VSim Examples

Release 10.1.0-r2780

Tech-X Corporation

Mar 12, 2020

CONTENTS

CHAPTER

ONE

OVERVIEW

These are examples for illustrating the capabilities of VSim.

VSim *[VSi]* is an arbitrary dimensional, electromagnetics and plasma simulation code consisting of two major components:

- VSimComposer, the graphical user interface.
- Vorpal *[NC04]*, the VSim Computational Engine.

VSim also includes many more items such as Python, MPI, data analyzers, and a set of input simplifying macros.

CHAPTER

TWO

VSIM FOR BASIC PHYSICS EXAMPLES

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

These examples can be run with any license.

2.1 Basic Examples

2.1.1 Cylindrical Capacitor (cylindricalCapacitor.sdf)

Keywords:

`cylindrical, capacitor, electrostatic`

Problem description

The Cylindrical Capacitor simulation solves for the potential between two cylinders with a ring of charge.

This simulation can be performed with a VSimBase license.

Opening the Simulation

The Cylindrical Capacitor example is accessed from within VSimComposer by the following actions:

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim for Basic Physics* option.
- Expand the *Basic Examples* option.
- Select *Cylindrical Capacitor* 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 Grid element and select or deselect the box next to `Grid`.

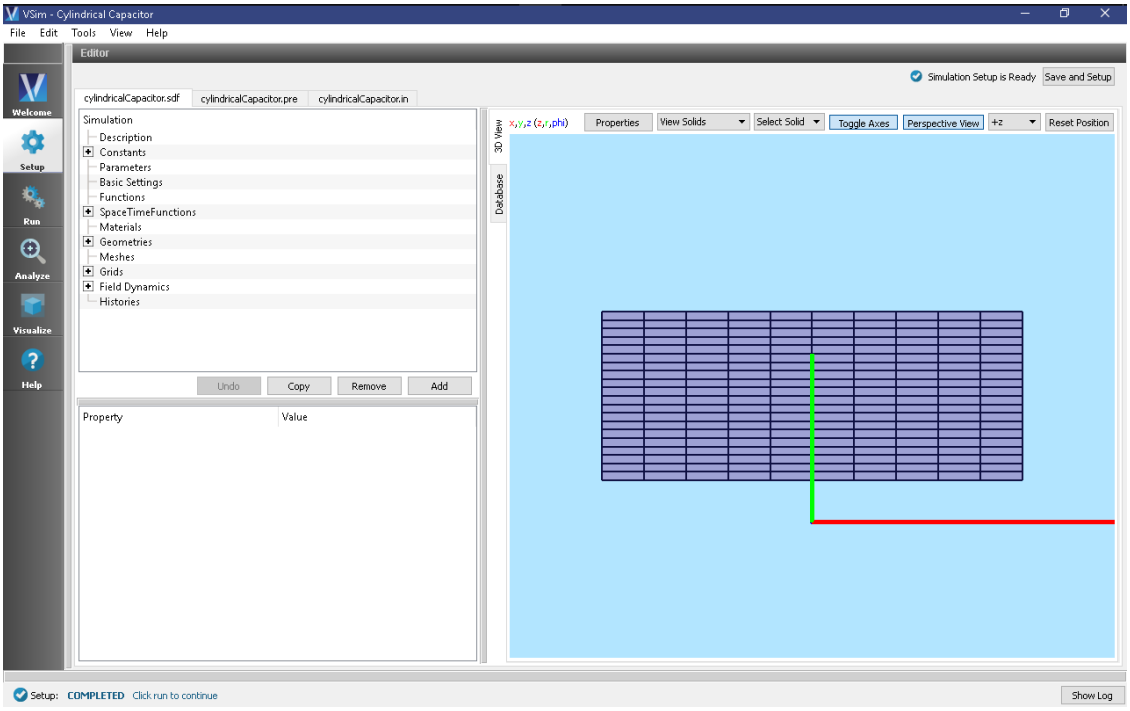


Fig. 2.1: Setup Window for the Cylindrical Capacitor example.

Simulation Properties

In this simulation there is a backgroundChargeDensity0 field which is given by an expression, chargedDist. That expression is defined as a SpaceTimeFunction. The variables x and y in the expression are place holders for the actual variables, Z and R, in the simulation. So this is a ring of charge, centered at R = 0.3, with a Gaussian fall off.

There are Dirichlet boundary conditions on the lower and upper R boundaries, with the lower bound set to 10 volts.

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.
- Here you can set run parameters, including how many cores to run with (under the Parallel Run Options).
- When you are finished setting run parameters, click on the *Run* button in the upper left corner of the window. 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 is shown in Fig. 2.2.

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.

This particular run computes the electrostatic potential, which we see by opening the scalar data and checking the box next to Phi, which is shown in the right of the visualization tab. See Fig. 2.3.

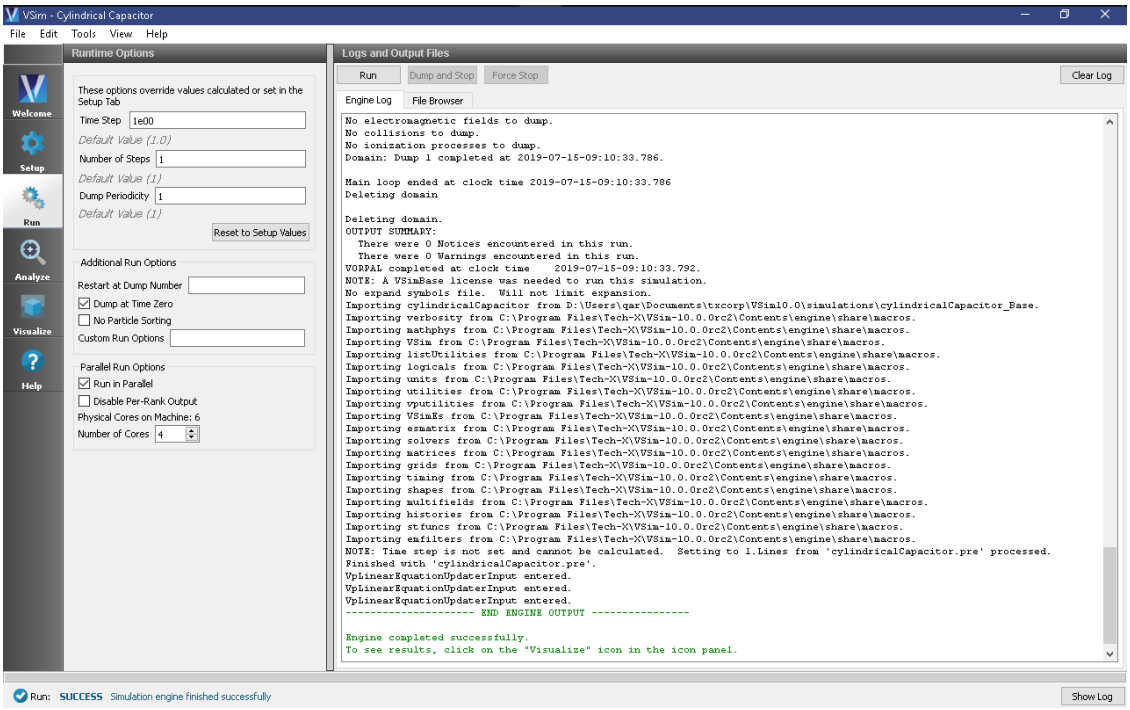


Fig. 2.2: The Run Window at the end of execution.

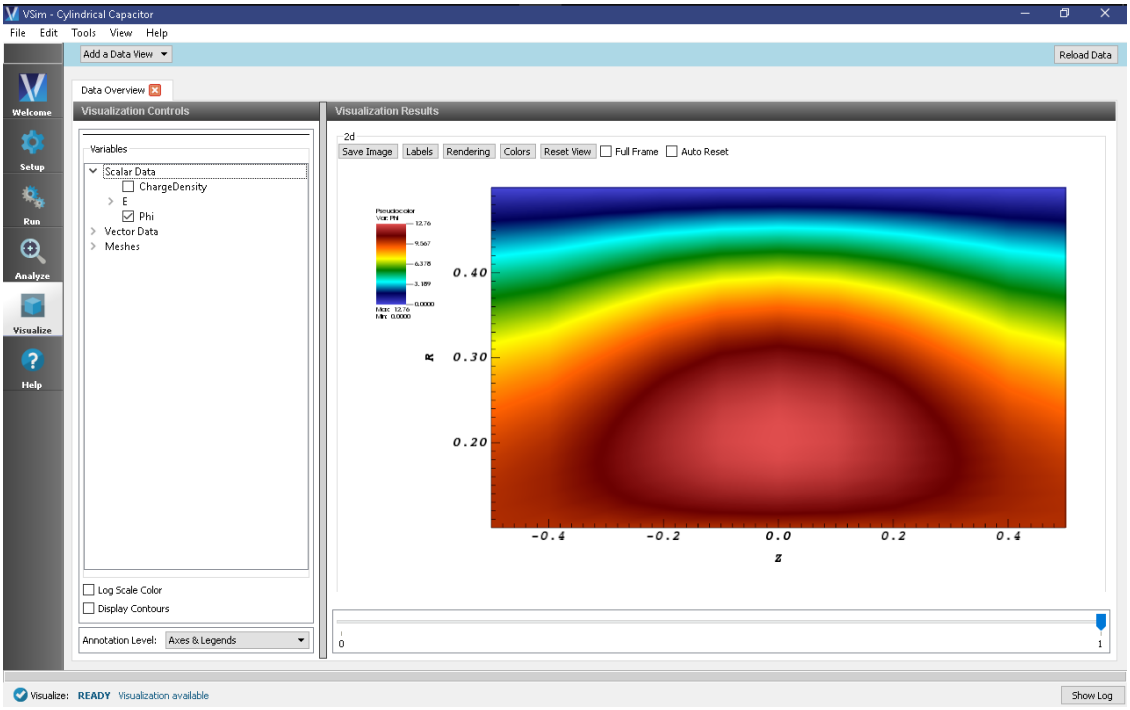


Fig. 2.3: Visualization of the electrostatic potential

Further Experiments

Looking inside the field boundary conditions, and highlighting dirichlet0, you can see that 10V was put on the lower R boundary. You can try experimenting with this, going to run and visualize with each change. The more voltage, the less the background charge should matter.

You can take the charge out of the system. Highlight the backgroundChargeDensity0 label. In the property editor below, double click on chargedDist, hit delete, and type 0.0. Then run and viz, and you will see a potential that is independent of Z.

Click the *Add a Data View* dropdown below the menu bar and select Field Analysis. For the Field, choose E_r with the Vertical Lineout Settings, then hit "Perform Lineout". You will see, as expected, that the radial electric field is positive (pointing outward) and falling off with the expected 1/r behavior.

2.1.2 Oscillating Dipole Above Conducting Plane (emOscDipoleAboveConductor.sdf)

Keywords:

`emOscDipoleAboveConductor, radiation`

Problem Description

This problem consists of an infinitesimally short dipole located a variable height and orientation above a conducting plane. This simulation consists of electric and magnetic fields that can be visualized to see how the distance between, and orientation of the dipole relative to the antenna effects these fields.

This simulation can be performed with a VSimBase license.

Opening the Simulation

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

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim for Basic Physics* option.
- Expand the *Basic Examples* option.
- Select *Oscillating 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. 2.4. 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`.

Simulation Properties

This example includes several constants for easy adjustment of simulation properties, Including:

- AMPLITUDE: The amplitude of the dipole current
- FREQUENCY: The operating frequency

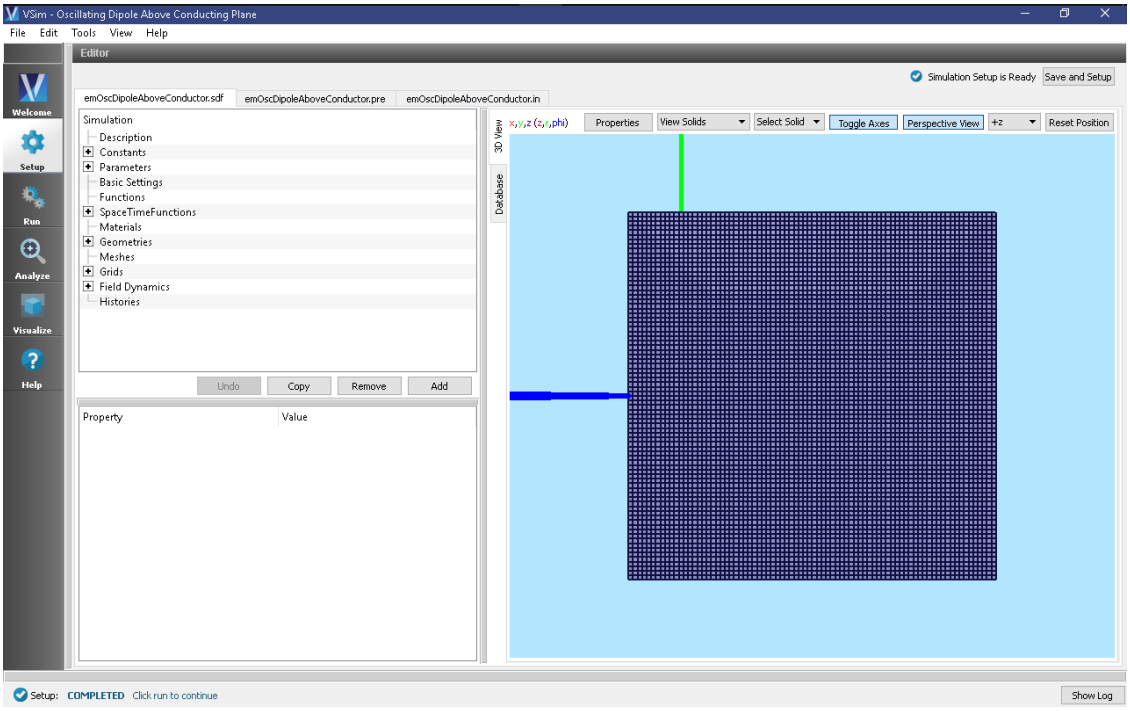


Fig. 2.4: Setup Window for the Dipole Above Conducting Plane example.

There is also a SpaceTimeFunction to define the current driver of the dipole source

Other properties of the simulation include open boundaries on all sides except for the lower x boundary, which is a perfect electric conductor. A *Dipole Current* source is used to set the location of the dipole source.

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.
- 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 the right pane. The run has completed when you see the output, "Engine completed successfully." This is shown in Fig. 2.5.

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 electric and magnetic field components can be found in the scalar data variables of the data overview tab. To create the plot shown in Fig. 2.6 do the following:

- Expand *Scalar Data*
- Expand *E*
- Select *E_y*
- Select the box next to *Display Contours* and set the # of contours to 12

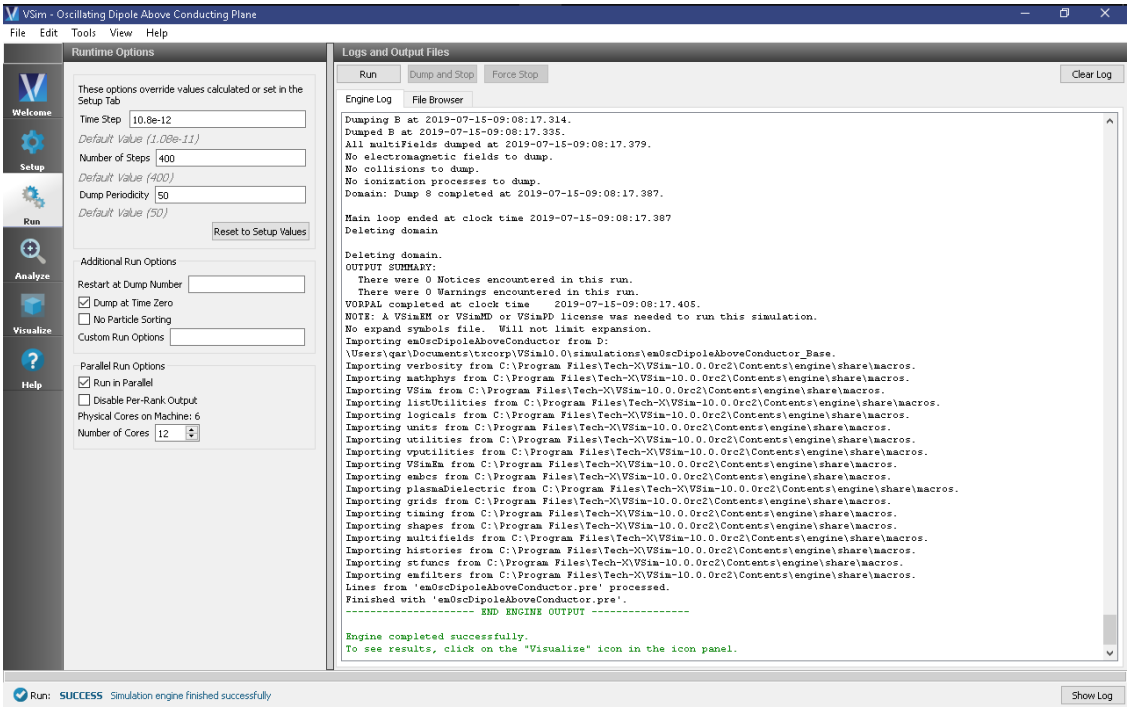


Fig. 2.5: The Run Window at the end of execution.

- Select the box next to *Clip All Plots*
- Drag the slider at the bottom of the *Visualization Results* pane to the right to show the 8th data dump
- If desired, rotate the plot by clicking and dragging your mouse

Further Experiments

In this example the "infinite" electric conductor is simulated by a physical conducting boundary at the bottom of the simulation. It would be possible to achieve the same results by having a second, equal infinitesimal dipole placed the same height "below" the conducting plane.

The number of "lobes" visible 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. This can be a bit difficult to visualize using just E-field data as it must be properly thresholded. The lobes will be easier to see in the example Advanced Dipole Above Conductor, a part of the VSimEM package.

2.1.3 Electromagnetic Plane Wave (emPlaneWave.sdf)

Keywords:

`electromagnetics, plane wave, periodic boundary conditions, wave launcher`

Problem Description

A linearly-polarized (with electric field in the z-direction) electromagnetic pulse with a sinusoidal amplitude on a plane wave is launched from the left side (x=0) to propagate in the x-direction. The transverse (y,z) boundary conditions are periodic.

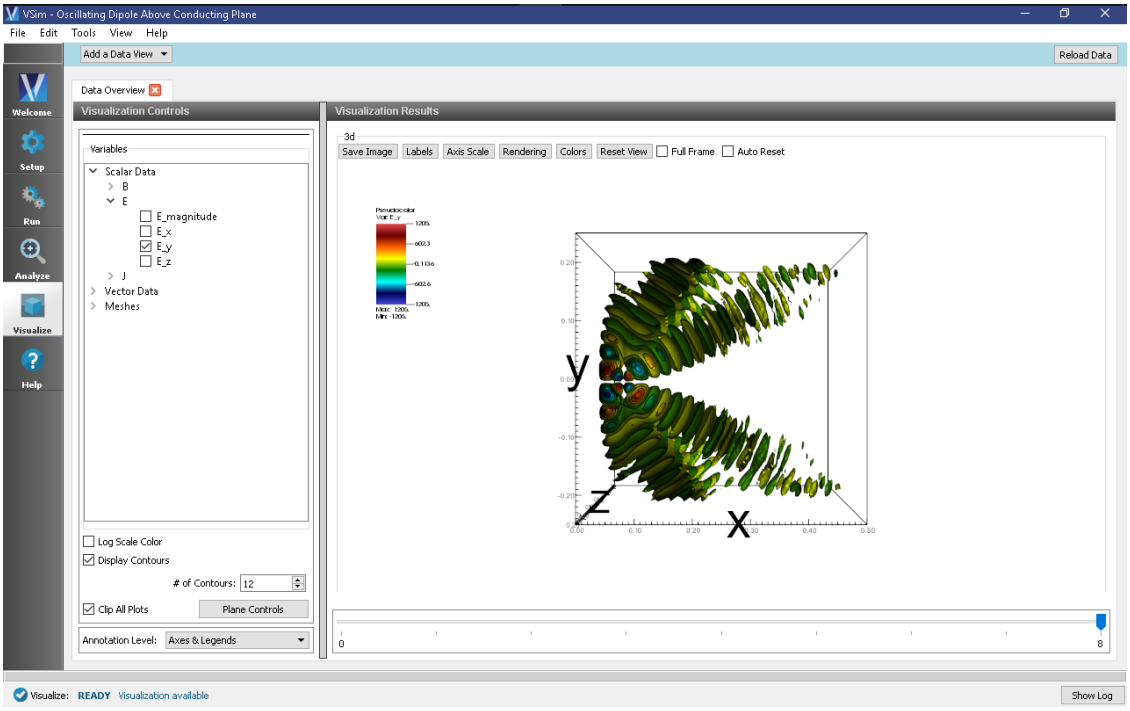


Fig. 2.6: The electric field

This simulation can be performed with a VSimBase license.

Opening the Simulation

The Electromagnetic Plane Wave example is accessed from within VSimComposer by the following actions:

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim for Basic Physics* option.
- Expand the *Basic Examples* option.
- Select *Electromagnetic Plane Wave* 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. 2.7. 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`.

Simulation Properties

This example includes several constants for easy adjustment of simulation properties. Those include:

- AMPLITUDE: The amplitide of the plane wave
- WAVELENGTHS: The number of wavelengths inside the domain

There is a SpaceTimeFunction to define the plane wave that is launched with a *Port Launcher* boundary condition.

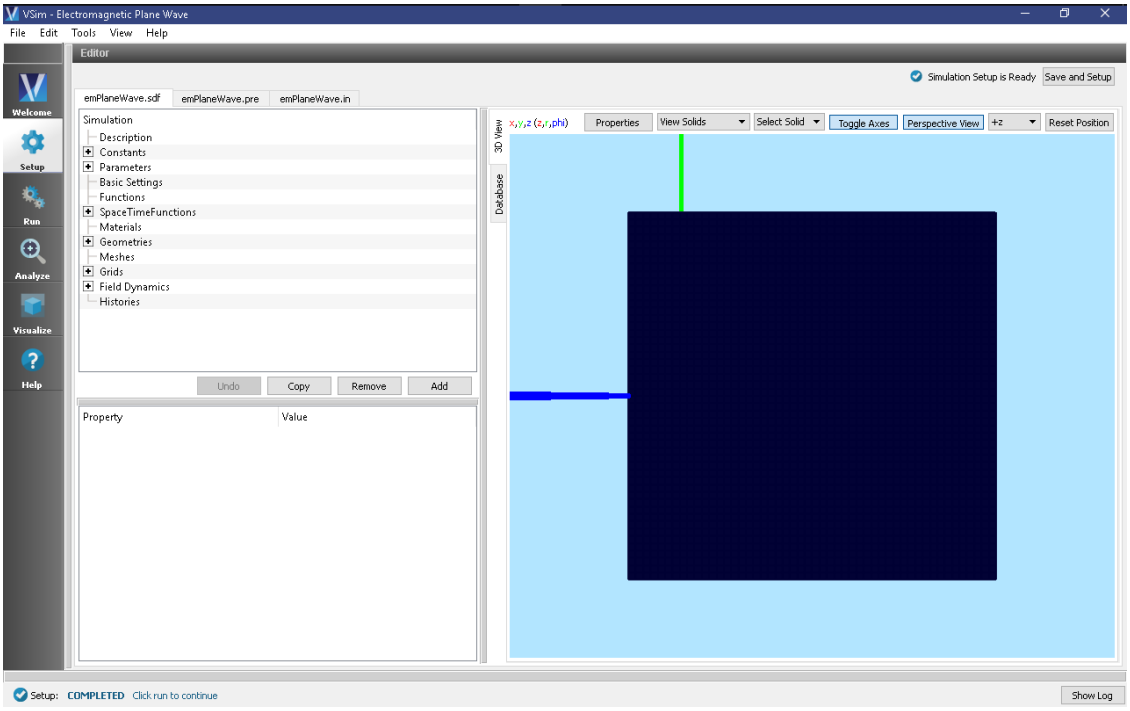


Fig. 2.7: Setup Window for the Electromagnetic Plane Wave example.

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.
- 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 the right pane. The run has completed when you see the output, "Engine completed successfully." This is shown in Fig. 2.8.

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 electric and magnetic field components can be found in the scalar data variables of the data overview tab.

- The Data Overview tab should be active. If it is not, click the *Add a Data

View* dropdown below the toolbar and select *Data Overview*. * Here you can see *Variables*. Expand the *Scalar Data*. * Expand E * Select E_z

Initially, no field will be seen, as one is looking at Dump 0, the initial dump, 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. The final time is shown in Fig. 2.9.

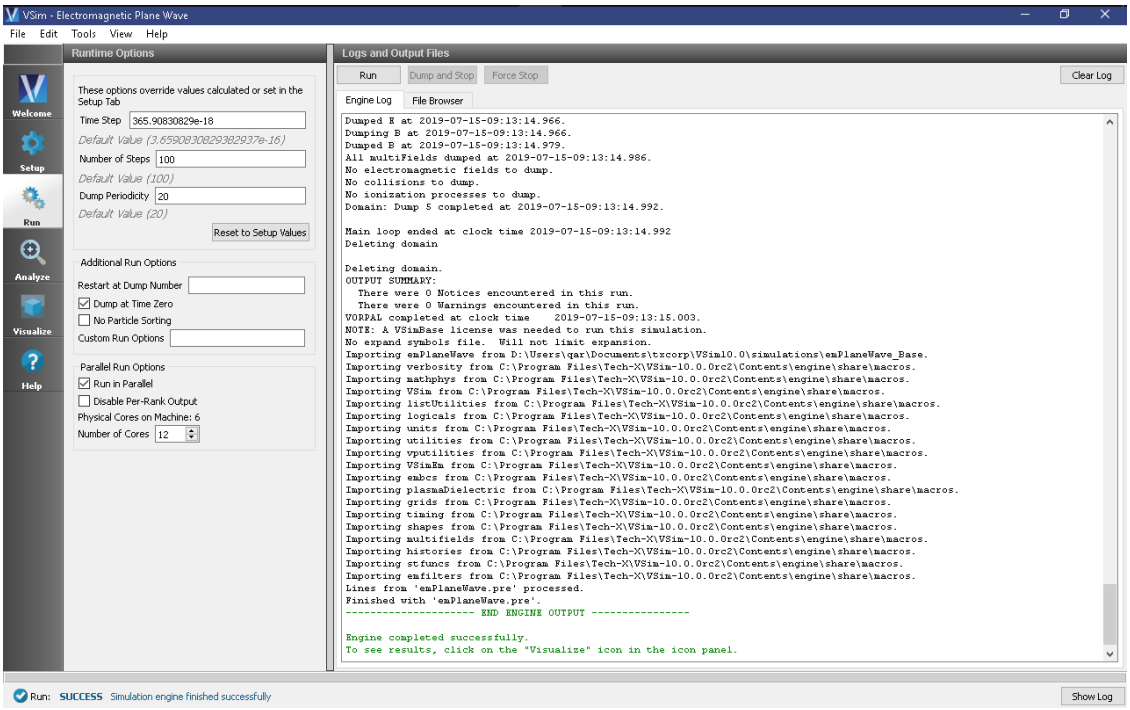


Fig. 2.8: The Run Window at the end of execution.

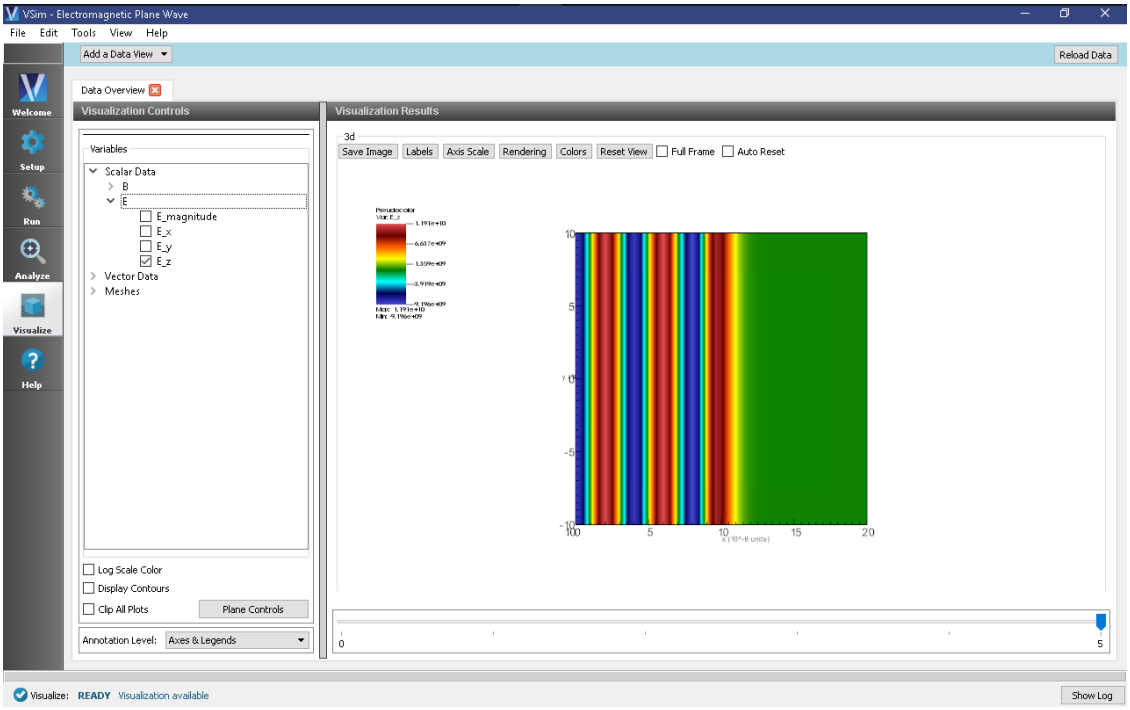


Fig. 2.9: Visualization of plane wave as a color contour plot.

Further Experiments

To see more wavelengths, change the value of the WAVELENGTHS variable. What happens to the waves when there are very few cells in a wavelength?

See the wave reflect off the right boundary by running for more time steps.

Try rotating the visualization by left-clicking and dragging with the mouse to see the how the simulation is uniform across the z- dimension.

2.1.4 Electromagnetic Particle In Cell (emPtclInCell.sdf)

Keywords:

`electromagnetics, particle in cell, sheath`

Problem description

A dipole antenna launches a wave from a point that is midway in x and y. The simulation is periodic in y and open in x. The electromagnetic field and plasma respond self consistently to the antenna current.

This simulation can be performed with a VSimBase license.

Opening the Simulation

The Electromagnetic Particle In Cell example is accessed from within VSimComposer by the following actions:

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim for Basic Physics* option.
- Expand the *Basic Examples* option.
- Select "Electromagnetic Particle In Cell" 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. 2.10. 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`.

Simulation Properties

This simulation includes several constants for easy adjustment of simulation properties including:

- N_X, N_Y: The number of cells in each direction
- LEN_X, LEN_Y: The length of the domain in each direction
- PPC: The number of macroparticles per cell
- FREQUENCY: The frequency of the dipole antenna

The *Parameters* element contains several parameters useful for calculating basic plasma physics properties such as the plasma frequency and Debye length.

There are 2 SpaceTimeFunctions that are used later in the setup to describe the thermal velocity of the electrons and the antenna current profile.

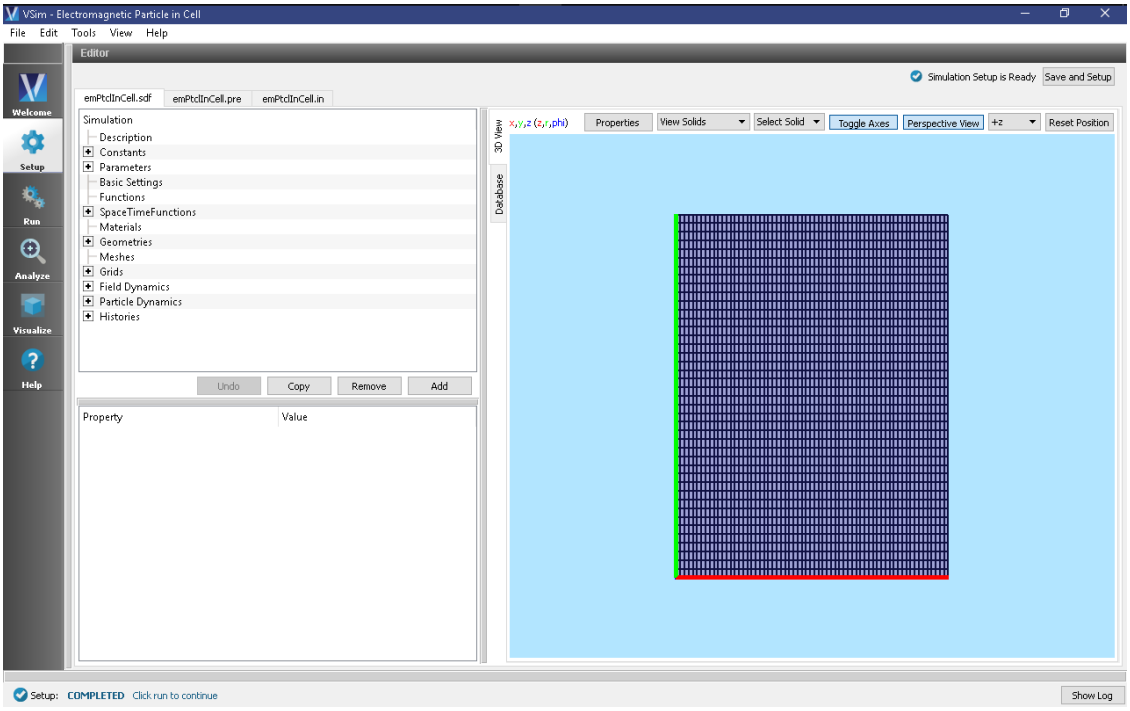


Fig. 2.10: Setup Window for Electromagnetic Particle in Cell.

The simulation has open boundary conditions in x, and periodic in y.

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.
- 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 the right pane. The run has completed when you see the output, "Engine completed successfully." This is shown in Fig. 2.11.

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 and particles as shown in Fig. 2.12, do the following:

- Expand *Particle Data*
- Expand *electrons0*
- Select *electrons0*
- Expand *Scalar Data*
- Expand *E*
- Select *E_z*

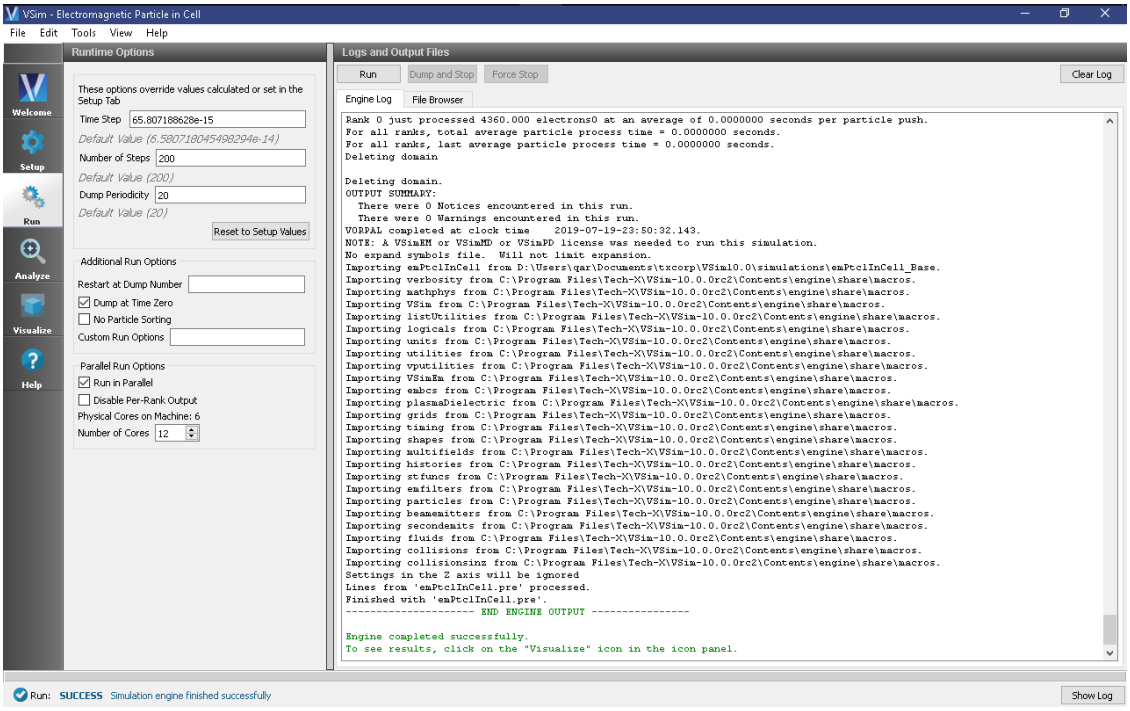


Fig. 2.11: The Run Window at the end of execution of Electromagnetic Particle in Cell.

Initially the field is at zero and particles are evenly distributed throughout the simulation. Move the dump slider forward in time to view the results.

Further Experiments

Vary the antenna amplitude, to find out how low it can be before the signal is swamped by the plasma noise.

Add in a magnetic field in the plane.

2.1.5 Vacuum Electromagnetic Pulse (emPulseInVacuum.sdf)

Keywords:

`electromagnetics, laser, plane wave pulse, field energy monitoring`

Problem description

A linearly-polarized (with electric field in the z-direction) electromagnetic pulse with a sinusoidal amplitude on a plane wave is launched from the left side (x=0). The transverse (y, z) boundary conditions are periodic, but the pulse has finite transverse extent.

This simulation can be performed with a VSimBase license.

Opening the Simulation

The Vacuum Electromagnetic Pulse example is accessed from within VSimComposer by the following actions:

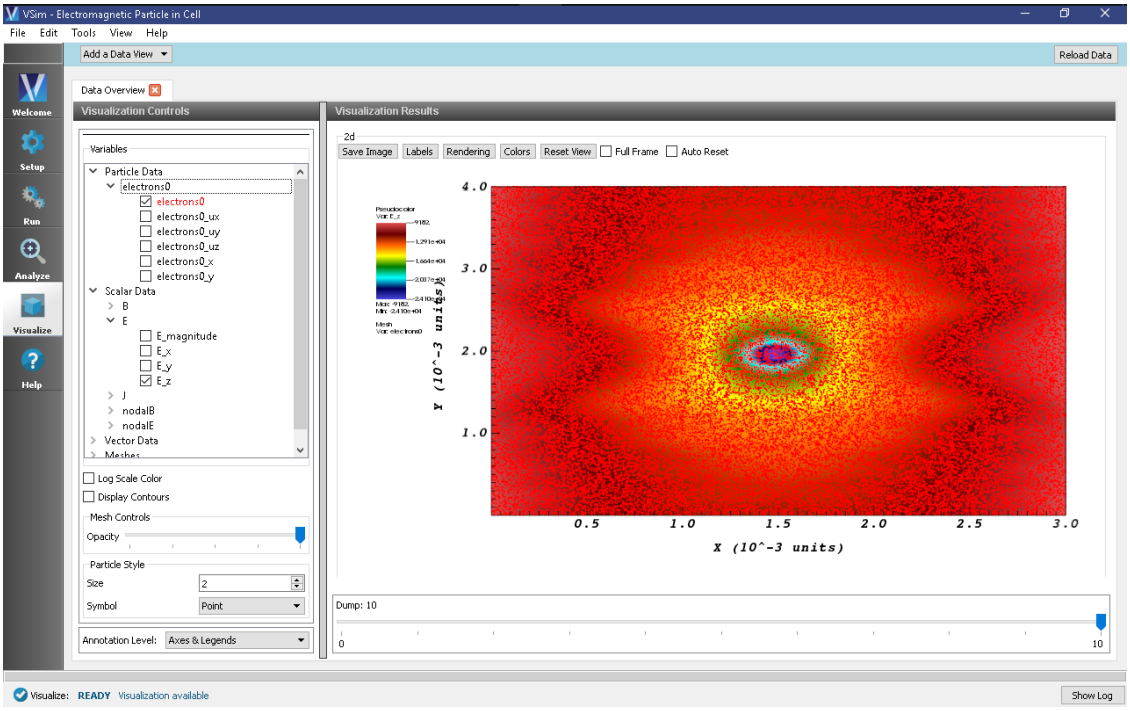


Fig. 2.12: Visualize Window with electric field and particles

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim for Basic Physics* option.
- Expand the *Basic Examples* option.
- Select *Vacuum Electromagnetic Pulse* 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. 2.13. 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`.

Simulation Properties

The Vacuum Electromagnetic Pulse example includes several constants for easy adjustment of simulation properties. Those include:

- AMPLITUDE: The amplitude of the pulse
- WAVELENGTH: The wavelength of the pulse
- PULSELENGTH: The length of the pulse in the propagation direction
- PULSEWIDTH: The width of the pulse in the transverse direction

There is also a SpaceTimeFunction defined for the pulse shape and is used in the *Port Launcher* boundary condition.

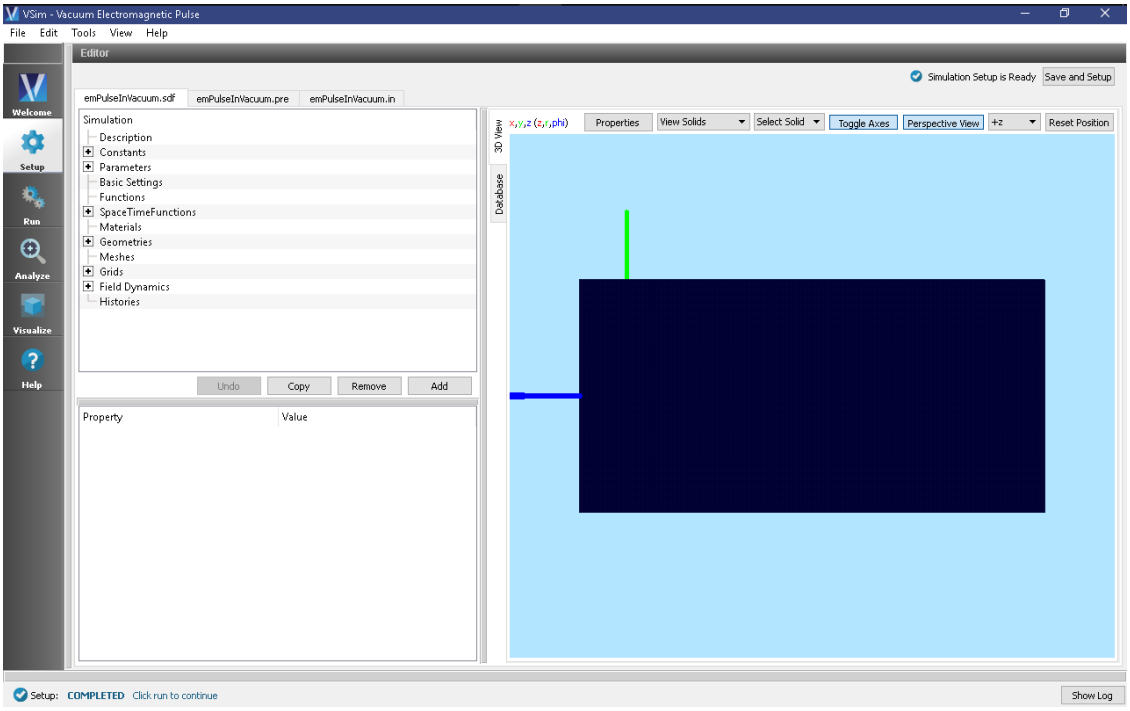


Fig. 2.13: The Setup Window for the electromagnetic pulse.

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

The Run Window, showing settable parameters with the engine output in the right pane, is shown in Fig. 2.14.

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 electric and magnetic field components can be found in the scalar data variables of the data overview tab.

- Make sure the *Data View* drop down is set to *Data Overview*.
- Here you can see *Variables*. Expand the *Scalar Data*.
- Expand E
- Select E_z
- Check the box next to *Clip All Plots*
- Check the box next to *Display Contours* and set the # of contours to 5
- Click and drag with your mouse to rotate the view

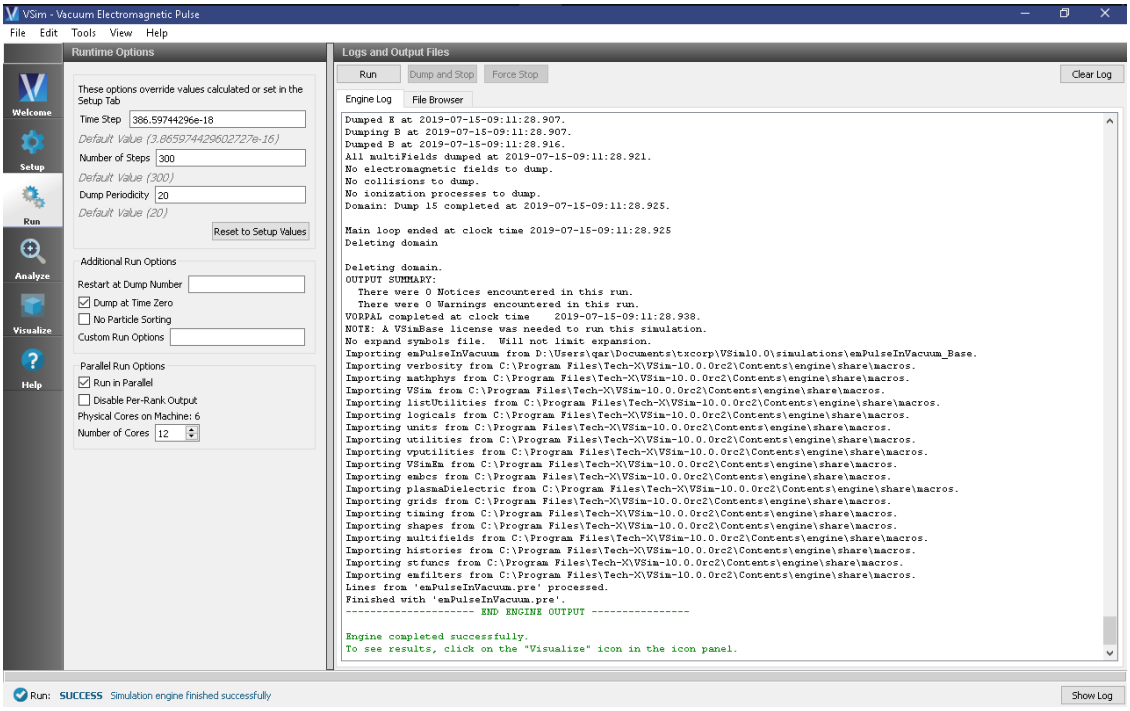


Fig. 2.14: The Run Window for the electromagnetic pulse.

Initially, no field will be seen, as one is looking at Dump 0, the initial dump, when no fields are yet in the simulation. Move the slider at the bottom of the right pane to see the magnetic field at different times.

Further Experiments

Increase NX to better resolve the wave and see whether it slips less with respect to the box.

Increase the pulse and box widths (you will also need to increase the number of cells in the transverse directions) to reduce diffraction.

2.1.6 Electrostatic Particle In Cell (esPtclInCell.sdf)

Keywords:

`electrostatics, particle in cell, sheath`

Problem description

This Electrostatic Particle in Cell example computes the electrostatic potential and field in a box with conducting walls and particle absorbers and with an immobile, background neutralizing charge density. The electrons move to the wall by the potential, creating a sheath.

This simulation can be performed with a VSimBase license.

Opening the Simulation

The electrostatic particle in cell example is accessed from within VSimComposer by the following actions:

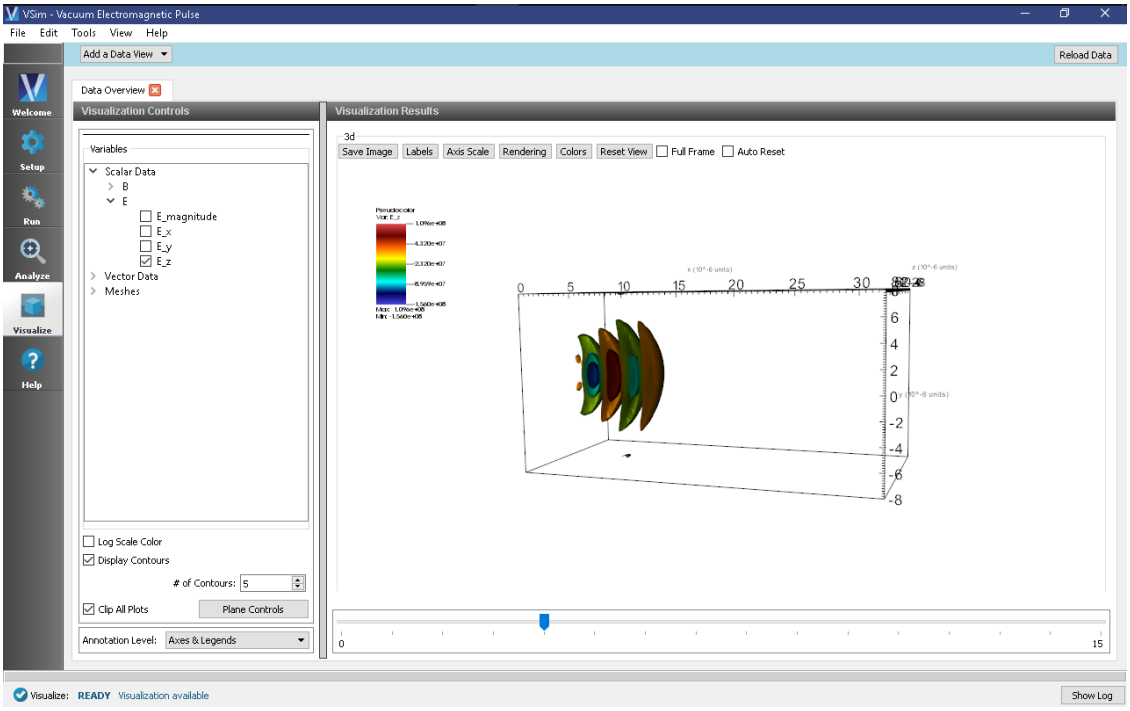


Fig. 2.15: Ez field at Dump 5.

