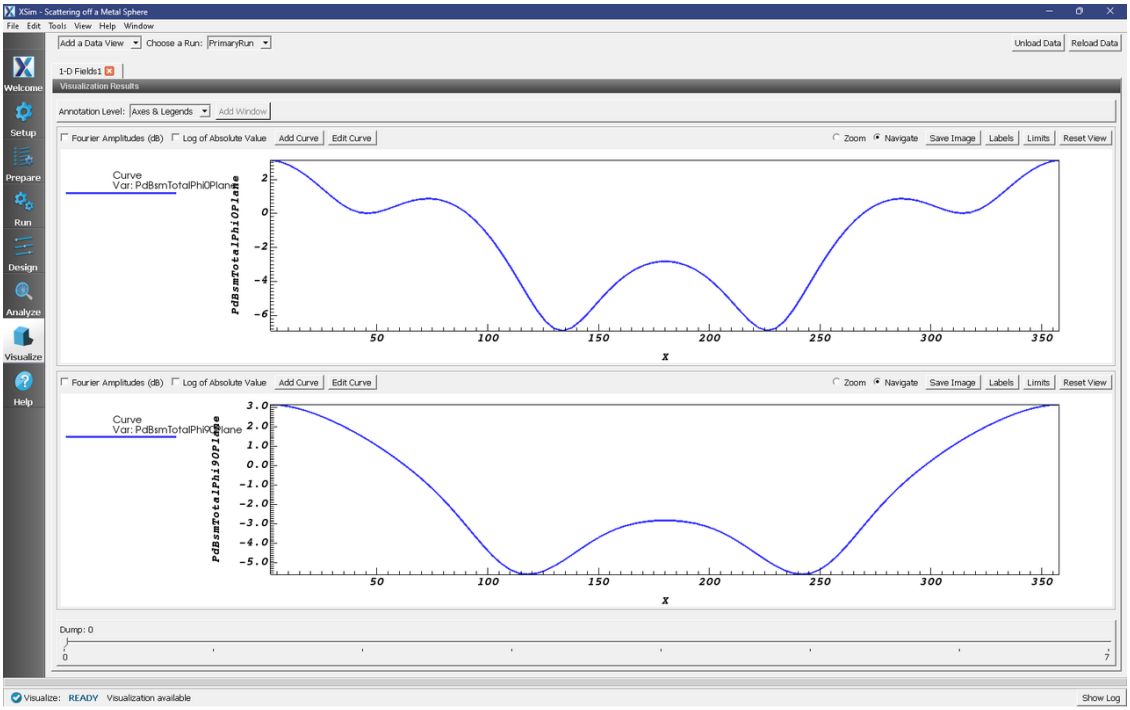

Fig. 2.133: The Visualize Window at the end of execution.

Further Experiments

You may try the example that includes a dielectric coating over the same metal sphere to compare the effects of such a coating over the RCS.

2.4.2 Scattering off a Metal Sphere with a Dielectric Coating (dielecCoatedMetalSphere.sdf)

Keywords:

`Mie Scattering Dielectric Coated Metal Sphere`

Problem description

The example describes the scattering of electromagnetic waves off a metal sphere with a dielectric coating, which is a modification of the previous example that described scattering off a pure metal sphere.

An incident plane wave is launched toward the sphere. XSim computes the resulting fields in the vicinity of the sphere, within a computational domain by applying the proper boundary conditions around the surface of the sphere. The waves that exit the computational domain are absorbed into MAL layers.

The fields for points far away from the sphere, beyond the computational domain, are computed with the help of an analyzer that is part of the XSim distribution. The histories of the electric and magnetic fields are recorded along a closed surface known as a Kirchhoff box that lies within the MAL layers. This field information is then used to compute fields far away from the sphere center by applying the Kirchhoff integral theorem.