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim for Basic Physics* option.
- Expand the *Basic Examples* option.
- Select *Electrostatic Particle in Cell* 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. 2.16. 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`.

Simulation Properties

This simulation includes several constants for easy adjustment of simulation properties including:

- N_X, N_Y: The number of cells in each direction
- W_X, W_Y: The length of the domain in each direction
- PPC: The number of macroparticles per cell

The Parameters element contains several parameters useful for calculating basic plasma physics properties such as the plasma frequency and Debye length.

There is a SpaceTimeFunction used later in the setup to describe the thermal velocity of the electrons.

The simulation is periodic in y with Dirichlet boundary conditions in x set to zero.

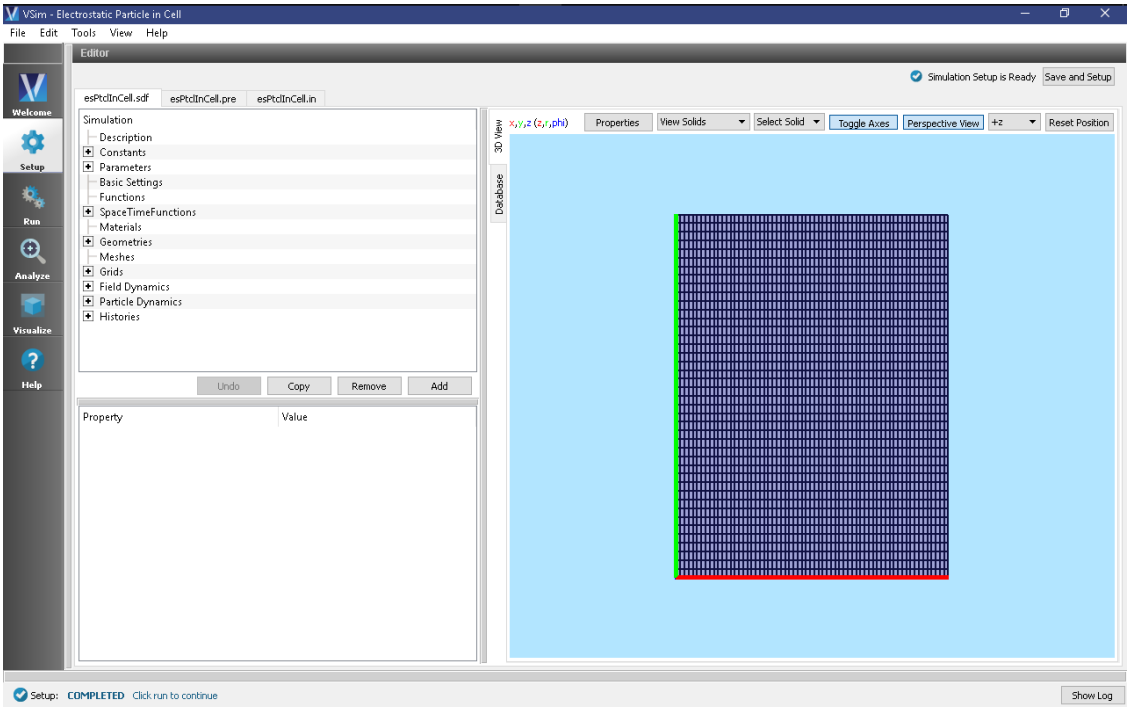


Fig. 2.16: Setup Window for the electrostatic particle in cell example.

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.
- 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 the right pane. The run has completed when you see the output, "Engine completed successfully." A snapshot of the simulation run completion is shown in Fig. 2.17.

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 potential as shown in Fig. 2.18, do the following:

- Expand *Scalar Data*
- Select *Phi*

Move the dump slider forward in time to see the evolution of the field.

Further Experiments

Change the plasma density and see whether the frequency in the histories changes.

Use the computePtclNumDensity analysis script in the *Analyze* Tab to calculate the electron density at each dump and view the sheath formation.

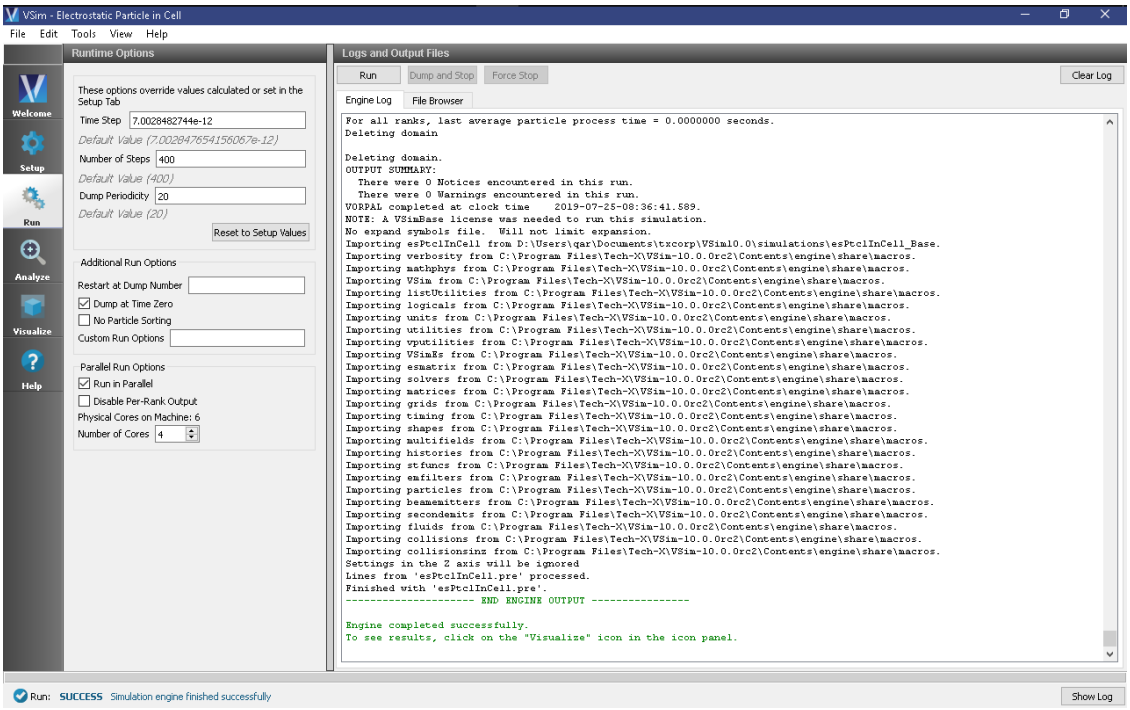


Fig. 2.17: The Run Window at the end of execution.

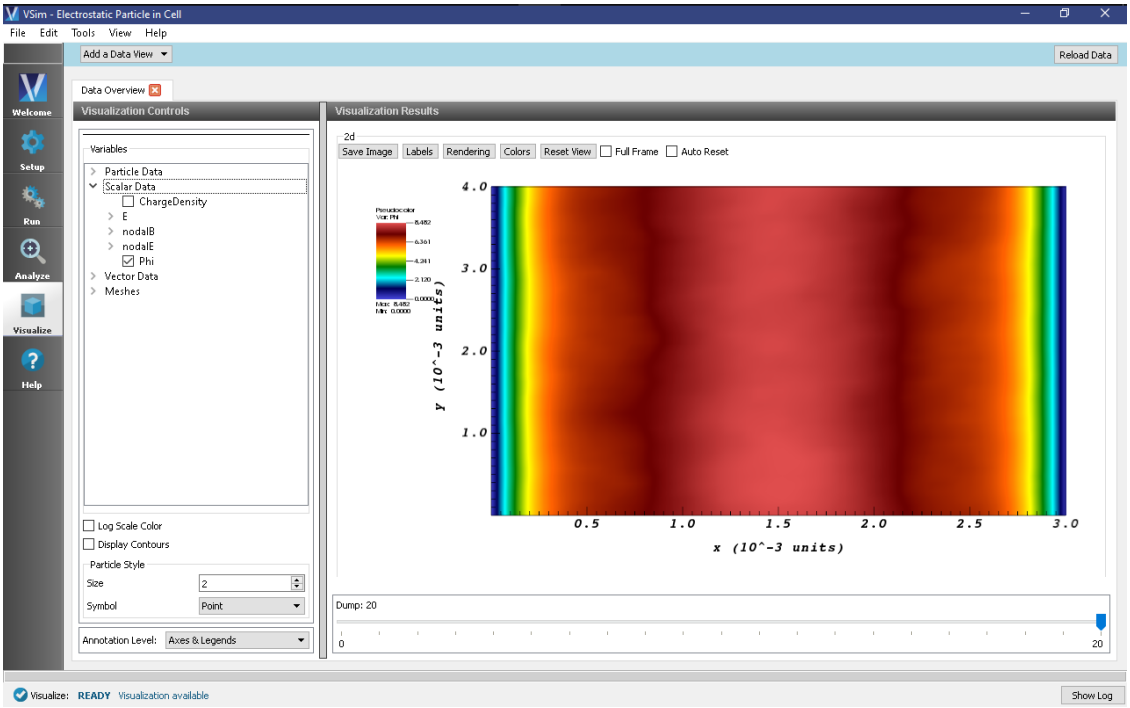


Fig. 2.18: The Visualize Window showing the electric potential, *Phi*, at dump 20.

2.1.7 Half-wave Antenna (halfWaveAntenna.sdf)

Keywords:

`electromagnetics, antennas`

Problem Description

The half wave antenna example describes a simple box source in a vacuum.

This simulation can be performed with any license.

Opening the Simulation

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

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim for Basic Physics* option.
- Expand the *Basic Examples* option.
- Select *Half Wave 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. 2.19. 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`.

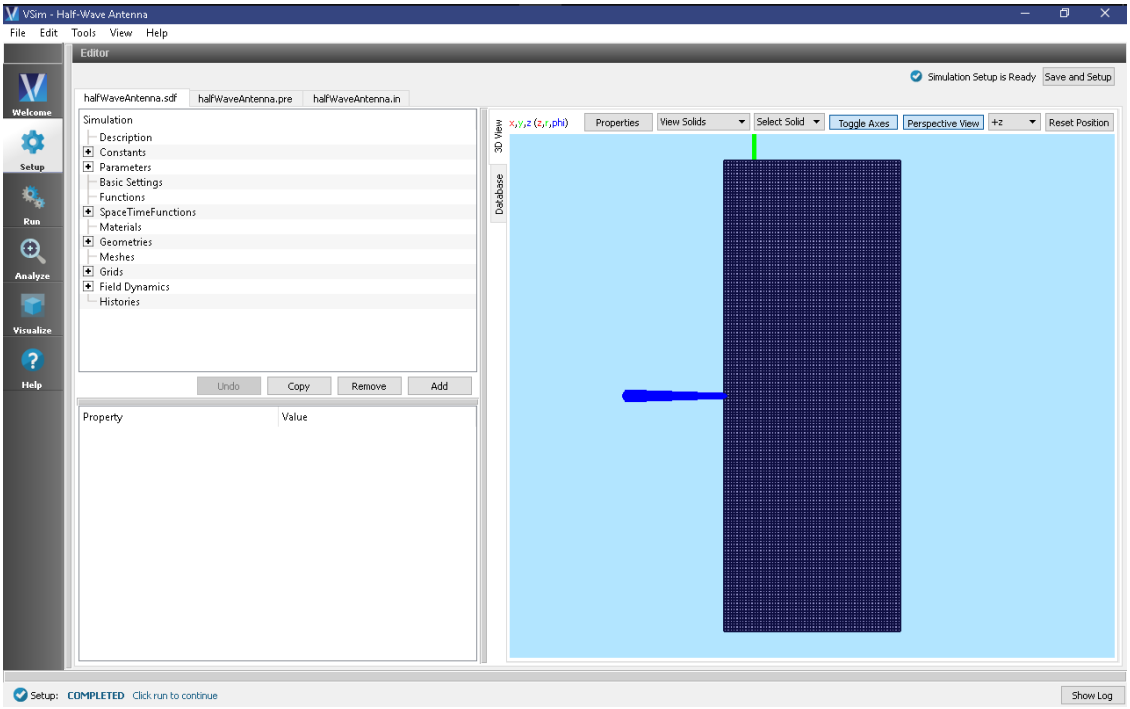


Fig. 2.19: Setup Window for the Half Wave Antenna example.

Simulation Properties

This example includes several constants for easy adjustment of simulation properties, Including:

- WAVELENGTH: The wavelength of the antenna

There is also a SpaceTimeFunction to define the current driver of the half wavelength source.

Other properties of the simulation include port boundaries on all sides except for the lower x boundary, which is a perfect electric conductor. A Distributed Current source is used to set the current of the half wavelength source.

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.
- 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 the right pane. The run has completed when you see the output, "Engine completed successfully." This is shown in Fig. 2.20.

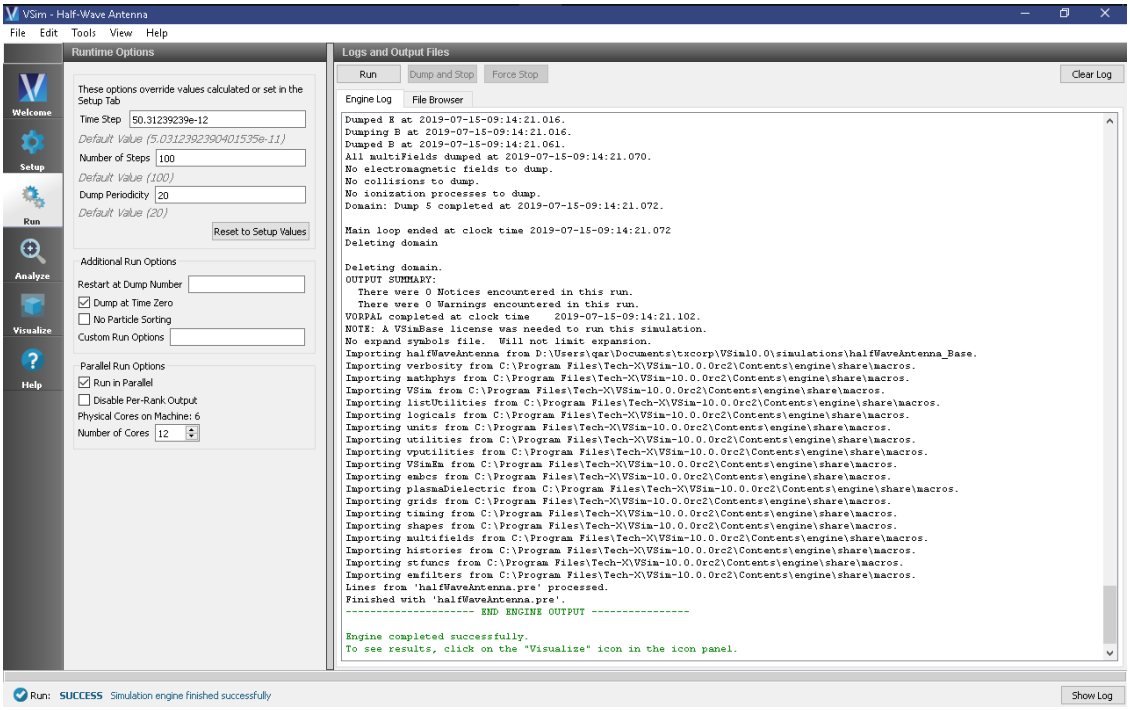


Fig. 2.20: The Run Window at the end of execution.

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 field pattern as shown in Fig. 2.21, do the following:

- Expand *Scalar Data*
- Expand *B*

- Select *B_y*
- Select *Display Contours* and set the # of Contours to 100
- Move the dump slider forward in time to the last dump
- Rotate the plot by clicking and dragging with your mouse

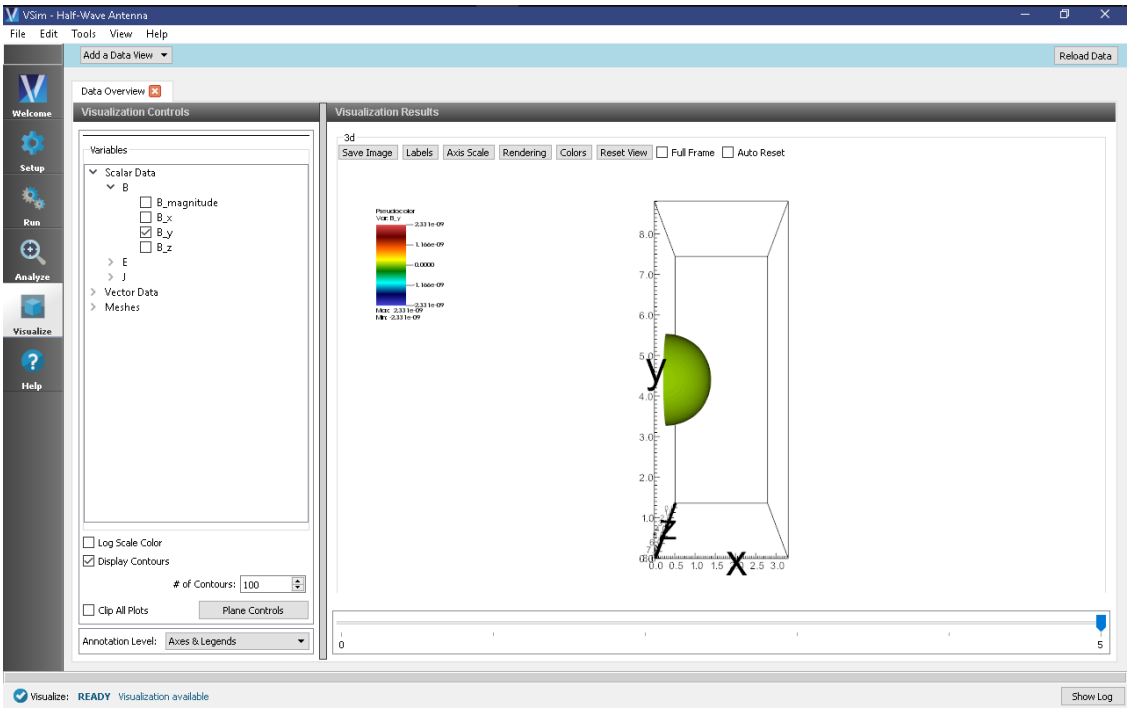


Fig. 2.21: Visualization of the wave pattern as a color contour plot.

Further Experiments

Additional experiments worth investigating are:

- Change the frequency of the source.

2.1.8 Parallel Plate Capacitor (parPlateCapacitor.sdf)

Keywords:

`electrostatics, parallel plate capacitor`

Problem description

This Parallel Plate Capacitor simulation computes the electrostatic potential and field for a parallel plate capacitor. It can be run in any number of dimensions. It is periodic in the y and z directions when they are present.

This simulation can be performed with a VSimBase license.

Opening the Simulation

The Parallel Plate Capacitor example is accessed from within VSimComposer by the following actions:

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim for Basic Physics* option.
- Expand the *Basic Examples* option.
- Select *Parallel Plate Capacitor* 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. 2.22. 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`

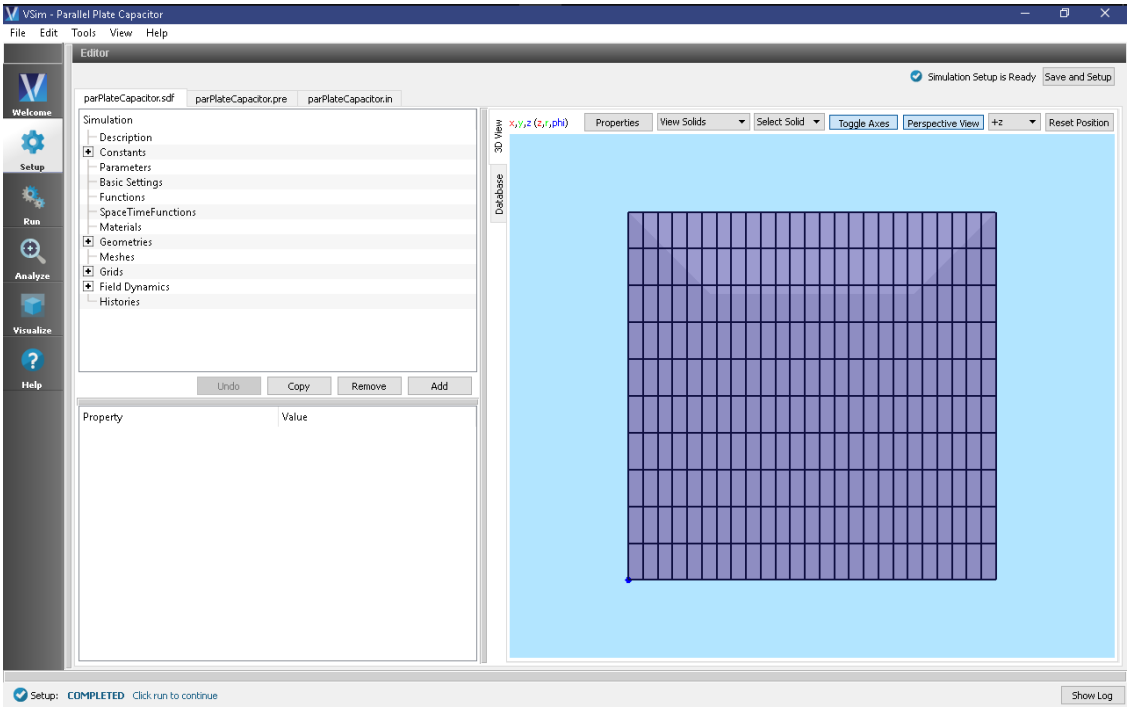


Fig. 2.22: Setup Window for the Parallel Plate Capacitor example.

Simulation Properties

The Simulation Elements Tree and Property Editor allow one to choose the distance between the plates, width of the plates, voltage of the positive plate and the length of a time step (which is irrelevant as this is an electrostatic 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.

- 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 the right pane. The run has completed when you see the output, "Engine completed successfully." This is shown in Fig. 2.23.

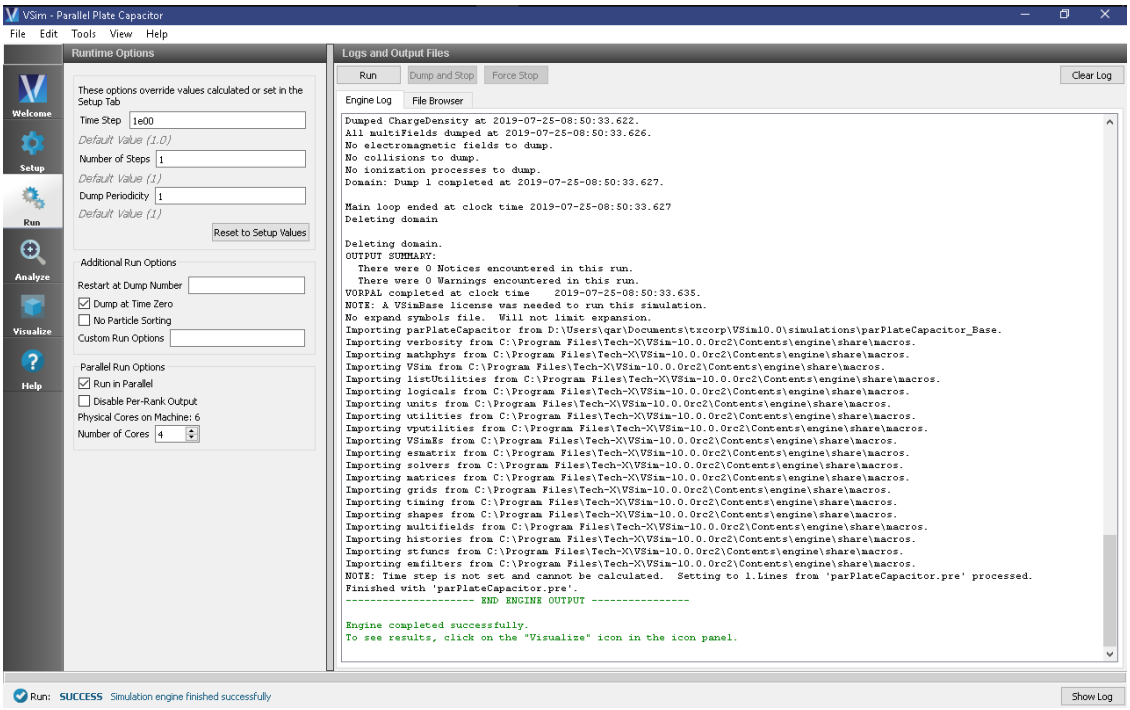


Fig. 2.23: The Run Window at the end of execution.

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 create the image shown in Fig. 2.24, proceed as follows:

- Expand *Scalar Data* and select the box next to *Phi*

Further Experiments

Change the gap between the plates by changing the length of the grid in the z direction and see how the electric field changes.

Change a width, e.g., LY and see whether it has an effect on the electric field.

Change the voltage on the right plate and see if it affects the electric field.

Set the periodic directions under *BasicSettings* to *none*, add boundary conditions on those directions and see how the field changes.

2.1.9 Two-Stream Instability (twoStream.sdf)

Keywords:

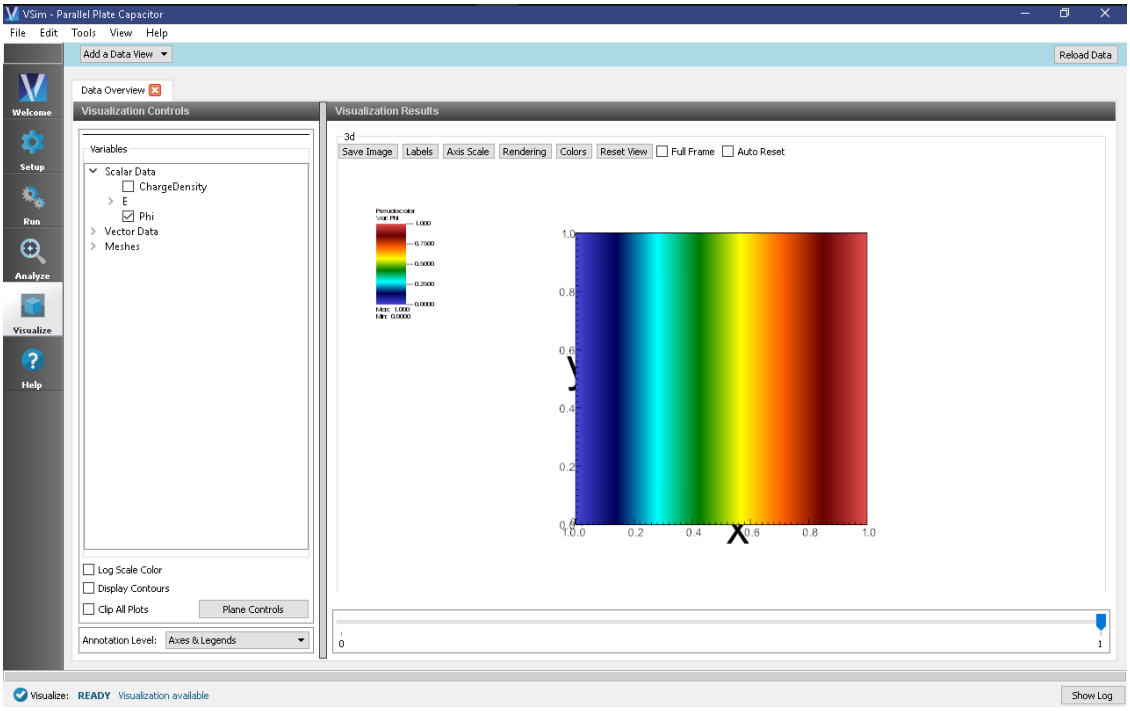


Fig. 2.24: Visualization of plane wave as a color contour plot.

`electromagnetics, two-stream instability`

Problem Description

The two-stream instability is a rapidly growing collision-less plasma instability arising from small charge imbalances. A local imbalance leads to the acceleration or deceleration of particles in its vicinity, which in turn leads to an even stronger imbalance. One setup that allows to easily observe the instability is two counter-streaming beams of identical charge in a periodic system. The advantage of this configuration is that the generated plasma wave becomes a standing wave, thus allowing to easily observe the formation of the phase space vortices.

In this example, we use two electron streams. At $t = 0$ the streams have drift velocities of magnitude 7.78×10^6 m/s. In order to accelerate the onset of the instability, the two particle beams are given a small sinusoidal perturbation in velocity space.

This simulation can be performed with a VSimBase license.

Opening the Simulation

The Two-Stream Instability example is accessed from within VSimComposer by the following actions:

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim for Basic Physics* option.
- Expand the *Basic Examples* option.
- Select *Two-Stream Instability* 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. 2.25. 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`.

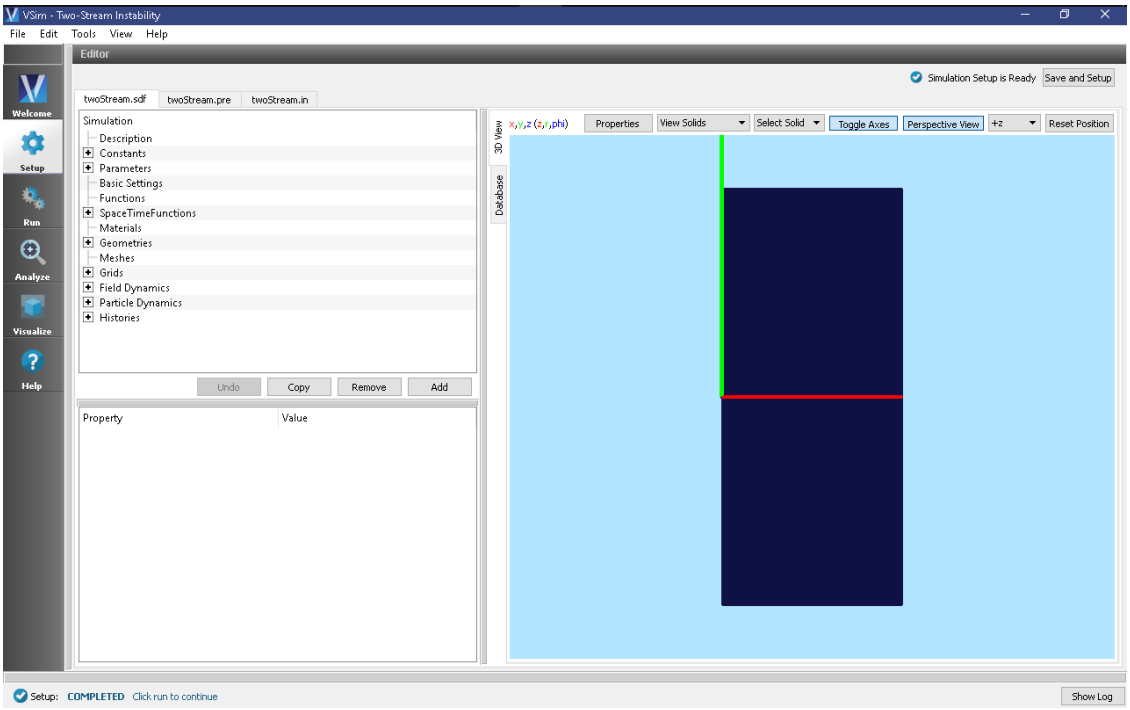


Fig. 2.25: Setup Window for the Two-Stream Instability example.

Simulation Properties

There are a number of *Constants* in this simulation to help make modifying the simulation even easier. Those include:

- XCELLS: The number of cells
- NOM_DEN_E: The electron density
- VBAR: The average velocity
- WAVELENGTHS: The number of wavelengths in the domain to simulate
- PPC: The number of particles per cell.

SpaceTimeFunctions are used to set the velocities of each particle stream.

The simulation is 1 dimensional and periodic in x.

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.
- To run the file, click on the *Run* button in the upper left corner of the window. 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 is shown in Fig. 2.26.

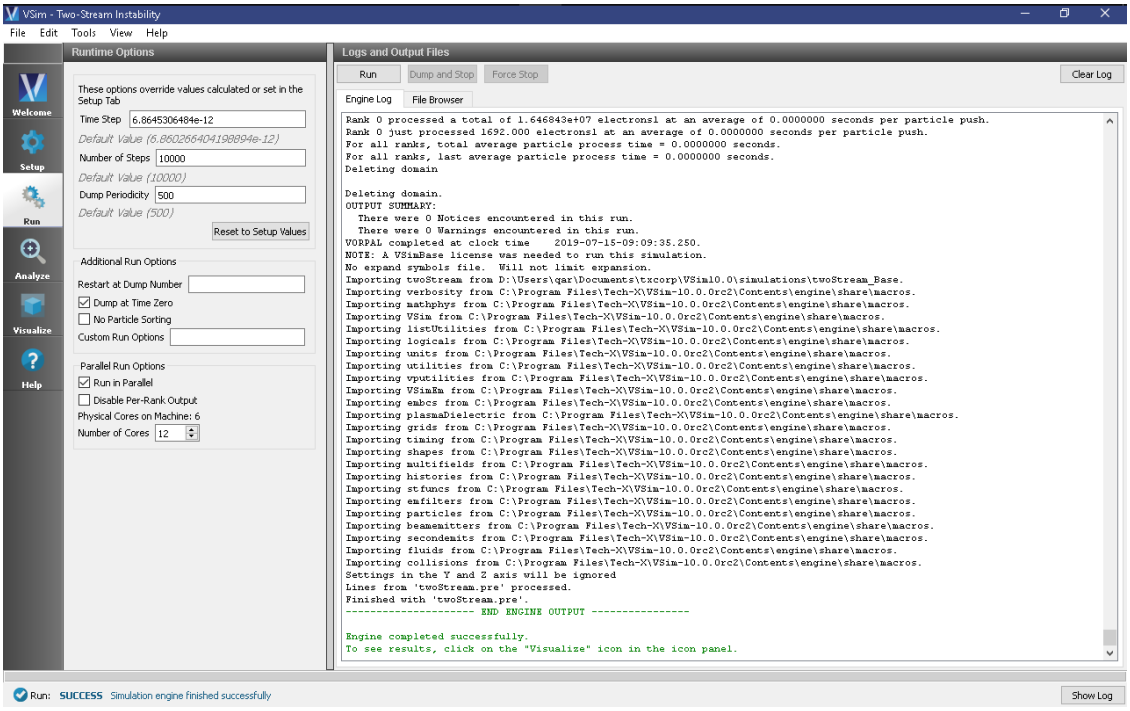


Fig. 2.26: The Run Window at the end of execution.

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 instability as shown in Fig. 2.27, do the following:

- Select the *Phase Space* option from the *Data View* menu
- In the Plot 1 box, change the *X-axis* to *electrons0_x*, and the *Y-axis* to *electrons0_ux*
- Click the *Enable Second Plot* box
- In the Plot 2 box, change the *Base Variable* to *electrons1*, the *X-axis* to *electrons1_x*, and the *Y-axis* to *electrons1_ux*
- Click the *DRAW* button at the bottom, then move the *Dump* slider forward in time.

Further Experiments

Change the average velocity and velocity modulation and see how the speed at which the instability sets in depends on the modulation.

View the particle density by using the *computePtclNumDensity* script in the Analyze Window.

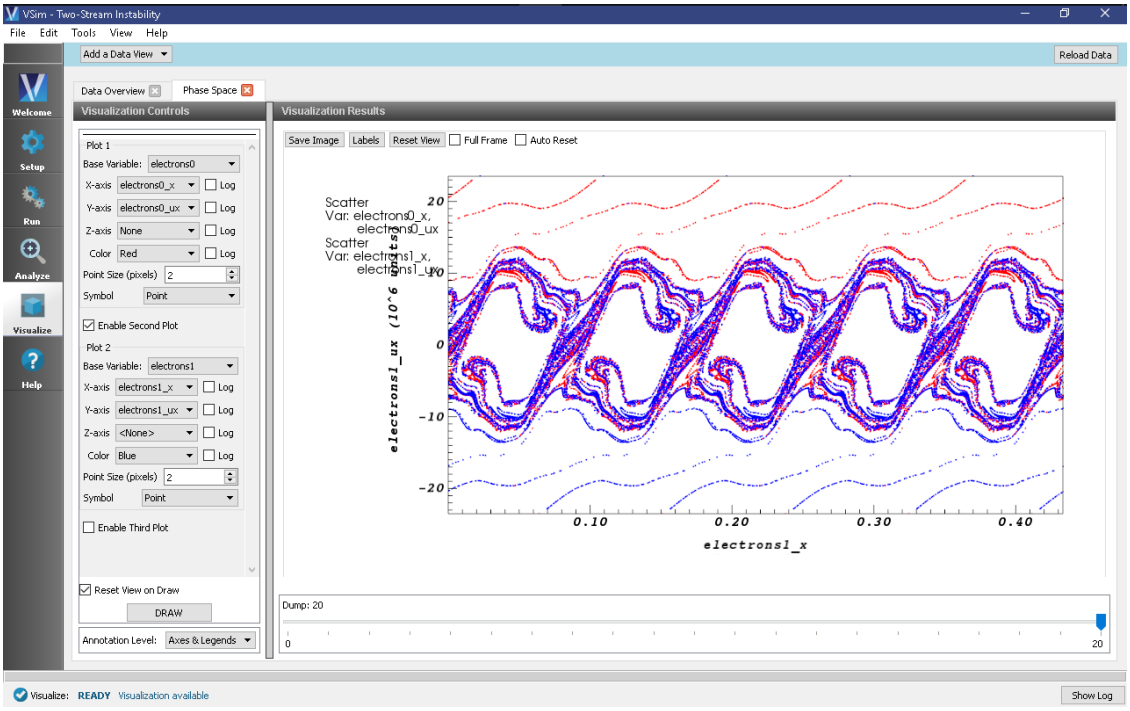


Fig. 2.27: Visualization of the two-stream instability developing in particle phase space.

2.2 Basic Examples (text-based setup)

2.2.1 Magnetic Fields of Wire (bFieldByJT.pre)

Keywords:

`Calculating A vector by a current carrying long linear wire.`

Problem description

This simulation illustrates how to model magnetostatics. A straight current, J_o, is directed along the z-axis. The example solves Poisson's equation for the vector potential, A. The 0th dump of the simulation is the analytical solution for the purpose of comparison.

This simulation can be performed with a VSimBase license.

Opening the Simulation

The magnetic field by a current example is accessed from within VSimComposer by the following actions:

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim for Basic Physics* option.
- Expand the *Basic Examples (text-based setup)* option.
- Select "Magnetic Fields of Wire (text-based setup)" 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 basic variables of this problem should now be alterable via the text boxes in the left pane of the *Setup Window*, as shown in Fig. 2.28.

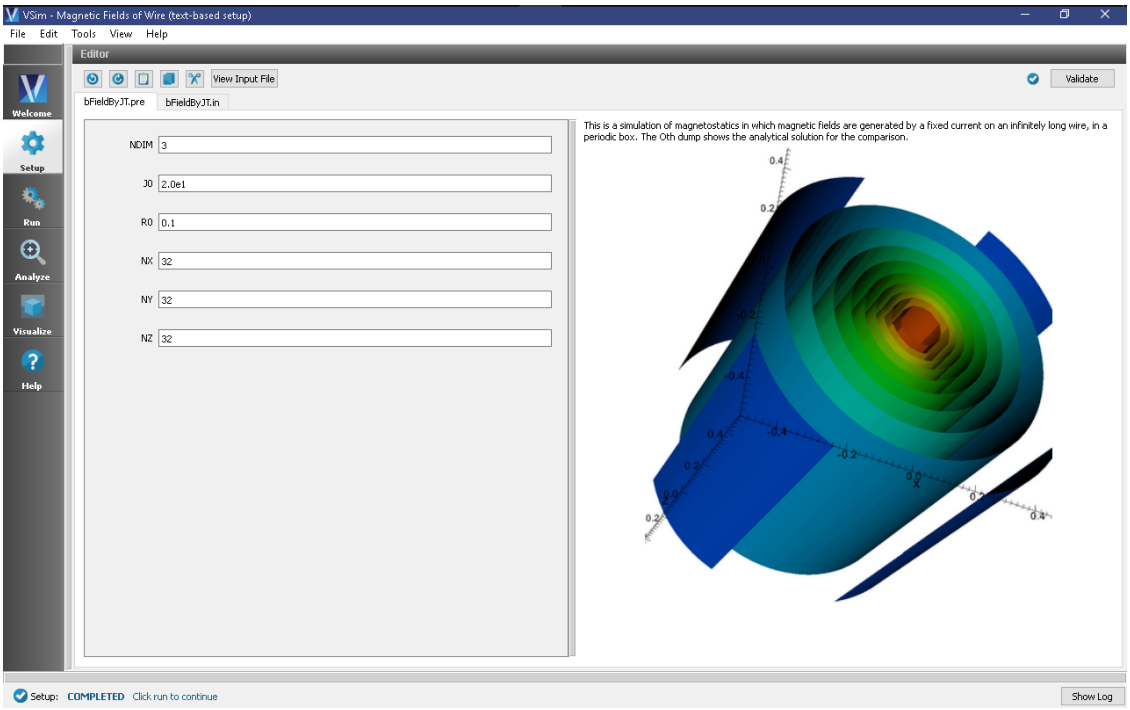


Fig. 2.28: Setup Window for the magnetic field by a current.

Input File Features

VSimComposer allows the user to vary the applied current and a radius of the wire. These are the main parameters that affect the A vector. By changing these parameters, a user can run simulations to explore how the A vector depends on the applied current and radius. The input file, when viewed in the editor, also exposes all the grid sizes of the simulation that are used by the implemented models.

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.
- 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 the right pane. The run has completed when you see the output, "Engine completed successfully." This is shown in Fig. 2.29.

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.

For this example, one can see the A vector fields. To see the A vector, continue as follows:

- Make sure the *Data View* drop down is set to *Data Overview*.

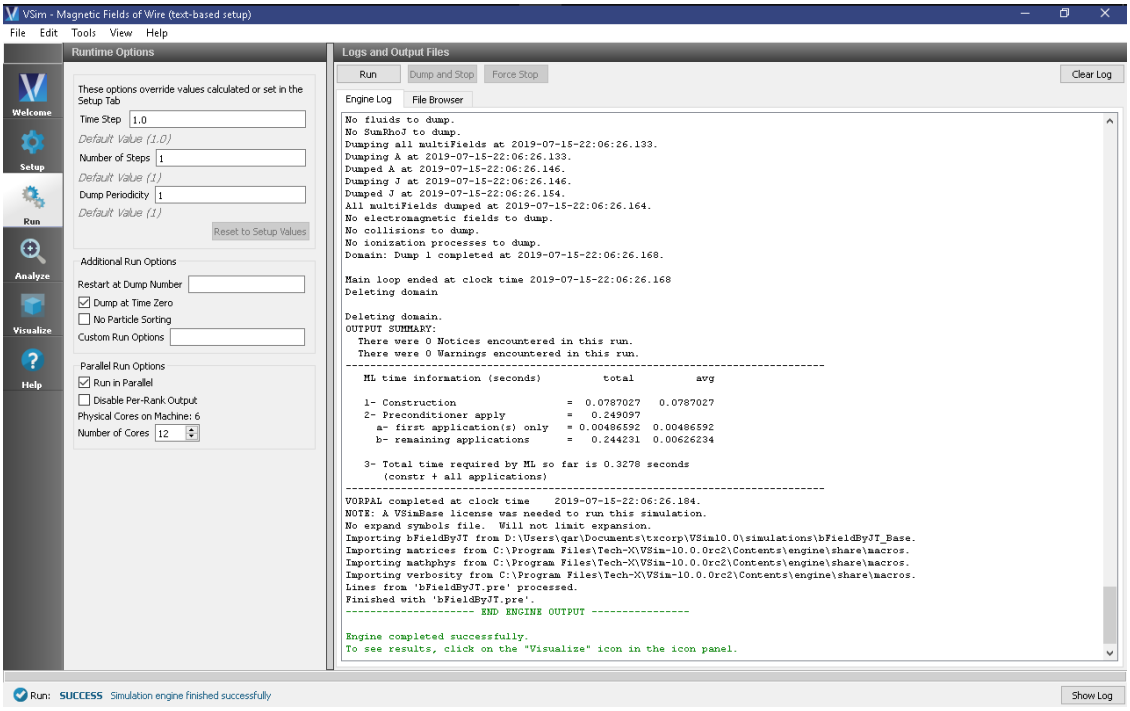


Fig. 2.29: The Run Window at the end of execution.

- Here you can see *Variables*. Expand the *Scalar Data*. Then expand *A*
- Four variables are available, the three components of *A* and the magnitude of the vector. Select A_magnitude.
- Check the box next to *Display Contours*
- Rotate the view by clicking and dragging your mouse.

Fig. 2.30 shows the visualization seen for A_magnitude.

Further Experiments

This input file can be modified to test different current, and wire radius. This will allow users to study how to use the magnetostatics capability in Vorpal.

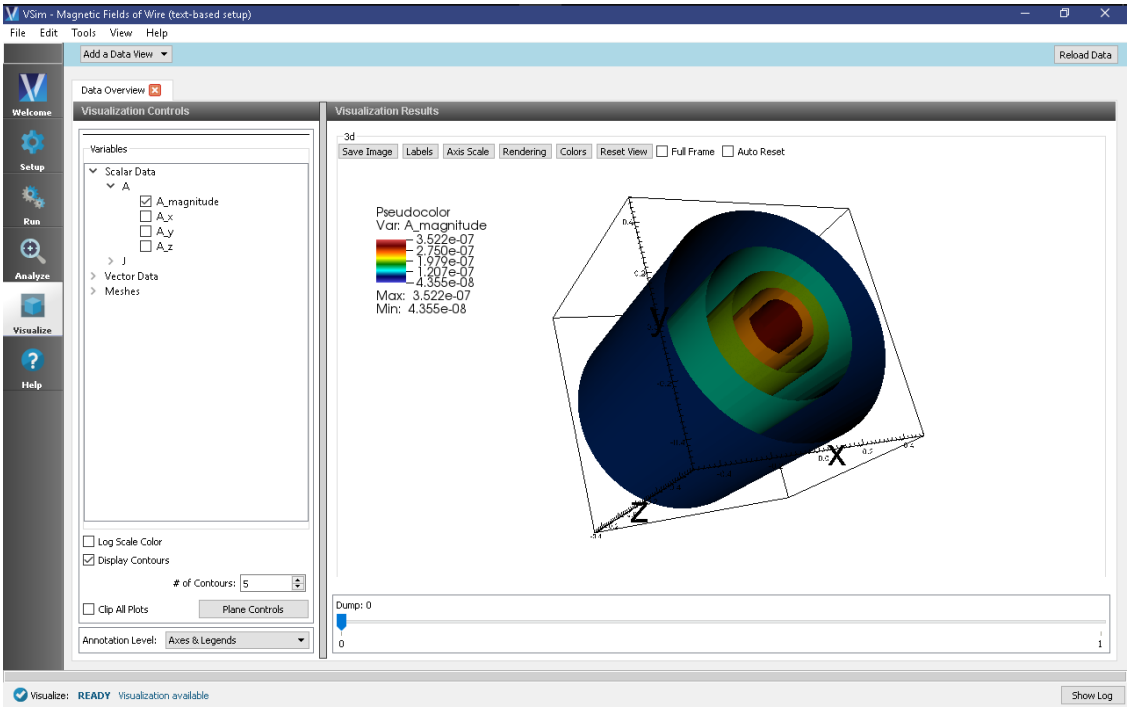


Fig. 2.30: Visualization of magnetic fields by the current as a color contour plot.

CHAPTER

THREE

VSIM FOR ELECTROMAGNETICS EXAMPLES

These examples illustrate how to solve complex problems in electromagnetics.

These examples can be run with a VSimEM license.

3.1 Antennas

3.1.1 Yagi-Uda (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.

This simulation can be performed with a VSimEM license.

Opening the Simulation

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

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim 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. 3.1. 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`.

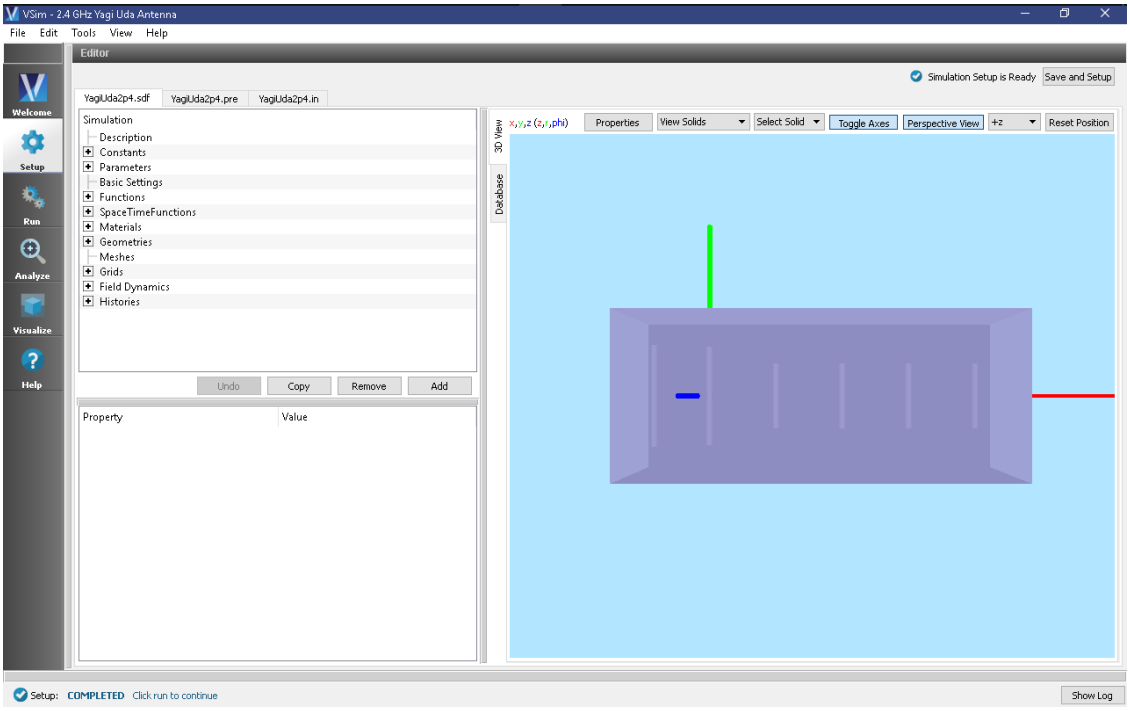


Fig. 3.1: Setup Window for the Yagi-Uda example.

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.
- Here you can set run parameters, including how many cores to run with.
- When you are finished setting run parameters, 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 the right pane. The run has completed when you see the output, "Engine completed successfully." This is shown in Fig. 3.2.

Analyzing the Results

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

Select "computeFarFieldFromKirchhoffBox.py" from the analyzer list, and click "Open."

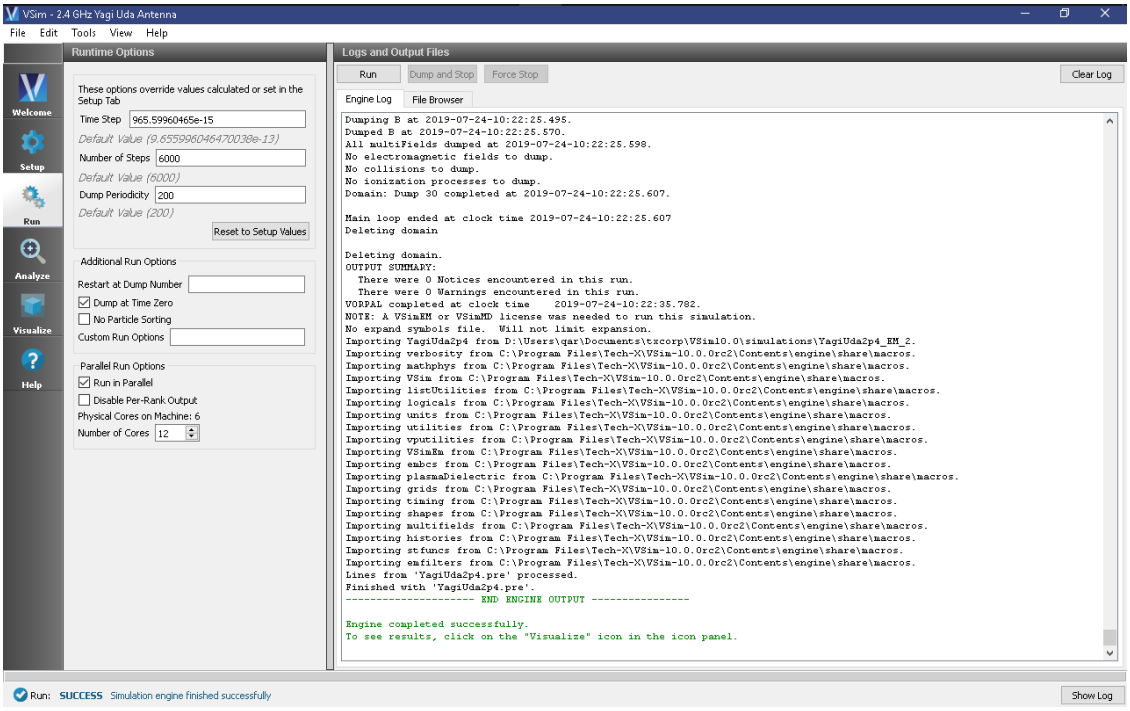


Fig. 3.2: The Run Window at the end of execution.

The default parameters are sufficient for this problem. Input 10.0 for the *farFieldRadius* parameter and run the analyzer by clicking the "Analyze" button.

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*
- Click *Colors*
- Check the *Fix Minimum* box and set the value to -0.1
- Check the *Fix Maximum* box and set the value to 0.1, then click "OK"
- Expand *Geometries*
- Select *poly (YagiUda2p4PecShapes)*
- Select *Clip All Plots*
- 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. Check the farE_magnitude box, remove the minimum and maximum restrictions on colors, and uncheck *Clip All Plots*.

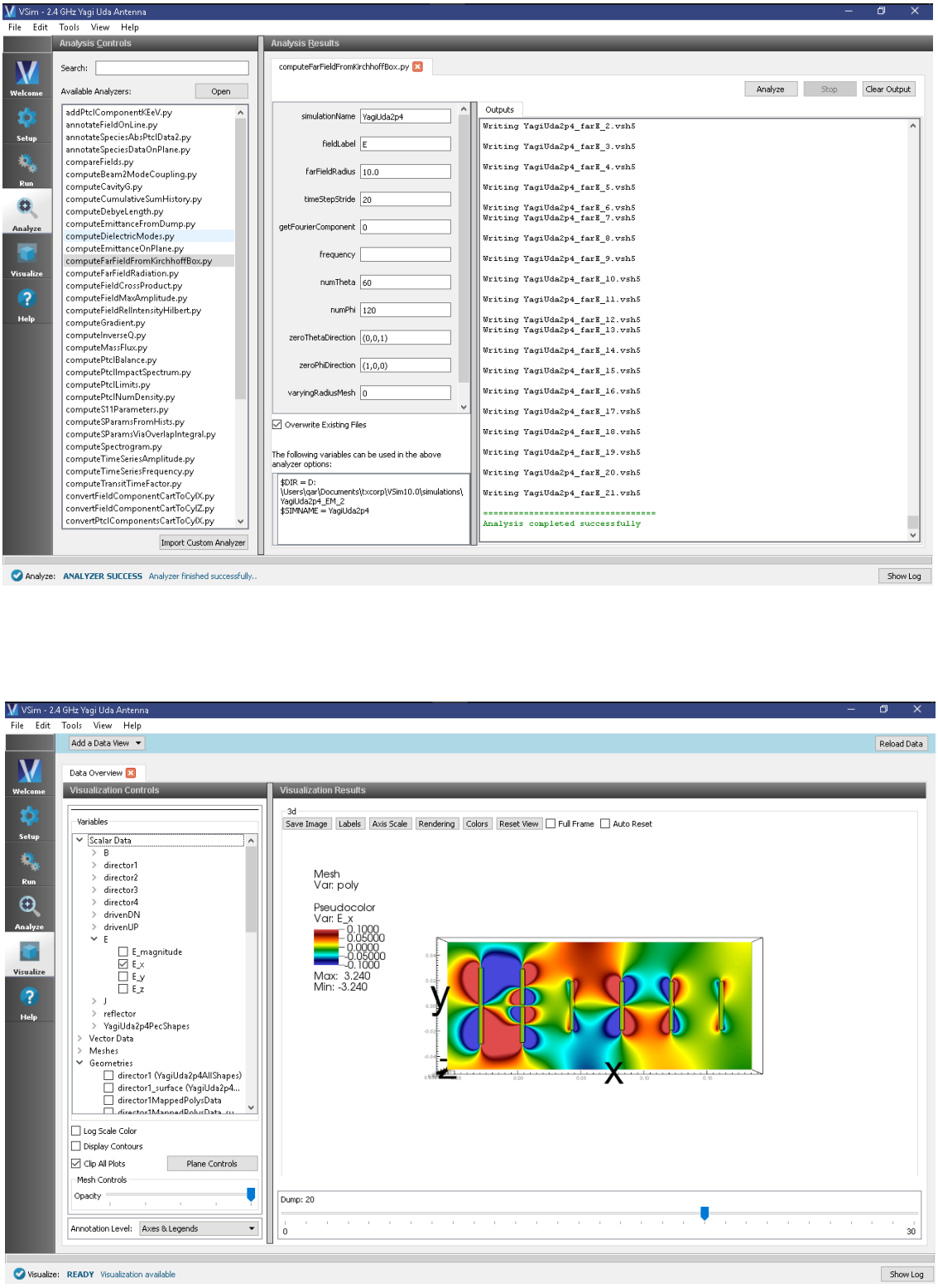


Fig. 3.3: The electric field near-field pattern.

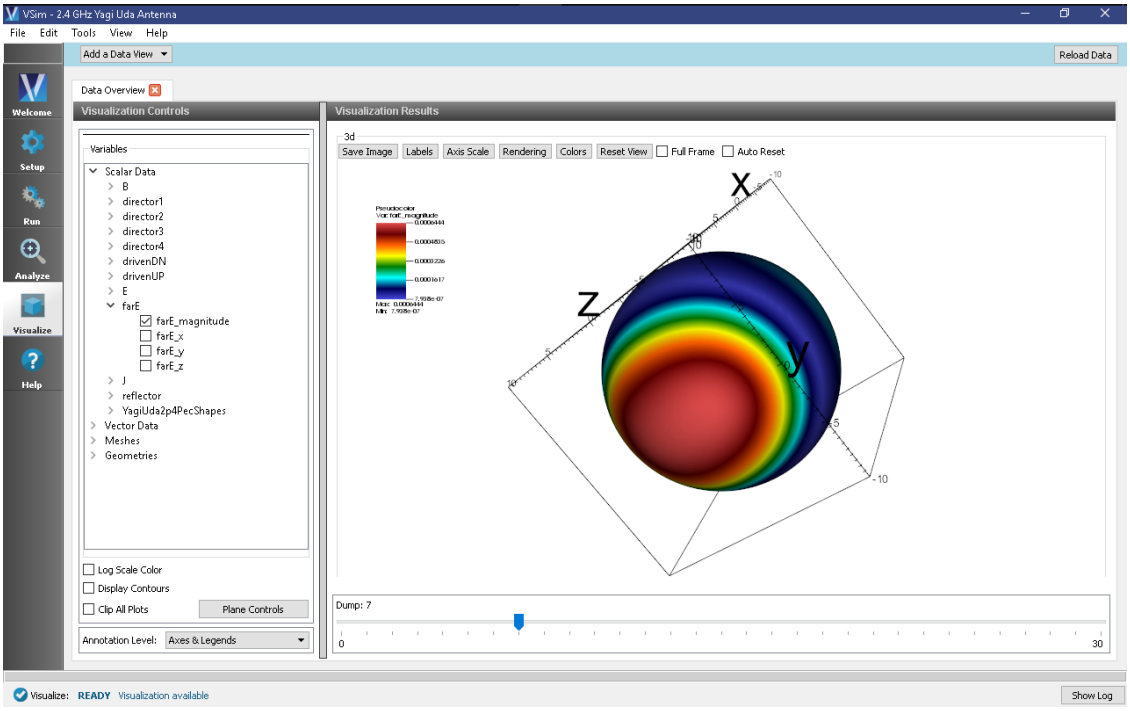


Fig. 3.4: The electric field manifestation of the far field pattern.

Further Experiments

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

3.1.2 Antenna Array 2D (antennaArray2D.sdf)

Keywords:

keywordone, keyword two, keywordthree, keyword four

Problem Description

This set of 2-D VSimEM 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*.

This simulation can be run with a VSimEM license.

Opening the Simulation

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

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim 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. 3.5.

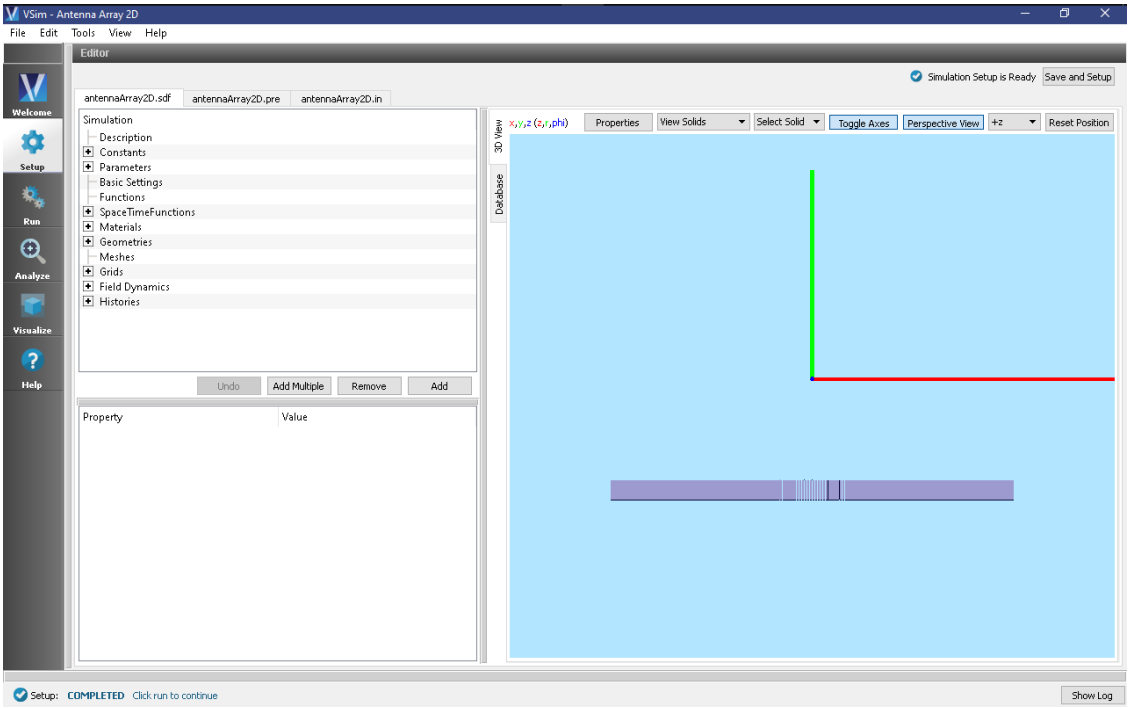


Fig. 3.5: Setup Window for the Antenna Array 2D example.

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 beteen 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 Attay. 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.

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 Stepts: 6000 * *Dump Periodicity: 3000 * *Dump at Time Zero: box checked

After the first run completes, proceeds as follows:

- Second run settings: * *Change *Number of Stepts* to 1800 * *Change *Dump Periodicity* to 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. 3.6.

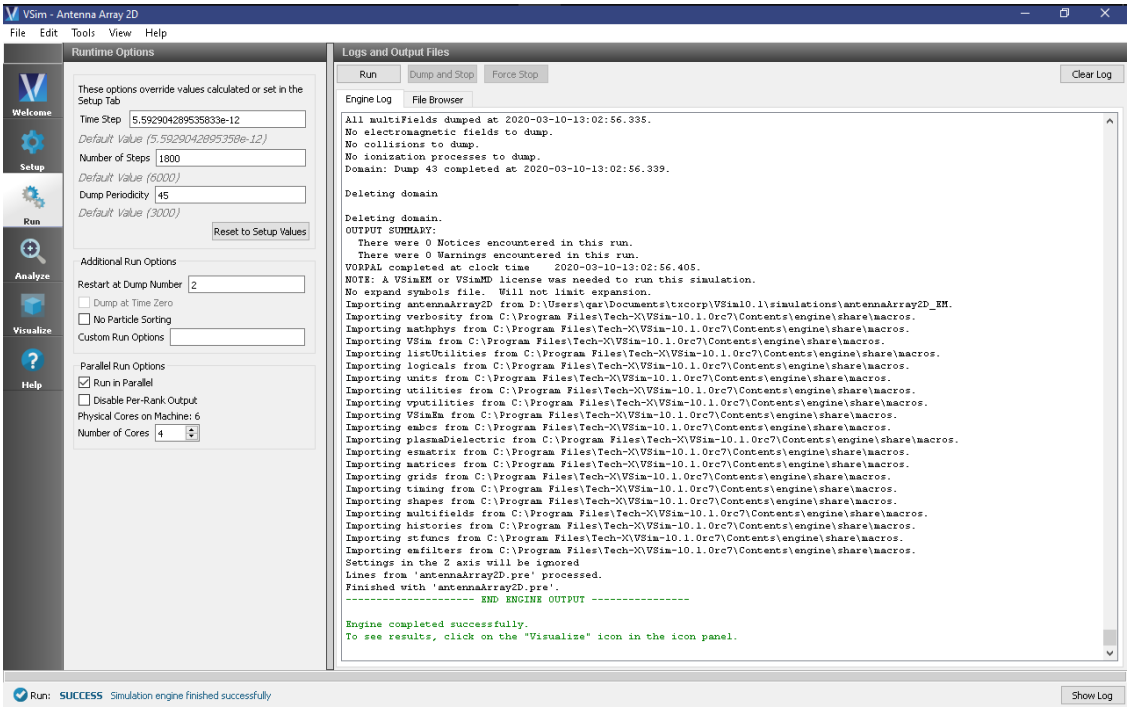


Fig. 3.6: The Run Window at the end of the second run execution.

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*
- Click on *Colors* in the *Visualization Results* pane
- Check the box for *Fix Minimum* and set it to -100
- Check the box for *Fix Maximum* and set it to 100
- Click OK

- 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. 3.7.

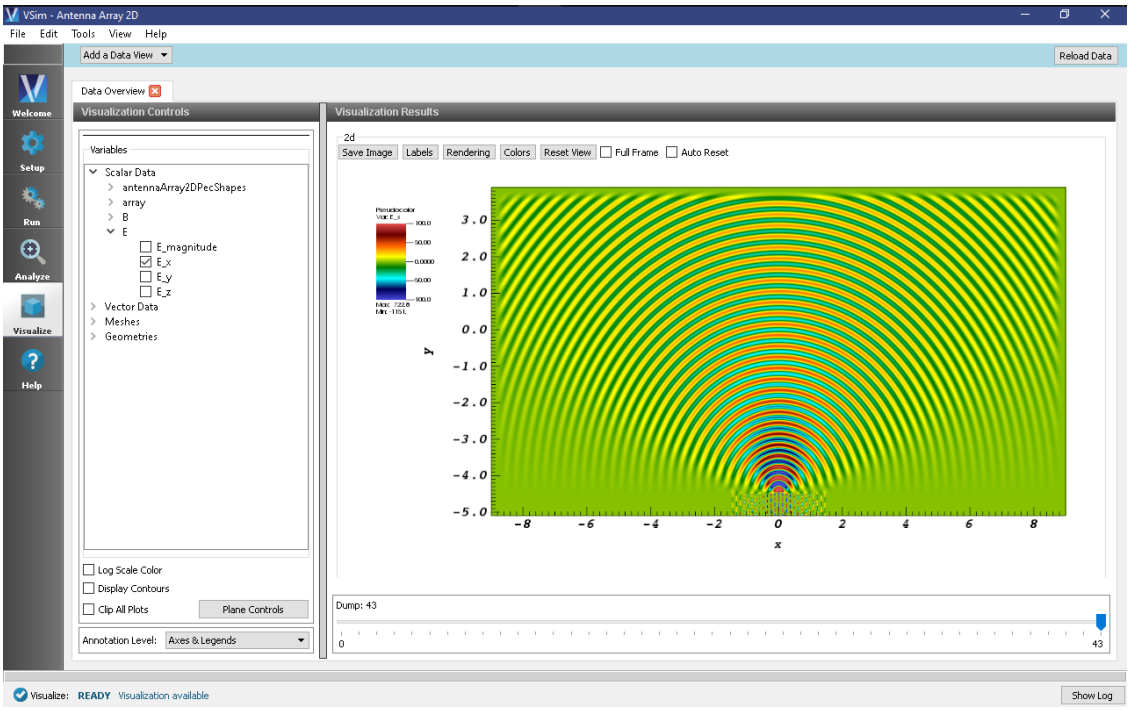


Fig. 3.7: The near and far electric fields in the x-direction at the end of the simulation.

Figure Fig. 3.7 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.

Single Element Antenna

- Expand Constants
- Change constants N_ELEM to 1
- Change N_EXCITED_ELEM to 1
- Expand Geometries
- Expand CSG
- Remove *array*
- Remove *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 viz is shown in `antennaarray2dvizwinrun2`.

:

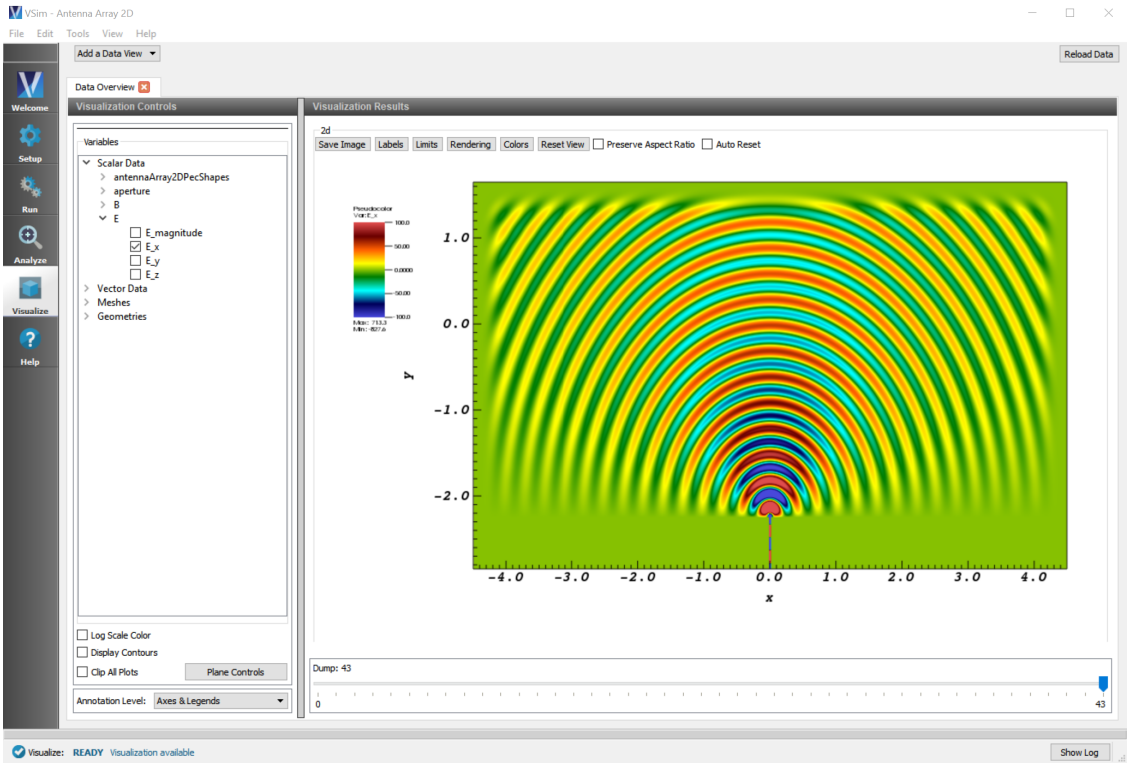


Fig. 3.8: The near and far electric fields in the x-direction for a 1-element antenna.

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 work *Calibration* before *.sdf*

- Click *Save*
- Expand *Geometries*
- Expand *CSG*
- Remove *array*
- Remove *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., waveguide).

- 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.
- The *compute2DantennaGainAndPhase.py* analyzer will open by default
- The default analyzer fields are the following:
  - *simulationName*: antennaArray2D
  - *dumoNr*: 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. 3.9.

The S-paramters 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 VSim in dB, the

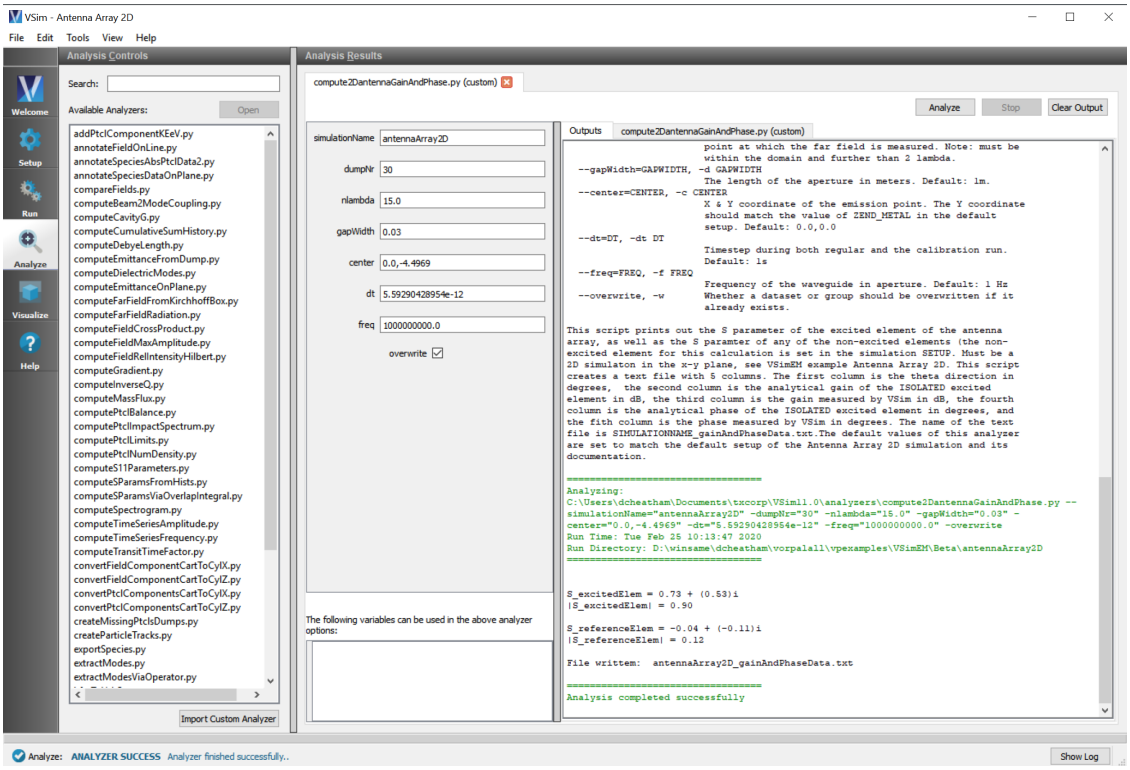


Fig. 3.9: The S-paramters for the excited element as well as the reference element associated with the constant S_PARAM_ELEM in the simulation setup are shown at the end of the analyzer run.

fourth column is the analytical phase of the ISOLATED excited element in degrees, and the fith column is the phase measured by VSim 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. 3.10.

Plotting the fourth and thith columns (analytical and measured field phases) against the first column (as a function of theta) will give the results shown in Fig. 3.11.

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.

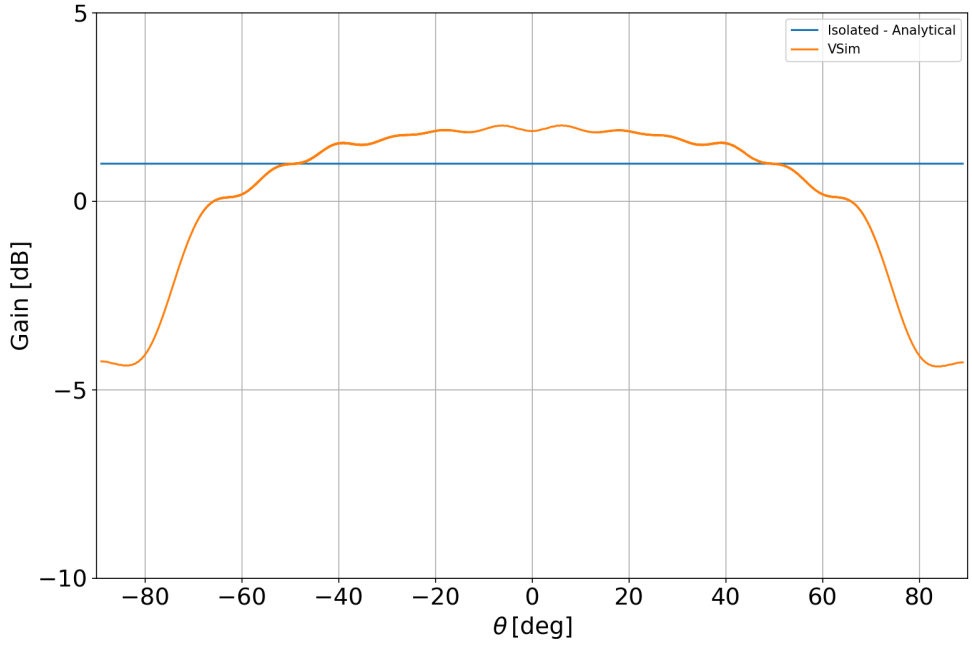


Fig. 3.10: The gain pattern of a 25-element array with the center excited element.

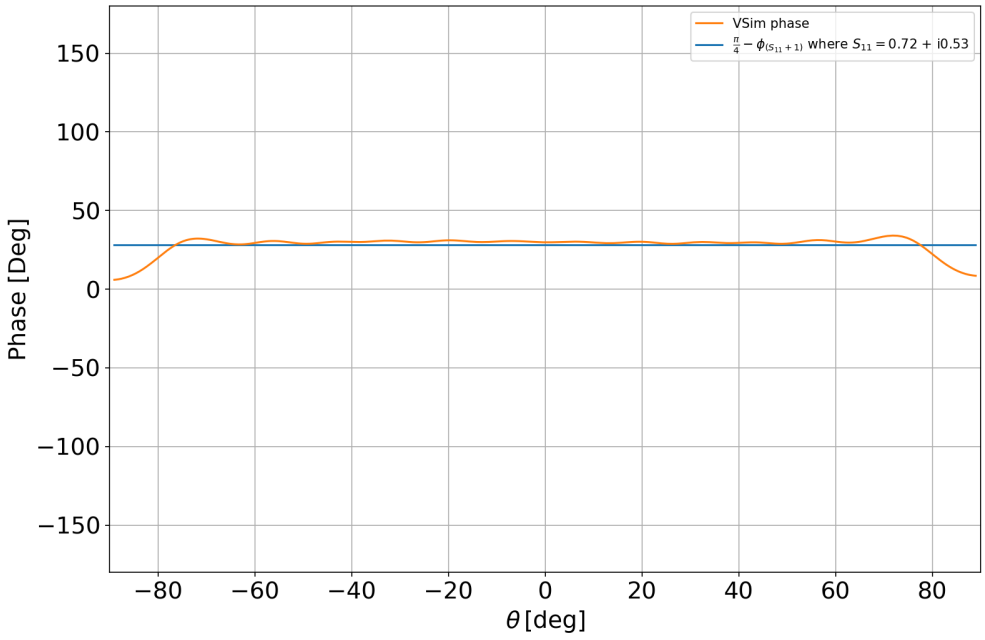


Fig. 3.11: The phase pattern of a 25-element array with the center excited element.

Repeating the analysis steps for a 1-element antenna (N_ELEM = 1 in the simulation setup) will give the results shown in Fig. 3.12 and Fig. 3.13.

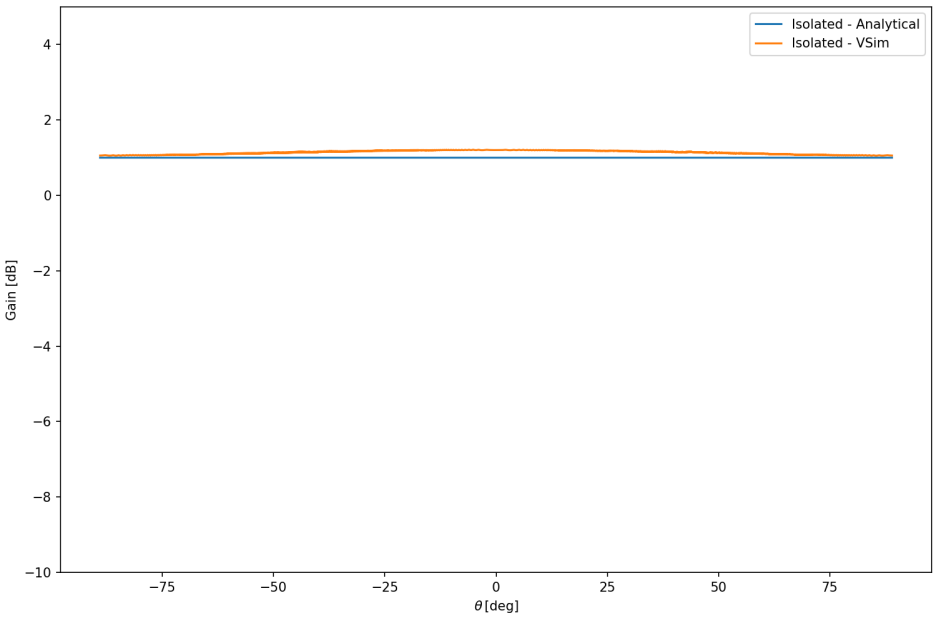


Fig. 3.12: The gain pattern of a 1-element array.

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

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. 3.14 and Fig. 3.15.

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

Keywords:

`antennaOnHand, far field, radiation`

> **Warning:** Due to a known issue parallel runs, we suggest limiting the run to 8 cores.

Problem Description

This problem calculates the far-field radiation pattern of a small (cellular mobile) antenna mounted on a small curved dielectric (plastic/PVC). The fields interact with the human hand for which the bone structure was approximated by long thin cylinders. The antenna feed is a 850 MHz source.

This simulation can be performed with a VSimEM license.

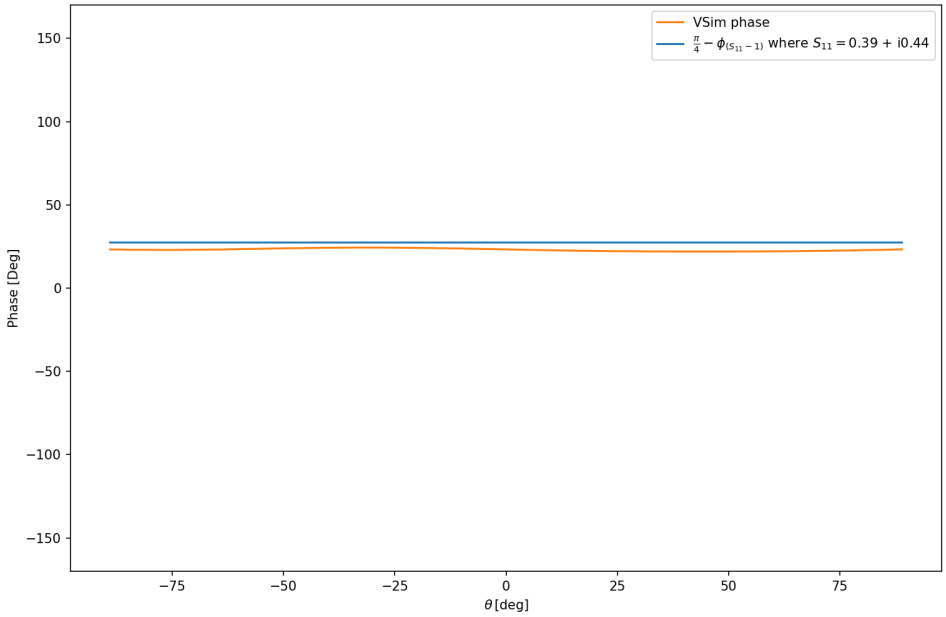


Fig. 3.13: The phase pattern of a 1-element array.

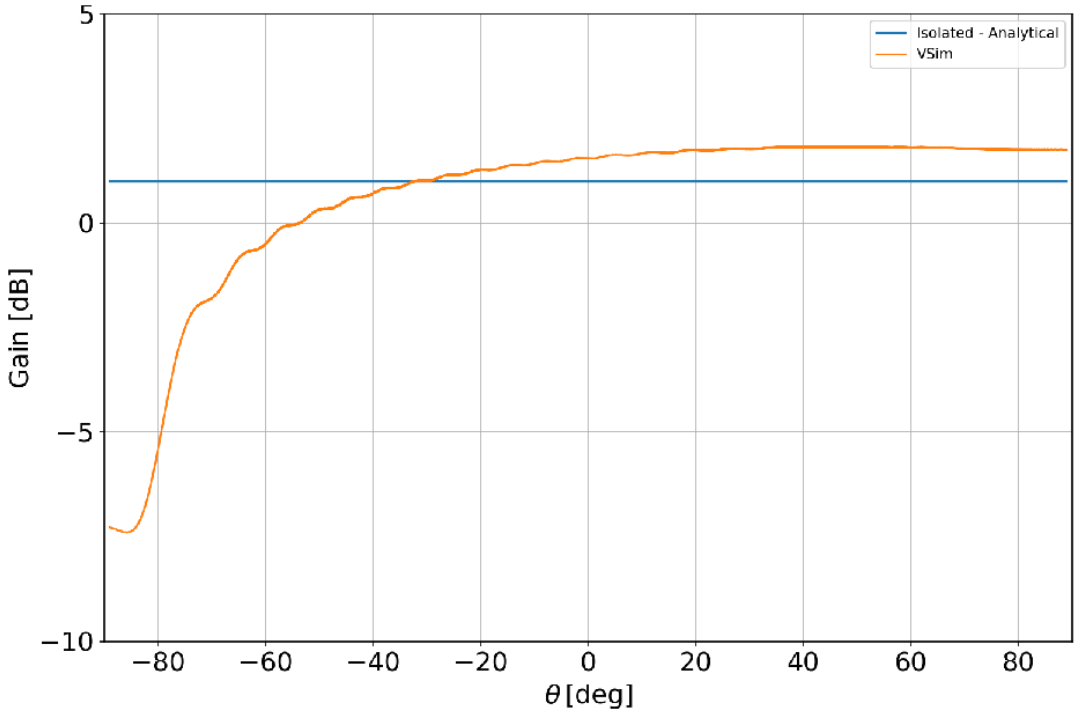


Fig. 3.14: The gain pattern of a 25-element array with the edge excited element.

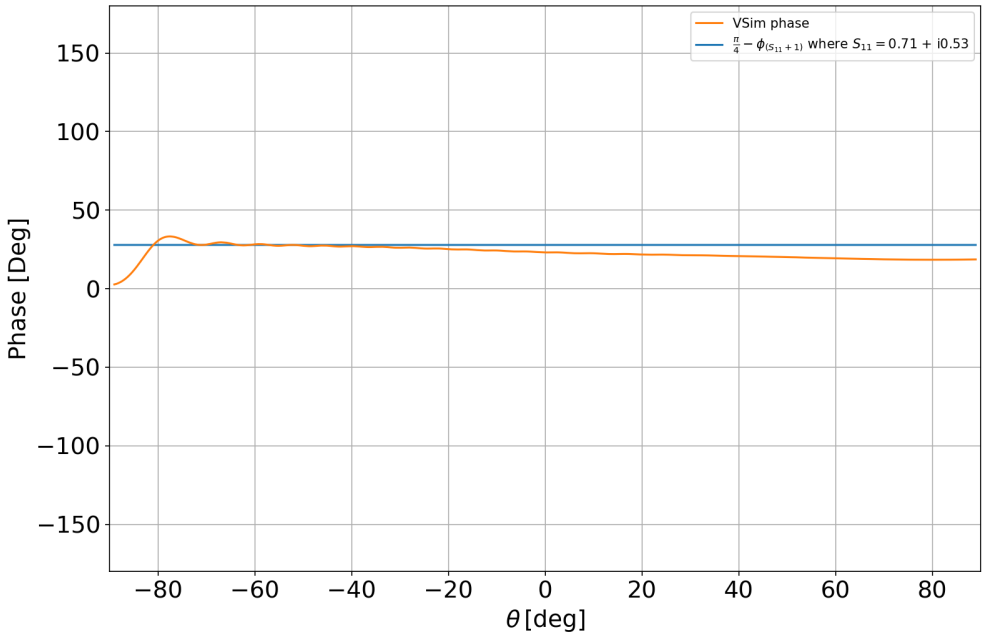


Fig. 3.15: The phase pattern of a 25-element array with the edge excited element.

Opening the Simulation

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

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim 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. 3.16.

Simulation Properties

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

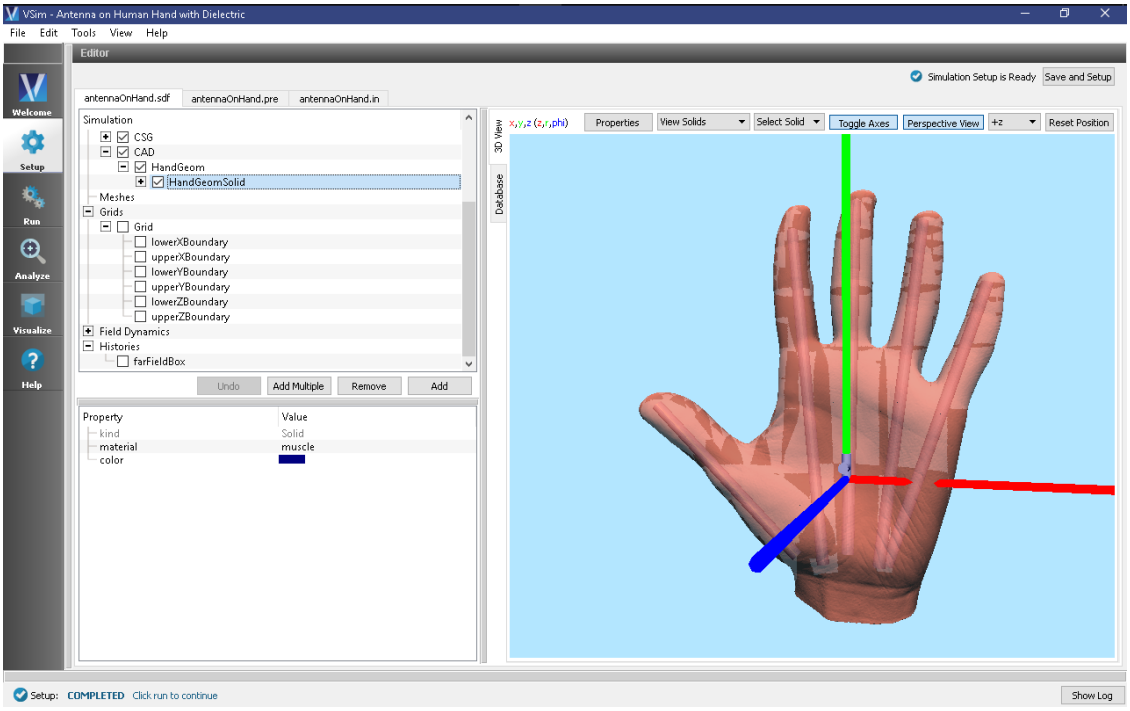


Fig. 3.16: Setup Window for the Antenna on Human Hand with Dielectric example, with Grid and farFieldBox History hidden.

- 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 when you see the output, "Engine completed successfully." This is shown in the window below.

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. 3.18)
- Input values for the analyzer parameters. The analyzer may be run multiple times, allowing the user to experiment with different values.
  - simulationName - antennaOnHand (name of the input file)
  - fieldLabel - E (name of the electric field)
  - farFieldRadius - 0.5 (distance to far field in m, 10.0 is a good value)
  - timeStepStride - 150 (number of timesteps between far field calculations; determines how many far fields are output)
  - getFourierComponent - 0, do not integrate assuming single fourier frequency
  - frequency - not used because getFourierComponent is false
  - numTheta - 60 (number of theta points in the far field, 30 for a quick calculation, 60 for finer resolution)
  - numPhi - 120 (number of phi points in the far field, 60 for a quick calculation, 120 for finer resolution)
  - zeroThetaDirection - (0,0,1) (determines orientation of far field coordinate system)

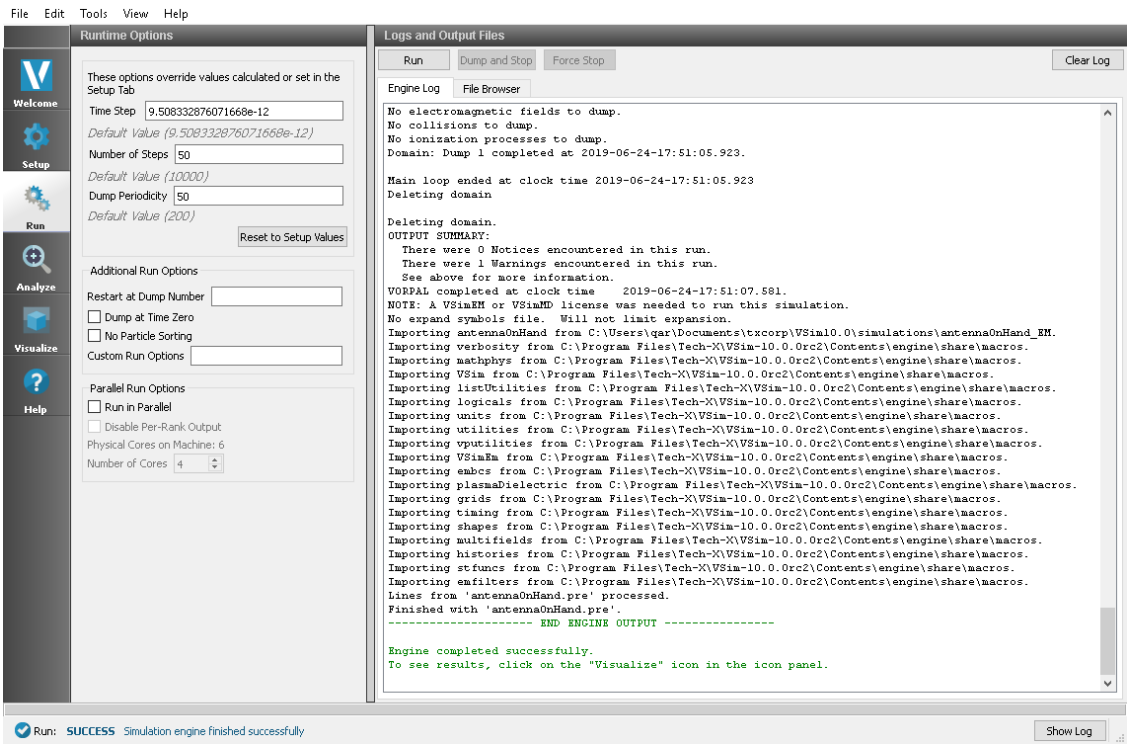


Fig. 3.17: The Run Window at the end of execution.

  - zeroPhiDirection - (1,0,0) (determines orientation of far field coordinate system
  - varyingRadiusMesh - 1 (Set to 1 in order to make far field mesh adapt to magnitude of far field solution: the classic lobe view - Note: using a varying mesh option will make the analyzer run very slow.)
- Click “Analyze”
- Depending on the values of numTheta, numPhi, and timeStepStride, the script may need to run for several minutes or longer.

Visualizing the Results

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 *farE* and check the *farE_Magnitude* box. The poly_surface (HandGeomSolid) under *Geometries*, can also be plotted. Fig. 3.19 shows the visualization at last far-field time.

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.

3.1.4 Coaxial Loop Antenna (coaxialLoopAntenna.sdf)

Keywords:

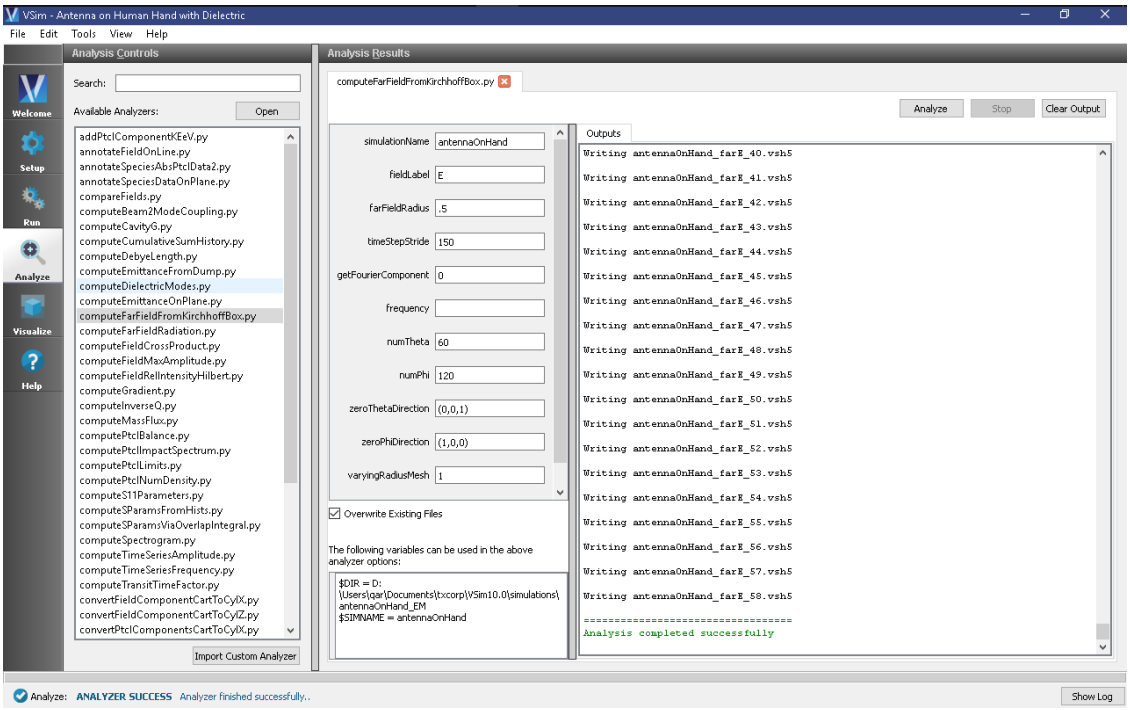


Fig. 3.18: The **Analyze** panel after running computeFarFieldFromKirchhoffBox.py.