In this example, the radius of the metal sphere is set to equal 0.3367 times the wavelength of the incident radiation, which is 1m in length. The thickness of the coating is 0.1 times the wavelength. The computational domain extent is set to three times the wavelength in all directions along the three coordinate axes. The thickness of the MAL layer is twice the wavelength. The resolution of the grid corresponds to 24 cells per wavelength. This parameter is chosen such that the thickness of the dielectric layer is resolved. The time step is chosen to be very close to the Courant condition limit. The wave is launched from the positive z direction, and the incident wave electric field is polarized along the x direction. Care needs to be taken so that the simulation is performed for a sufficient number of time steps, so that the wave reaches the surface of the Kirchhoff box where the histories are recorded. Thus, changing the frequency or the resolution of the grid would alter the time step, and the number of steps required to complete the run will need to be changed accordingly.

Opening the Simulation

The Mie Scattering, Dielectric Coated Metal Sphere example is accessed from within XSimComposer by the following actions:

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *XSim for Electromagnetics* option.
- Expand the *Scattering* option.
- Select "Scattering off a Metal Sphere with a Dielectric Coating" and press the *Choose* button.
- In the resulting dialog, create a New Folder if desired, and press the *Save* button to create a copy of this example.

All of the properties and values that create the simulation are now available in the Setup Window as shown in Fig. 2.134. You can expand the tree elements and navigate through the various properties, making any changes you desire. The right pane shows a 3D view of the geometry, if any, as well as the grid, if actively shown. To show or hide the grid, expand the Grid element and select or deselect the box next to `Grid`. While the setup is similar to the previous

example of a pure metal sphere, the additional components in this example may be found as follows: Expand the Menu under `Material` to see the element `COATING`. As showed in Fig. 2.134, clicking on `COATING` shown the properties of the dielectric material. In addition, expand the menu under `Geometries`, followed by the menu under `CSG` to see the additional geometry element, `coating`.