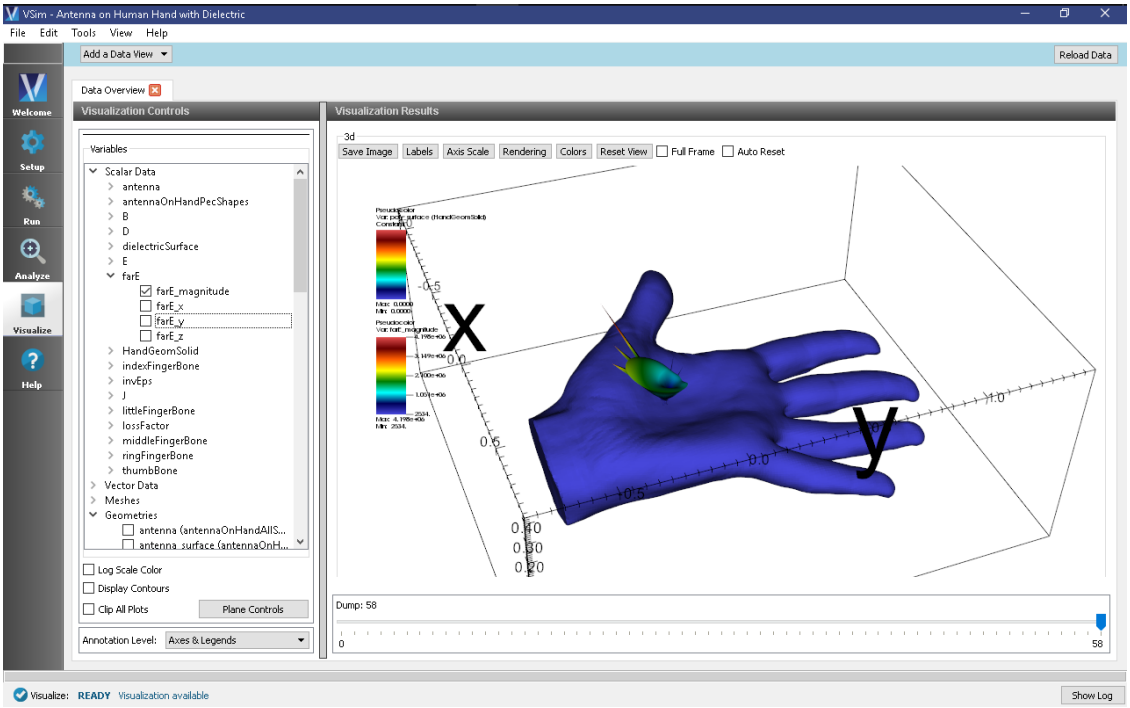


Fig. 3.19: The Far Field Radiation Pattern.

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.

This simulation can be run with a VSimEM, VSimMD, VSimPD, or VSimPA license.

Opening the Simulation

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

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim 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. 3.20.

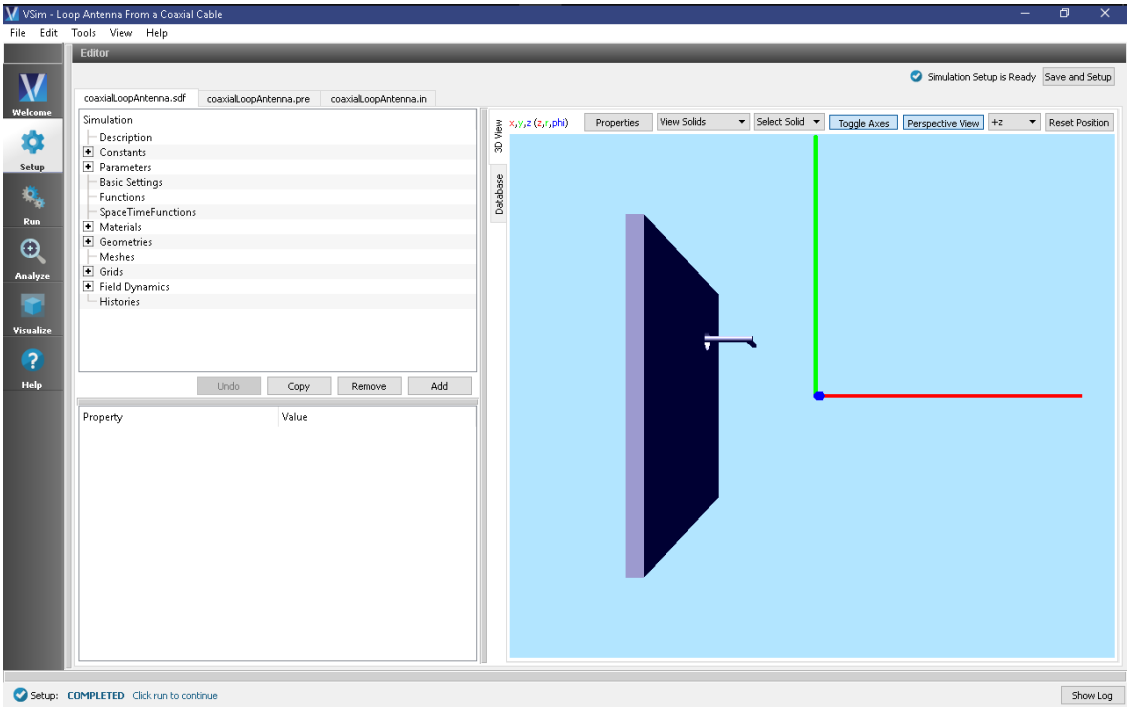


Fig. 3.20: Setup Window for the Coaxial Loop Antenna example.

Simulation Properties

This simulation makes use of the new coaxial waveguide Field Boundary Condition in VSim 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 exist 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

Once finished with the problem setup, continue as follows:

- Proceed to the Run Window by pressing the Run button in the left column of buttons.
- Select appropriate settings in the Parallel Run Options section
- 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."

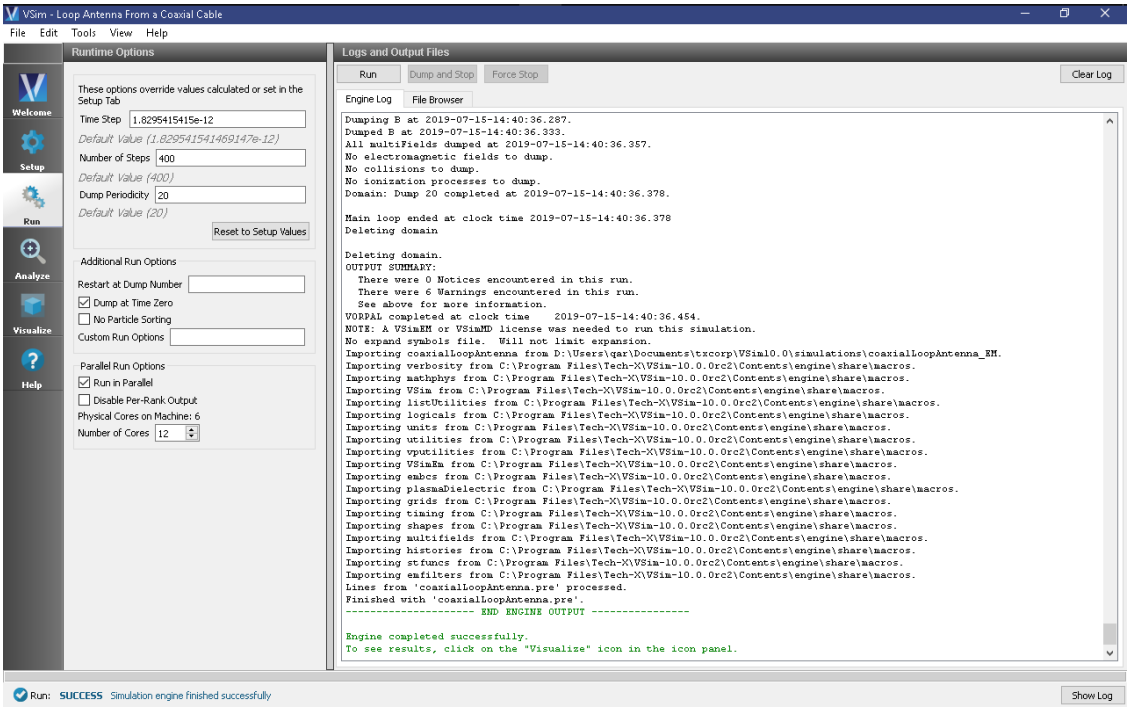


Fig. 3.21: The Run Window at the end of execution.

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*, and select *E_z*.

- To slice inside the field, select *Clip All Plots* in the lower left hand corner, then click on *Plane Controls* and change the clip plane normal to Y instead of Z, and adjust the origin of the normal vector to Y = 0.05.
- Access the Color Options by clicking the *Colors* button, check the *Fix Minimum* and *Fix Maximum* boxes, and 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.

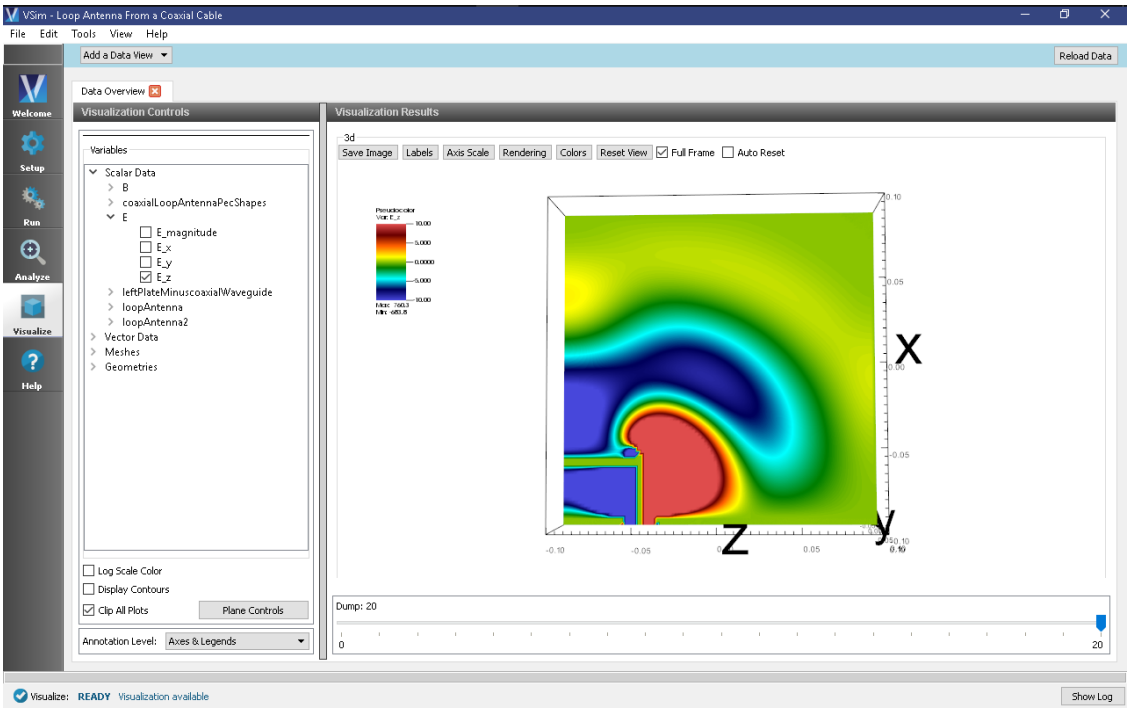


Fig. 3.22: The E_z field propagating off of the loop antenna.

3.1.5 Dipole Antenna (dipoleAntenna.sdf)

Keywords:

`antenna, electromagnetics, current source`

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

This simulation can be run with a VSimEM, VSimMD, or VSimPD license.

Opening the Simulation

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

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim 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. 3.23.

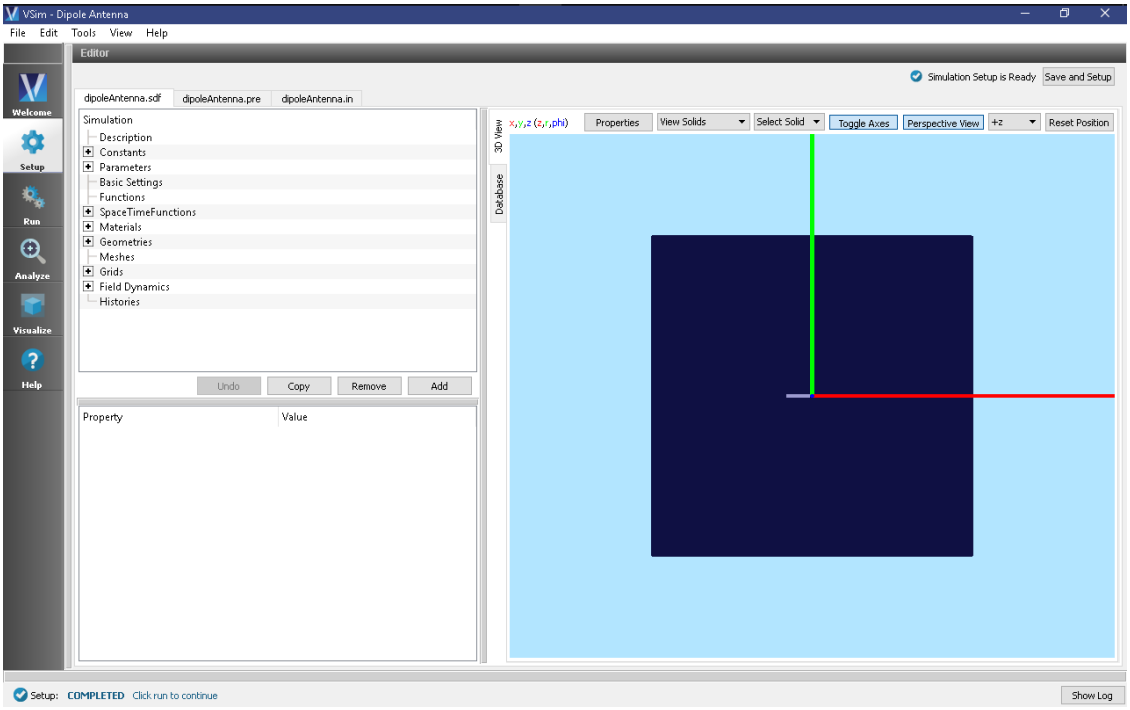


Fig. 3.23: Setup Window for the Dipole Antenna example.

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* → *DrivingCurrent*. 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 *drivingCurrent* spacetime function. The *drivingCurrent* 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

To run the simulation:

- Proceed to the run window by pressing the Run button in the left column of buttons.
- Here you can set run parameters, including how many cores to run with (under the MPI tab).
- When you are finished setting run parameters, 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." This is shown in Fig. 3.63.

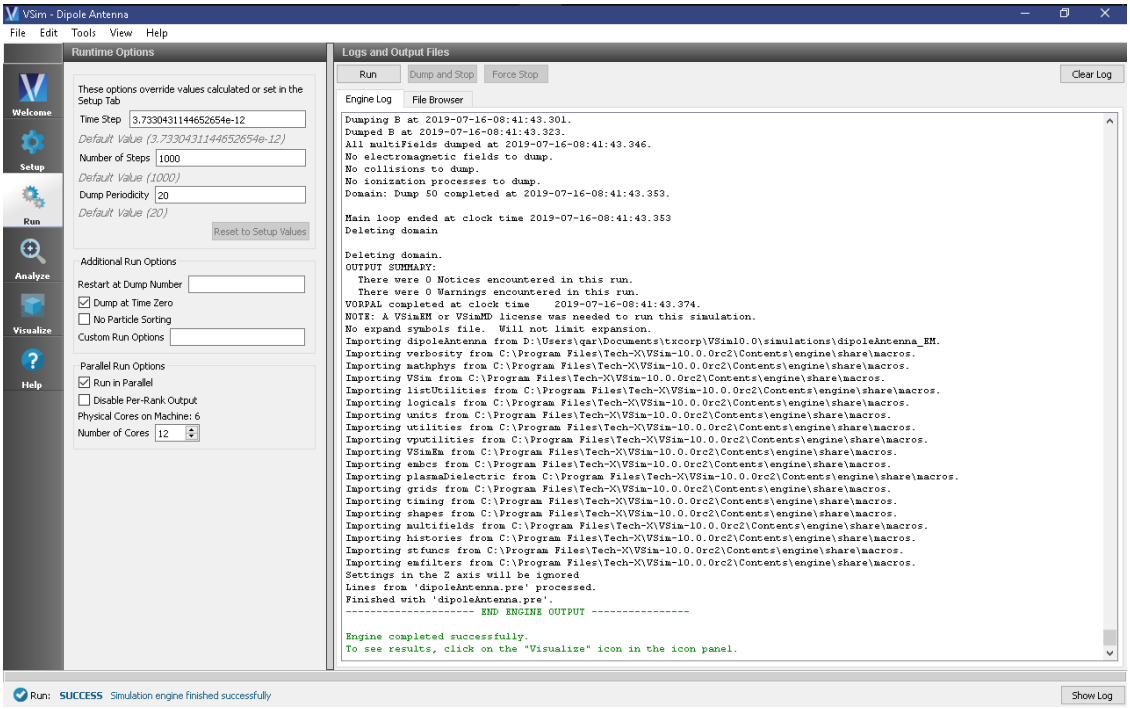


Fig. 3.24: The Run window at the end of execution.

Visualizing the Results

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

1. Proceed to the Visualize Window by pressing the *Visualize* button in the navigation column.
2. With the *Data Overview* open, expand *Scalar Data* then expand *E*.
3. Check the box for *E_y*. This will plot the y-component of the electric field.
4. Click on the *Colors* button found in the *Visualization Results* Pane.
5. In the *Color Options* window that opens, set *Fix Minimum* to -0.5, and *Fix Maximum* to 0.5.
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. 3.25.

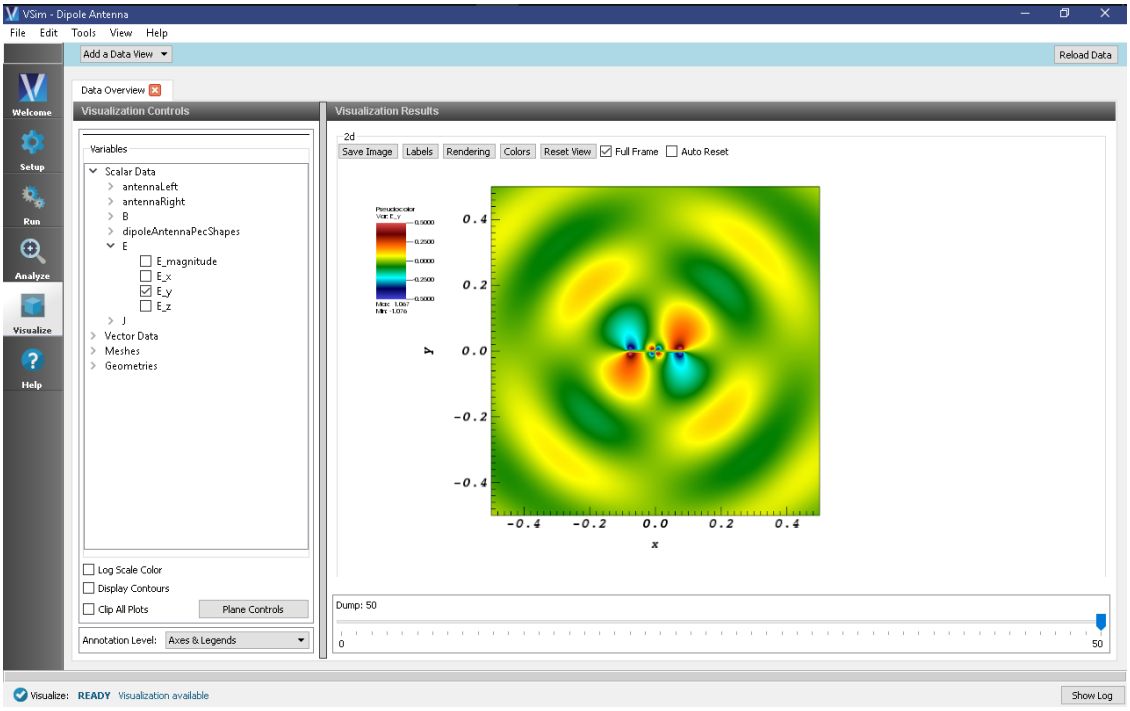


Fig. 3.25: The y-component of the Electric Field shows a 4-lobe pattern.

Un-check the *E_y* box, then check the box for *E_magnitude*. Then go back to the *Colors Options* and switch *Fix Minimum* to -.1. The magnitude of the electric field at the end of the simulation is shown in Fig. 3.26.

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.

3.1.6 Dipole Above Conducting Plane (dipoleOnConductingPlane.sdf)

Keywords:

`dipoleOnConductingPlane, far field, radiation`

Problem Description

This problem illustrates how to obtain far fields within VSim 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 VSimBase, but modified with functionality available as part of the VSimEM 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.

This simulation can be performed with a VSimEM license.

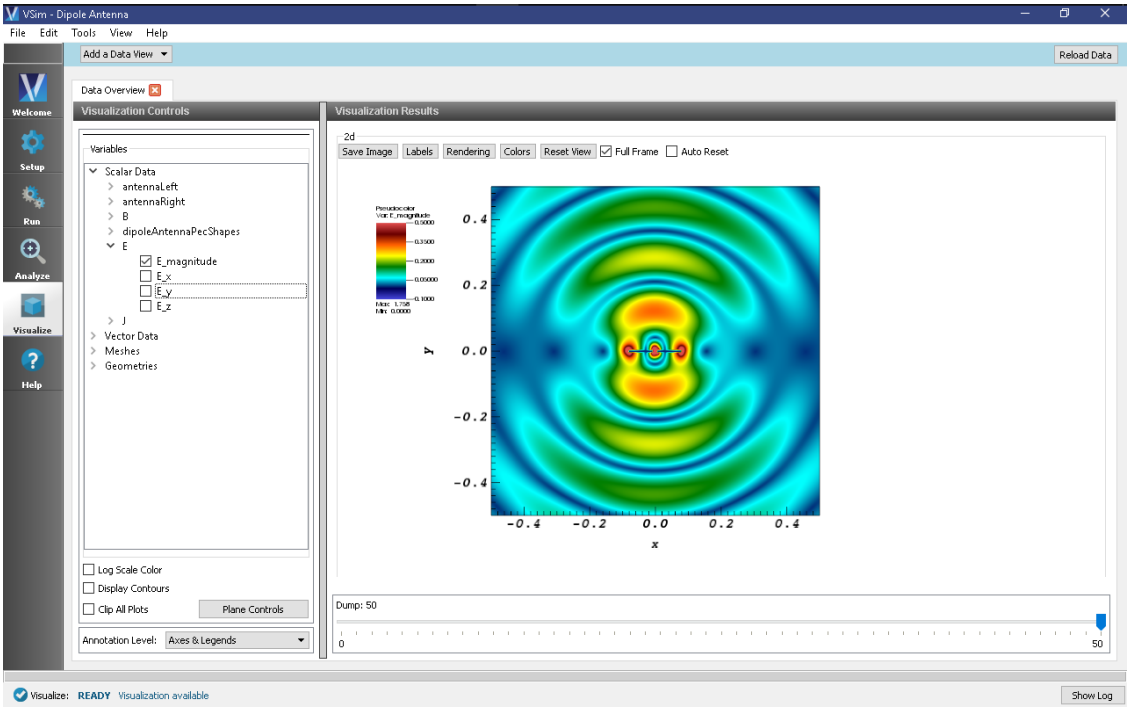


Fig. 3.26: The magnitude of the electric field shows the full 2-lobed dipole antenna pattern.

Opening the Simulation

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

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim 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. 3.27. 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`.

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.

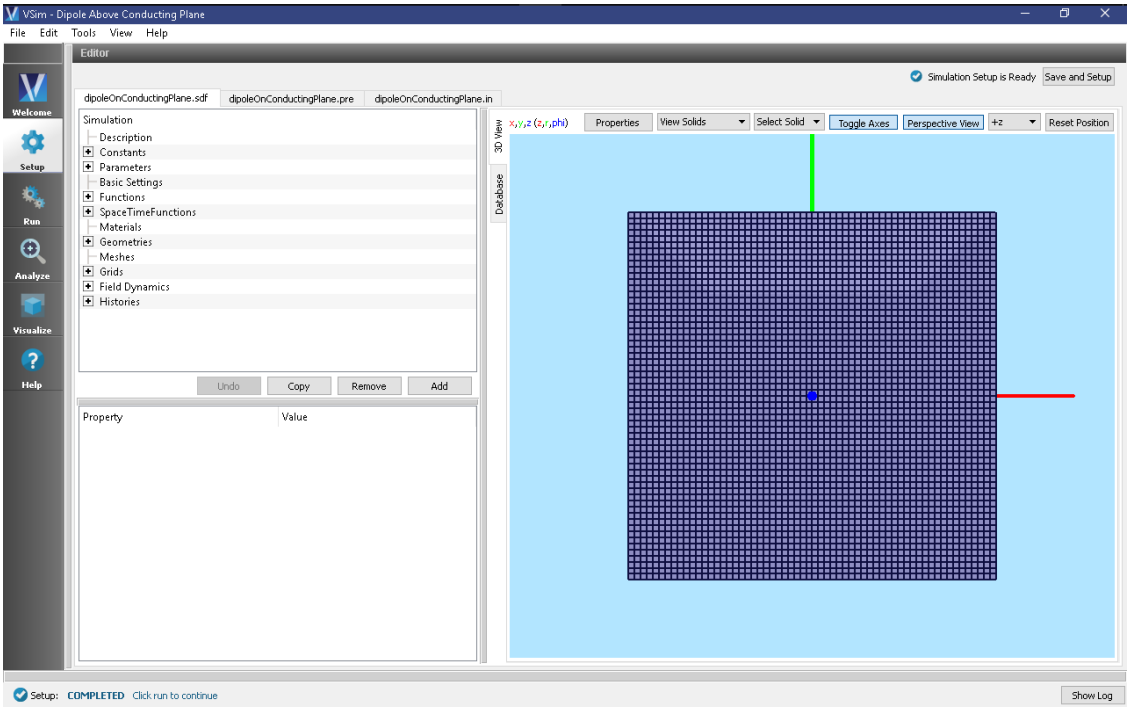


Fig. 3.27: Setup Window for the Dipole Above Conducting Plane example.

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.
- To run the file, click on the *Run* button in the upper left corner of the window. 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 is shown in Fig. 3.28.

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 - 15.
  - timeStepStride - 15
  - getFourierComponent - 0
  - frequency - 3e9
  - numTheta - 45 (number of points in the theta direction)

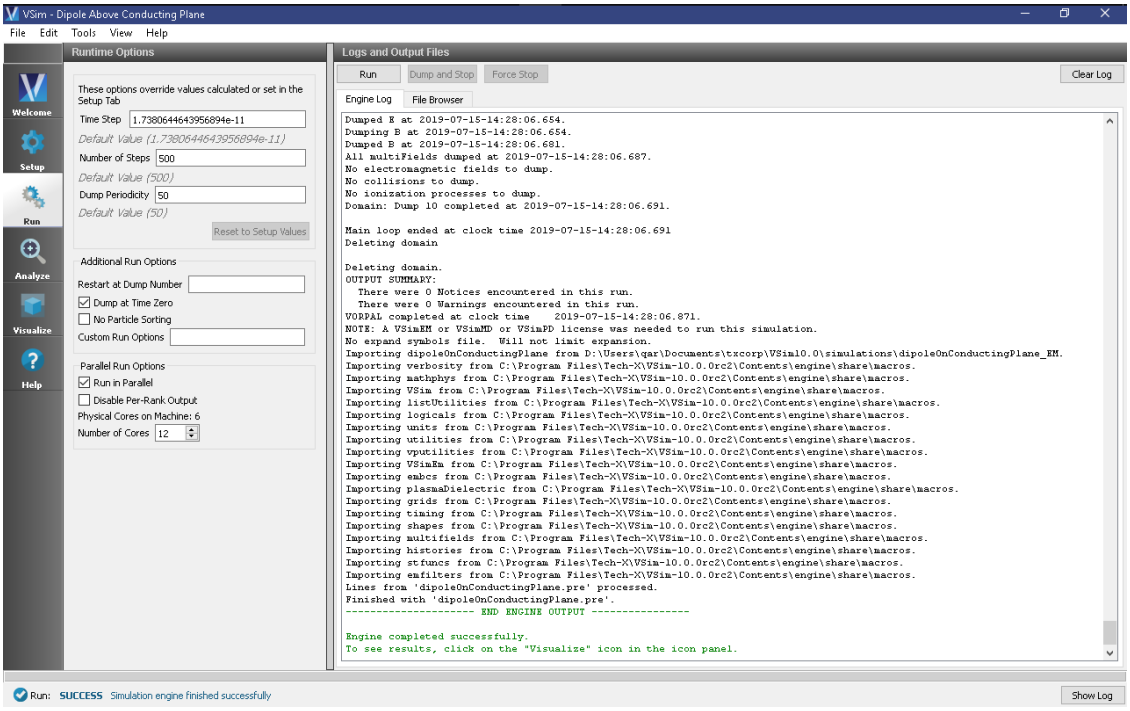


Fig. 3.28: The Run Window at the end of execution.

  - numPhi - 90 (number of points in the phi direction)
  - zeroThetaDirection - (0,0,1)
  - zeroPhiDirection - (1,0,0)
  - varyingRadiusMesh - 1 (varies the radius of the field based on field magnitude)
  - simpsonIntegration - 0
- Click the *Analyze* button near the top right of the window.

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 All Plots* 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.

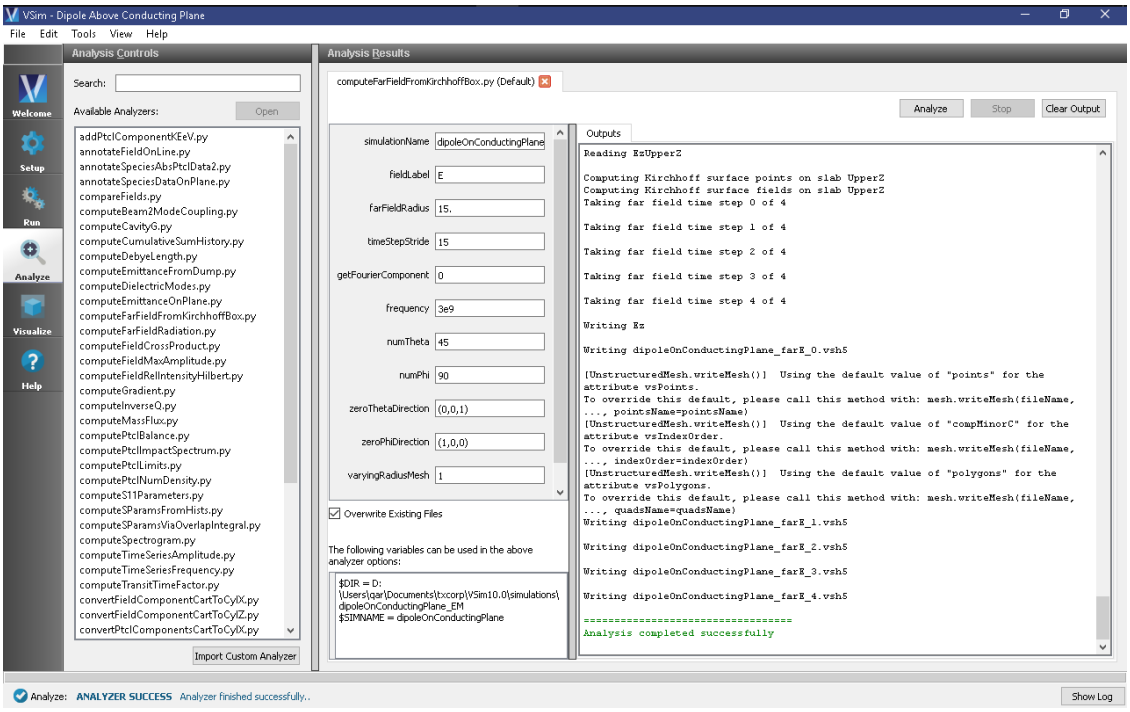


Fig. 3.29: The Analyze Window at the end of execution.

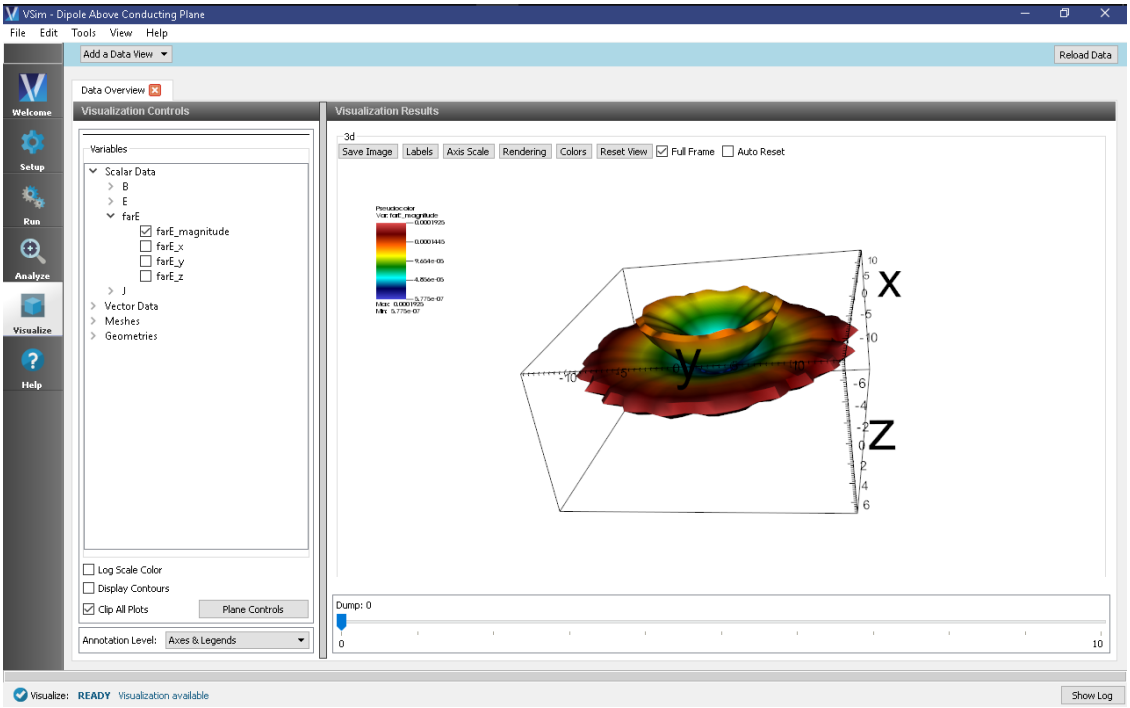


Fig. 3.30: The far field radiation pattern

3.1.7 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.

This simulation can be performed with a VSimEM, VSimMD or VSimPD license.

Opening the Simulation

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

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim 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. 3.31. 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`.

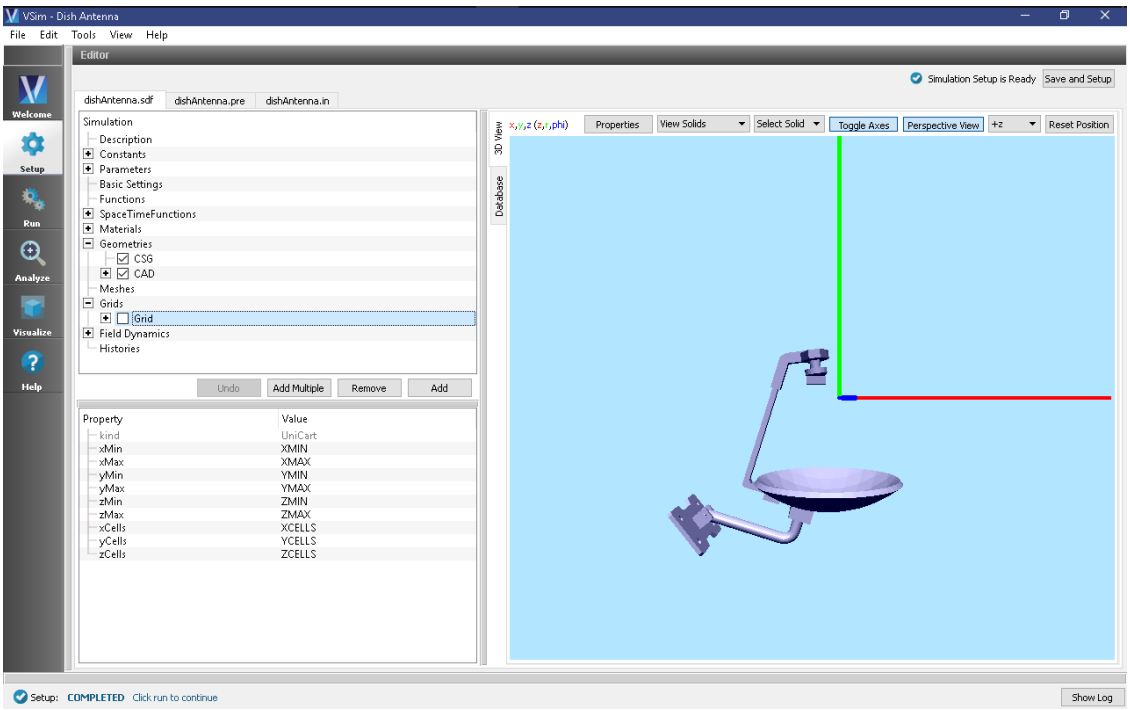


Fig. 3.31: Setup Window for the Dish Antenna example.

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.
- 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 the *Logs and Output Files* pane. The run has completed when you see the output, "Engine completed successfully." This is shown in Fig. 3.32.

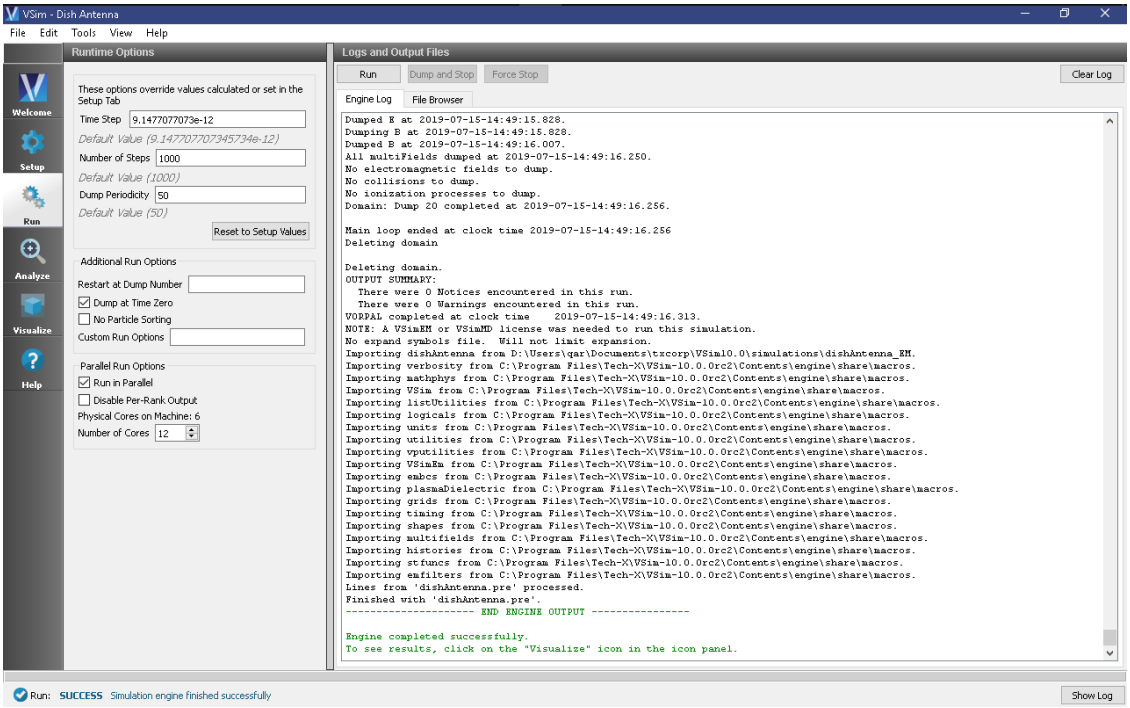


Fig. 3.32: The Run Window at the end of execution.

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. 3.33, do the following:

- Expand *Scalar Data*
- Expand *E*
- Select *E_x*
- Expand *Geometries*
- Select *poly (dishAntennaPecShapes)*

- Select *Clip All Plots*

It is easier to see the fields if you change the color scale minimum and maximum. To do so, click on the *Colors* button, 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.

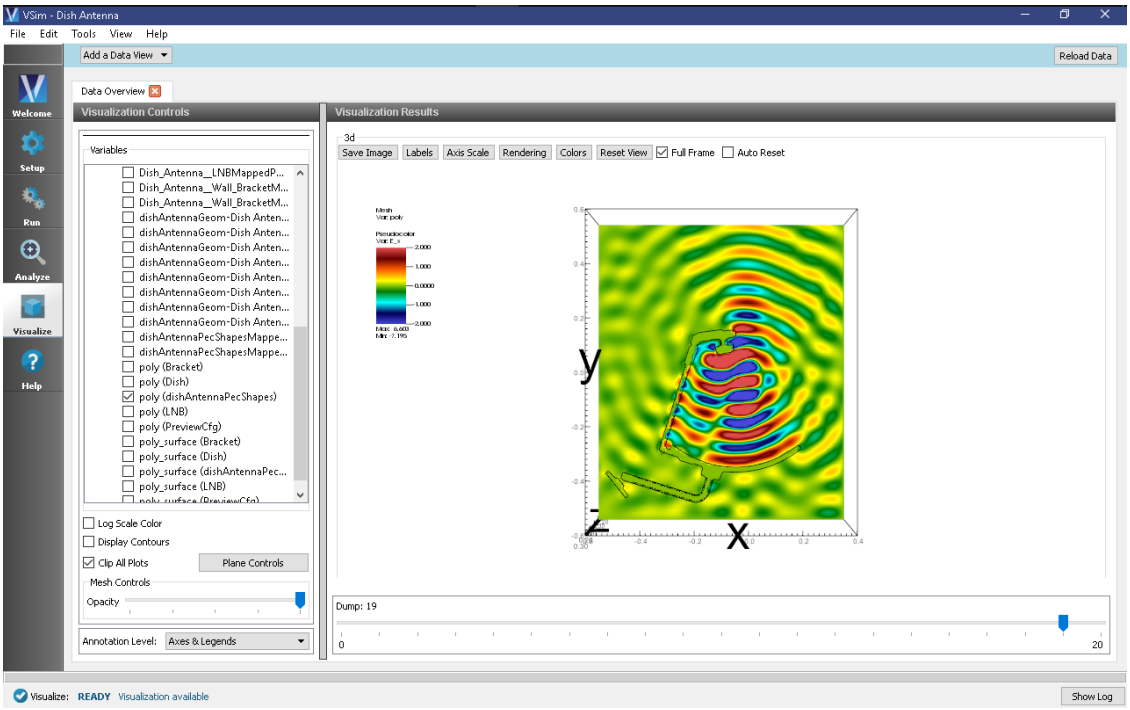


Fig. 3.33: Visualization of a slice of the electric field as a color contour plot at dump 19.

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.

3.1.8 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 VSim simulation data. The simulation itself consists of a half-wavelength long current source in free space.

This simulation can be performed with a VSimEM license.

Opening the Simulation

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

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim 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. 3.34. 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`.

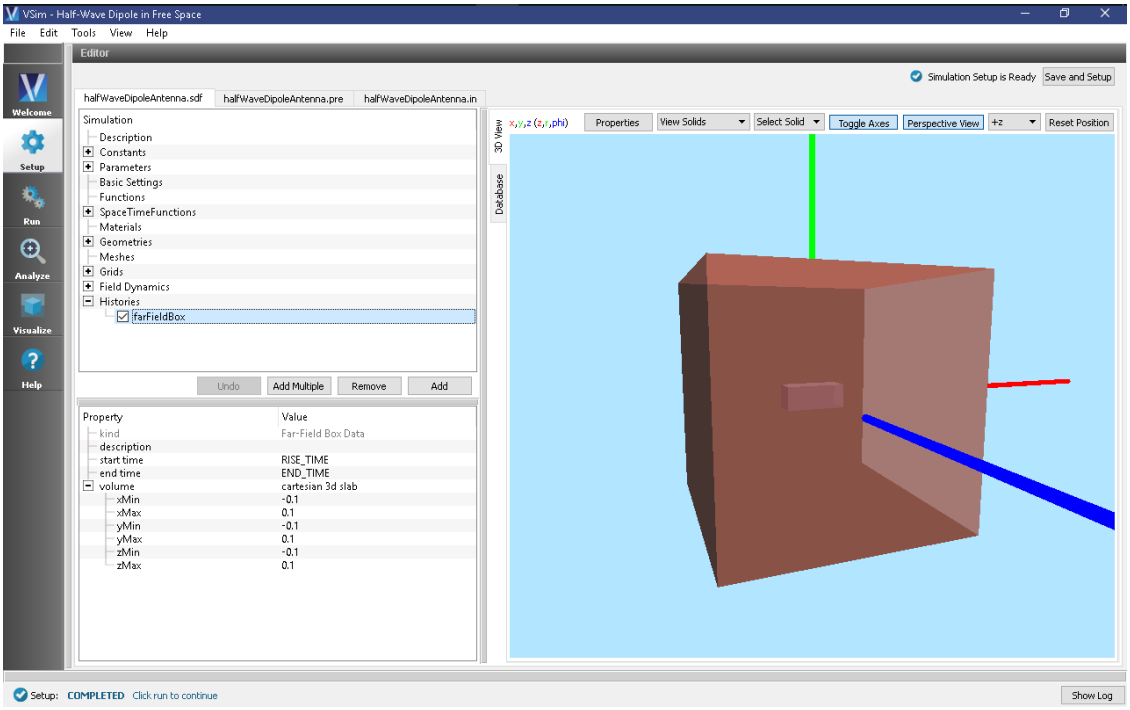


Fig. 3.34: Setup Window for the Half Wave Dipole Antenna example.

Simulation Properties

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

- WAVELENGTH: The wavelength of the antenna
- 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.
- To run the file, click on the *Run* button in the upper left corner of the right panel. 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 is shown in Fig. 3.35.
- NOTE: the correct elements will not appear in the visualization step if the analysis step has not been performed first.

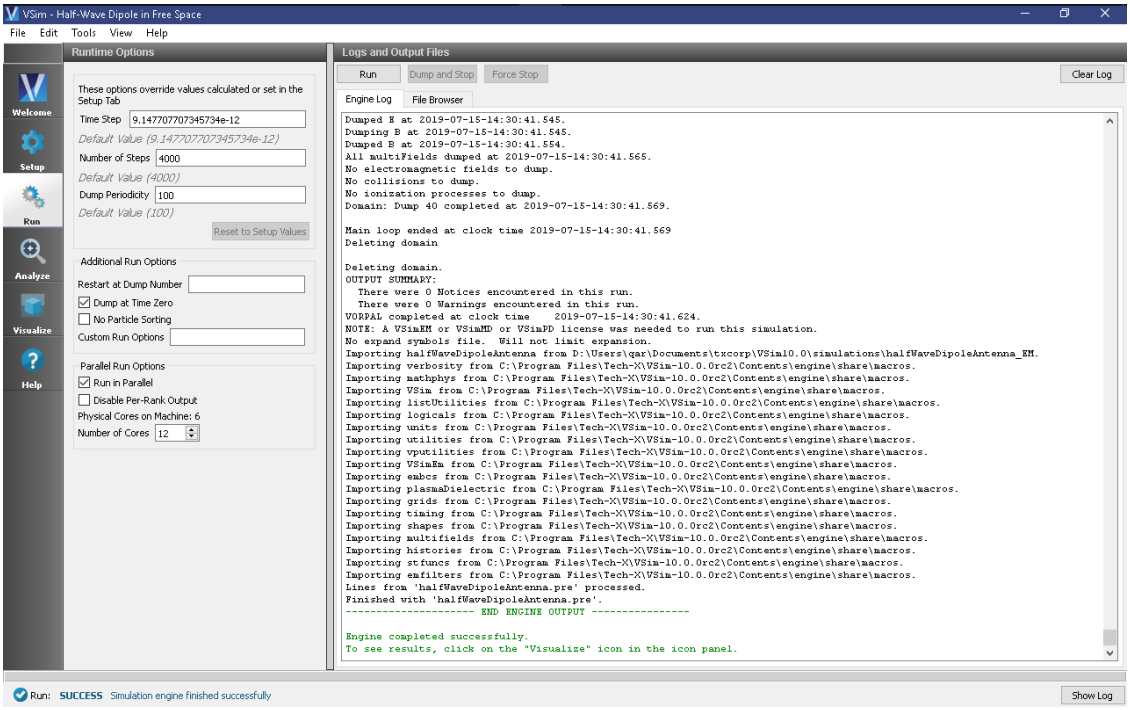


Fig. 3.35: The Run Window at the end of execution.

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 (the rest should remain at their default setting):
  - simulationName - halfWaveDipoleAntenna (name of the input file)
  - fieldLabel - E (name of the electromagnetic field)
  - farFieldRadius - 0.05 (radius of the far sphere, i.e., distance to the far zone)
  - numTheta - 45 (number of points in the theta direction)
  - numPhi - 90 (number of points in the phi direction)
- Click the *Analyze* button in the upper right corner.