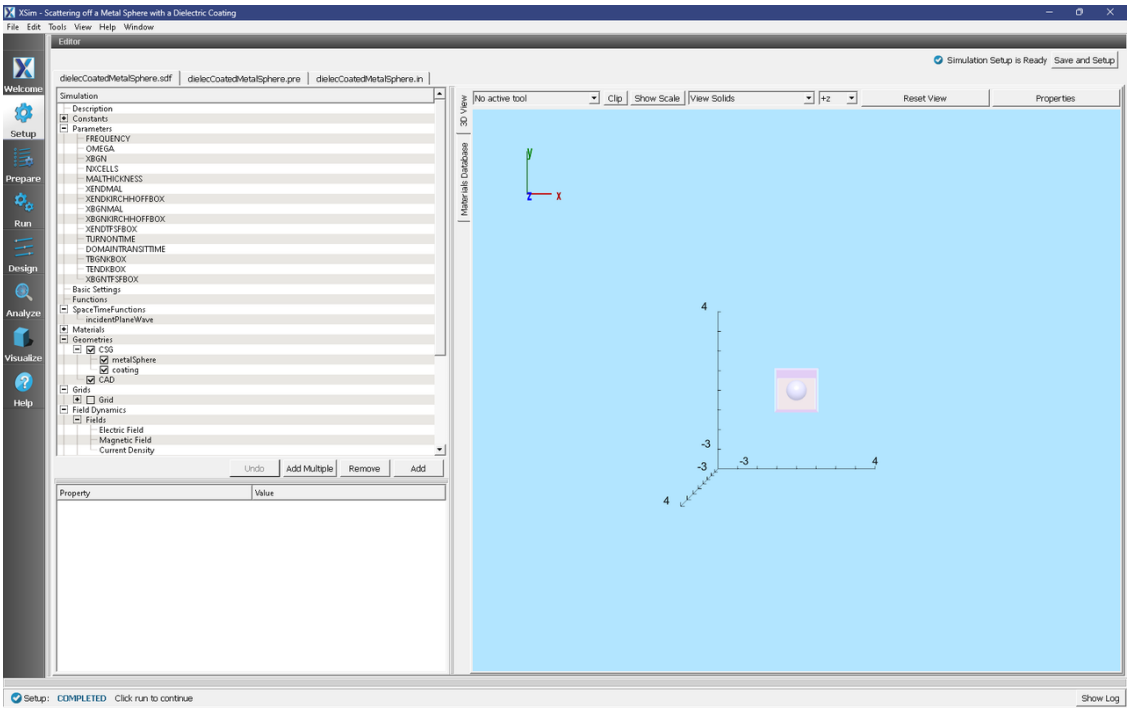

Fig. 2.134: Setup Window for the Mie Scattering example.

The Setup Window shows the Kirchhoff box, with the metal sphere at the center. To see the grid, click on the drop down menu of "Grids" on the left side panel. Check the box beside "Grid". In order to view or change the wave frequency, click on the drop down menu "Parameters" and then click on "Frequency". To view or change the direction of the incident wave or polarization, you can click on the drop down menu "Field Dynamics", then the drop down menu "RCSBox". Following this, check the box beside "rcsBox0". The lower left panel displays a table with the wave property and its corresponding value.

Running the simulation

After performing the above actions, continue as follows:

- Proceed to the run window by pressing the Run button in the left column of buttons.
- Check that you are using these run parameters:
  - *Time Step*: 3.8115448780607225e-11
  - *Number of Steps*: 4000
  - *Dump Periodicity*: 400
  - *Dump at Time Zero*: Checked
- Click on the *Run* button in the upper left corner of the right pane.

You will see the output of the run in the right pane. The run has completed when you see the output, "Engine completed successfully". This result is shown in Fig. 2.135.

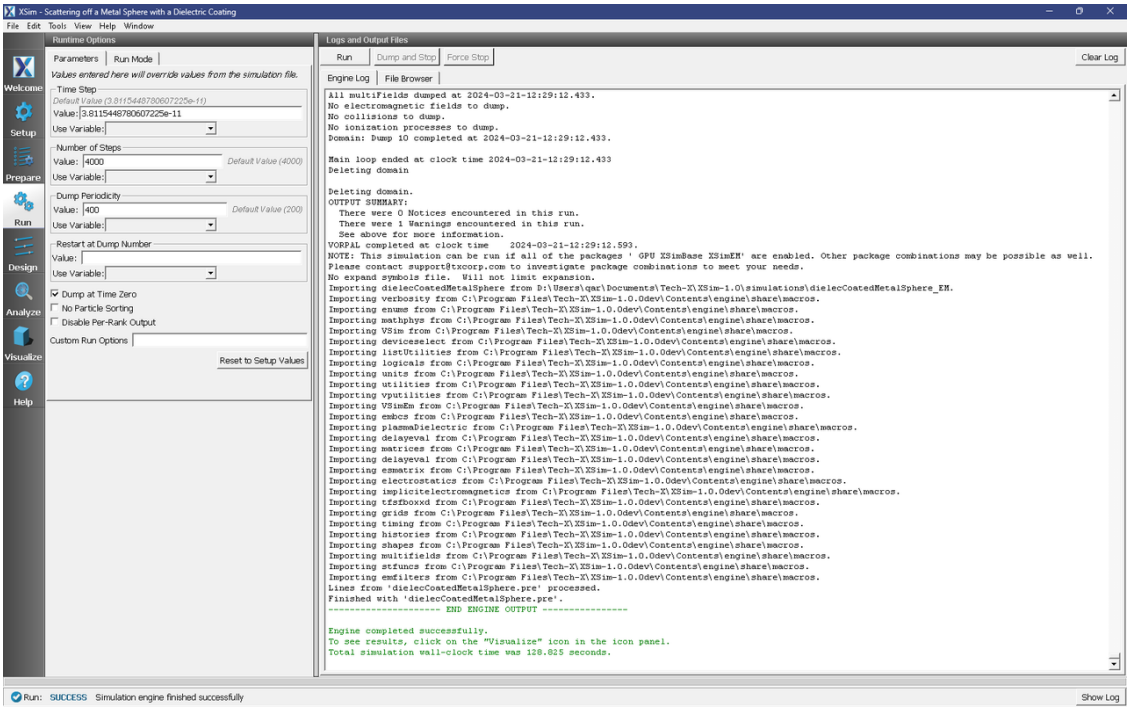

Fig. 2.135: The Run Window at the end of simulation.