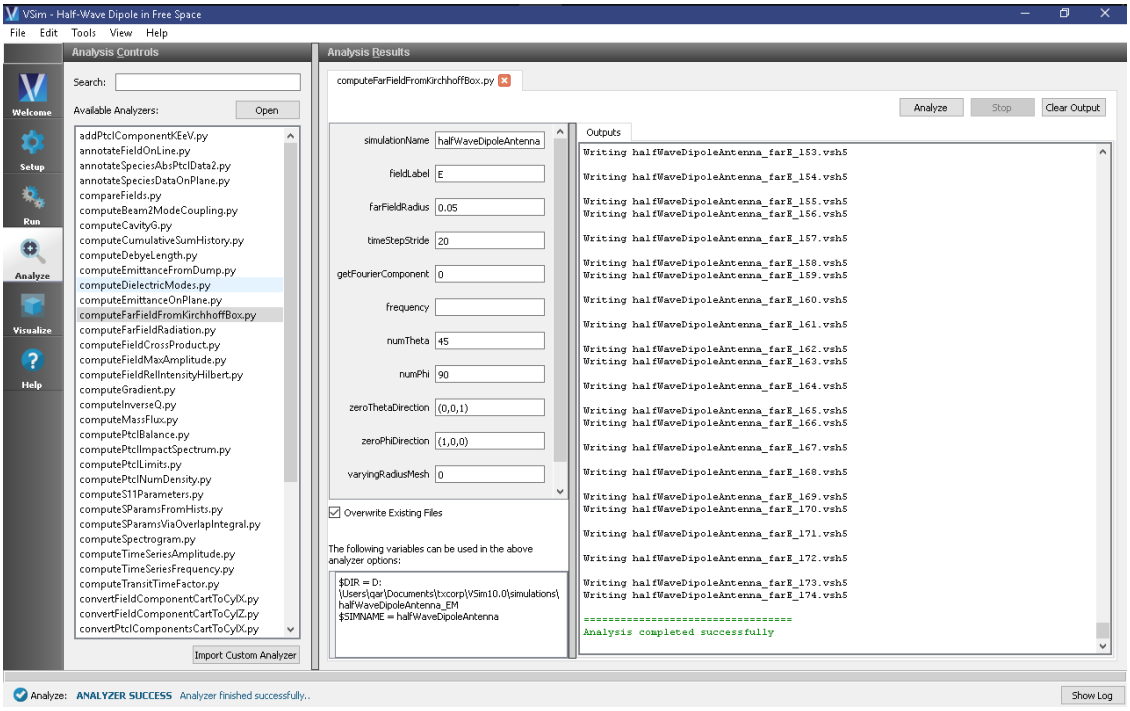


Fig. 3.36: The Analysis window at the end of execution.

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

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.

3.1.9 Horn Antenna (hornAntenna.sdf)

Keywords:

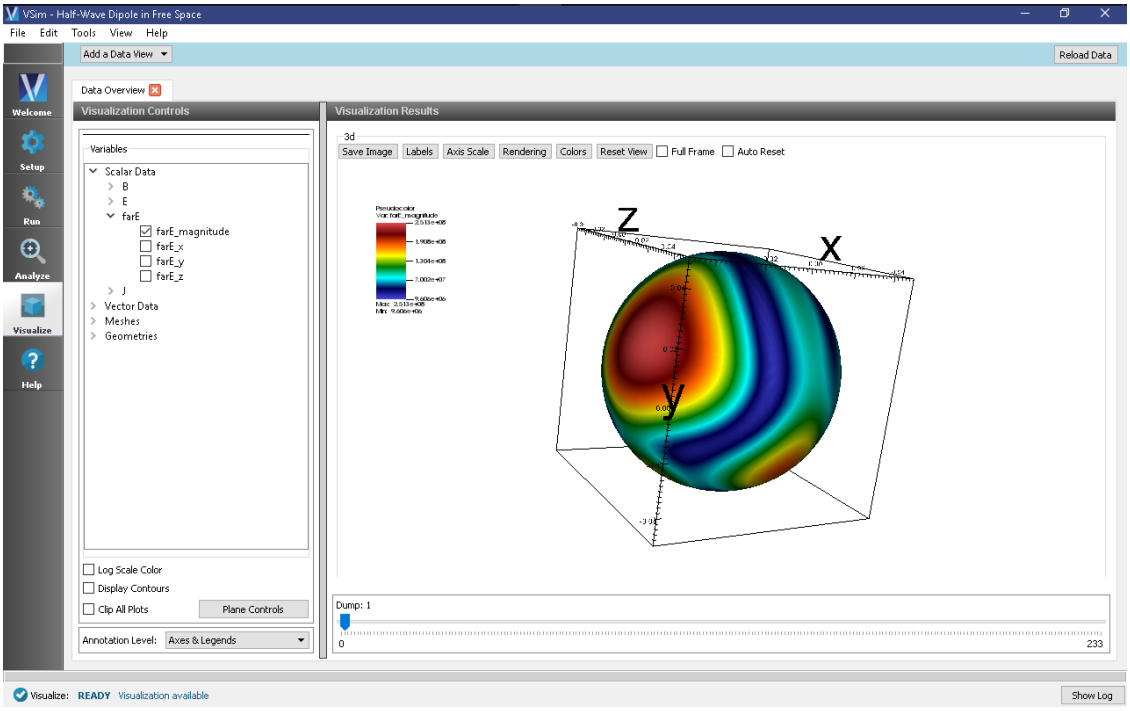


Fig. 3.37: The far field radiation pattern

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.

This simulation can be run with a VSimEM license.

Opening the Simulation

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

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim 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. 3.38. 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.

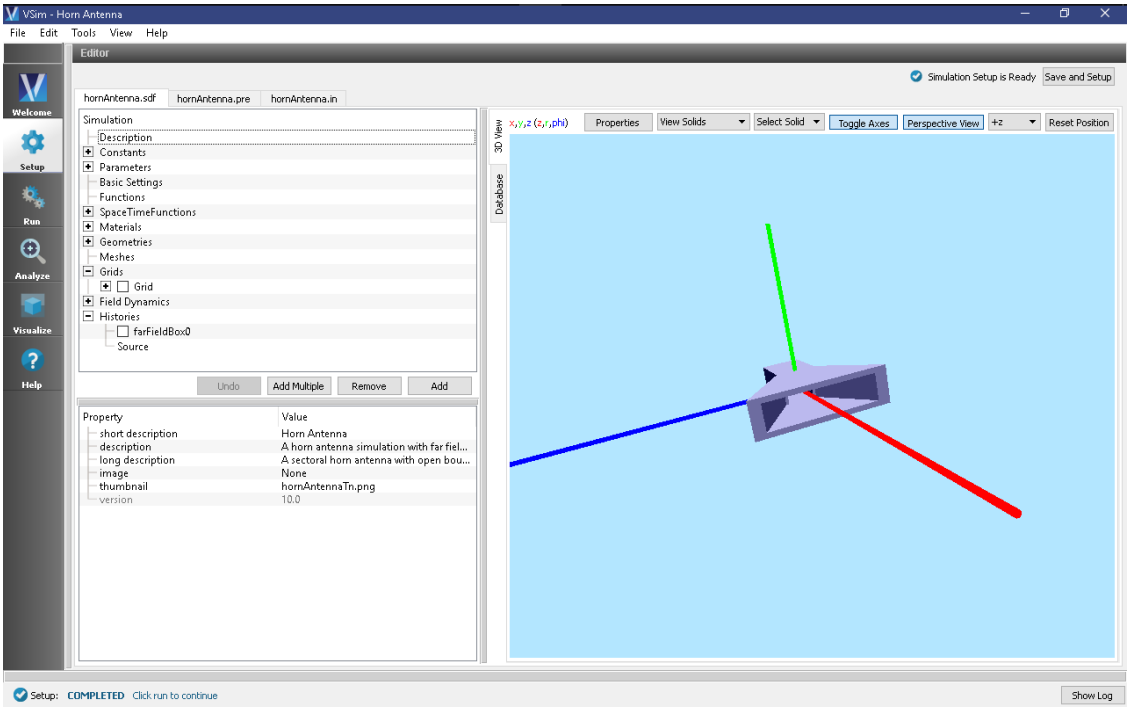


Fig. 3.38: Setup Window for the Horn Antenna example.

Simulation Properties

The antenna geometry in this example has been setup 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 setup to resolve both scales.

Running the Simulation

Once finished with the problem setup, continue as follows:

- Proceed to the Run Window by pressing the **Run** button in the left column of buttons.
- Choose parallel computing options on the MPI 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 that pane. The run has completed when you see the output, "Engine completed successfully." This is shown in the window below.

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.
- Click 'Show All Analyzers'

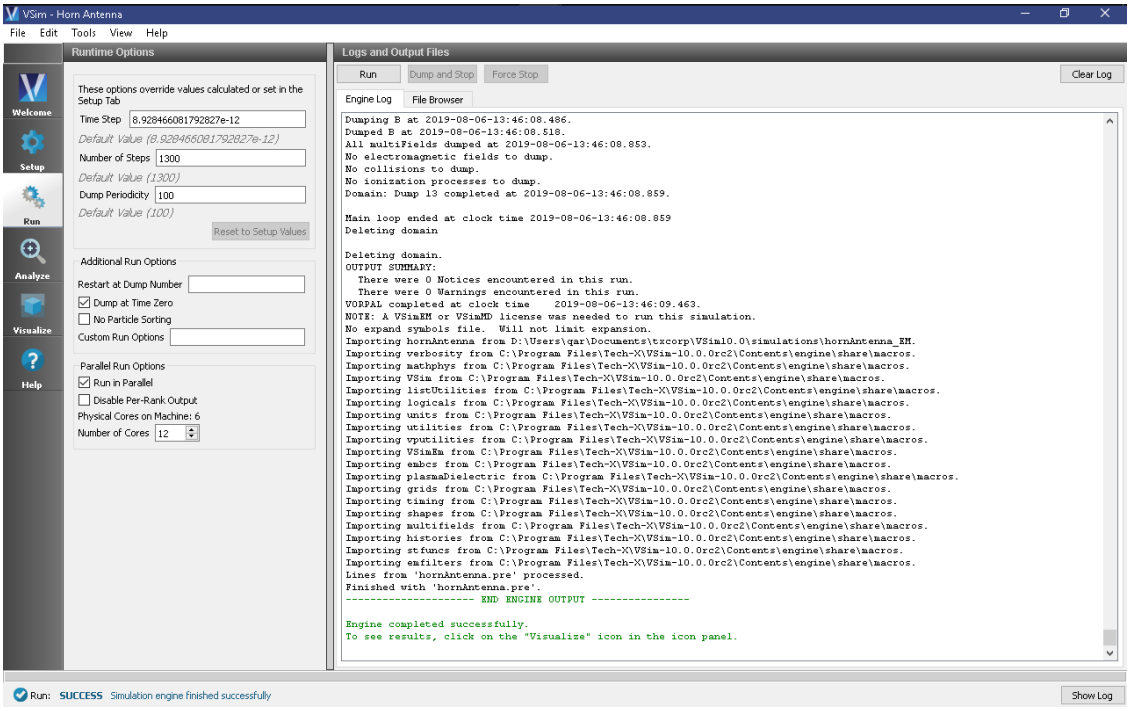


Fig. 3.39: The Run Window at the end of execution.

- In the resulting dialog, select computeFarFieldFromKirchhoffBox.py (Fig. 3.40) 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)
  - timeStepStride - 20 (number of timesteps between far field calculations; determines how many far fields are output; 20 steps should yield 4 far fields in this case)
  - getFourierComponent - [0 (default) use] [1 do not use] fourier analysis for a particular frequency
  - frequency - the frequency to use in the fourier analysis (if 'getFourierComponent' is not 0)
  - numTheta - 18 (number of theta points in the far field, 18 for a quick calculation, 45 for finer resolution)
  - numPhi - 36 (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
  - varyingRadiusMesh - 1 (Set to 1 in order to make far field mesh adapt to magnitude of far field solution: the classic lobe view)
  - simpsonIntegration - 0 (Set to 0 for more accurate integration)
- 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.

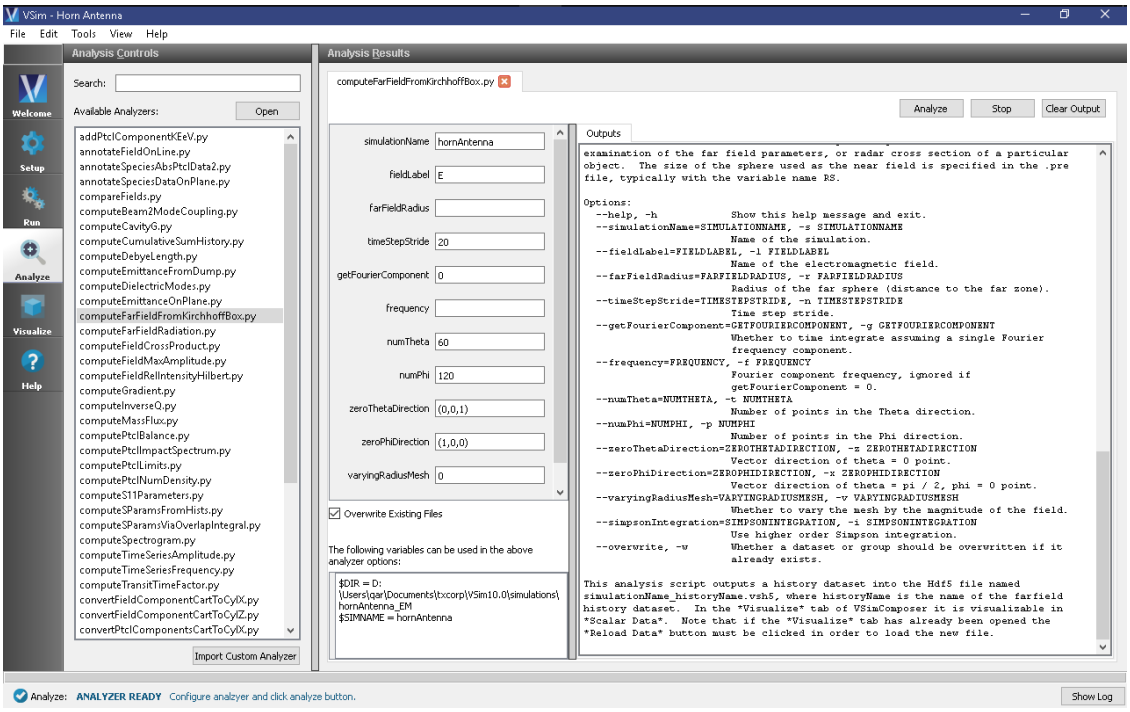


Fig. 3.40: Add the computeFarFieldFromKirhhoffBox.py script to your simulation.

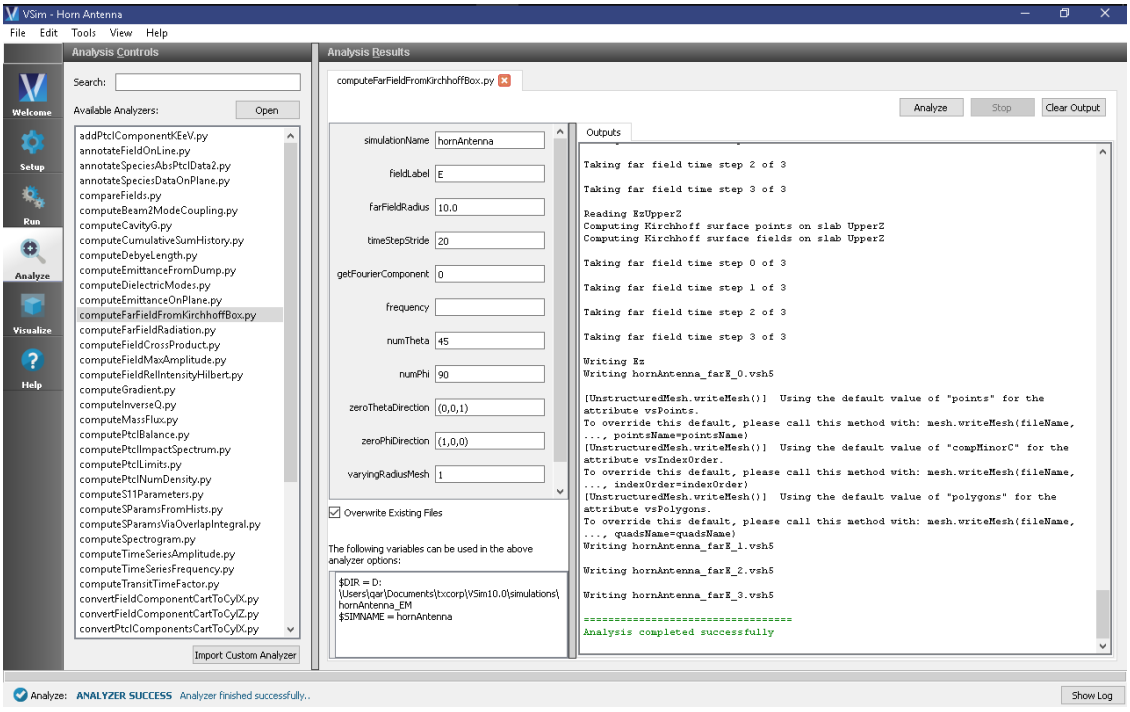


Fig. 3.41: The **Analyze** panel after running computeFarFieldFromKirhhoffBox.py.

Visualizing the Results

Under *Scalar Data* plot $E_{magnitude}$. To slice inside the horn, select *Clip All Plots* 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 and then click the *Colors* button above the 3D view. Try adjusting the min and max until the signal is well resolved (see `_hornantennavizwin1`).

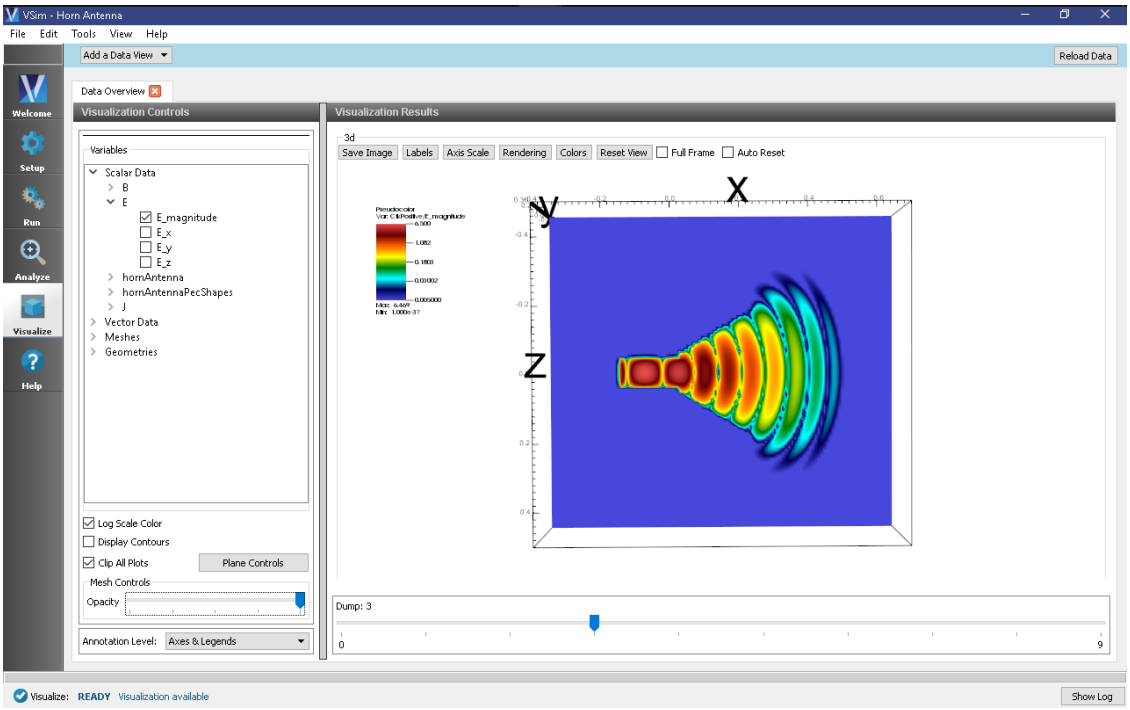


Fig. 3.42: The $E_{magnitude}$ field propagating out of the horn at dump 3. The color scale has been log scaled and the min and max have been fixed to 0.005 and 6.5, respectively. The optimal min/max values will depend on the dump selected. The view has been rotated to show the x-z plane.

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*.

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.

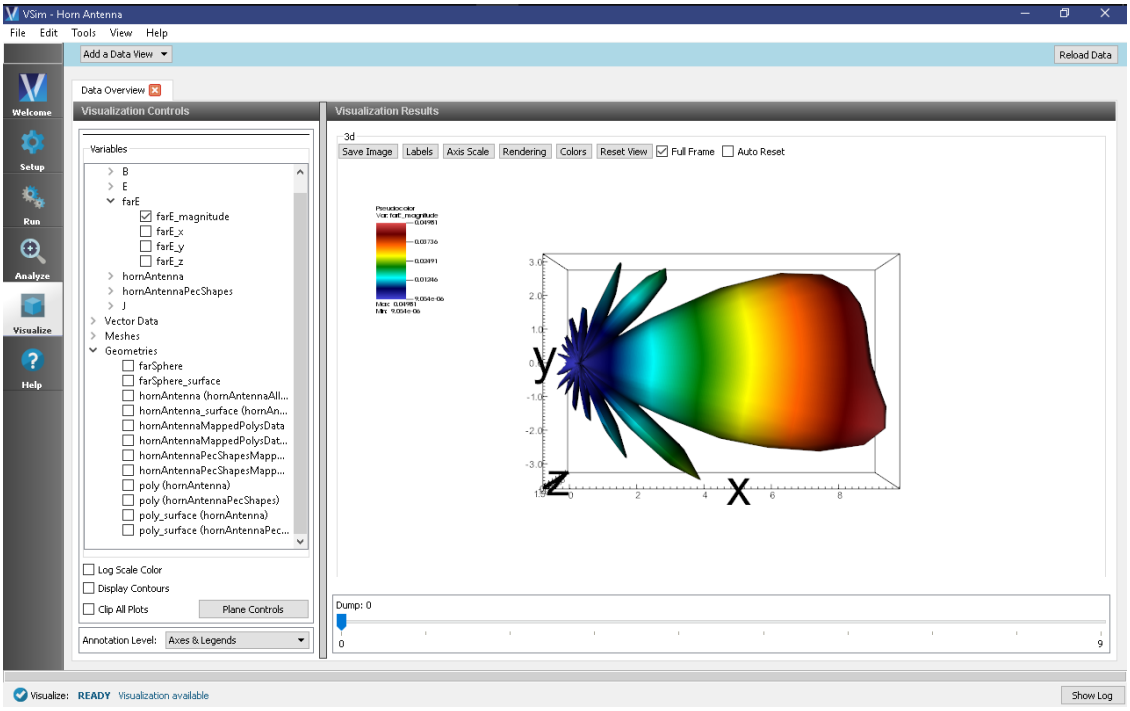


Fig. 3.43: The far field radiation pattern.

3.1.10 Patch Antenna with Far Fields (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.

This simulation can be performed with a VSimEM license.

Opening the Simulation

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

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim 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. 3.44.

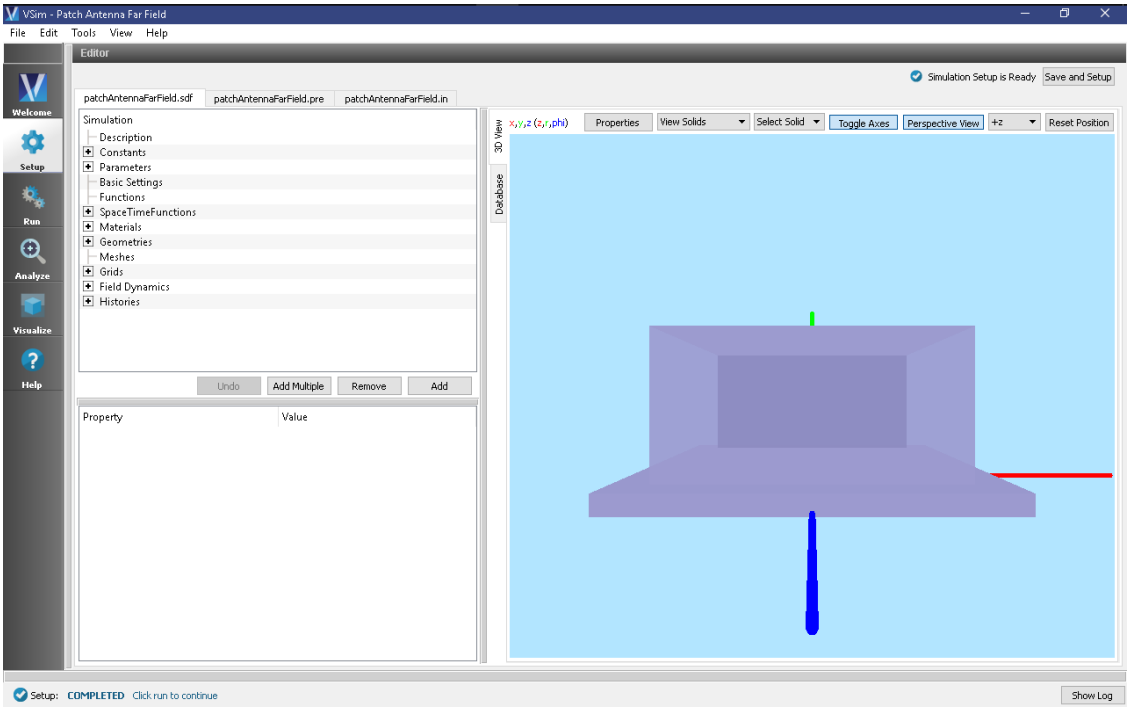


Fig. 3.44: Setup Window for the Patch Antenna example.

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.
- 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 when you see the output, "Engine completed successfully." This is shown in the window below.

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. 3.46)
- Input values for the analyzer parameters. The analyzer may be run multiple times, allowing the user to experiment with different values.
  - simulationName - patchAntennaFarField (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)

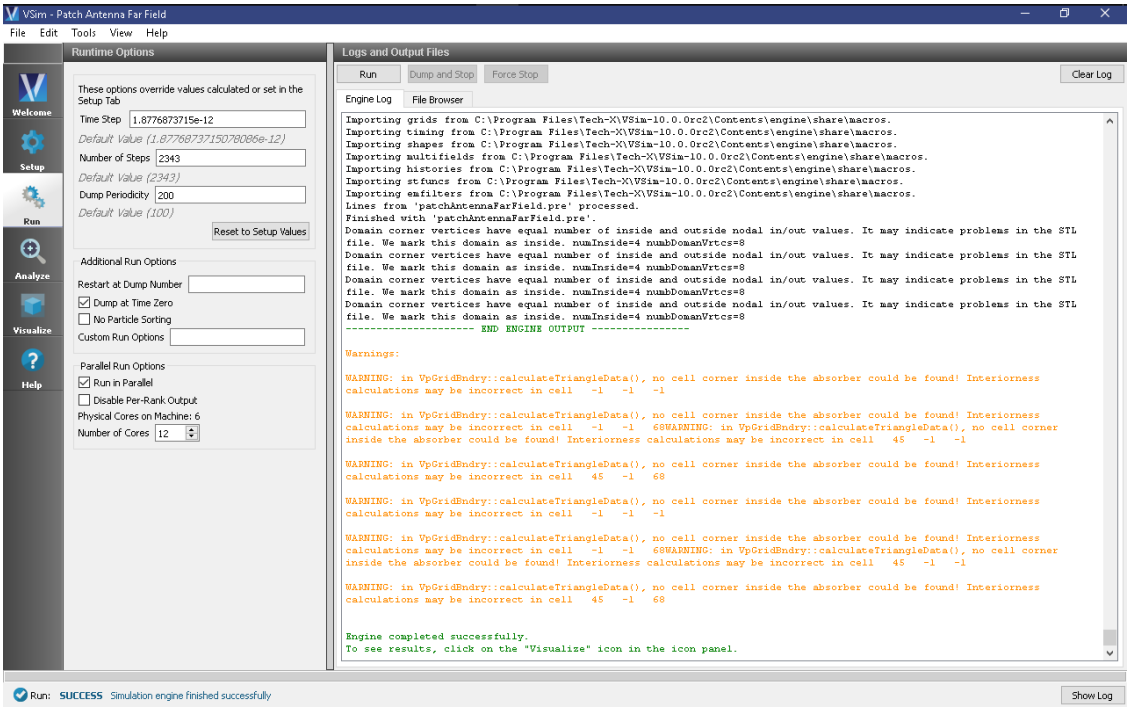


Fig. 3.45: The Run Window at the end of execution.

- timeStepStride - 80 (number of timesteps between far field calculations; determines the number of outputted farfields)
- getFourierComponent - 0, do not integrate assuming single fourier frequency
- frequency - not used because getFourierComponent is false
- numTheta - 60 (number of theta points in the far field, 30 for a quick calculation, 60 for finer resolution)
- numPhi - 120 (number of phi points in the far field, 60 for a quick calculation, 120 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
- varyingRadiusMesh - 0 (Set to 1 in order to make far field mesh adapt to magnitude of far field solution: the classic lobe view)

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*.

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.

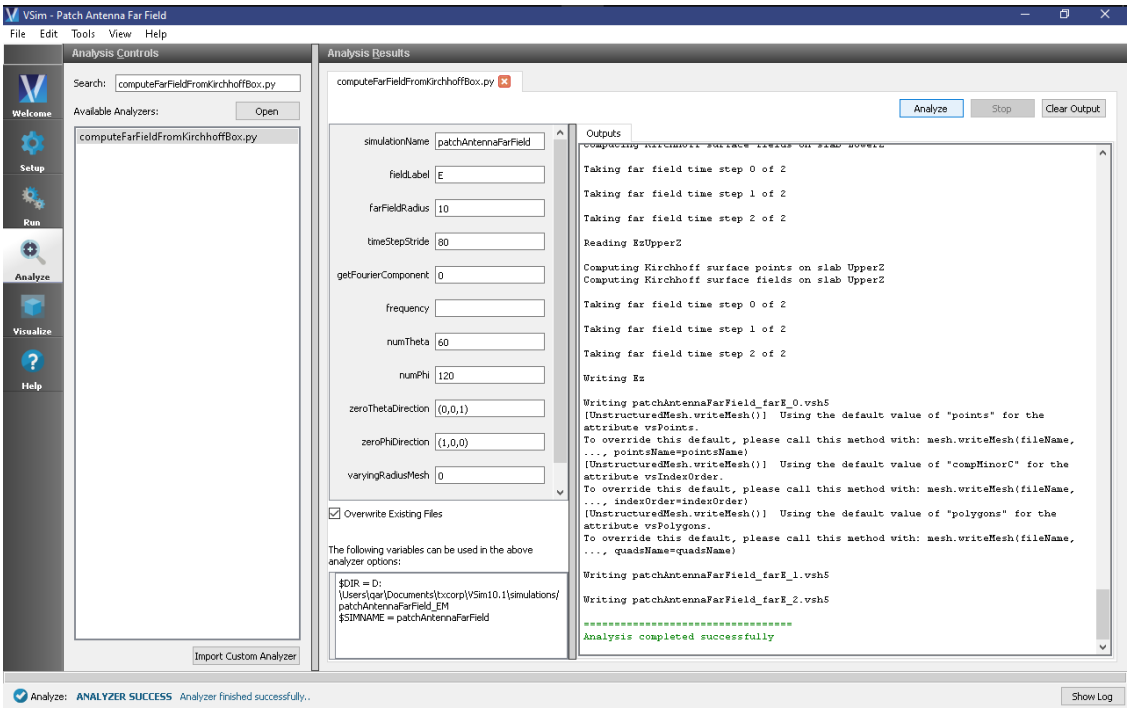


Fig. 3.46: Add the computeFarFieldFromKirhhoffBox.py script to your simulation.

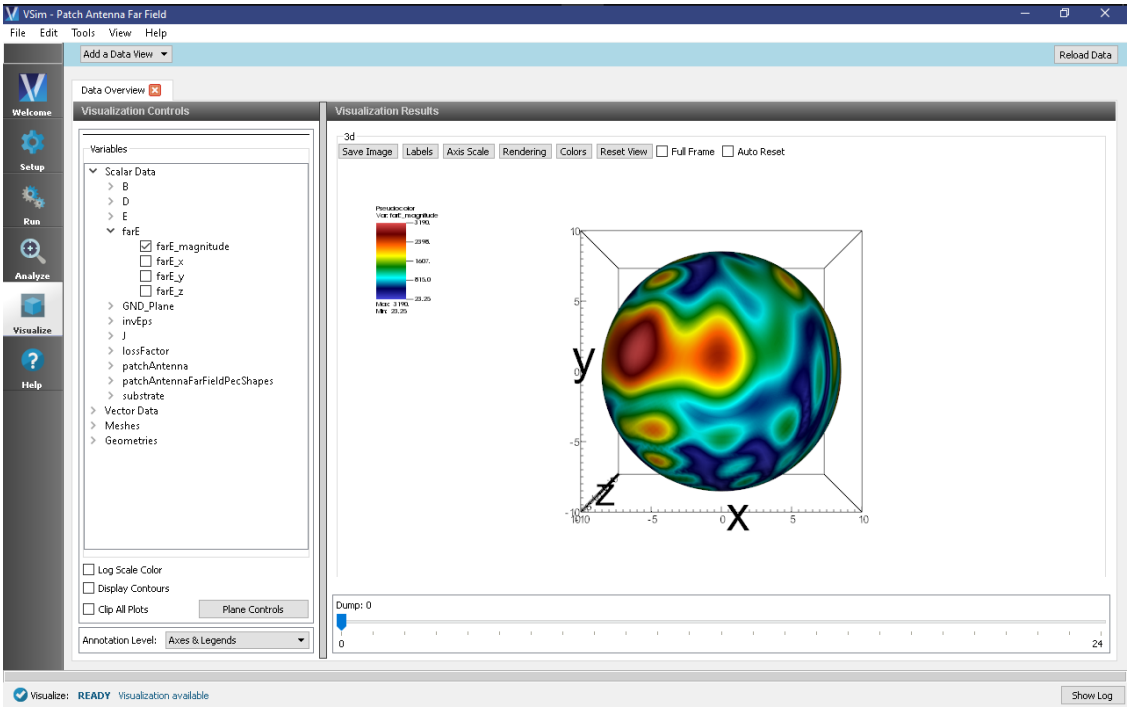


Fig. 3.47: The Far Field Radiation Pattern

3.1.11 Phased Array (phasedArrayAntenna.sdf)

Keywords:

keywordone, keyword two, keywordthree, keyword four

Problem Description

This VSimEM example illustrates how to setup a phased array simulation and analyze the far field results. Phased array antennas are a vastly-expanding field of research and development due to the fact that going from a one element antenna to an N-element antenna provides more directive beamforming characteristics and, most importantly, non-mechanical steering. Creating a multiple-element antenna results in an array pattern composed of wires, apertures, or other element types. Directive patterns are obtained via constructive interference in the desired direction and destructive in the other directions. Applications of phased array antennas range from commercial (5G, wireless & mobile, satellite telecommunication), military & defense (RADAR, acoustics) to research: atmospheric, space.

This simulation can be run with a VSimEM license.

Opening the Simulation

The phasedArrayAntenna example is accessed from within VSimComposer by the following actions:

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim for Electromagnetics* option.
- Expand the *Antennas* option.
- Select *Phased Array* 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. 3.48.

Simulation Properties

This example consist of 15×15 array of small metal antenna elements excited by a distributed current source. The separation between the elements is $\frac{\lambda}{6}$.

Note that the SPACING parameter CANNOT be used in the CSG array setup settings. To recreate an array with a different spacing between elements, the spacing needs to first be calculated and then typed into the array setup window.

The current source is results in an outging wave centered at $(x, y) = (0, 0)$. The wave amplitude has a Gaussian profile with the standard deviation σ. The phase and amplitude are tied to the azimuthal angle θ, and polar angle ϕ. In this example, the azimuthal angle θ is fixed at $\pi/4$ via the Function dphiFunc. The polar angle ϕ goes from 0 to 2π throughout the simulation via the Function thetaFunc.

The simulation domain contains a far field box history that is later used for the computeFarFieldFromKirchhoffBox analyser.

The current excitation formula, $F(x, y, \phi, \theta, t)$, is:

$$F(x, y, \phi, \theta, t) = A(x, y, \phi, \theta) \times \sin \Phi(x, y, \phi, \theta) \times X(x) \times Y(y)$$

Where the aplitude $A(x, y, \phi, \theta)$, the phase $\Phi(x, y, \phi, \theta)$, and locations in the x-y plane $X(x)$ and $Y(y)$ are defined by:

$$A(x, y, \phi, \theta) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2\sigma}\left[\sin^2 \phi(x \sin\theta - y\cos\theta)^2 + \cos^2 \phi\left(x^2+y^2\right)\right]}$$

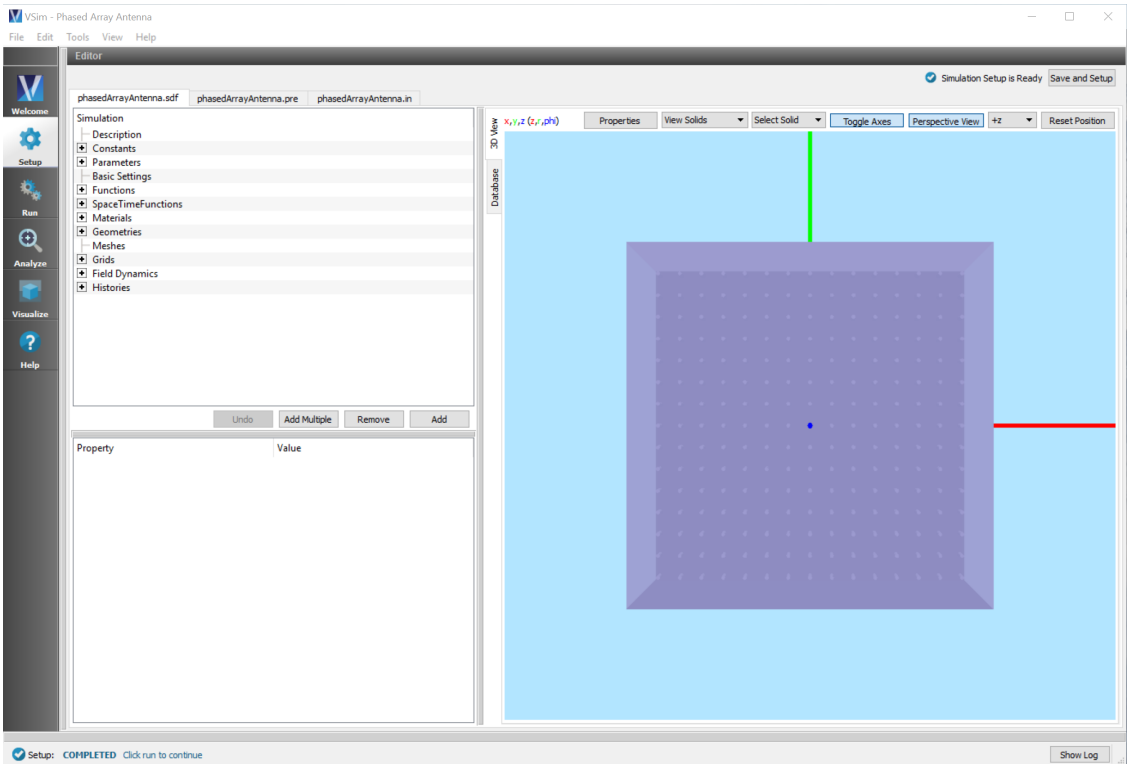


Fig. 3.48: Setup Window for the Phased Array example.

$$\Phi(x, y, \phi, \theta) = \omega t + \frac{\omega}{c} \sin\phi \left(x \cos\theta + y \sin\theta\right)$$

$$X(x) = H\left[\cos\frac{2\pi x}{d} - 0.9\right]$$

$$Y(y) = H\left[\cos\frac{2\pi y}{d} - 0.9\right]$$

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." This is shown in Fig. 3.49.

Analyzing the Results

After performing the above actions, for a quick rough estimation of the far fields, continue as follows:

- Proceed to the Analysis Window by pressing the *Analyze* button in the navigation column.
- Click *Show All Analyzers*
- In the resulting list, select *computeFarFieldFromKirchhoffBox* and press *Open*
- The analyzer fields should be filled as below:

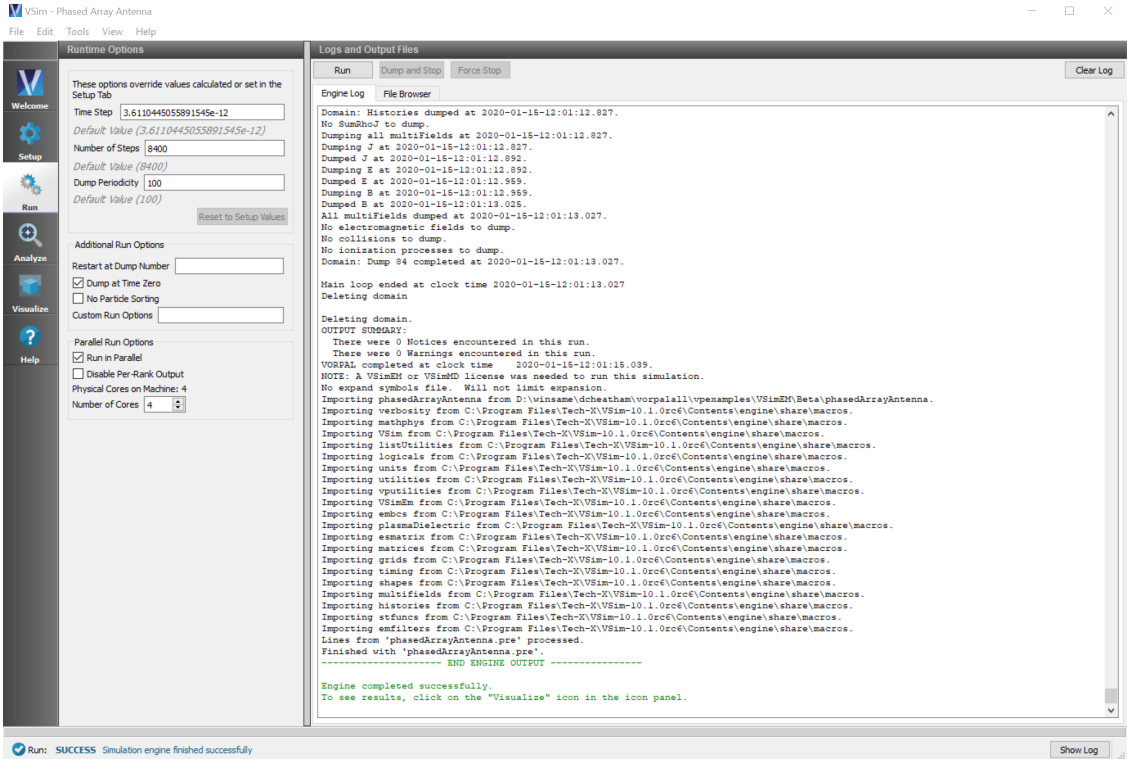


Fig. 3.49: The Run Window at the end of execution.

- *simulationName*: phasedArrayAntenna
- *fieldLabel*: E
- *farFieldRadius*: 30
- *timeStepStride*: 21
- *getFourierComponent*: 0
- *frequncy*: 1e9
- *numTheta*: 15
- *numPhi*: 30
- *zeroThetaDirection*: (0,0,1)
- *zeroPhiDirection*: (1,0,0)
- *varyingRadiusMesh*: 1
- *simpsonIntegration*: 0

Check the *Overwrite Existing Files* box.

- Click *Analyze* in the top right corner.
- The analysis is completed when you see the output shown in Fig. 3.50.

For more accurate results, use the following input paramters in the analyzer:

- *simulationName*: phasedArrayAntenna
- *fieldLabel*: E

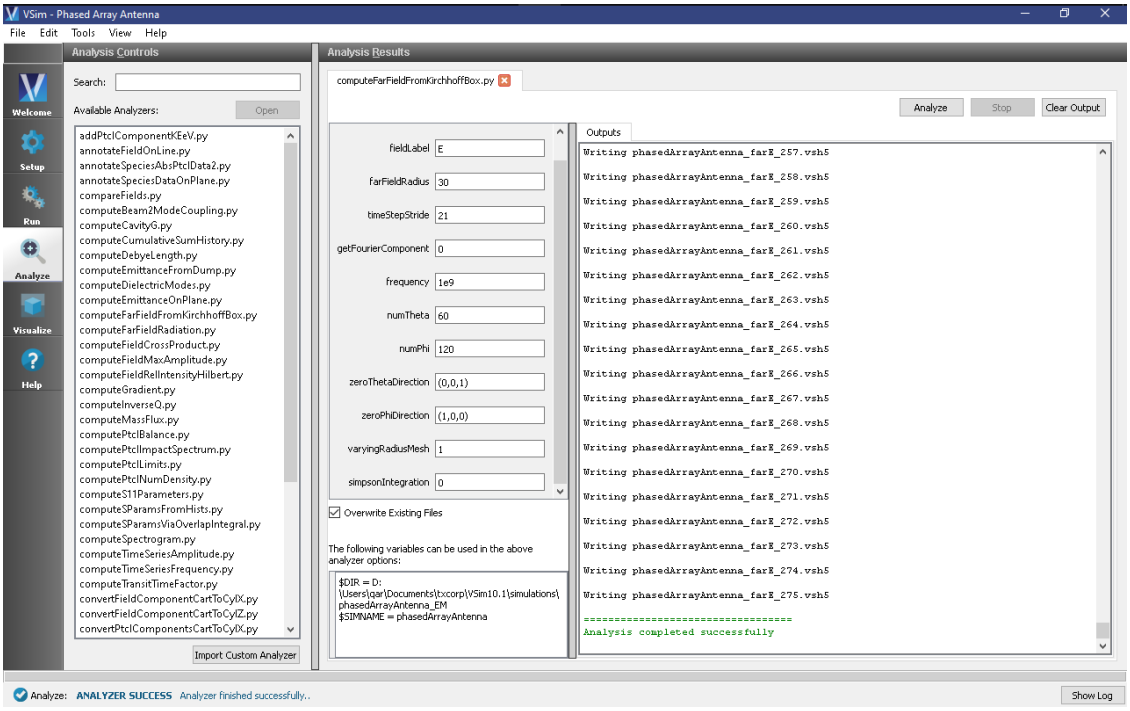


Fig. 3.50: The computeFarFieldFromKirchhoffBox at the end of a successful run.

- *farFieldRadius*: 30
- *timeStepStride*: 21
- *getFourierComponent*: 0
- *frequncy*: 1e9
- *numTheta*: 60
- *numPhi*: 120
- *zeroThetaDirection*: (0,0,1)
- *zeroPhiDirection*: (1,0,0)
- *varyingRadiusMesh*: 1
- *simpsonIntegration*: 0

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.
- From the *Data View* dropdown menu, select *Data Overview*.
- Expand *Scalar Data* variables.
- Expand *farE* and check the *farE_Magnitude* box.
- Move the dump slider to see the evolution of the far fields in time.

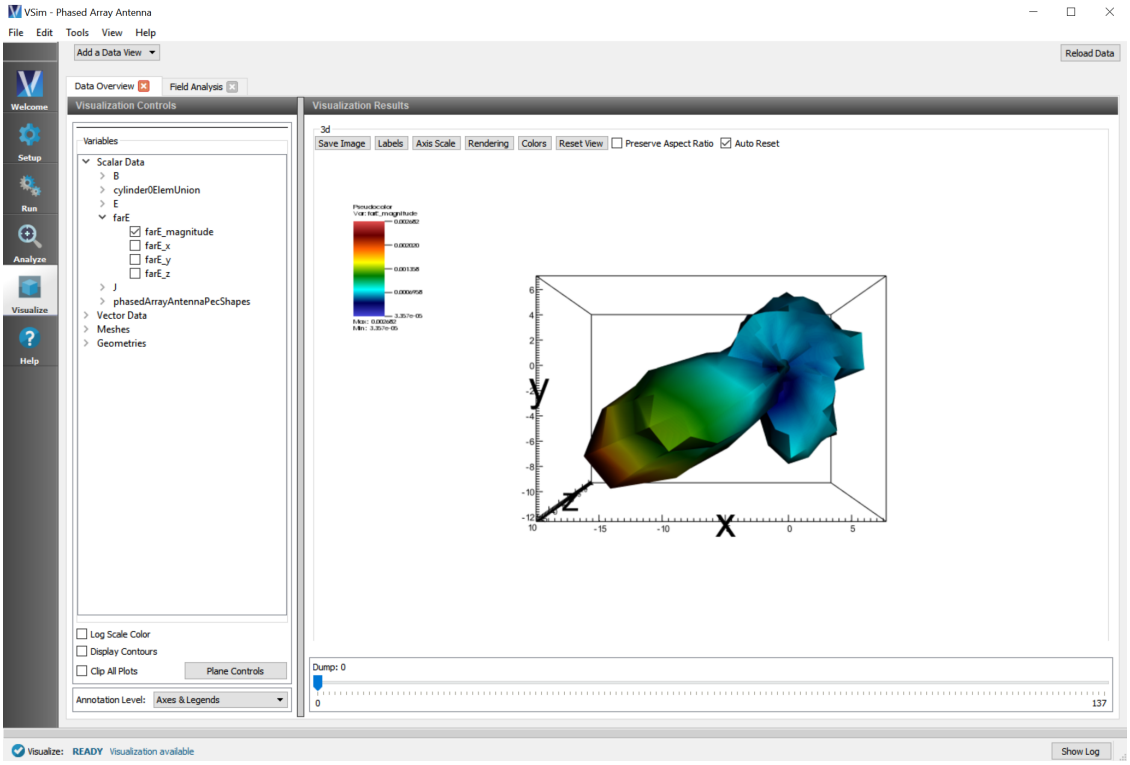


Fig. 3.51: Far field pattern 30 m away from the antenna.

The resulting visualization is shown in Fig. 3.51.

To visualize the 2D far fields proceed as follows:

- Proceed to the Visualize Window by pressing the *Visualize* button in the navigation column.
- From the *Data View* dropdown menu, select *Field Analysis*
- Under *Field* choose *farE_magnitude*
- Move the dump slider to see the evolution of the 3D far fields as well as the 2D cross-sections in time.

The resulting visualization is shown in Fig. 3.52.

This method of visualizing the far fields can be used for studying properties such as directivity, main and side lobe pattern, radiation strength, etc.

3.1.12 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.

This simulation can be performed with a VSimEM license.

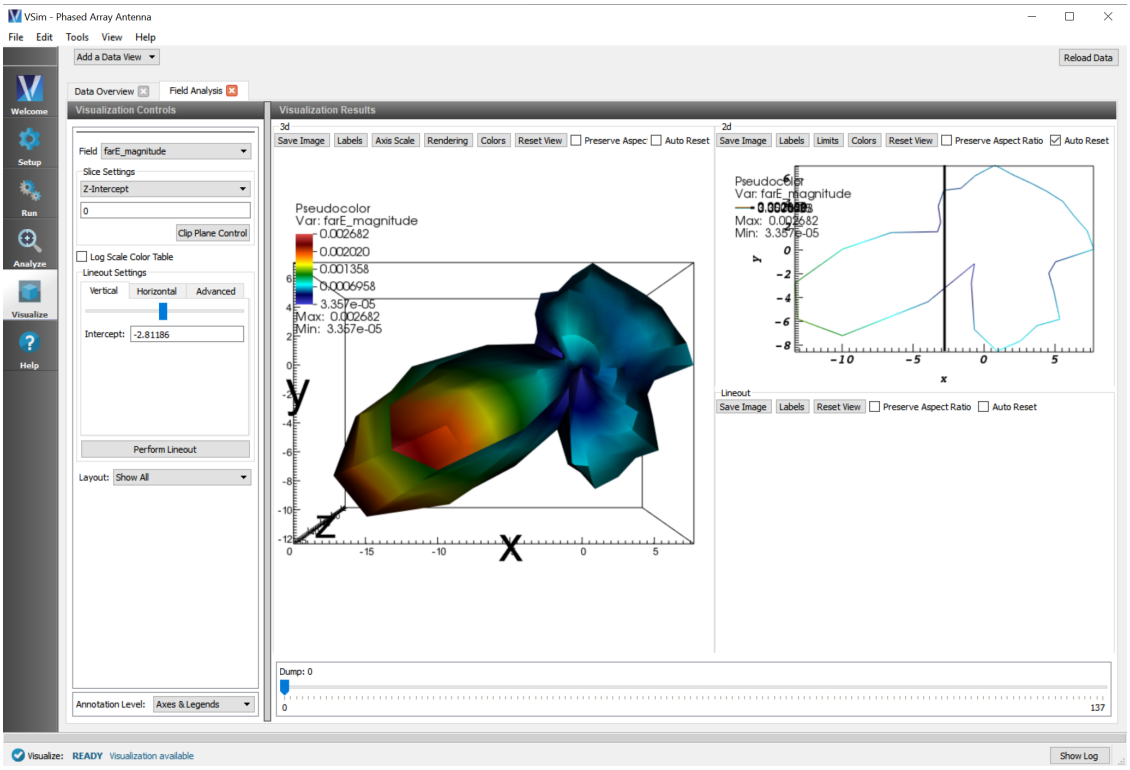


Fig. 3.52: 2D cross-section of the far field patern measured 30 m away from the source.

Opening the Simulation

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

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim 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. 3.53.

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:

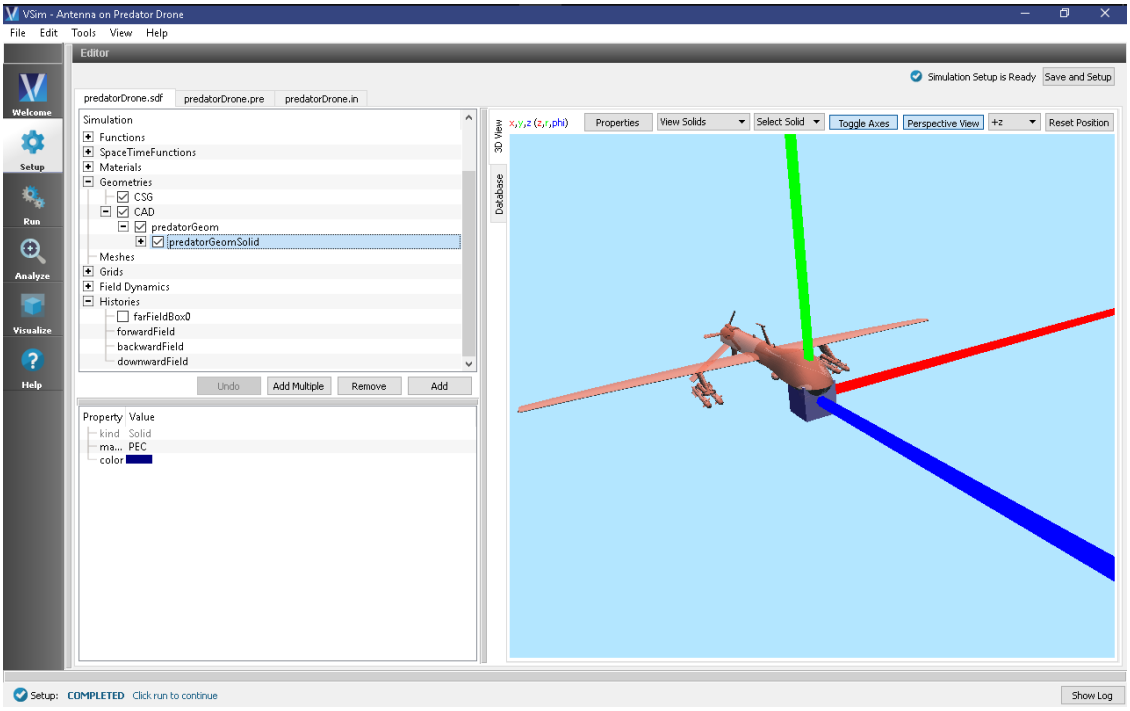


Fig. 3.53: Setup Window for the Predator Drone example.

- Proceed to the Run Window by pressing the *Run* button in the left column of buttons.
- 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 the right pane.
- The run has completed when you see the output, "Engine completed successfully." This is shown in Fig. 3.54.

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 the *Show All Analyzers* button, then click *computeFarFieldFromKirchhoffBox.py*, then click on the *Open* button as shown in Fig. 3.55.

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.

The default values should all be good, but one must set *farFieldRadius*. Since the box size is one, it is sufficient to choose 10.

The analysis script will compute the instantaneous far fields at the timeStepStride. You may also optionally compute the Fourier amplitude at a single frequency. As this is a single frequency excitation, at 1 GHz, the Fourier amplitude is indeed of interest, and so you should also set *getFourierComponent* to 1, and set *frequency* to 1.0e9.

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

- You will also usually want to choose *Overwrite Existing Files*.
- Hit the *Analyze* button in the top left of the window.

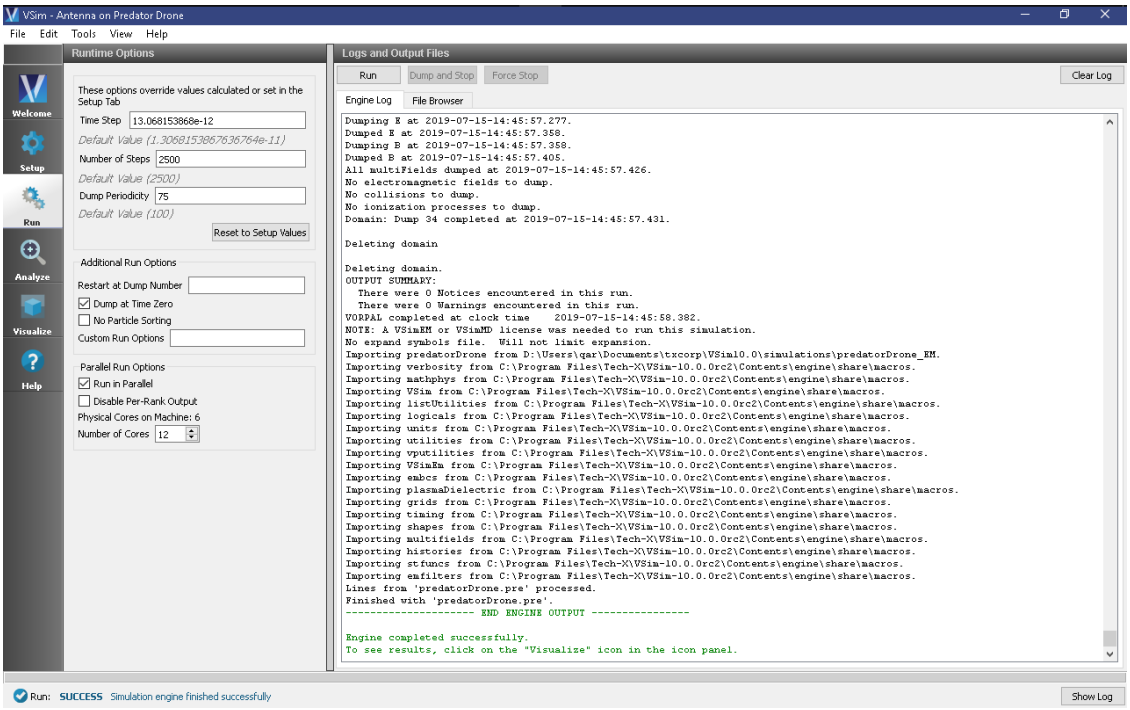


Fig. 3.54: The Run Window at the end of execution.

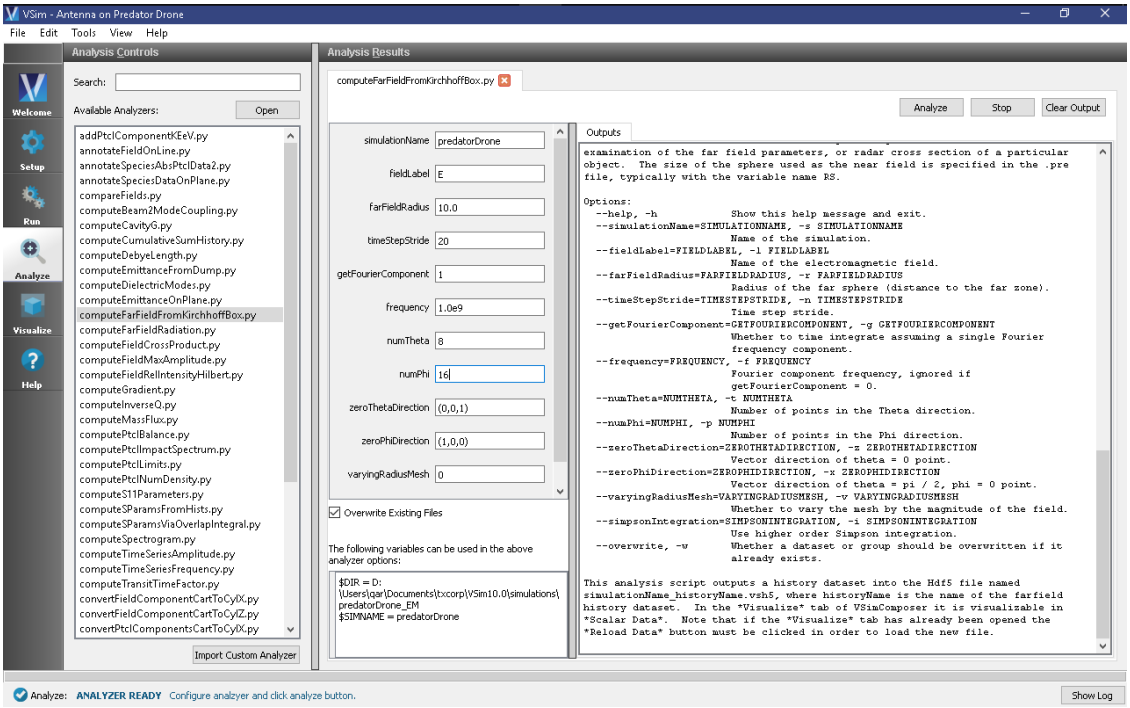


Fig. 3.55: Choosing the Kirchhoff Box Analyzer.

At completion, you will see Fig. 3.56. The far field data is written to vsh5 files in the simulation directory.

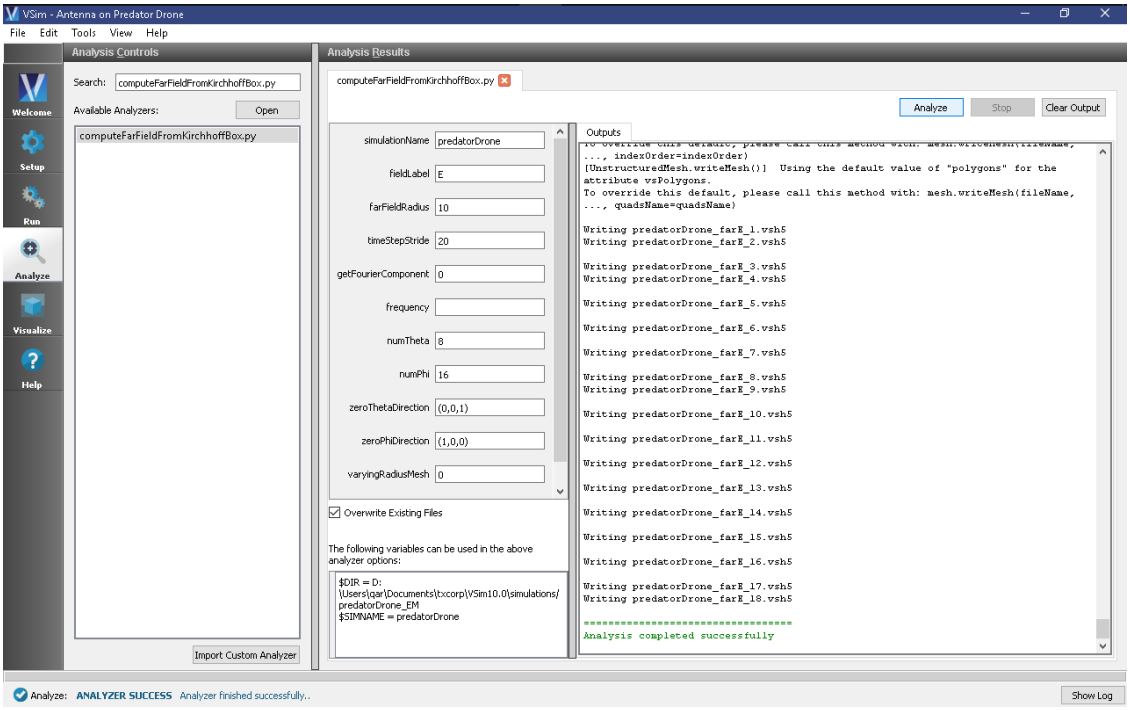


Fig. 3.56: The Analysis window at the end of execution.

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
- Move the Dump slider to see the far field pattern at different times.

It is useful to reset the color minimum and maximum using the Colors button and the number of Contours, to give a pleasing pattern, as shown in Fig. 3.57. 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 *PdBi* 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. 3.58.

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

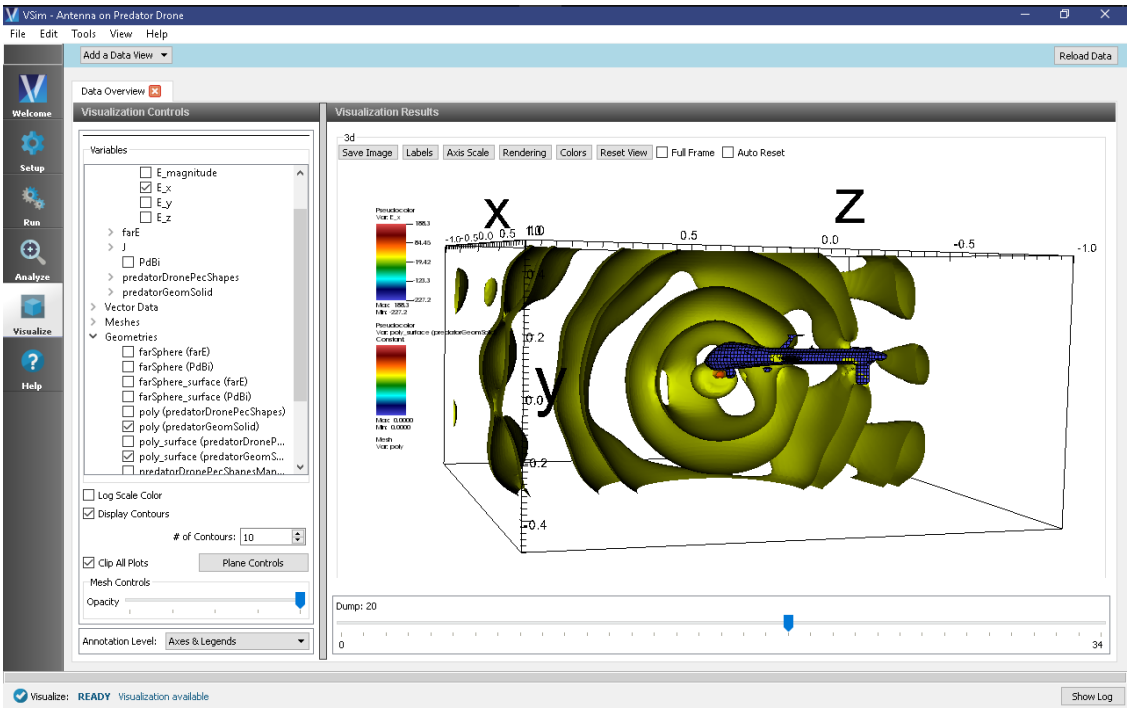


Fig. 3.57: The radiation pattern in real space

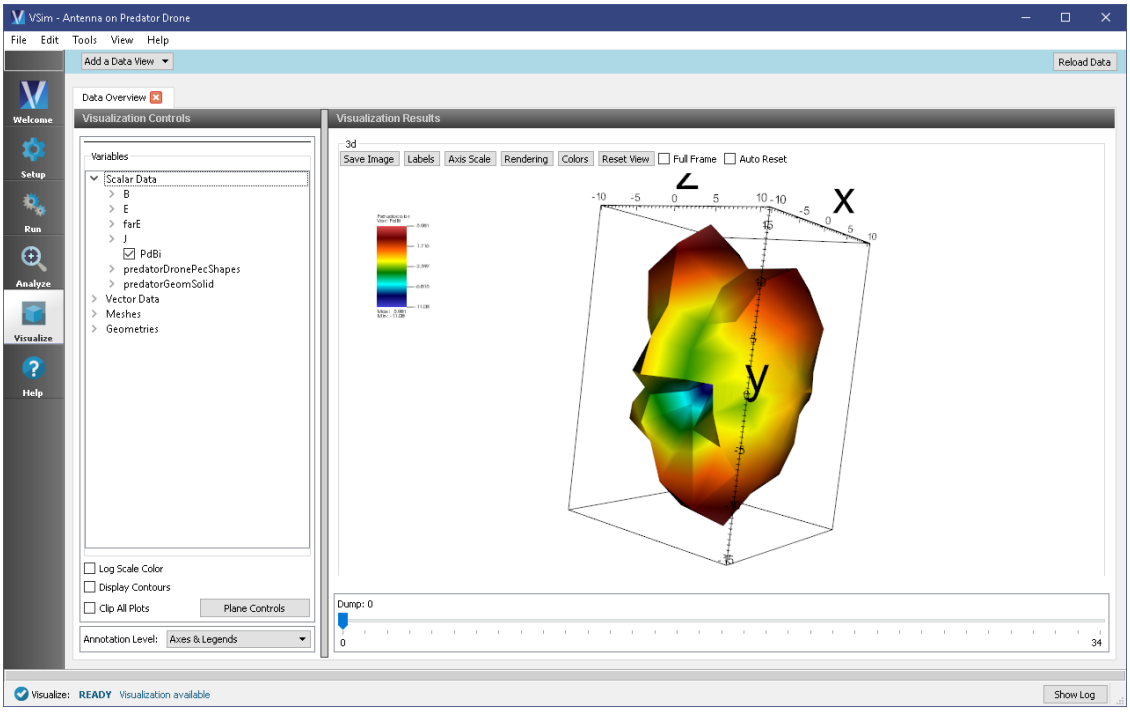


Fig. 3.58: The far field radiation pattern

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.

3.2 Electrostatics

3.2.1 Like-Charge Dipole (esChargedSpheres.sdf)

Keywords:

`electrostatics, like-charge dipole`

Problem description

This Like-Charge Dipole simulation computes the electrostatic potential and field for a dipole of two spheres with given radius at the same potential.

This simulation can be performed with a VSimEM or VSimPD license, with Composer licensed for Visual Setup.

Opening the Simulation

The Like-Charge Dipole example is accessed from within VSimComposer by the following actions:

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim for Electromagnetics* option.
- Expand the *Electrostatics* option.
- Select *Like-Charge Dipole* 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. 3.59. 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`.

Simulation Properties

This simulation uses visual setup to create a simple dipole. To do this we employ a few simple techniques such as Constructive Solid Geometries (CSG), and field Boundary Conditions. The dipole is constructed as two spheres of identical size. A Dirichlet boundary condition with the desired voltage is applied on both spheres.

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.

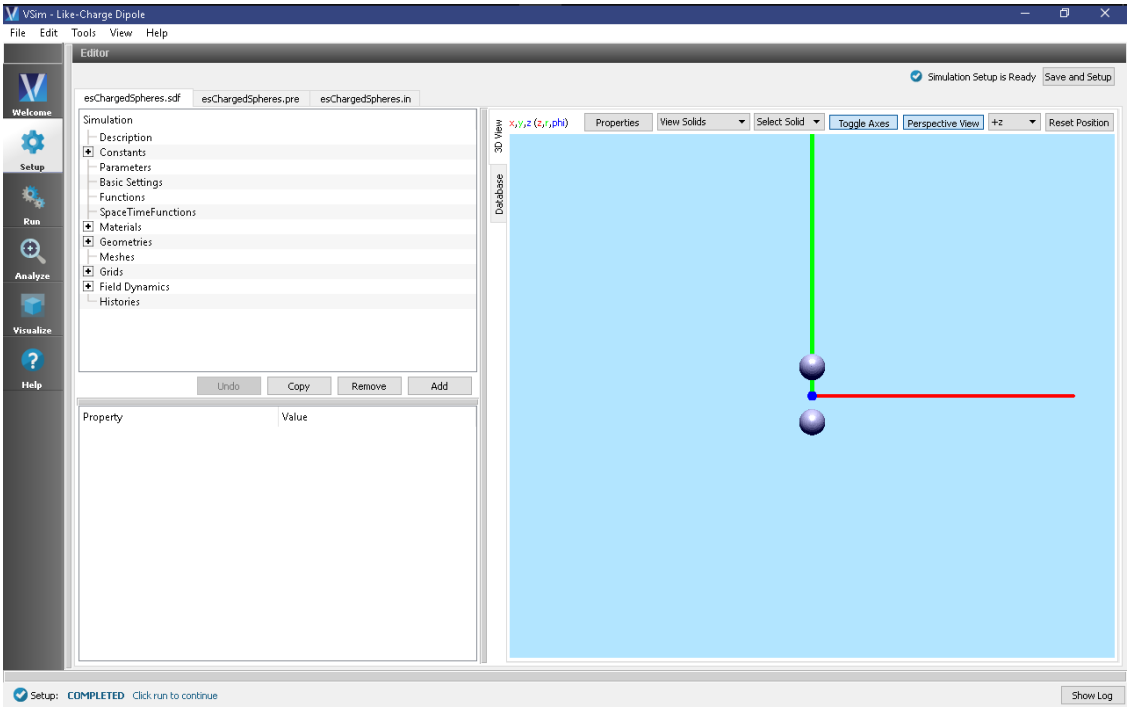


Fig. 3.59: Setup Window for the Like-Charge Dipole example.

- 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." This is shown in Fig. 3.60.

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 electrostatic potential of the dipole as seen in Fig. 3.61 do the following:

- Expand *Scalar Data*
- Select *Phi*
- Select *Display Contours*
- Select *Clip All Plots*

Further Experiments

Change the distance between spheres and see how the electric field changes.

Change the sphere radius and see how the electric field changes.

Change the sphere's potential to observe a change in the electric field.

Determine how the electric field changes with varying resolution.

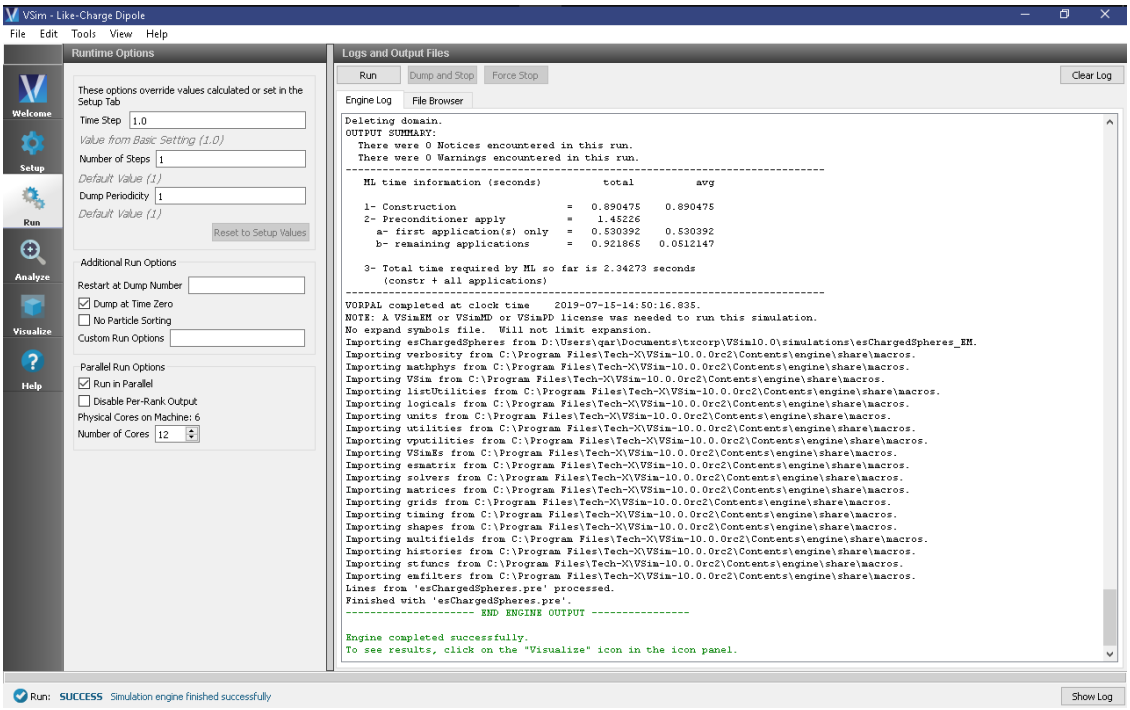


Fig. 3.60: The Run Window at the end of execution.

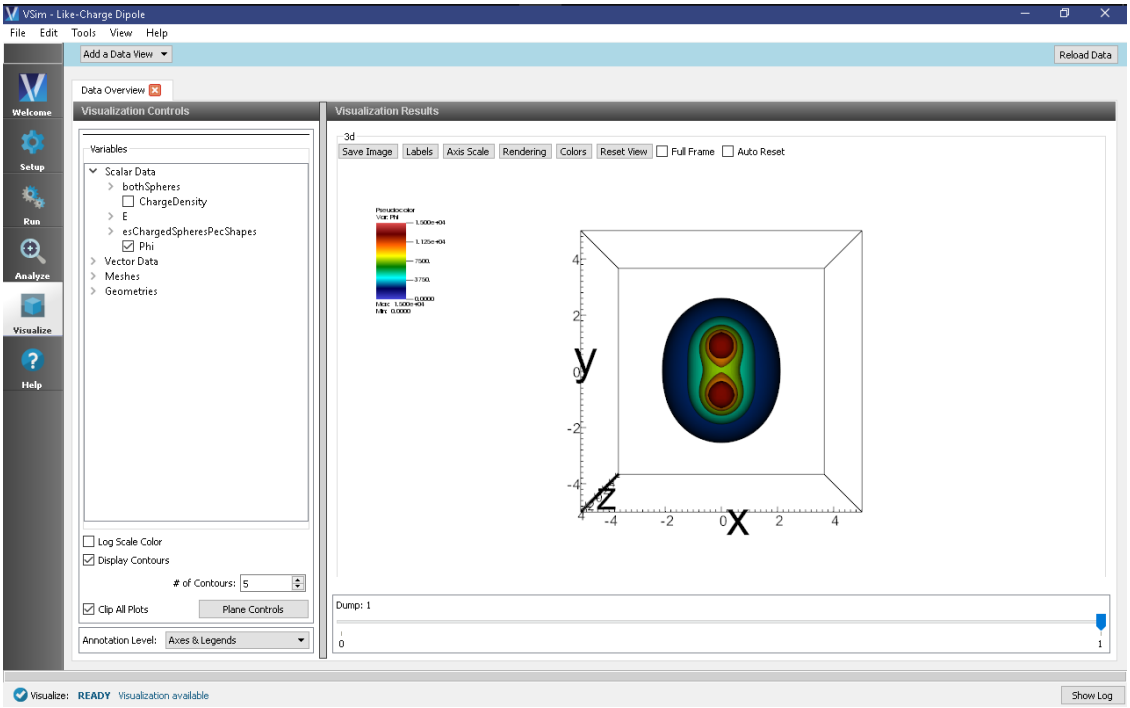


Fig. 3.61: Visualization of Like-Charge Dipole as a pseudocolor plot.

3.2.2 Dielectric in Electromagnetics (dielectricInEM.sdf)

Keywords:

dielectric, electromagnetics, capacitor

Problem Description

This simulation and the *Dielectric in Electrostatics (dielectricInES.sdf)* demonstrate the differences between simulating dielectric materials with electrostatic and electromagnetic solvers. They also demonstrate some of the more general differences between electrostatic and electromagnetic simulations.

Both of these simulations represent the same physical system: a slab of dielectric between two metal plates. The simulation grid is one square meter with a .75m x .25m slab of dielectric centered between the plates.

Unlike in the electrostatics simulation, in this simulation the full, coupled set of Maxwell's equations are considered, so more nuanced and complete physics will be represented in this simulation. The trade-off is that the simulation will take longer to run, and a user has to have a deeper, more fundamental understanding of the physics relevant to the simulation.

For situations when the nuanced and detailed physics is important, an electromagnetic solver is often required. For other cases, the nuances and details are not important and an electrostatic approximation is an appropriate simplification of the system.

A rule of thumb to consider when deciding between electromagnetic and electrostatic solvers is how the wavelength of an EM wave, λ_{EM} compares to the length scale of the simulation domain, L_s. If λ_{EM} is smaller than L_s, an electromagnetic solver will be more appropriate. In other words, if many EM waves of interest fit inside the simulation grid, an electromagnetic solver will be necessary.

This criterion, $\lambda_{EM} < L_s$, is equivalent to a criterion on the period of the EM oscillation T_{EM} and the time it takes light to cross the simulation domain, T_γ. Divide both sides by the speed of light and we get $T_{EM} < T_\gamma$. In this limit, information about changes to fields will not be able to travel across the simulation domain within one timestep, so we must pick a solver that respects relativity.

This simulation can be run with a VSimEM, VSimMD, or VSimPD license.

Opening the Simulation

The Dielectric in Electromagnetics example is accessed from within VSimComposer by the following actions:

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim for Electromagnetics* option.
- Expand the *Electrostatics* option.
- Select *Dielectric in Electromagnetics* 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. 3.62.

Simulation Properties

In full electromagnetics, the electric potential is a quantity that must be derived from line integrals of the electric field since it is not directly calculated (as is the case for electrostatics). This makes setting up potential differences between

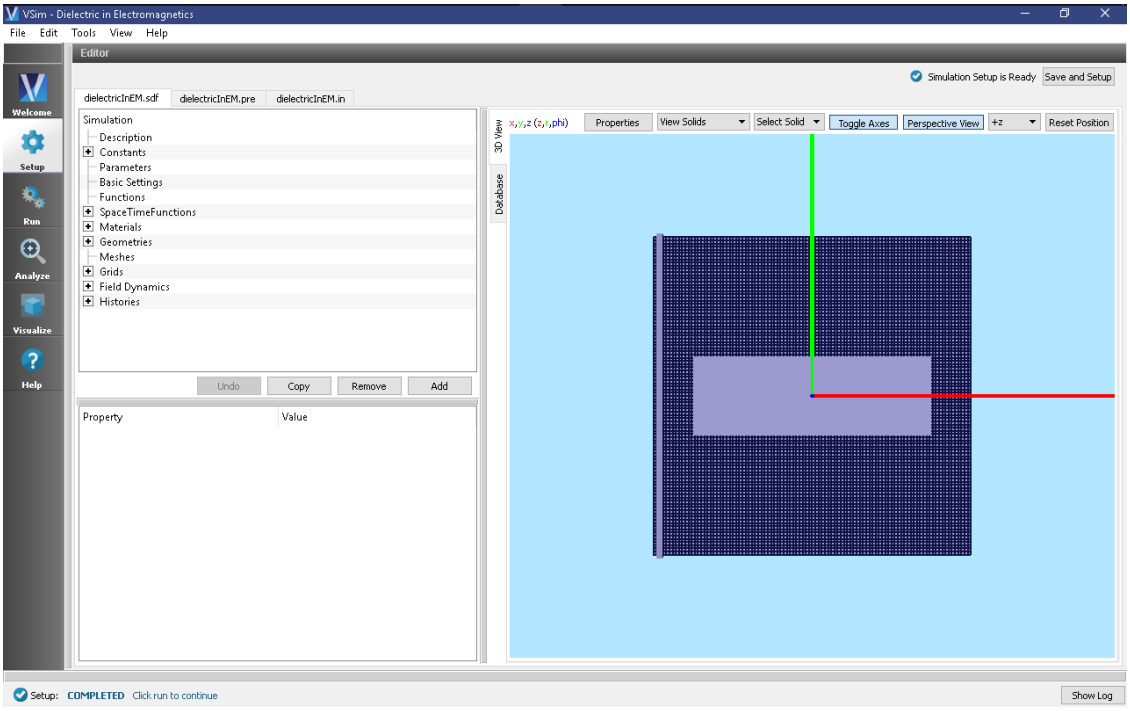


Fig. 3.62: Setup Window for the Dielectric in Electromagnetics example.

objects a little more complicated. So, to set a potential difference between the top and bottom plates in this simulation, a Distributed Current is set to run from the bottom of the simulation (lower y) to top (upper y).

The upper and lower y boundaries are set as a "PEC", a perfect electrical conductor, under the Field Boundary Conditions element. The left and right walls (lower and upper x), are set as open boundaries, and so will let EM waves pass through unaffected.

The current effectively moves positive charge from the bottom boundary of the simulation to the top boundary making the top plate positively charged and the bottom plate negatively charged. This will result in an electric field that points down towards the lower y boundary, similar to the *Dielectric in Electrostatics (dielectricInES.sdf)* example.

The functional form of the current is a Gaussian, $I(t) = e^{-(t-t_0)^2/(2*\sigma^2)}$. The constants t_0 and σ are set with the constants DRIVE_PEAK_TIME and DRIVE_SPREAD, respectively. The default values for these constants were chosen somewhat arbitrarily, but so that the simulation results will resemble those from the electrostatic simulation.

The DRIVE_SPREAD (standard deviation of the current Gaussian), will set a time scale for the electromagnetic waves that will propagate through this simulation, so DRIVE_SPREAD was chosen to be much larger than the timestep so that the EM wavelength was much larger than the simulation domain.

Ideally, the PEAK_DRIVE_TIME would be much larger than the DRIVE_SPREAD because whenever a function is used to set a current, the current should be initialized so that $I(0) = \frac{dI}{dt}|_{t=0} = 0$. That is, the current and its derivative are both zero when the simulation begins. This will reduce any "ringing" that may occur by an abrupt turn-on. The default value for PEAK_DRIVE_TIME for this simulation is 0, so at least the derivative of the current is zero. The *Future Experiments* section outlines how to do a higher fidelity simulation, and it will demonstrate the power of the electrostatic approximation. The *Dielectric in Electrostatics (dielectricInES.sdf)* simulation takes only one timestep to run.

To add the dielectric slab to the simulation, first a "Box" primitive shape was added under *Geometries* → *CSG*. Then the `Sapphire` material was added to the simulation and set as the material for the boxDielectic material.

Running the Simulation

To run the simulation:

- Proceed to the run window by pressing the Run button in the left column of buttons.
- Here you can set run parameters, including how many cores to run with (under the MPI tab).
- When you are finished setting run parameters, 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." This is shown in Fig. 3.63.

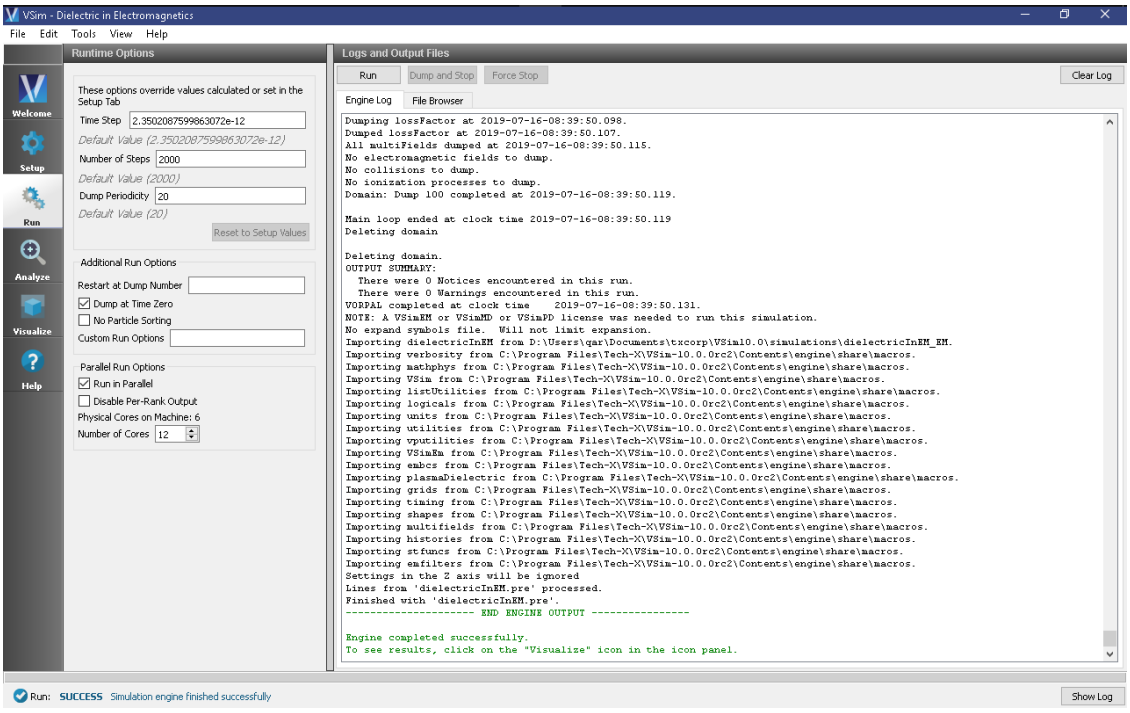


Fig. 3.63: The Run window at the end of execution.

Visualizing the Results

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

1. Proceed to the Visualize Window by pressing the *Visualize* button in the navigation column.
2. With the *Data Overview* tab open, expand *Scalar Data* then expand *E*.
3. Check the box for *E_y*. This will plot the y-component of the electric field.
4. Scroll through the dumps to see how the y-component of the electric field changes with time. The last dump is shown in Fig. 3.64.

As you scroll through the dumps, you will notice that a wave front propagates through the simulation from left to right. We see this wave front because electromagnetics respects relativity. Any electromagnetic signal can travel no faster than the speed of light. When the simulation begins, there is no voltage difference between the upper and lower plates. At $t = 0$, the current turns on and then a signal begins traveling (at the speed of light) from the left to the right that a voltage difference has been established.

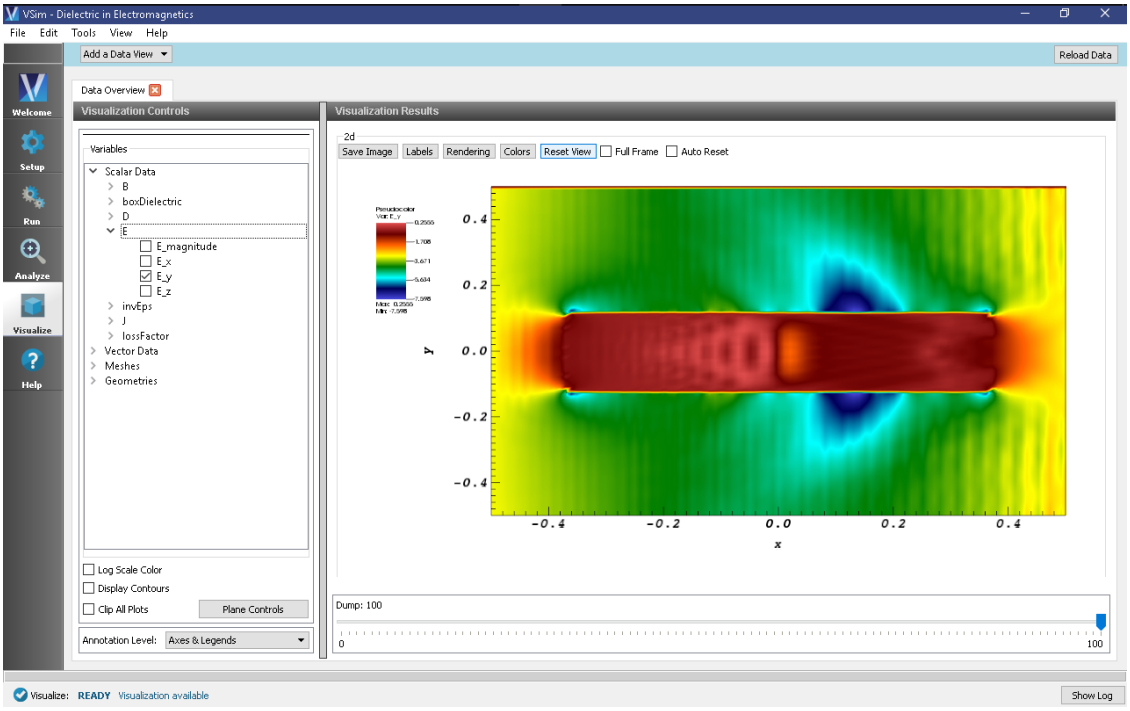


Fig. 3.64: The y-component of the Electric Field.

Additionally, you can see reflections of this wave off the vacuum-dielectric interface, and refracted waves entering dielectric. There is also some "ringing" which appears as ripples in the electric field from the abrupt current turn on.

Further Experiments

1. Experiment with different values of the DRIVE_PEAK_TIME and DRIVE_SPREAD constants that tune the Gaussian current. Try a value of 10^-7 for the DRIVE_PEAK_TIME while leaving the DRIVE_SPREAD at its default value of 10^-8. This will shift peak of the Gaussian far beyond the end of the simulation, so that the smooth, flat, far left tail of the Gaussian, with values close to zero, sets the current. What do you notice about the noisiness of the y-component of the Electric Field? How much longer would you have to run before the current ramps up to its maximum value then drops back down to zero to produce the image below?

2. Then put the DRIVE_PEAK_TIME back to the original, default value of 10^-10 and change the DRIVE_SPREAD to a much smaller value, like 10^-10. Given the discussion about timescales in the *Problem Description* and *Simulation Properties* sections what changes would you expect to see?

3. Normalize the current Gaussian so that you can control the final value of the y-component of the electric field.

3.2.3 Dielectric in Electrostatics (dielectricInES.sdf)

Keywords:

`dielectric, electrostatics, capacitor`

Problem Description

This simulation and the *Dielectric in Electromagnetics (dielectricInEM.sdf)* demonstrate the differences between simulating dielectric materials with electrostatic and electromagnetic solvers. They also demonstrate some of the more

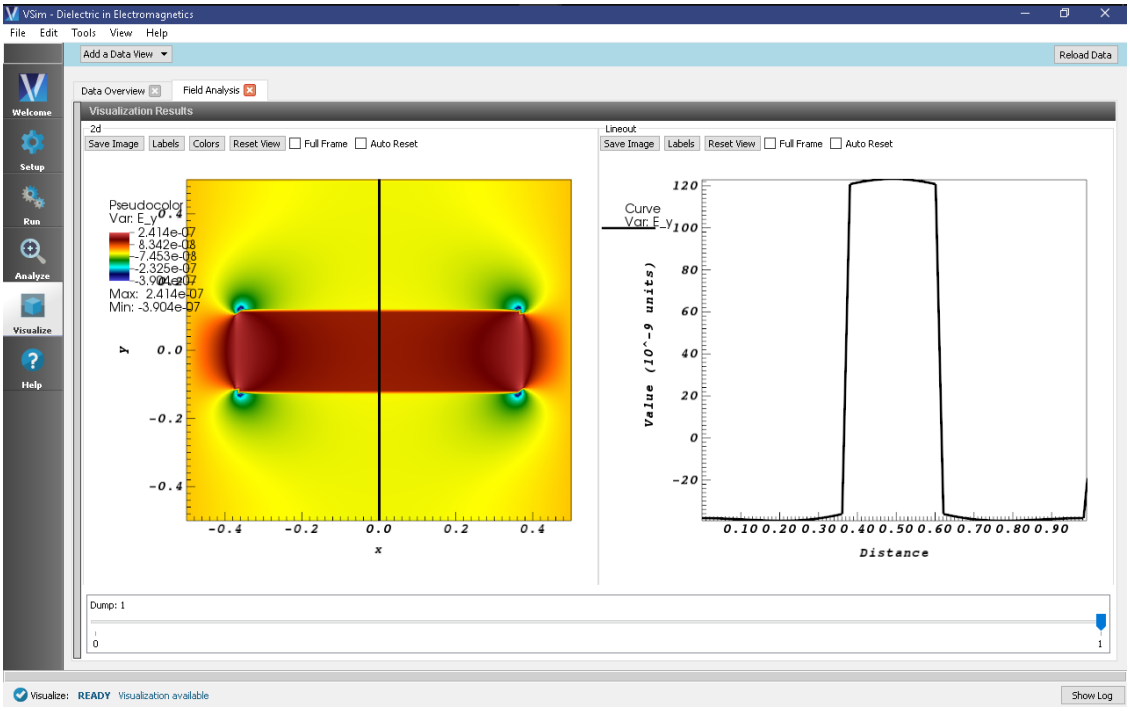


Fig. 3.65: The y-component of the electric field after the current ramped up and then back down to zero.

general differences between electrostatic and electromagnetic simulations.

Both of these simulations represent the same physical system: a slab of dielectric between two metal plates. The simulation grid is one square meter with a .75m x .25m slab of dielectric centered between the plates. The electric potential and electric field are solved over the entire domain.

Electrostatics is an approximation of the full set of Maxwell's equations. According to Faraday's law,

$$\nabla \times \vec{E} = -\frac{\partial \vec{B}}{\partial t}.$$

In the electrostatic limit, fields change slowly. More precisely, $\frac{\partial B}{\partial t} \approx 0$, so that any curling electric field is negligible compared to the full electric field. When curling electric fields can be neglected, the electric field is a conservative field and can be written as the gradient of a scalar function. This scalar function is the electric potential, or voltage, and satisfies Poisson's equation

$$\nabla^2 \phi = -\frac{\rho}{\epsilon_0}.$$

Electrostatics is appropriate if the shortest wavelength of interest, λ_{EM} is much longer than the length scale of the simulation L_s. This criterion, $\lambda_{EM} > L_s$ respects the "fields change slowly" heuristic. Divide both sides of the expression by the speed of light and we get that the period of an electromagnetic oscillation is longer than the time it takes light to cross the simulation domain. In this limit, any change to a field that occurs within a timestep will have time to propagate throughout the simulation domain within the same timestep.

As can be seen in Poisson's equation, electrostatics does not respect relativity, since any change in ρ will instantaneously change the electric potential everywhere in the simulation. So, it is important to respect the $\lambda_{EM} > L_s$ criterion when doing electrostatics, otherwise, the simulation might neglect some important physical effects.

This simulation can be run with a VSimEM, VSimMD, or VSimPD license.

Opening the Simulation

The Dielectric in Electrostatics example is accessed from within VSimComposer by the following actions:

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim for Electromagnetics* option.
- Expand the *Electrostatics* option.
- Select *Dielectric in Electrostatics* 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. 3.66.