Running in 3D, this simulation uses around 4000 time steps. The run takes about 15 minutes on a 4-core 2.3 GHz processor.

Running the Analyzer

After completing the run, one can run the analyzer to compute field values at far away points. To bring up the analyser script, Click on the Analyze icon. The panel that appears will be named computeFarFieldFromKirchhoffBox.py, along with a set of text boxes in which the necessary parameters need to be filled. For the default settings of the file you may use the following:

- *simulationName* : dielectricCoatedMetalSphere
- *fieldLabel* : E
- *farFieldRadius* : 50.0
- *numPeriods* : 0.25
- *numFarFieldTimes* : 2
- *frequency* : 0.3e9
- *numTheta* : 31
- *numPhi* : 64
- *zeroThetaDirection* : (0,0,1)
- *zeroPhiDirection* : (1,0,0)
- *incidentWaveAmplitude* : 1.0
- *incidentWaveDirection* : (0,0,1)

- *varyingMeshMaxRadius* : 1
- *principalPlanesOnly* : checked

After entering the above parameters, also shown in Fig. 2.136, press the "Analyze" button that appears on the upper right side.

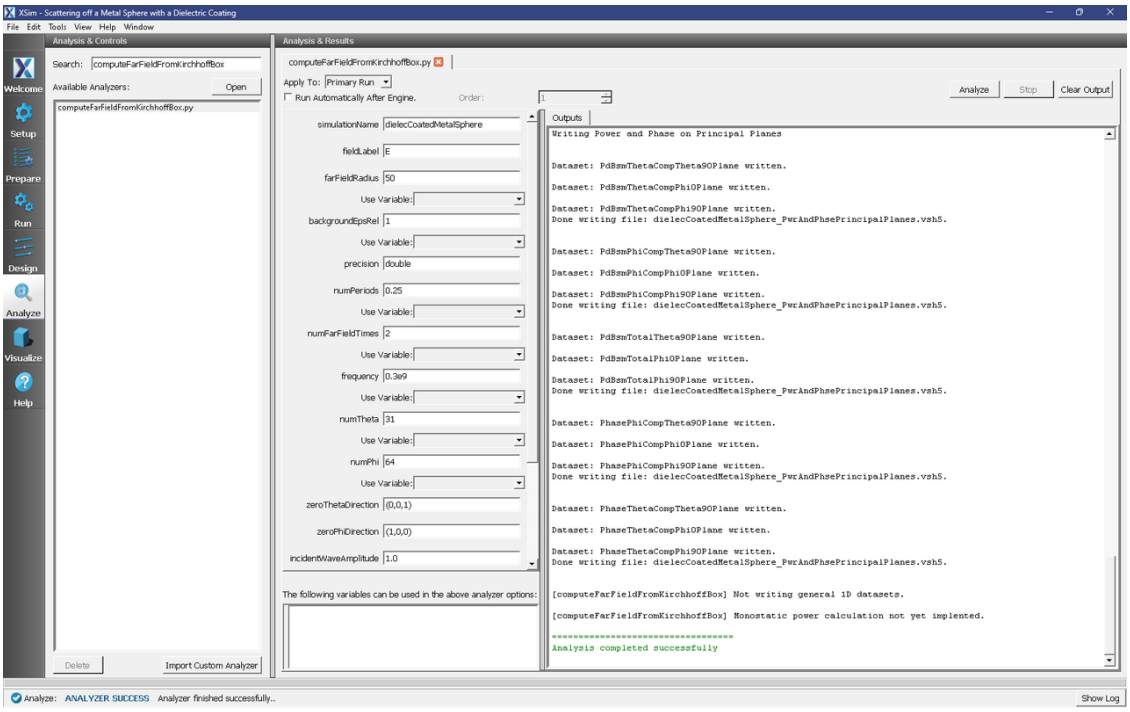

Fig. 2.136: The Analyze Window at the end of analyzer execution.