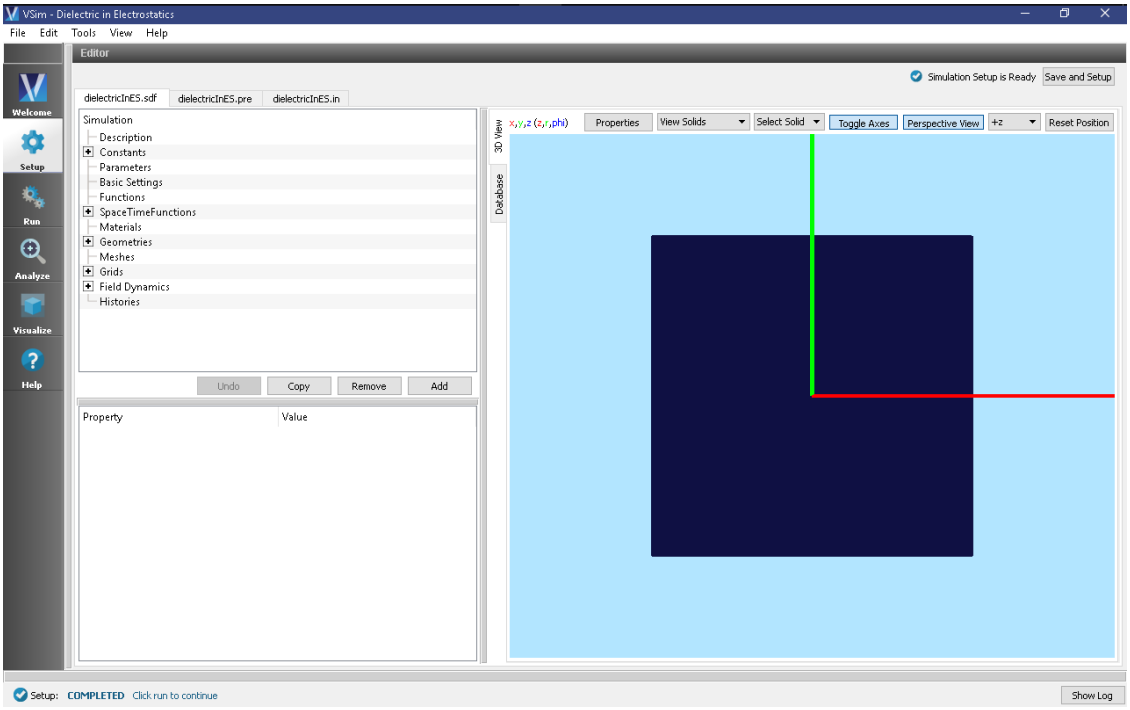


Fig. 3.66: Setup Window for the Dielectric in Electrostatics example.

Simulation Properties

In this simulation, voltages of 10.0 V and 0.0 V are set on the upper and lower boundaries (respectively) with Dirichlet boundary conditions. Neumann boundaries are used on the left and right walls of the simulation domain. In vacuum, this would set up an electric field of 10 V/m pointing down.

A dielectric is introduced to the simulation using a spacetime function `dielectricSapphire`. Sapphire has a relative dielectric constant, $\epsilon_r = \frac{\epsilon_{sapphire}}{\epsilon_0}$, of 9.8. Through the use of Heaviside functions, we set the relative permittivity to be 9.8 in a rectangular region between $x = \pm.375$ meters an $y = \pm.125$ meters and 1 everywhere else.

In the setup tree, the `dielectricSapphire` function is set as the *relative permittivity* under *Field Dynamics* → *PoissonSolver*.

Running the Simulation

To run the simulation:

- Proceed to the run window by pressing the Run button in the left column of buttons.
- Here you can set run parameters, including how many cores to run with (under the MPI tab).
- When you are finished setting run parameters, 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." This is shown in Fig. 3.67.

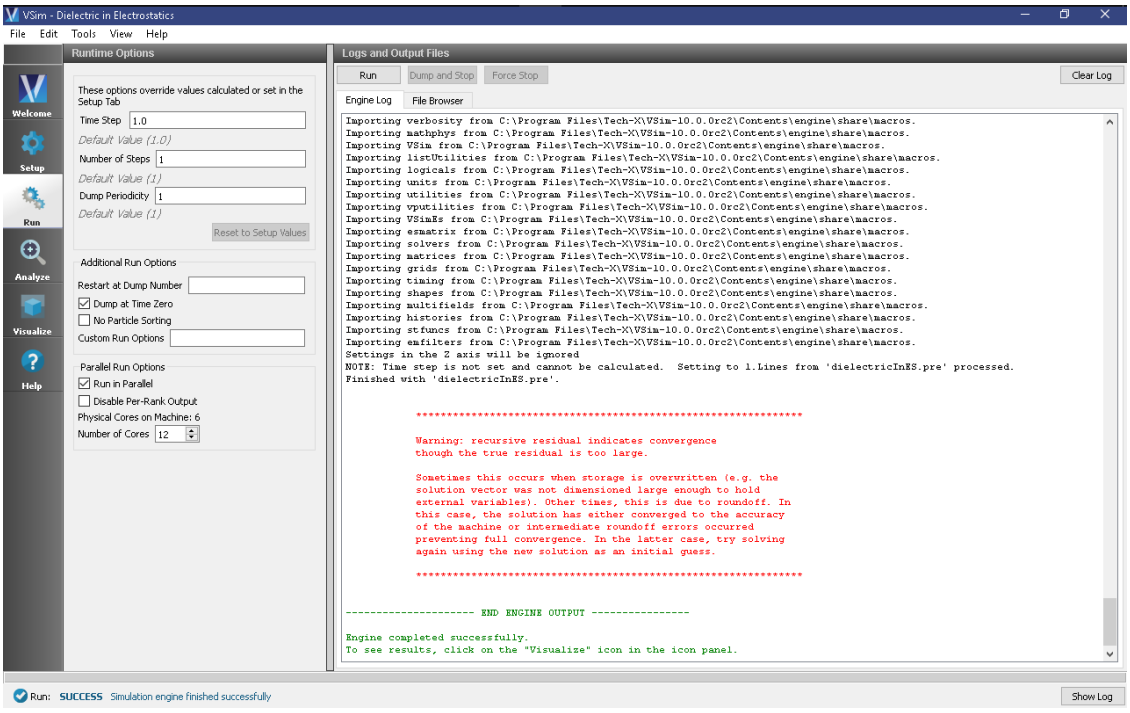


Fig. 3.67: The Run window at the end of execution.

Visualizing the Results

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

1. Proceed to the Visualize Window by pressing the *Visualize* button in the navigation column.
2. With the *Data Overview* tab open, expand *Scalar Data* then check the box for *Phi*.
4. For a closer look at the potential, open the *Field Analysis* visualization tab by selecting it from the *Add a Data View* drop down menu at the very top right of the VSim Window.
5. In the new *Field Analysis* tab, from the *Field* drop down menu select "Phi."
6. The user can switch the location of the line out with the options on the left side of the screen.
7. Shown below is a plot with a horizontal line out at a position $y = .18$ meters. Don't forget to press the *Perform Lineout* button after adjusting the

line out settings!

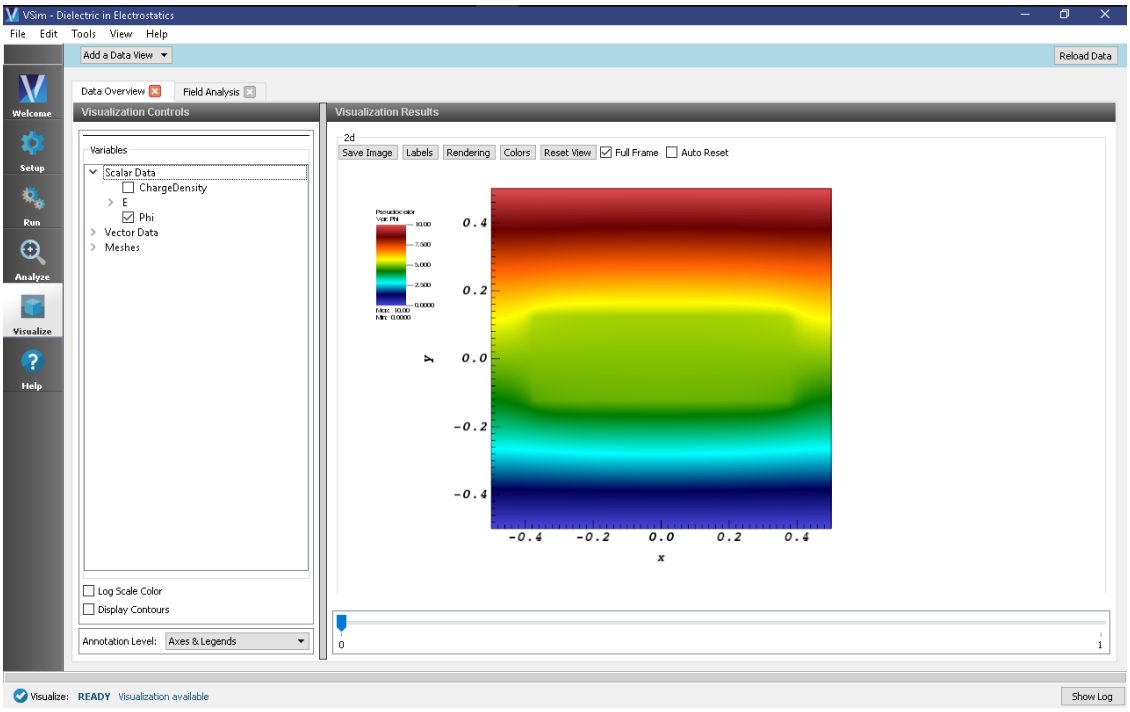


Fig. 3.68: The electric potential between the two parallel plates.

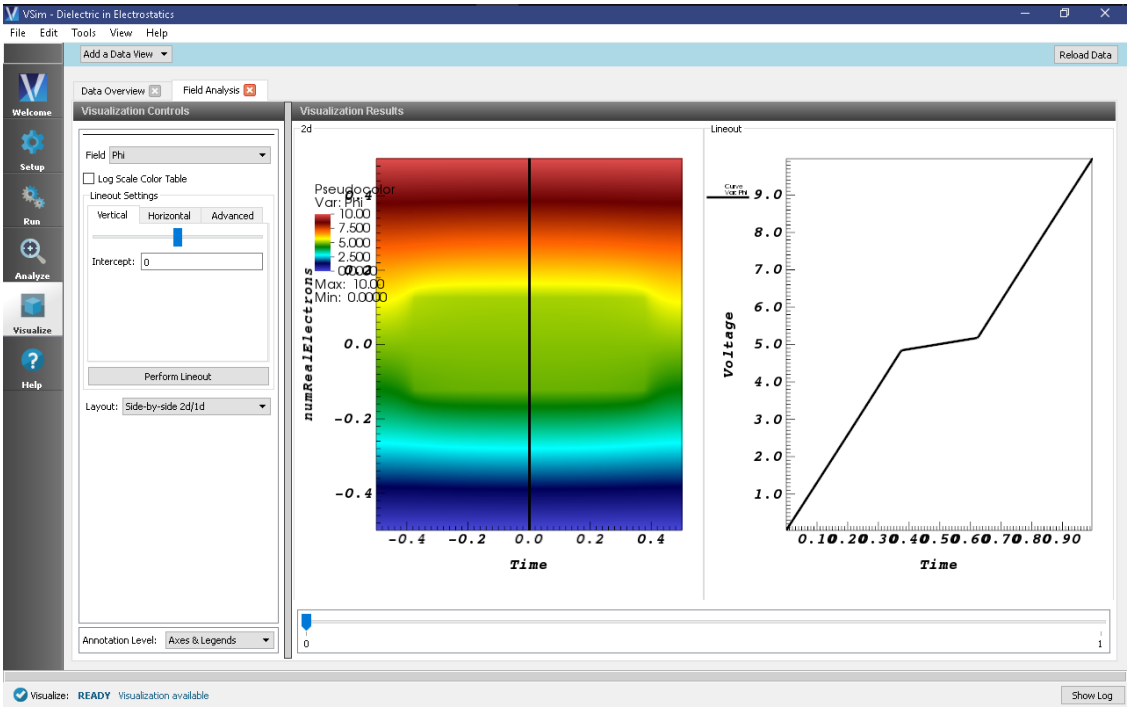


Fig. 3.69: Field Analysis tab showing the electric potential along the black vertical line.

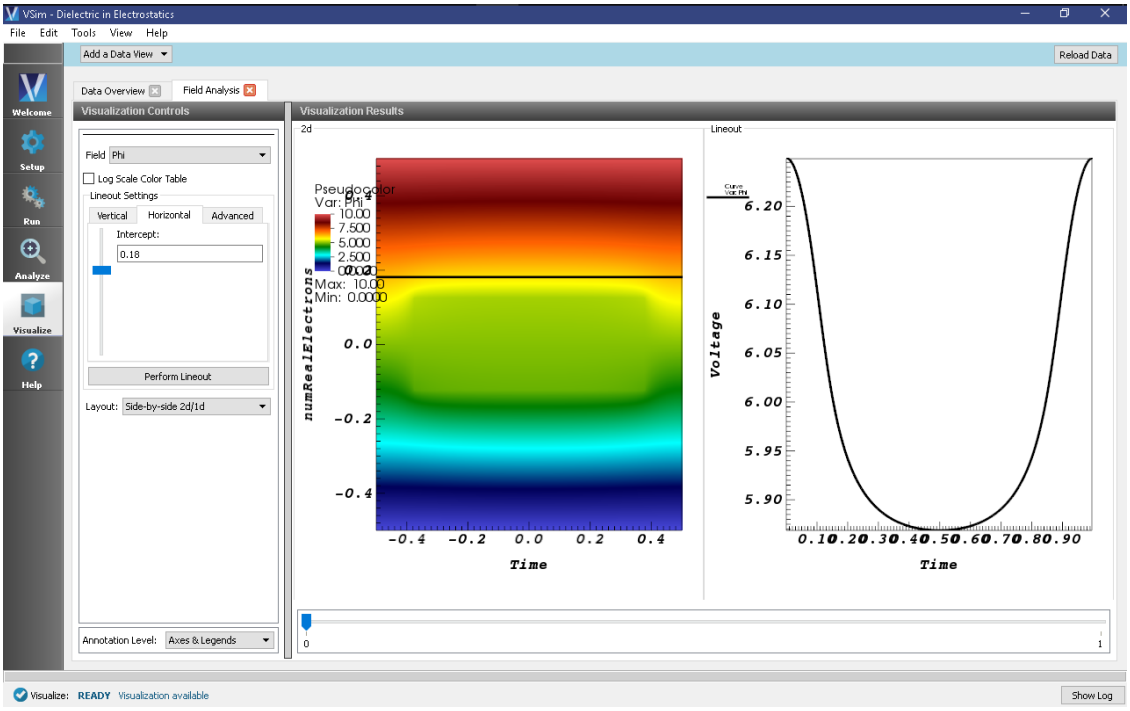


Fig. 3.70: Field Analysis tab showing line out in different position.

Further Experiments

Change the dielectric constant, or change the location and area of the dielectric. Add a sinusoidal voltage on the plates.

3.3 Photonics

3.3.1 Multimode Fiber Mode Calculation (multiModeFiberModeCalc.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 multiModeFiberModeCalc_invEps_0.h5 file to solve for the guided modes using the computeDielectricModes.py analyzer. This analyzer will find the entire basis set of modes for this fiber and output each into a separate .vsh5 file. These mode files can be used to launch the exact modes into your simulation. This process is shown in the multiModeFiberModeLaunchT example.

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 VSimEM license.

Opening the Simulation

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

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim for Electromagnetics* option.
- Expand the *Photonics* option.
- Select *Multimode Fiber 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

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

- WAVELENGTH_VAC: The vacuum wavelength (frequency divided by the speed of light) for the eigenmode of interest. In this example we scale all lengths by 1e6. Therefore, a typical 1.55 micron source is simply 1.55 in this example.
- NA: The numerical aperture of the fiber. This defines the background permittivity relative to the core permittivity.
- PERMITTIVITY_CORE: This is the permittivity of the core material as defined in the *Materials* branch and assigned in the *Geometries* branch.
- RADIUS_CORE: The radius of the fiber core.
- RESOLUTION_YZ: The number of cells per core radius in the transverse (y & z) directions.
- RESOLUTION_X: The number of cells per estimated wavelength in the propagation (x) direction.
- CFL_NUMBER: The time step, DT, will be this value times the limit for numerical stability.

Parameters Many parameters in this simulation are defined to assist with launching the mode in a subsequent example, multiModeFiberModeLaunchT. Some important parameters that are relevant to the mode extraction are given below.

- PERMITTIVITY_BG: The permittivity assigned to the background surrounding the core. This is assigned in the *Basic Settings* branch under *background permittivity*.
- NMODES: The approximate number of modes supported by the fiber. Depends on NA, RADIUS_CORE, & WAVELENGTH_VAC. Includes degenerate modes. More accurate for many modes (>20).
- BGNYZ_SOURCE: The starting position for where we will evaluate the transverse permittivity profile on the analyzer tab. Defined for convenience and clarity. This slice is shown as the "source" geometry in the *3D view*.
- ENDYZ_SOURCE: The ending position for where we will evaluate the transverse permittivity profile on the analyzer tab.

Setting up the 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, expand the *Basic Settings* branch and verify that the *dielectric solver* field is set to *permittivity averaging*. This algorithm is a powerful VSim feature. This setting is shown in Fig. 3.71.

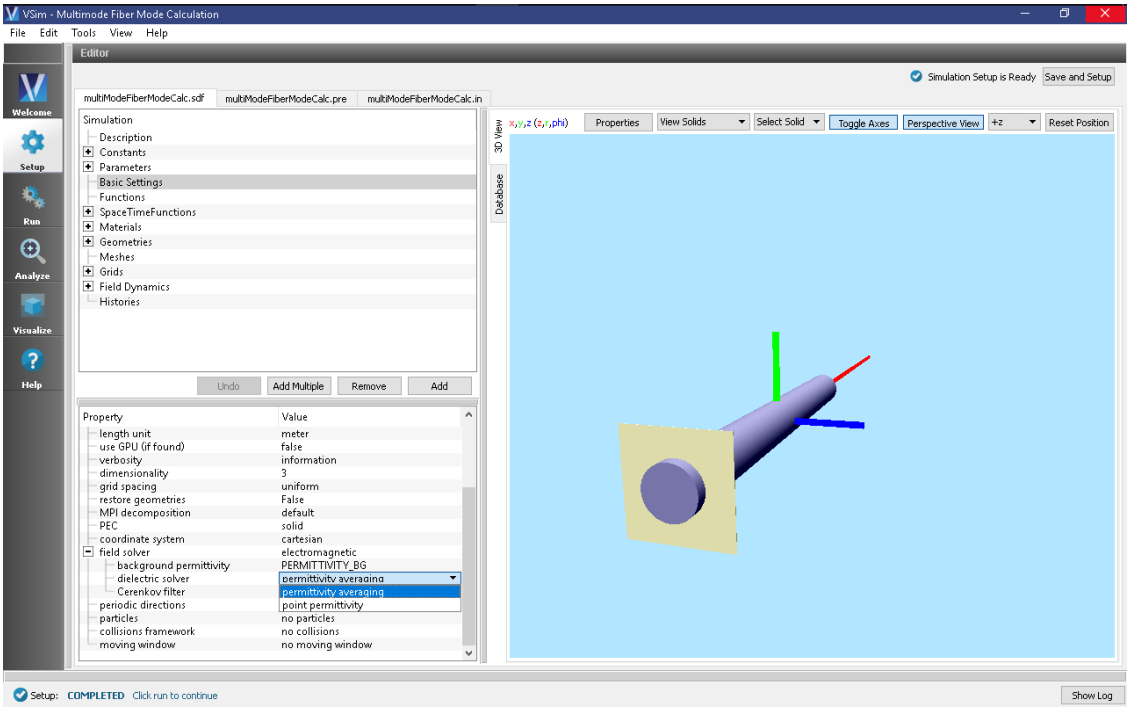


Fig. 3.71: Choosing the second order accurate, *permittivity averaging* for the *dielectric solver* field under *Basic Settings*.

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 change the *Number of Steps* and *Dump Periodicity* to 1.
- Under Additional Run Options select *Dump at Time Zero*.
- To run the file, 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." The result is shown in Fig. 3.72.

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 *computeDielectricModes.py* and click *Open* under the list.

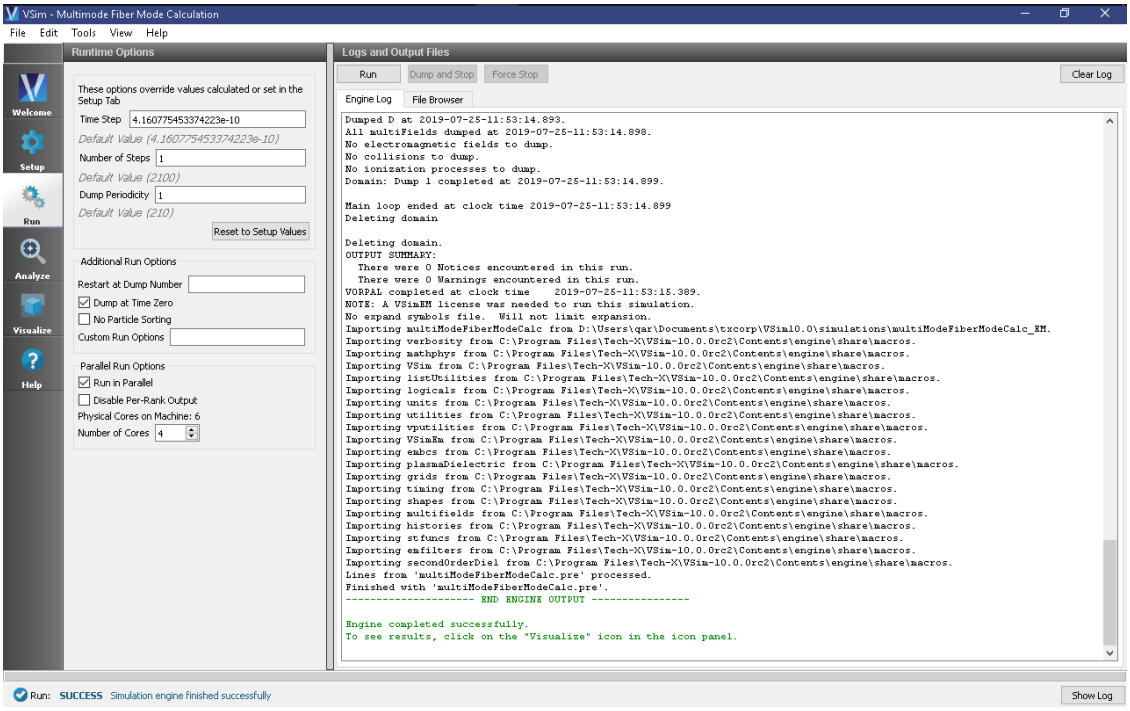


Fig. 3.72: Running the simulation for one step to get the permittivity data for the analyzer. Note that *Dump at Time Zero* is checked.

Now update the analyzer fields accordingly. Some of these parameters are described above under **Parameters**

- *transverseSlice*: 0,-10:10,-10:10
- *vacWavelength*: 1.55
- *nModes*: 25
- *writeFieldProfile*: H,E,D

We set the number of modes (*nModes*) to a value greater than the number of modes we expect. The analyzer will only find guided modes. Also check *Overwrite Existing Files*. Run the analyzer by clicking *Analyze* button in the upper right corner. The analyzer output should resemble Fig. 3.73. We see that the analyzer found 20 modes. They are listed in decreasing order of effective index. After the modes are calculated, the analyzer will dump the solutions to file so they can be visualized - this may take a few minutes.

Visualizing the Results

After performing the above actions 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 *Data View* dropdown, select *Data Overview*.
- Expand *Scalar Data*, expand *EigenD*, and select *EigenD_z*.
- At the top of the visualization window, tick the *Full Frame* box.
- Below the visualization, the dump slider will allow you to scroll between the modes. Scroll to dump number 13.

The resulting visualization pane should resemble Fig. 3.74.

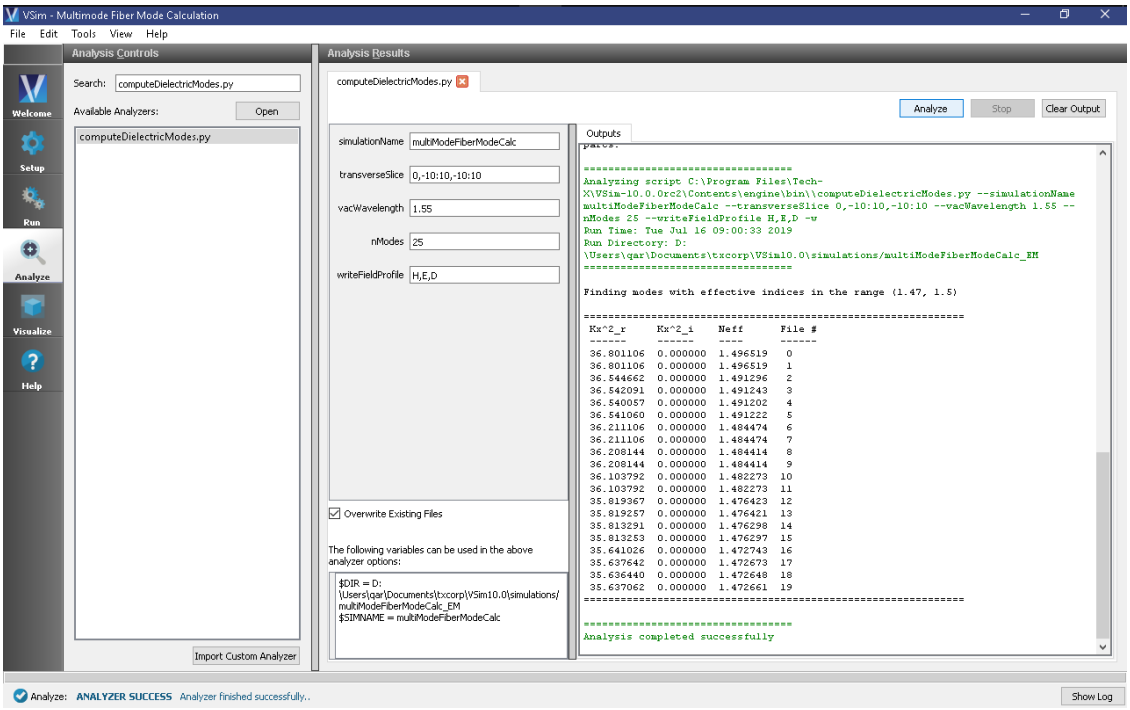


Fig. 3.73: The analyzer window after a successful run of computeDielectricModes.py.

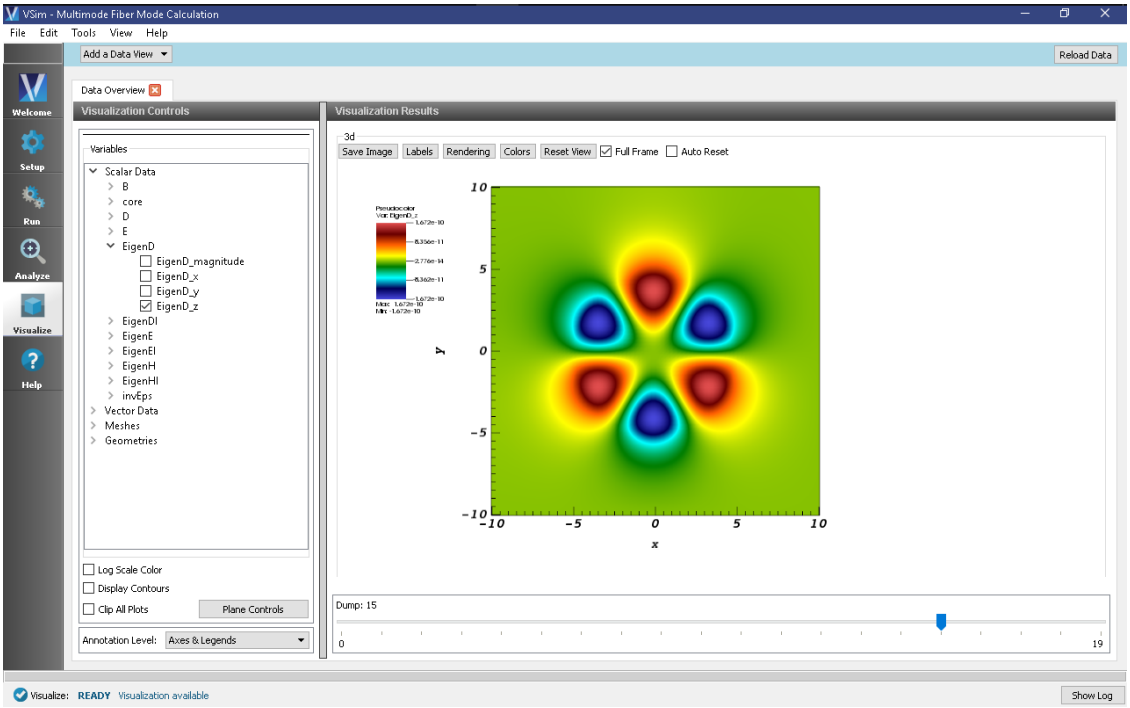


Fig. 3.74: The visualization pane showing the z component of the D field of the M=1, L=3 mode.

One can select other components of the H, E, or D field to see how they vary for the eigenmodes. These eigenmodes are now saved in .vsh5 files in the folder where the simulation was run.

Further Experiments

Increase the radius, decrease the wavelength, or increase the numerical aperture on the Setup window and rerun the simulation and analyzer to see higher order modes.

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

One can run a full convergence study of eigenmode effective indices by varying the RESOLUTION_YZ constant 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. 3.75. The linear relationship shows the second order accuracy of our dielectric algorithms.

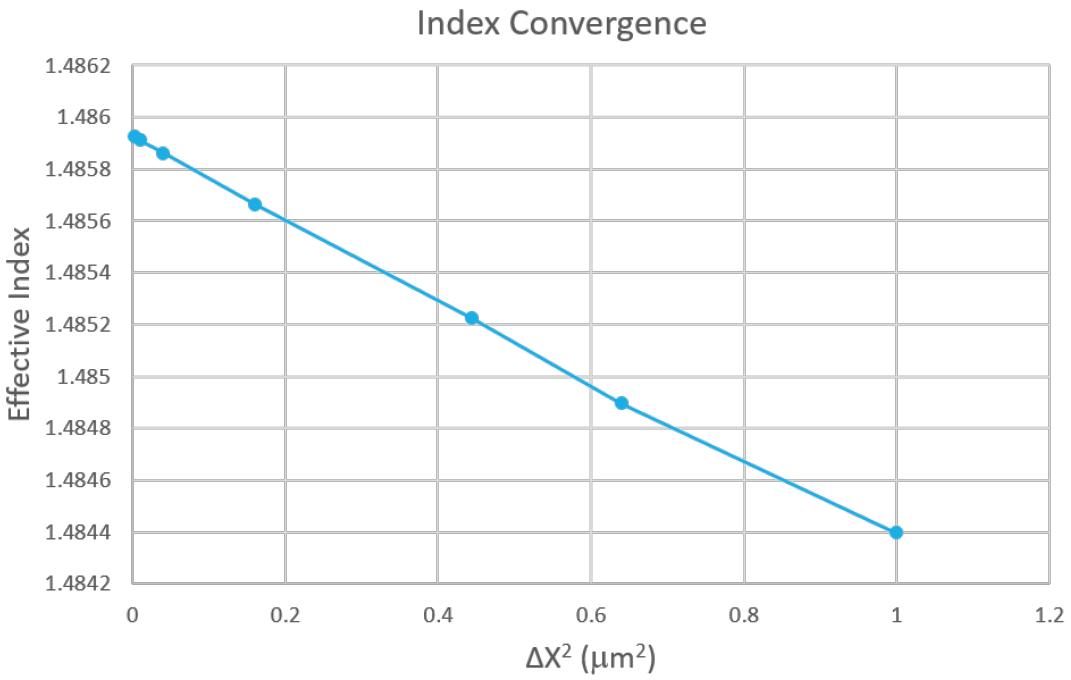


Fig. 3.75: The effective index as a function of transverse cell area for an eigenmode.

3.3.2 Multimode Fiber Mode Extraction (multiModeFiberModeExtract.sdf)

Keywords:

Photonics, dielectric fiber

Problem Description

This example illustrates how to compute the modes of a cylindrical fiber for a given propagation constant, β, which, because the primary direction of propagation in VSim is along the x-axis, is also denoted as k_x. The calculation is performed using excitation of a system with only two cells in the x-dimension. (The simulation can be performed with

one cell in x, but this cannot be easily visualized so two are used instead.) This document will show how to extract the modes and their frequencies, as well as how to get a text-based setup file for further exploration, including solving for the propagation constant as a function of the frequency.

This simulation can be performed with a VSimEM license.

Opening the Simulation

The Multimode Fiber Mode Extraction example is accessed from within VSimComposer by the following actions:

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim for Electromagnetics* option.
- Expand the *Photonics* option.
- Select *Multimode Fiber Mode Extraction* 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. 3.76. Expanding the Constants section of the Simulation Tree shows the user-defined constants of this simulation:

`LENGTH_UNIT` (real): The length-scale of the simulation. All lengths, such as that of the fiber radius and box size, are divided by this number to make the geometry lengths of order unity as needed by the geometry engine. In this case, `LENGTH_UNIT` is unity, so it has no effect.

`WAVELENGTH_VAC` (real): Wavelength of the signal in vacuum. Determines the excitation frequency.

`N_EFF` (real): Estimate of the value of n_{eff} of the expected modes. This determines the wavenumber, $k_x = 2\pi n_{eff}/\lambda_{vac}$, of the modes to be found.

`RESOLUTION` (real): The resolution of the simulation grid. The cell size is set to be this number multiplied by the smallest simulation feature, i.e., the fiber radius.

`XCELLS` (integer): The number of cells to simulate in the x direction.

`CFL_NUMBER` (real): This times the stable time step gives the time step chosen for the simulation.

`FREQ_GAP_REL` (real): This times the stable time step gives the time step chosen for the simulation.

Expanding the Parameters section of the tree shows how the other simulation parameters are computed from the constants. For example, the grid size `DX` along the x-axis is set to the resolution multiplied by the vacuum wavelength, divided by n_{eff} and scaled by `LENGTH_UNIT`. The excitation central frequency, `frequency`, is computed from the vacuum wavelength scaled by the `LENGTH_UNIT`. The longitudinal wavenumber, `KAY`, is computed from the desired n_{eff}, and from that the phase shift across cells along the x-axis is calculated.

The range of frequencies to be excited is [`FREQ_LOW`, `FREQ_HIGH`]. Outside of this range by `FREQ_GAP`, the excitation drops off to the suppression value. This requires a successively longer excitation time, `TIME_EXCITE` and so a successively larger `NSTEPS_EXCITE`, the number of steps for the excitation.

Absorbing layers have been placed at the y and z limits to damp out modes that would be outgoing for a fiber in infinite space.

In 3D View tab of the right pane of the Setup Window, the fiber and the grid are visible. Right-click and drag to rotate the view. The simulation has been constructed so that the fiber extends beyond the grid in both the positive and negative x-directions, with a fiber diameter one-half the perpendicular domain length.

Expanding the Materials section of the Simulation tree shows that the simulation includes `FiberMaterial`. This was created by importing a material from the Database tab of the right pane of the Setup Window, then changing the name of the material and changing its properties. A material can be changed arbitrarily once it is in the simulation, as shown in Fig. 3.77.

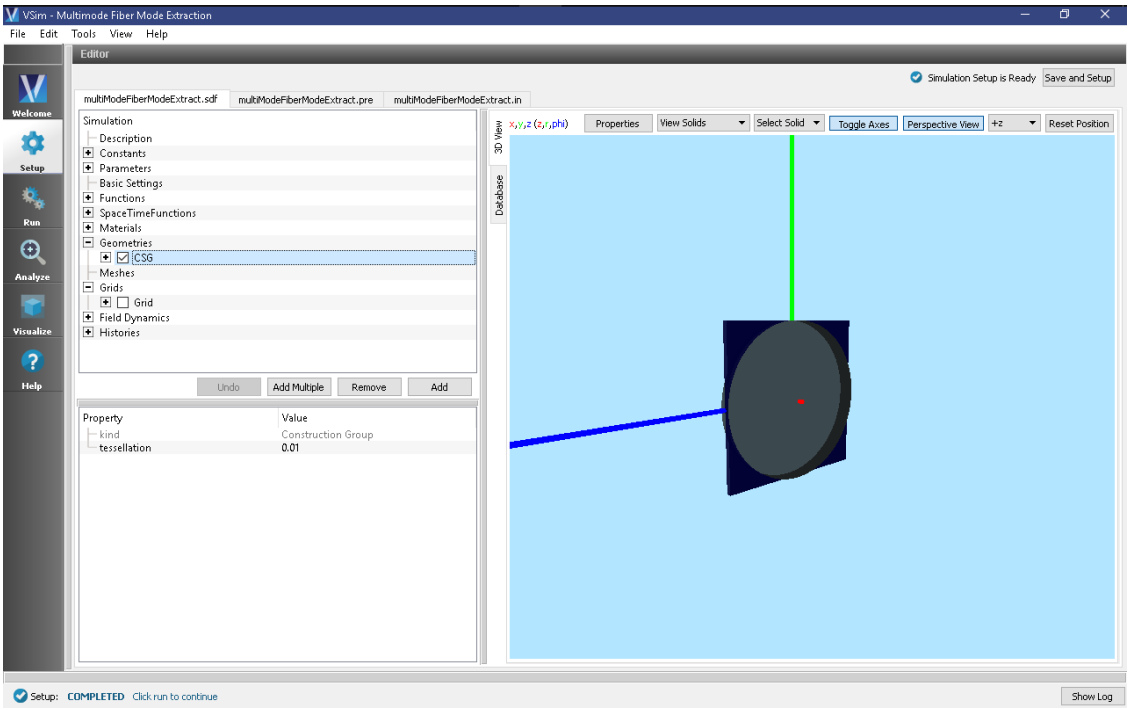


Fig. 3.76: Setup Window for the Multimode Fiber Mode Extraction example.

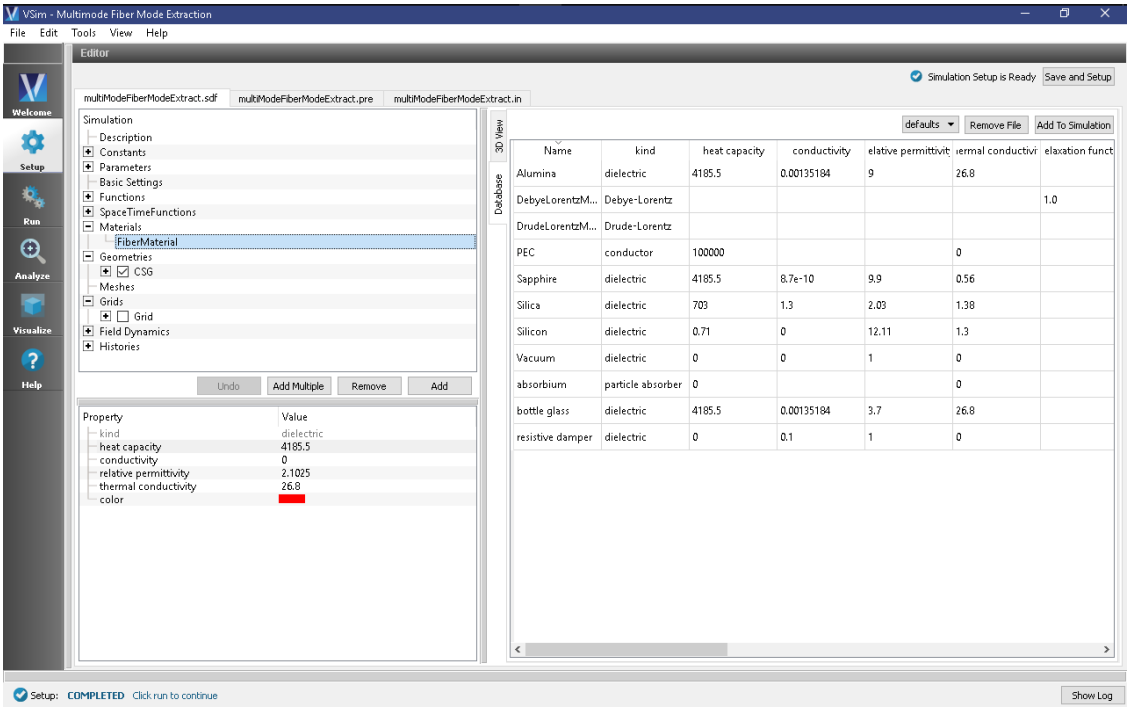


Fig. 3.77: Setup Window for the Multimode Fiber Mode Extraction materials.

Expanding the Geometries of the Elements Tree shows that the simulation includes one geometry, the fiber, and its material is `FiberMaterial`. This is seen in Fig. 3.78.

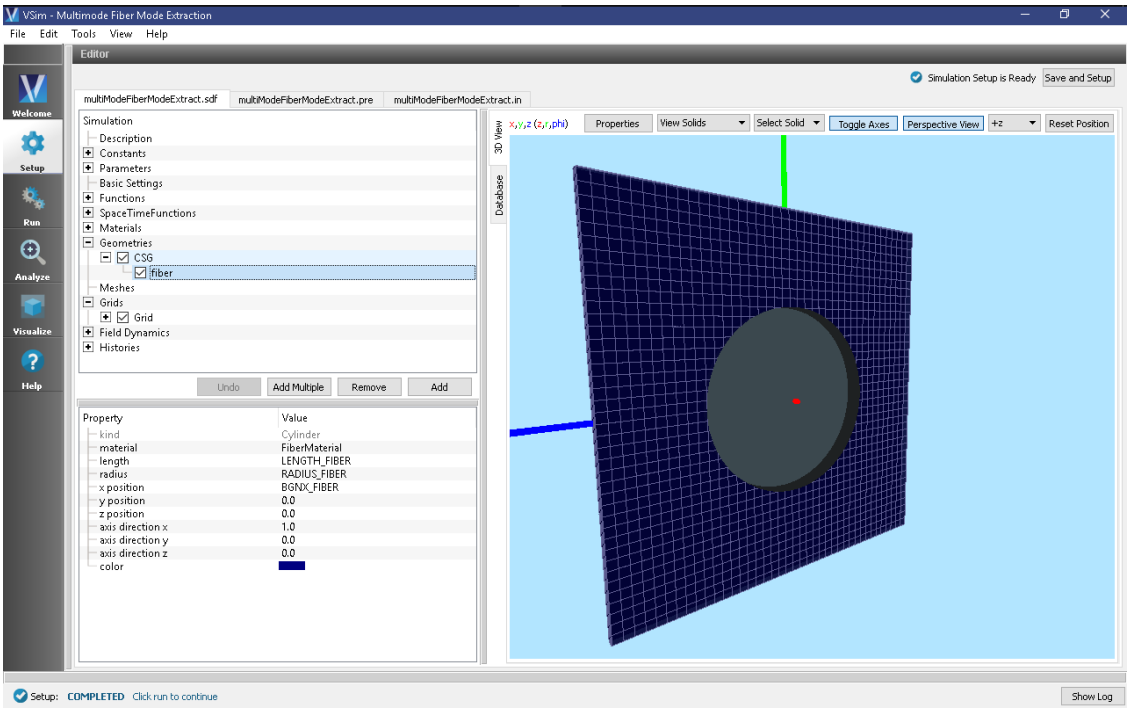


Fig. 3.78: Setup Window for the Multimode Fiber Mode Extraction geometries.

This simulation is excited with the `freqBand` function. This is a function that has a fairly uniform excitation over a band of frequencies, falling off steeply outside of the band. The band has been chosen to be centered near where we expect to find the modes.

A field history has also been implemented in this simulation, so that the Fourier transform of what has been excited can be seen.

As noted above, under the Parameters section of the Tree Elements is defined `NSTEPS_EXCITE` which specifies the number of steps to excite the desired frequency content. Because `FREQ_GAP` also distinguishes the peaks, this excitation time will distinguish the peaks.

Running the Simulation

Once finished with the problem setup, continue as follows:

- Proceed to the Run Window by pressing the **Run** button in the left column of buttons.
- For this run we choose 10000 steps, much greater than that (4000) required for the excitation. This further reduces any effect of the excitation on the signal in free oscillation.
- Choose parallel computing options on the MPI 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 that pane. The run has completed when you see the output, "Engine completed successfully." This is shown in Fig. 3.79.

This simulation takes approximately 10 seconds on 4 cores of a modern processor.

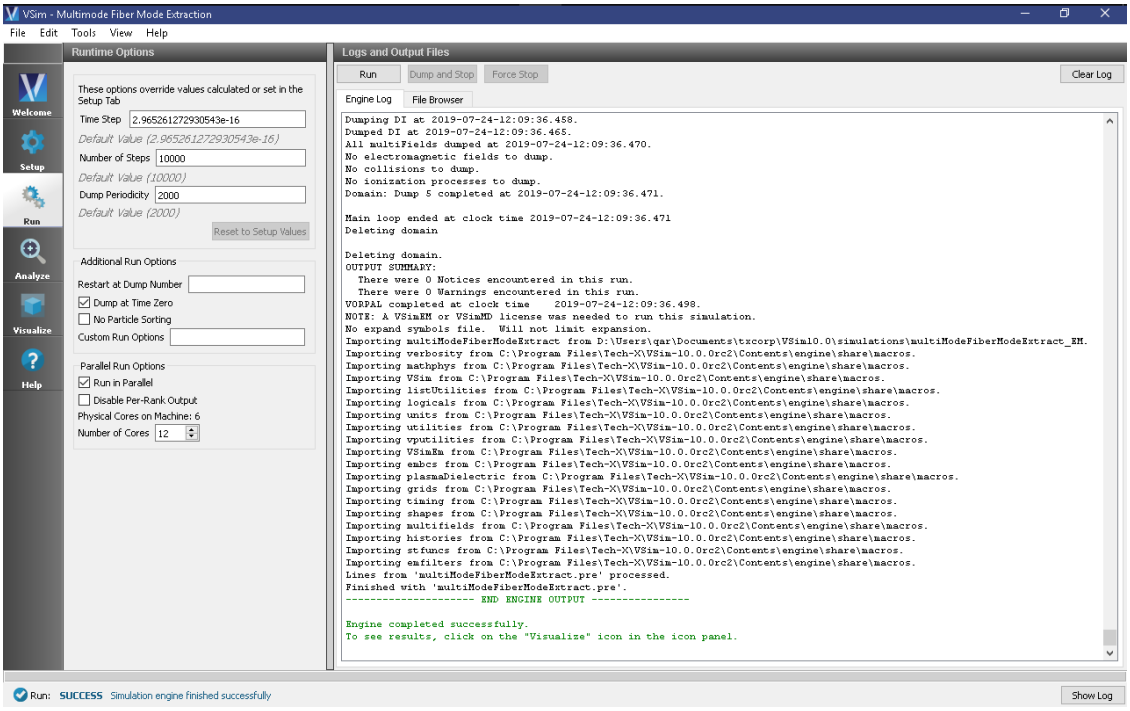


Fig. 3.79: The Run Window at the end of execution.

Visualizing the spectrum

- Proceed to the Visualize Window by pressing the Visualize button in the left column of buttons.
- In the Visualization Controls pane, in the dropdown menu for *Add a Data View* select *History*.
- For graph 1, set the quantity to be plotted to `driveCurrent_2`.
- For graph 2, set the quantity to be plotted to `midUpperRightE_2`.
- For graphs 3 and 4 set the quantity to be plotted to `<None>`.
- In the Visualization Results pane, for each plot, click the *Fourier Amplitudes (dB)* check box.
- Graph 1 shows the square window in frequency space.
- Graph 2 shows several peaks between 200 and 250 THz.
- To see this region in more detail, for each graph press the limits button and set the minimum to 1.7e14 and the maximum to 2.7e14. Peaks in the spectrum are seen at the frequencies, 198 THz, 205THz, 217THz, 233, and several around 250 THz in Fig. 3.80.

Analyzing the Results

Since 8 modes were seen, we look for 8. This will require 3 dumps per mode, or about 27 additional dumps. The dumps should be spread out over a few oscillations (35 steps each) or so, and it is good to do a few extra. To get these new dumps, return to the Run pane, set *Number of Steps* to 200, *Dump Periodicity* to 5, and *Restart at Dump Number* to 5. This will start a new simulation from where the first stopped. Click the *Run* button in the top left corner of the right pane.

Now go to the Analyze pane, select the *extractModes.py* analyzer, and press the *Open* button. Set the field to E, choose beginDump to be 6, endDump to be 36, nModes to be 8, sampleType of 1, construct to be 1, and 100 of each kind of

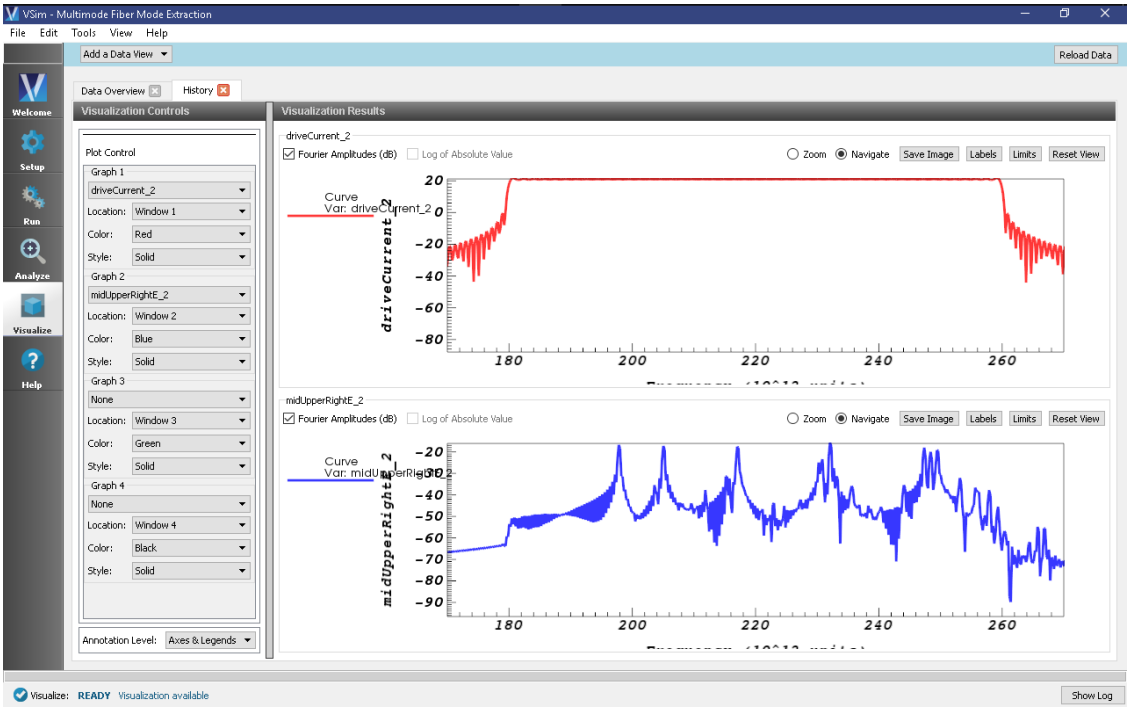


Fig. 3.80: The observed spectrum.

points. Upon hitting the Analyze button of the Analyze pane, we sees the list of detected frequencies, of which the first (mode 0) is the mode of interest, it having frequency of 1.98e14. This is seen in Fig. 3.81.