Visualizing the results

After performing the above actions, click on *Visualize* in the column of buttons at the left. The program will load the data and provide you with certain options.

One of the quantities that is of interest in such scattering phenomena is radar cross section (RCS) measured in dBsm. The RCS, sometimes designated as σ, having units of m^2, is given as

$$RCS = 4\pi R_s^2 \frac{P_r}{P_i}$$

where R_s is the radial distance from scatterer, P_r is the power flux received at the point of interest, and P_i is the incident power flux. In MKS units the power flux is measure in W/m^2. RCS in dBsm is given as

$$10log_{10}(RCS)$$

To obtain plots of this quantity, click on the drop down menu, *Add a Data View*. In this menu, choose 1-D Fields. There is a large list of options to choose from. Figure Fig. 2.137.

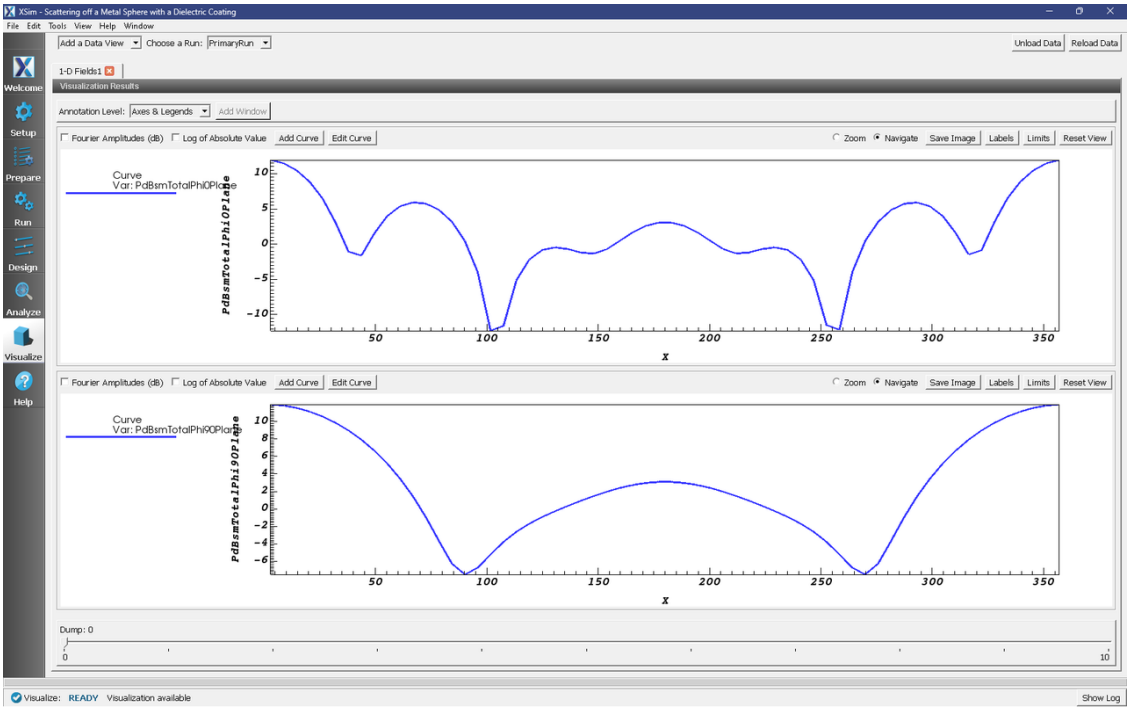

Fig. 2.137: The Visualize Window at the end of execution.

Further Experiments

Alter the thickness or the dielectric constant of the coating to see its effect on the computed RCS.

2.4.3 Scattering off Multiple Objects (dielecPlusMetalObjs.sdf)

Keywords:

`electromagnetics, pulse, dielectric`

Problem Description

The Scattering off Multiple Objects simulation illustrates the ability to define different materials with different dielectric properties (perfect electric conductor, sapphire, alumina) and have an electromagnetic pulse reflect off of both a complex metal surface and dielectric medium. It also illustrates a wave launcher to be used with different dielectric materials. This example can also be modified to calculate Radar Cross Sections.

Opening the Simulation

The Scattering off Multiple Objects example is accessed from within XSimComposer by the following actions:

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *XSim for Electromagnetics* option.
- Expand the *Scattering* option.
- Select "Scattering off Multiple Objects" and press the *Choose* button.
- In the resulting dialog, create a New Folder if desired, and press the *Save* button to create a copy of this example.

All of the properties and values that create the simulation are now available in the Setup Window as shown in Fig. 2.138. You can expand the tree elements and navigate through the various properties, making any changes you desire. The right pane shows a 3D view of the geometry, if any, as well as the grid, if actively shown. To show or hide the grid, expand the Grid element and select or deselect the box next to `Grid`.

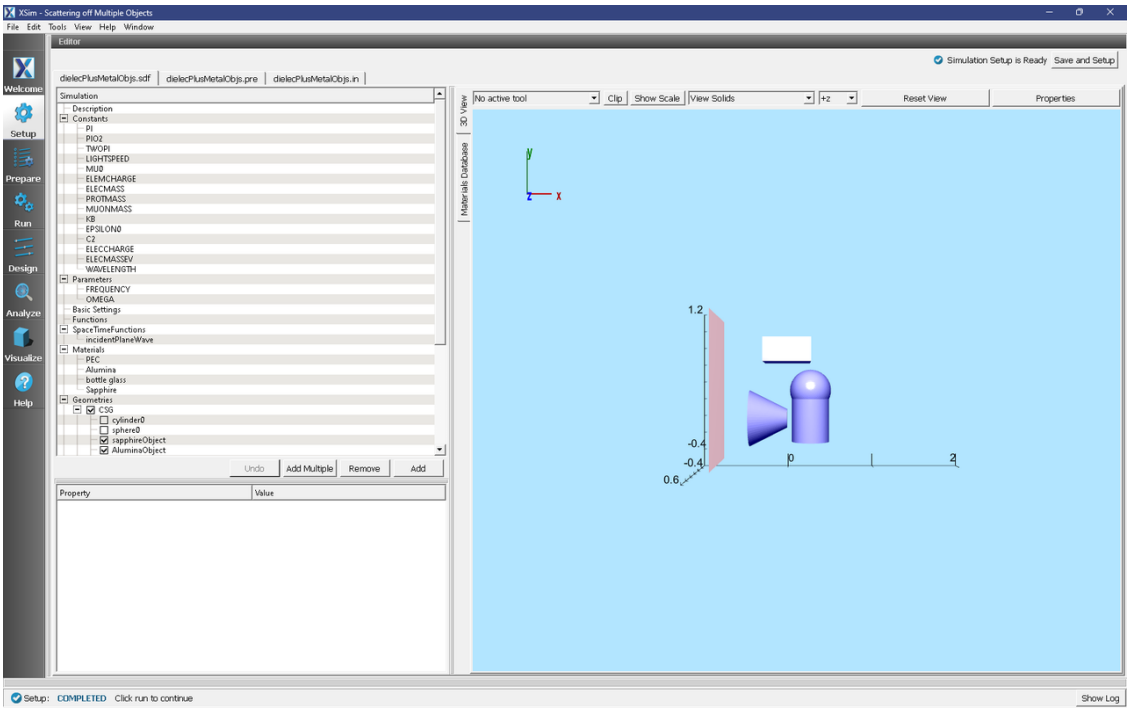

Fig. 2.138: Setup Window for the Scattering off Multiple Objects example.

Simulation Properties

This simulation includes just one user defined constant, WAVELENGTH, and just two user defined parameters, FREQUENCY and OMEGA. These three terms will define the incoming wave which is defined in the SpaceTimeFunctions element.

CSG shapes are used to define the geometries of the simulation. A sphere is unioned with a cylinder and given a material of sapphire. The box is an alumina structure and the truncated cone serves as a perfect electric conductor.

Placing all of these shapes and various materials in the same simulation shows how the electromagnetic wave can scatter off of different materials.

Running the Simulation

After performing the above actions, continue as follows:

- Proceed to the run window by pressing the Run button in the left column of buttons.
- Check that you are using these run parameters:
  - *Time Step*: 2.473991708760349e-11
  - *Number of Steps*: 200
  - *Dump Periodicity*: 20
  - *Dump at Time Zero*: Checked
- Click on the *Run* button in the upper left corner of the right pane.

You will see the output of the run in the right pane. The run has completed when you see the output, "Engine completed successfully". This result is shown in Fig. 2.139.

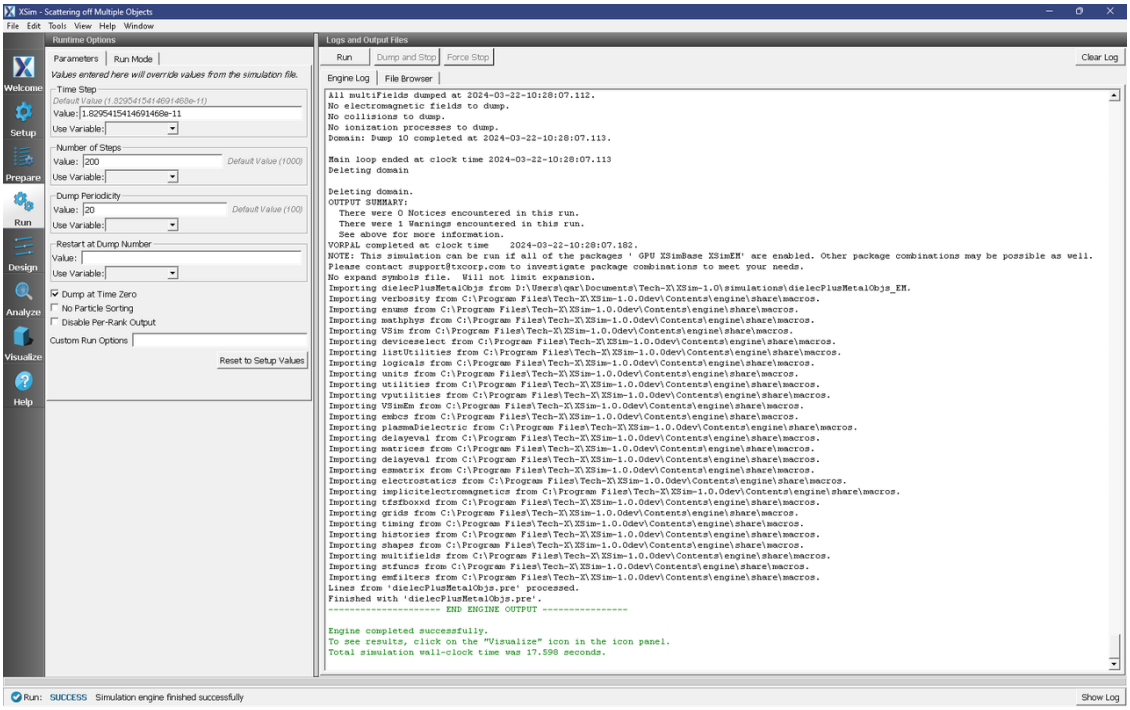

Fig. 2.139: The Run Window at the end of execution.

Visualizing the Results

After performing the above actions, continue as follows:

- Proceed to the Visualize Window by pressing the Visualize button in the left column of buttons.

To view the electric field, as well as the geometries as shown in Fig. 2.140, do the following:

- Expand *Scalar Data*
- Expand *E*
- Select *E_z*
- Check *Clip Plot*
- Set the color limits to -2 and 2
- Expand *Geometries*
- Select and check *Clip Plot* for: *AluminaObjectTriangles*, *PECObjectTriangles*, *sapphireObjectTriangles*

Initially, no field will be seen, as one is looking at Dump 1 when no fields are yet in the simulation. Move the slider at the bottom of the right pane to see the electric field at different times.

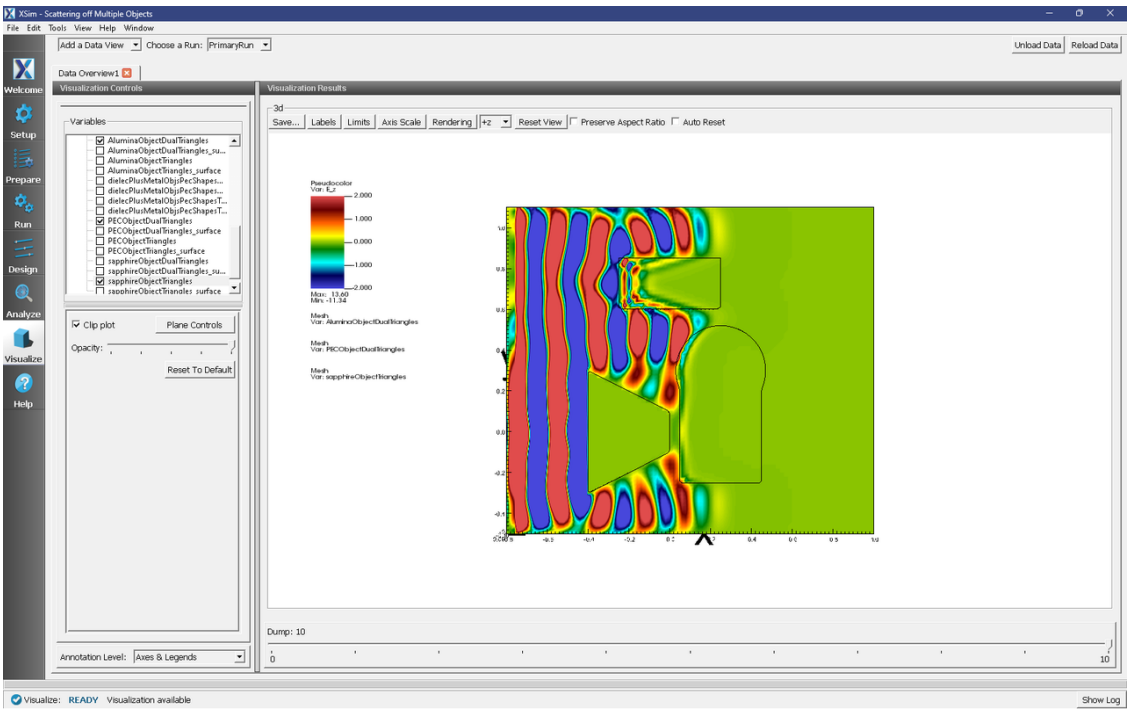

Fig. 2.140: Visualization of the wave as it hits the objects.

Further Experiments

One idea is to include radar cross section histories at setup time to be able to visualize the far fields.

This example is easily modifiable to include a different geometry and wave form.

Try changing the materials to see how it affects the wave.

INDEX

F

H

I

M

P

R

S

T

W