Visualizing the eigenmodes

Return to the Visualize Window, reload the data, open Scalar Data -> E, and click on E_z (EigenE). Keep the slider on position 0. With the mouse, turn the image sideways to see the cross section, as shown in Fig. 3.82.

Convergence

This simulation can be repeated for different values of `RESOLUTION` to see how the frequency varies with the meshing. We carried out this experiment with `RESOLUTION` varying over 0.1, 0.05, 0.02, 0.01 and plotted the frequency versus the inverse grid length in Fig. 3.83.

For each value of the resolution do the excitation run followed by the extraction run:

- Excitation run: Press *Reset to Setup Values* to get the correct value for the time step. Then set the number of step in the run panel to what is given by `NSTEPS_EXCITE`, and also modify the number of steps in the second run proportionately. E.g., for `RESOLUTION` = 0.05, `NSTEPS_EXCITE` = 7916, so in the Run panel choose *Number of Steps* = 8000 and *Dump Periodicity* = 2000. Clear the *Restart at Dump Number* box. Press the *Run* button.
- Extraction run: E.g., for `RESOLUTION` = 0.05, set the *Number of Steps* to 400, the *Dump Periodicity* to 10, and *Restart at Dump Number* to be 4.
- Analysis: Same as originally, as all numbers have been scaled.

The linear approach to the axis indicates that this is a first order accurate calculation. In other examples we will show higher-order accuracy. Even so, one can see that the frequency is obtained on the sub-percent level with the finest grid

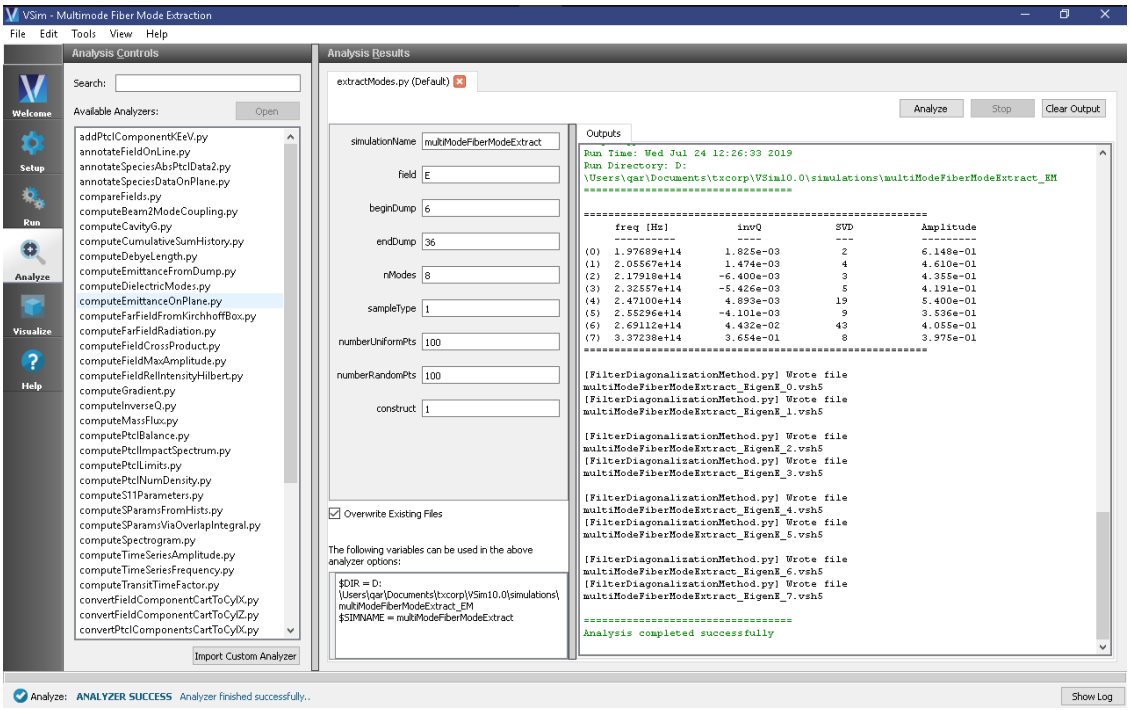


Fig. 3.81: Extraction of the mode frequencies.

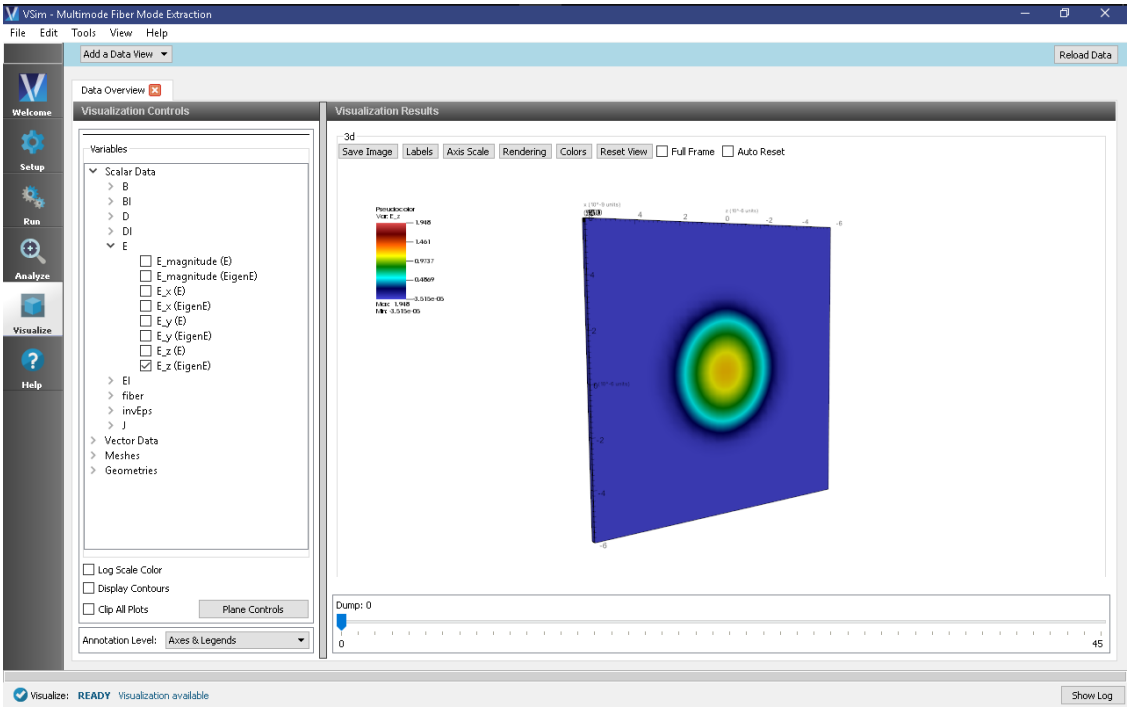


Fig. 3.82: The extracted eigenmode.

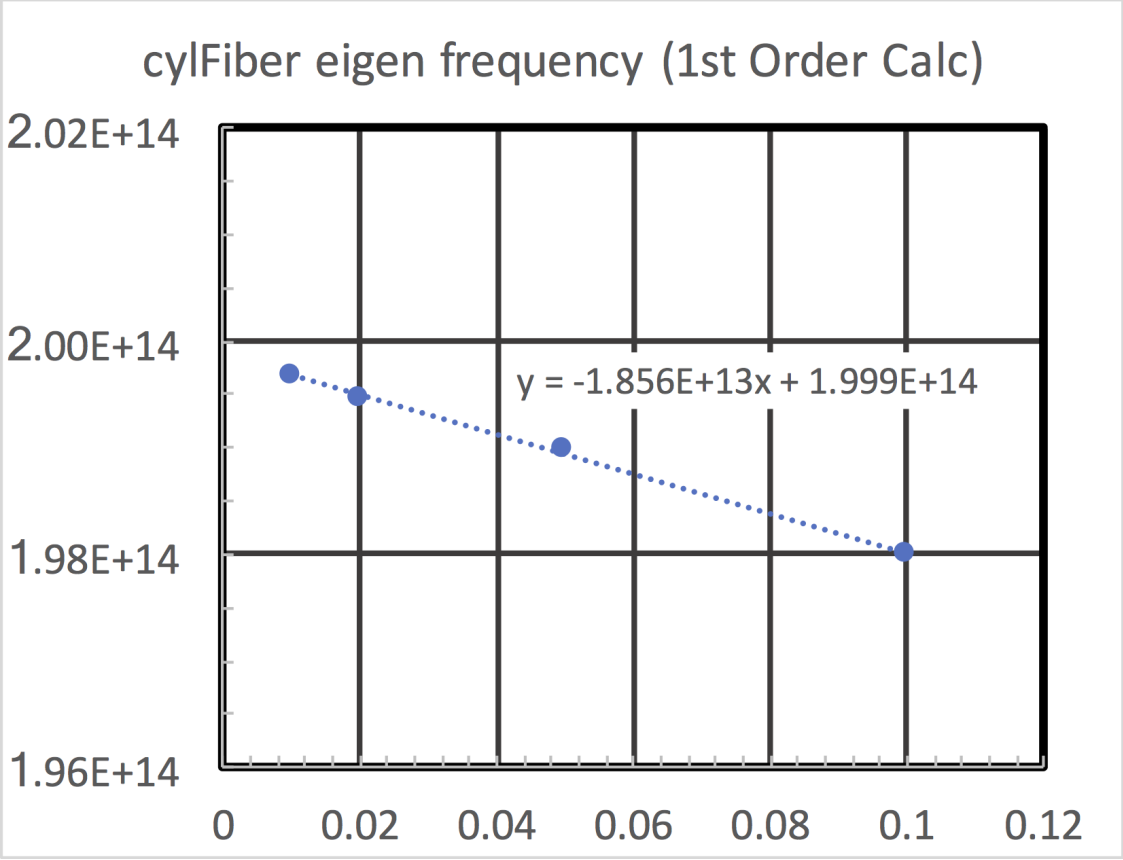


Fig. 3.83: Convergence of the first mode.

used.

Further Experiments

This same process can be used to get the frequency of modes of different wavelengths or of waveguides of different cross sections or made of different dielectrics.

3.3.3 Dielectric Waveguide with Gaussian Launcher (dielectricWaveguideGaussian.sdf)

Keywords:

Photonic Waveguide, Unidirectional Mode Launcher, MAL, Guided Mode, Semiconductor

Problem description

The dielectric waveguide consists of a single, straight silicon waveguide that is parallel to the x-axis and centered at the origin. The waveguide is surrounded by silica. Matched Absorbing Layers (MALs) are used to dampen the E and B fields near the boundary of the simulation to suppress reflected fields.

A gaussian approximation of the fundamental guided mode is launched in the silicon waveguide in the +x direction. The fundamental mode was extracted in the Dielectric Waveguide Mode Calculation Example.

This simulation can be performed with a VSimEM license.

Opening the Simulation

The dielectric waveguide example can be accessed from within VSimComposer through the following steps:

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window, expand the *VSim for Electromagnetics* option.
- Expand the *Photonics* option.
- Select *Dielectric Waveguide with Gaussian Launcher* 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 will now be available in the Setup window as shown in Fig. 3.84. 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. 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. Some relevant constants are listed below.

PERMITTVITY_WAVEGUIDE and PERMITTVITY_BACKGROUND: Relative permittivities of silicon and silica. These constants are used in multiple parameters and in the accompanying Python file for solving the waveguide modes.

LENGTH_UNIT: The constant factor by which VSim will scale all simulation lengths.

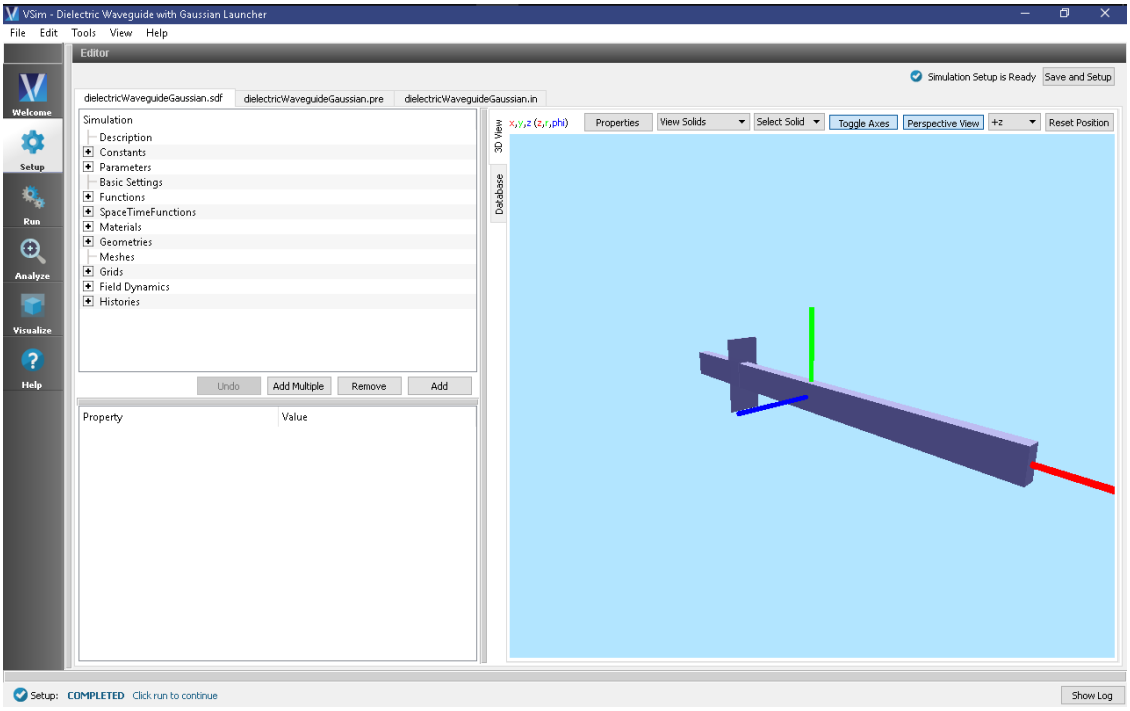


Fig. 3.84: The Setup window for the dielectric waveguide example showing some relevant constants.

WAVELENGTH_VAC: Wavelength of the input signal. This wavelength is also used for the calculation of the fundamental guided mode of the device.

NWAVELENGTH_MAL: Approximate number of wavelengths that can fit in a MAL region. The thickness of the MAL regions in this example are measured in wavelengths.

The *Materials* section contains just silicon and silica.

The *Geometries* 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. The gaussian approximation is defined under *SpaceTimeFunctions* and is set to drive the y-component of the *currentSource*.

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. You will be asked to Save. Click *Save*.
- To run the file, 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." The result is shown in Fig. 3.85.

Visualizing the Results

Then proceed to the Visualize window by pressing the *Visualize* button in the left column.

You can verify that the geometry is correct by visualizing the inverse permittivity as follows:

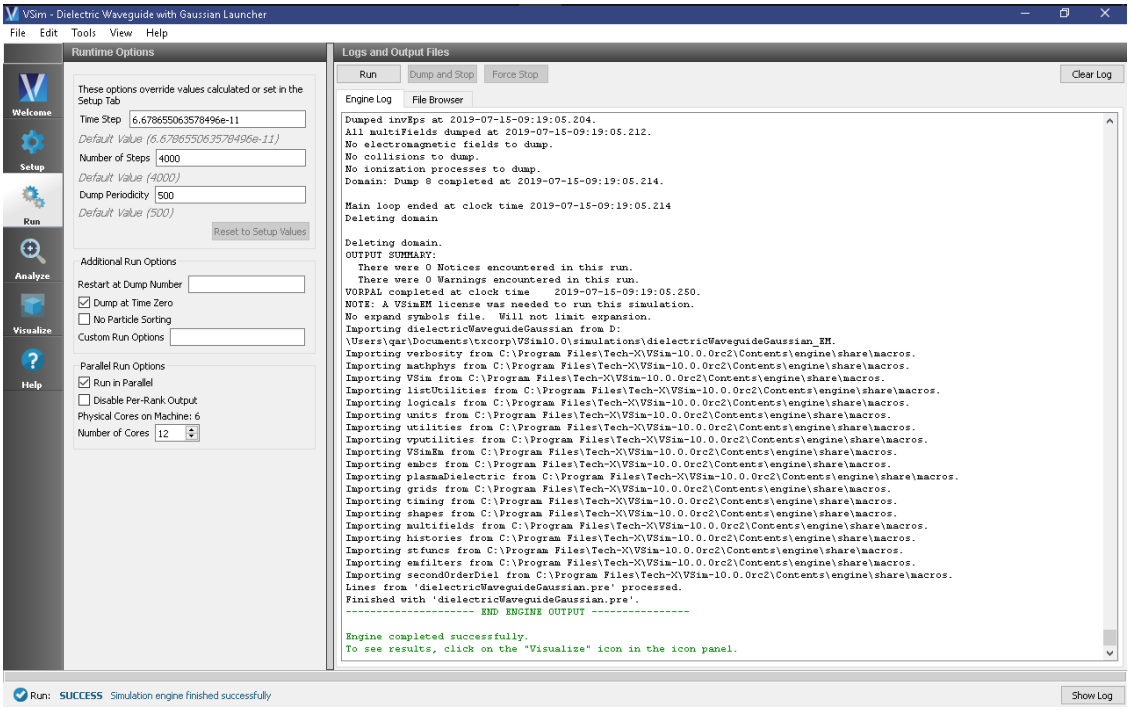


Fig. 3.85: The output after a successful run.

- Near the top left corner of the window, make sure *Data View* is set to *Data Overview*.
- Expand *Scalar Data*, expand *invEps*, and select *invEps_z*
- In the controls below the variables frame, select *Clip All Plots*.

By default, the clipping plane is at z = 0, which is in the middle of dielectric waveguide structure height-wise. As such, this will reveal the dielectric waveguide's 2D layout geometry as seen in Fig. 3.86.

A useful visualization of the dielectric waveguide would be to view the Z component of the B field to qualitatively see the mode propagate down the waveguide.

- Near the top left corner of the window, make sure *Data View* is set to *Data Overview*.
- Expand *Scalar Data*, expand *B*, and select *B_z*
- In the controls below the variables frame, select *Clip All Plots*.

Select the final dump step (dump 8) on the lower right of the screen using the slide bar. Fig. 3.87 shows an example of what one should expect if one has run the simulation for enough cycles.

Further Experiments

One can experiment by changing constants or introducing a different signal to drive the waveguide.

3.3.4 Dielectric Waveguide Mode Calculation (dielectricWaveguideModeCalc.sdf)

Keywords:

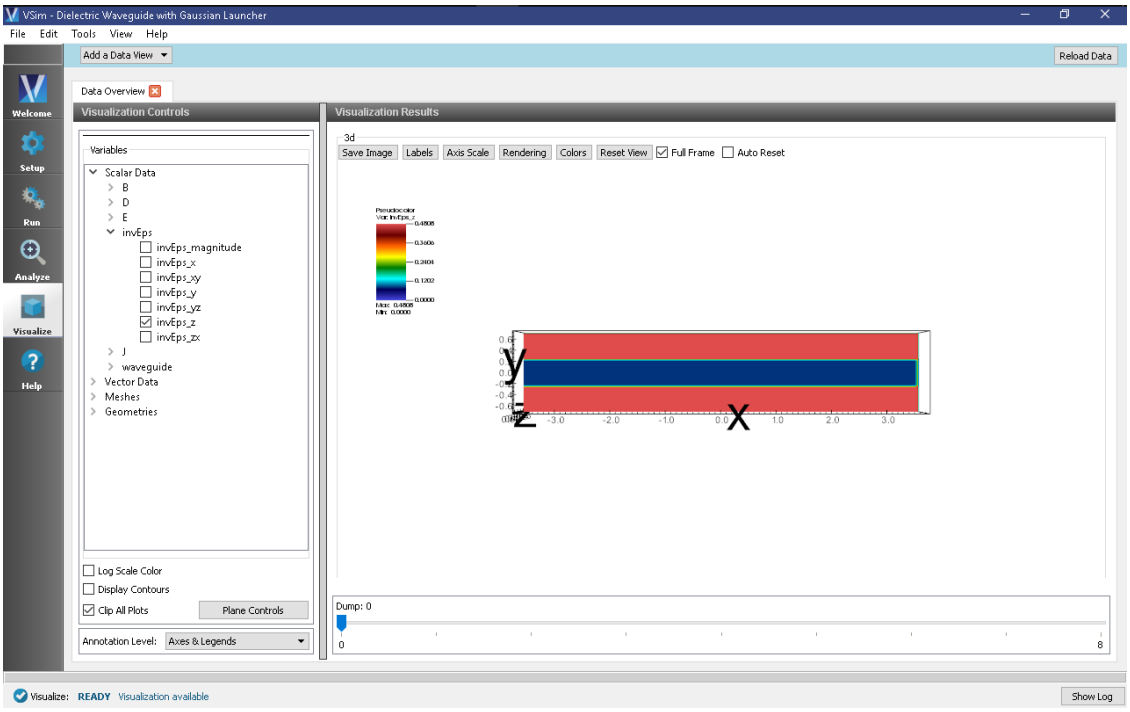


Fig. 3.86: Visualization of inverse epsilon field's Z component

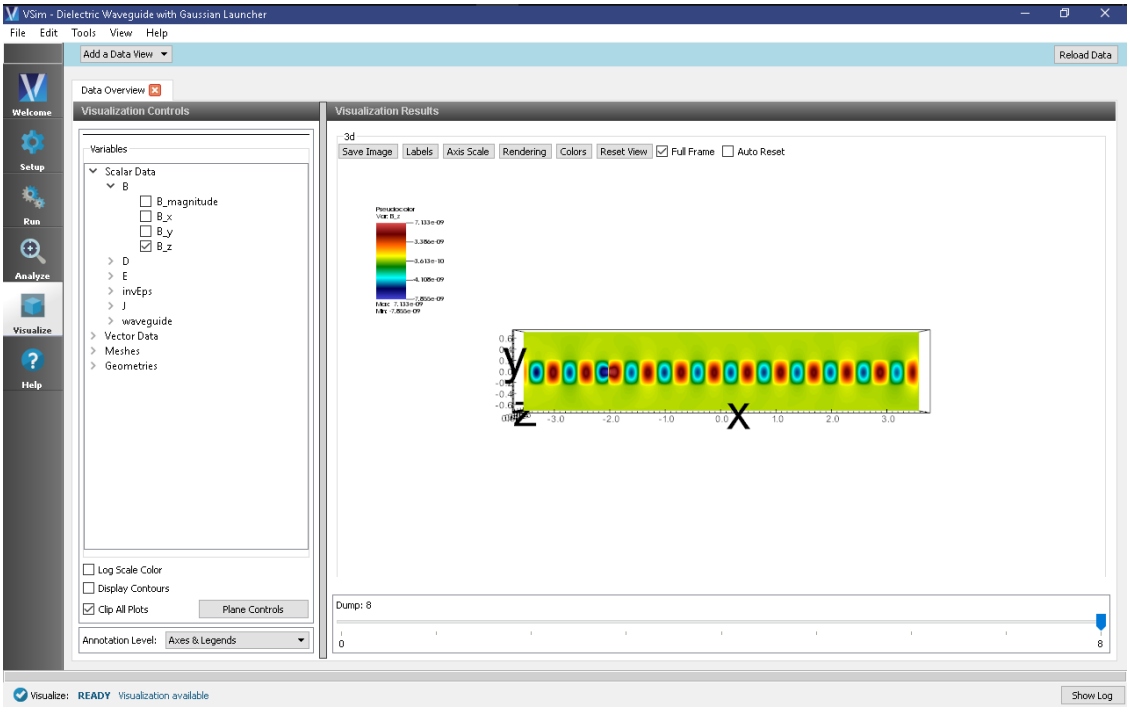


Fig. 3.87: Visualization of the B field's Z component

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 entire basis set of modes for this waveguide and output each into a separate .vsh5 file. These mode files can be used to launch the exact modes into your simulation. This process is shown in the multiModeFiberModeLaunchT example.

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 VSimEM license.

Opening the Simulation

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

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim 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.

- PERMITTVITY_WAVEGUIDE and PERMITTVITY_BACKGROUND: Relative permittivities of silicon and silica. These constants are used in multiple parameters and in the accompanying Python file for solving the waveguide modes.
- LENGTH_UNIT: The constant factor by which VSim will scale all simulation lengths.
- WAVELENGTH_VAC: Wavelength of the input signal. This wavelength is also used for the calculation of the fundamental guided mode of the device.
- NWAVELENGTH_MAL: Approximate number of wavelengths that can fit in a Matched Absorbing Layer (MAL) region. The thickness of the MAL regions in this example are measured in wavelengths.

The *Materials* section contains just silicon and silica. The *Geometries* 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. The gaussian approximation is defined under *SpaceTimeFunctions* and is set to drive the y-component of the *currentSource*.

Setting up the 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, expand the *Basic Settings* branch and verify that the *dielectric solver* field is set to *permittivity averaging*. This algorithm is a powerful VSim feature. This setting is shown in Fig. 3.88.

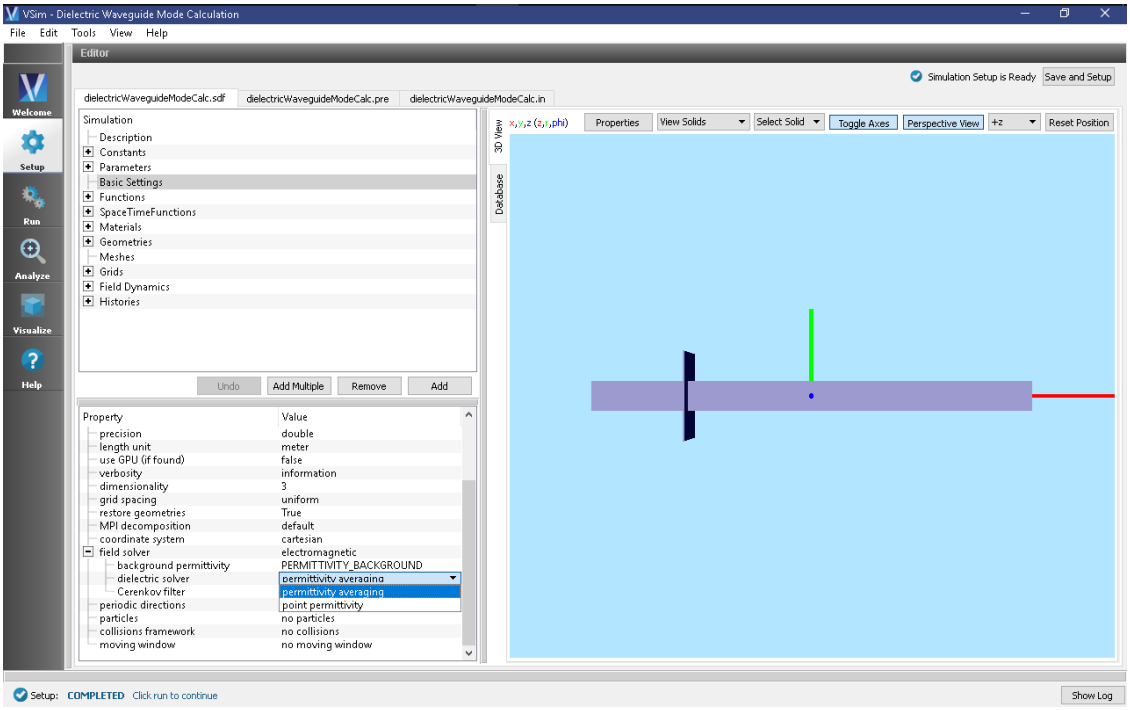


Fig. 3.88: Choosing the second order accurate, *permittivity averaging* for the *dielectric solver* field under *Basic Settings*.

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 change the *Number of Steps* and *Dump Periodicity* to 1.
- Under Additional Run Options select *Dump at Time Zero*.
- To run the file, 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. 3.89.

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 *computeDielectricModes.py* and click *Open* under the list.

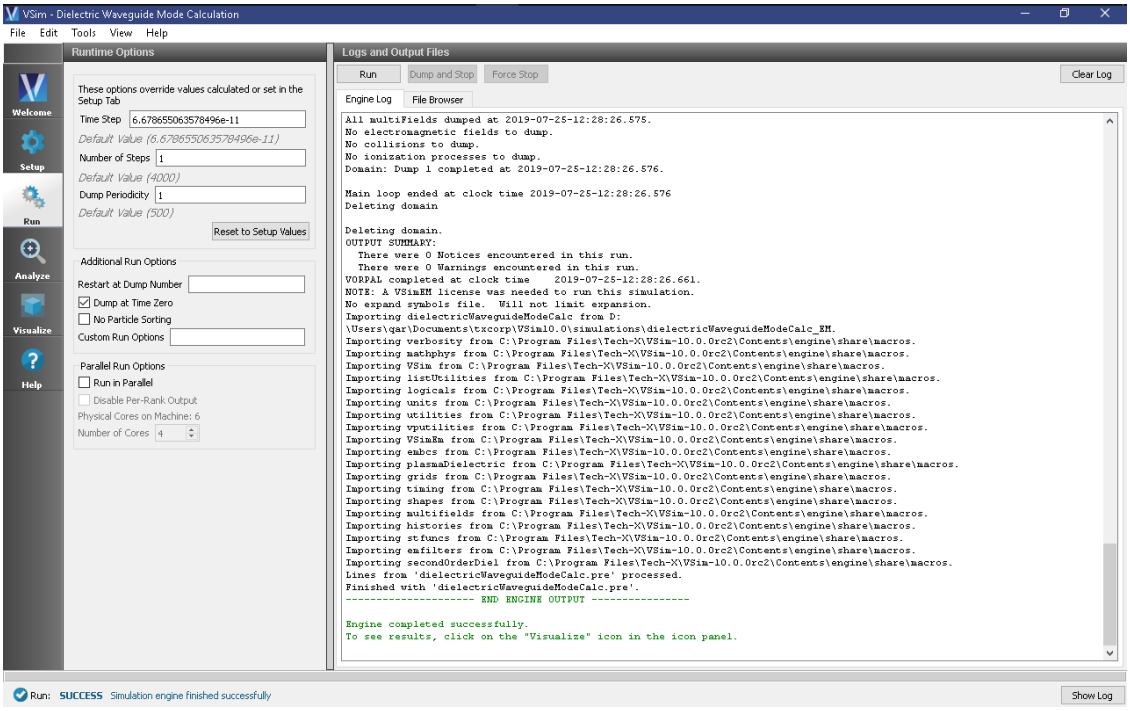


Fig. 3.89: Running the simulation for one step to get the permittivity data for the analyzer. Note that *Dump at Time Zero* is checked.

Now update the analyzer fields accordingly. Some of these parameters are described above under **Parameters**

- *transverseSlice*: 0,-.7:.7,-.5:.5
- *vacWavelength*: 1.55
- *nModes*: 10
- *writeFieldProfile*: H,E,D

We set the number of modes (*nModes*) to a value greater than the number of modes we expect. The analyzer will only find guided modes. Also check *Overwrite Existing Files*. Run the analyzer by clicking *Analyze* button in the upper right corner. The analyzer output should resemble Fig. 3.90. We see that the analyzer found 3 modes. They are listed in decreasing order of effective index.

Visualizing the Results

After performing the above actions 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*.
- Below the visualization, the dump slider will allow you to scroll between the modes.

The resulting visualization pane should resemble Fig. 3.91.

One can select other components of the H, E, or D field to see how they vary for the eigenmodes. These eigenmodes are now saved in .vsh5 files in the folder where the simulation was run.

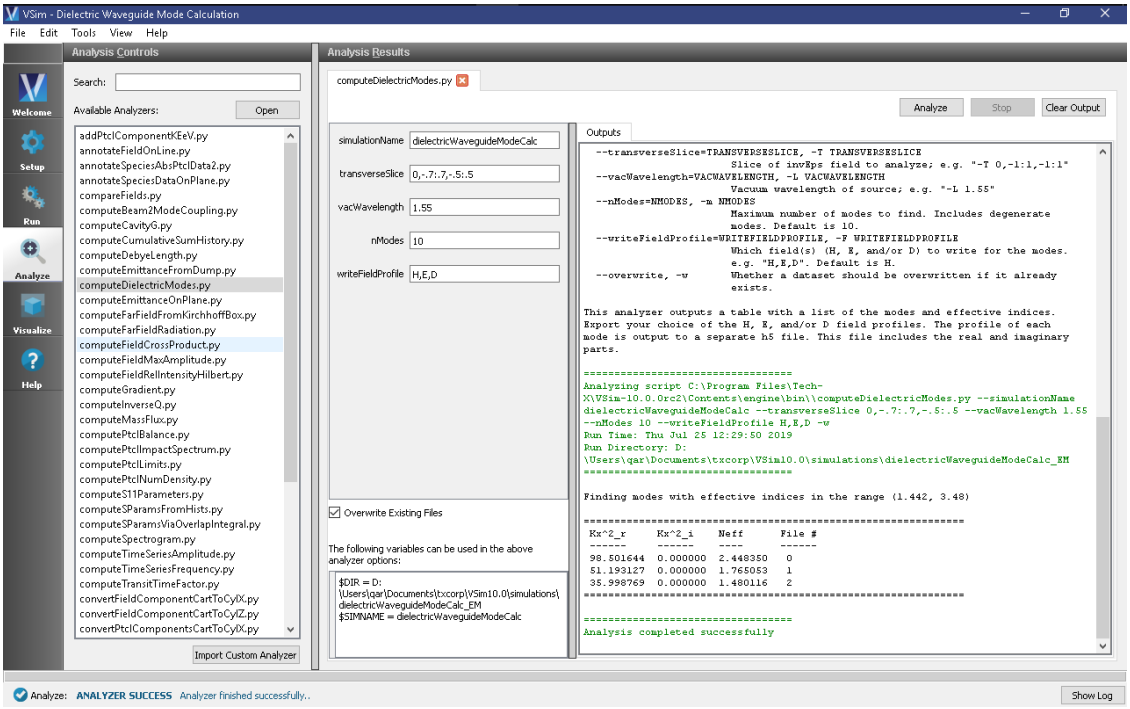


Fig. 3.90: The analyzer window after a successful run of computeDielectricModes.py.

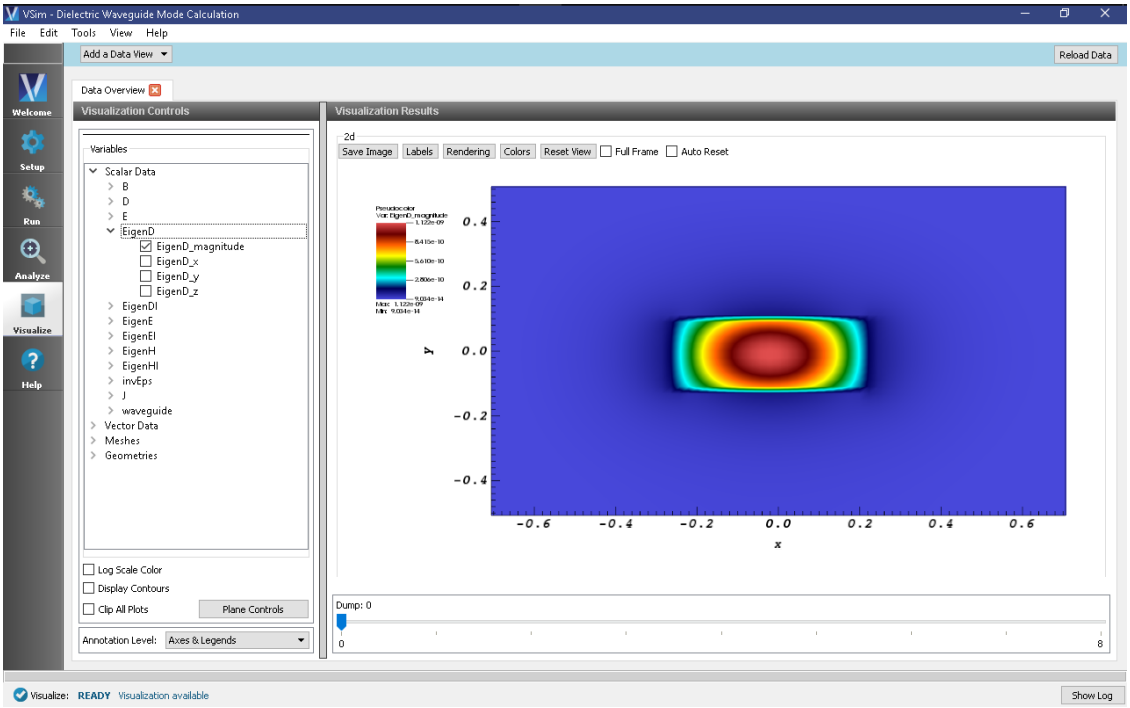


Fig. 3.91: The visualization pane showing the magnitude of the D field of the fundamental mode.

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 in the multiModeFiber-ModeLaunchT example.

One can run a full convergence study of eigenmode effective indices by varying the RESOLUTION constant 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. 3.92. The linear relationship shows the second order accuracy of our dielectric algorithms.

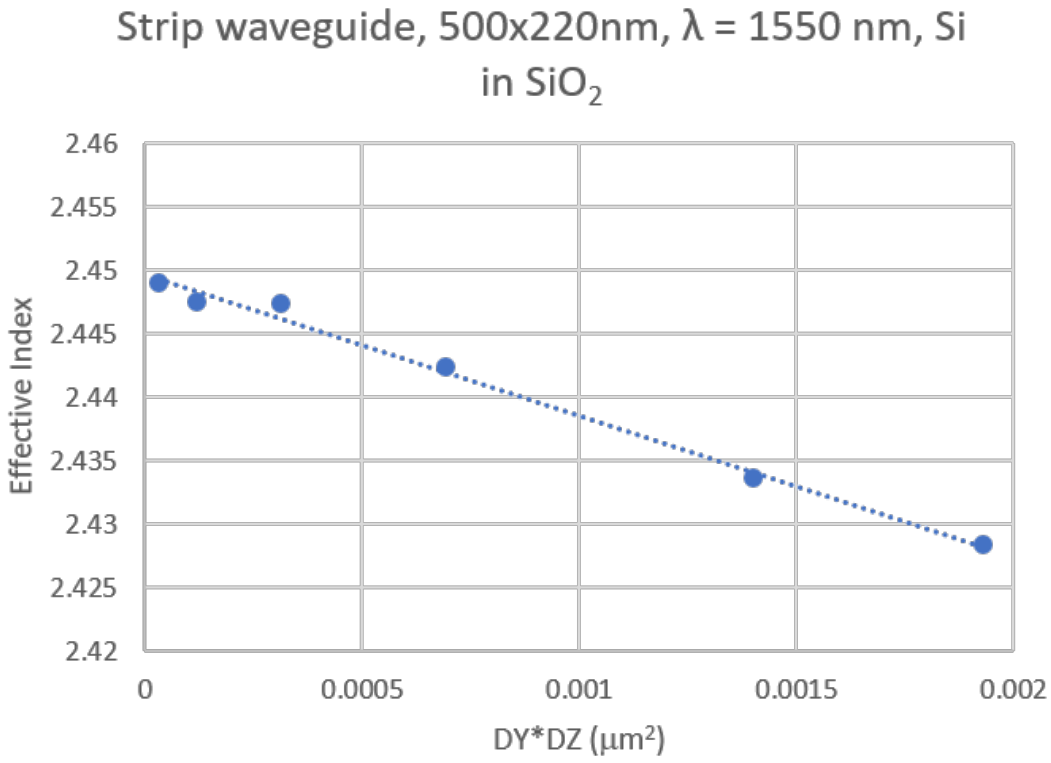


Fig. 3.92: The effective index as a function of transverse cell area for an eigenmode.

3.3.5 Ring Resonator (ringResonator.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 computeDielectricModes.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.

This simulation can be performed with a VSimEM license.

Opening the Simulation

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

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim 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. 3.93. 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*.

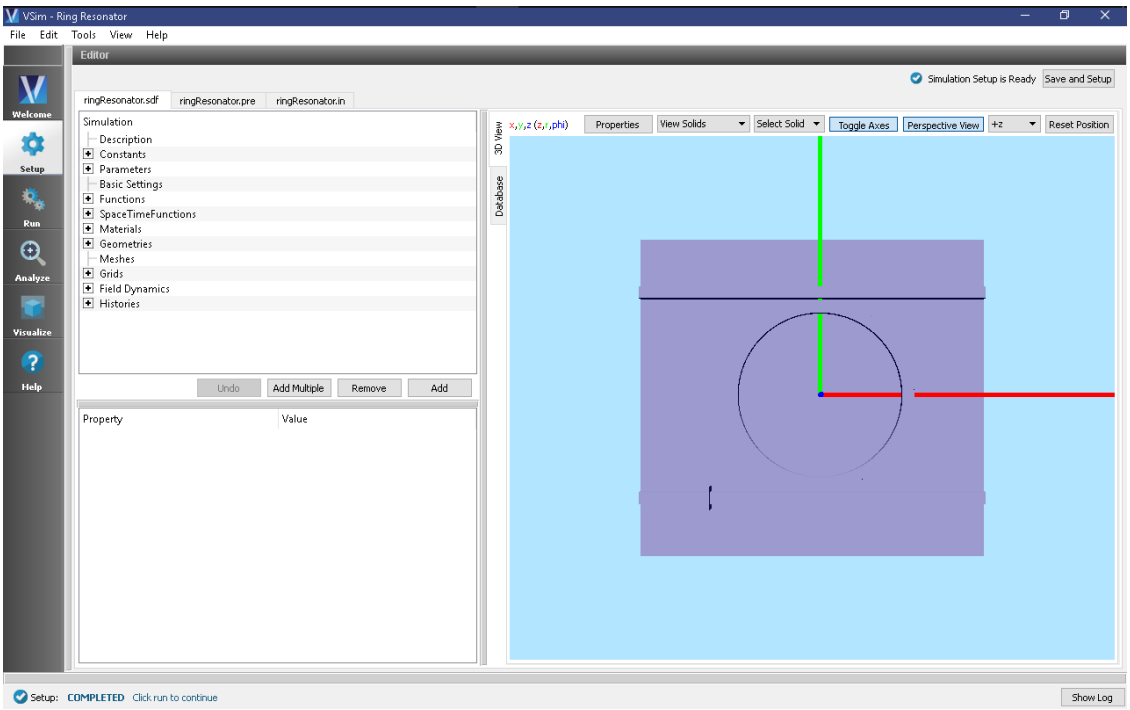


Fig. 3.93: The Setup window for the ring resonator example showing the external mode launching field.

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
- 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. 3.93.

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.
- When you are finished setting run parameters, 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. 3.94.

Analyzing the Results

Using post analysis scripts, one can extract the transmission coefficients. This is possible due to the field slab histories that are located at each port in the simulation. Each port has an E and B slab history in order to integrate over the poynting flux. This integration is done in a post analysis script called 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*: 1.7
- *minWavelength*: 1.4
- *inSLabE*: eSlab0
- *inSLabB*: bSlab0
- *outSLabE*: eSlab1

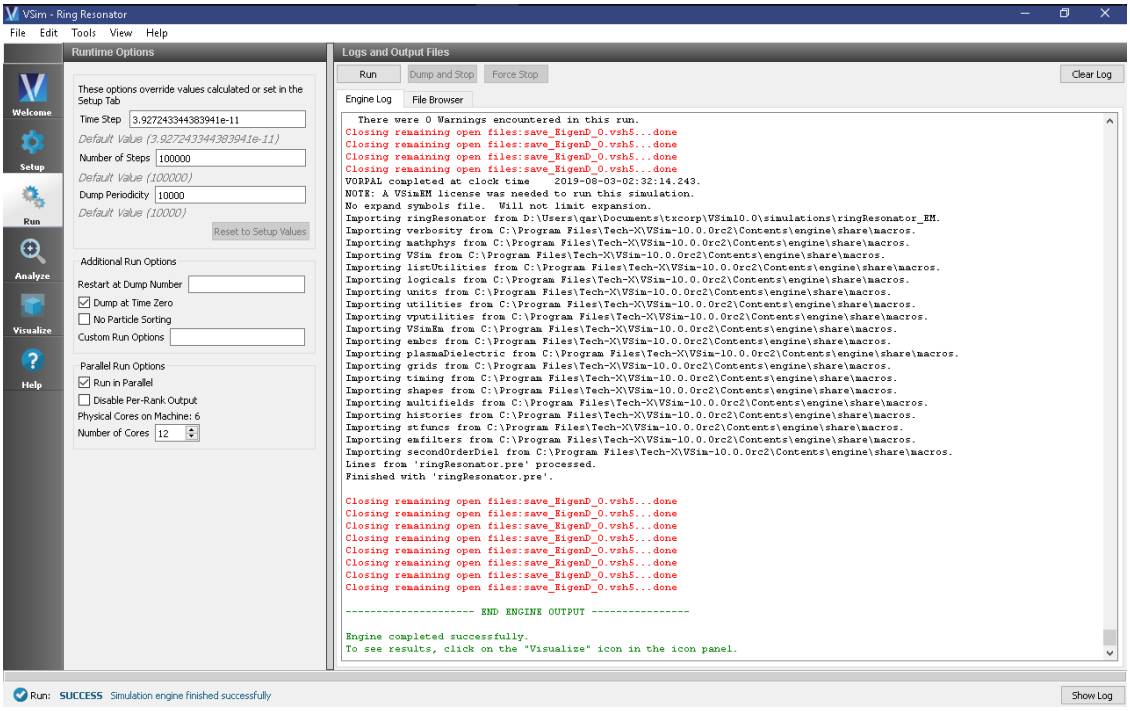


Fig. 3.94: Run window at completion.

- *outSLabB*: bSLab1

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. 3.95. Continue by analyzing the drop-port. To do this change the *outSlabE* and *outSlabB* fields to eSlab3 and bSlab3, respectively.

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=.1 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. 3.96.

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. 3.97. As expected, we see coupling of resonant wavelengths with the ring.

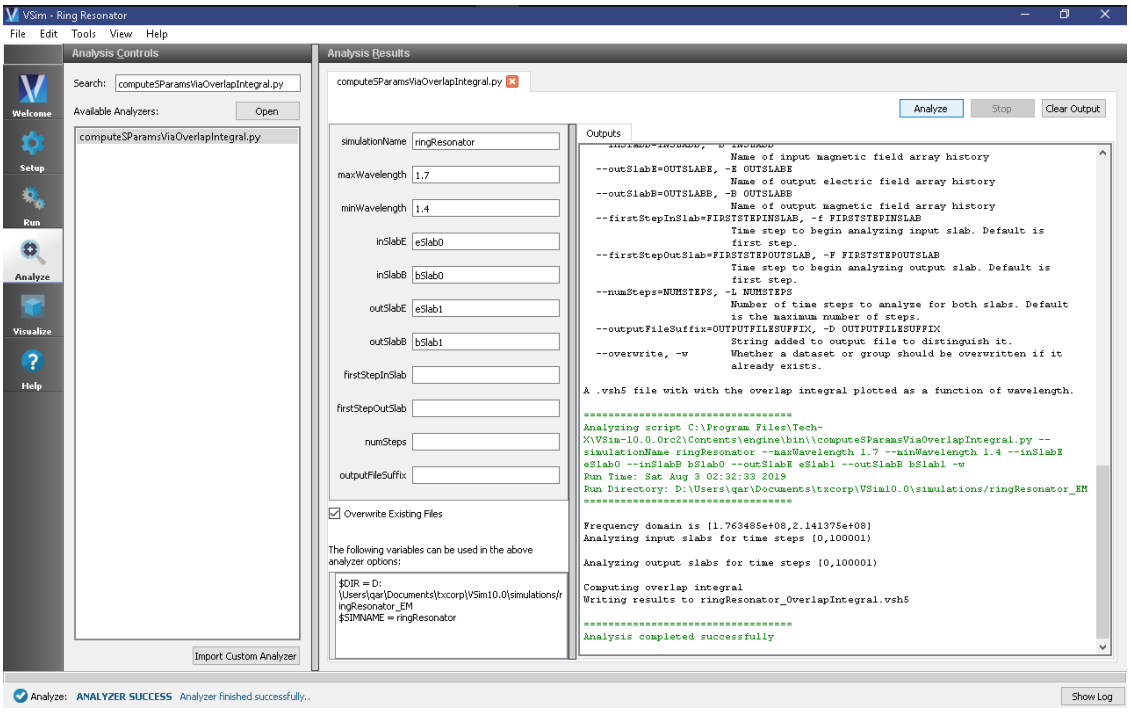


Fig. 3.95: Analyze window with output from computeSParamsViaOverlapIntegral.py.

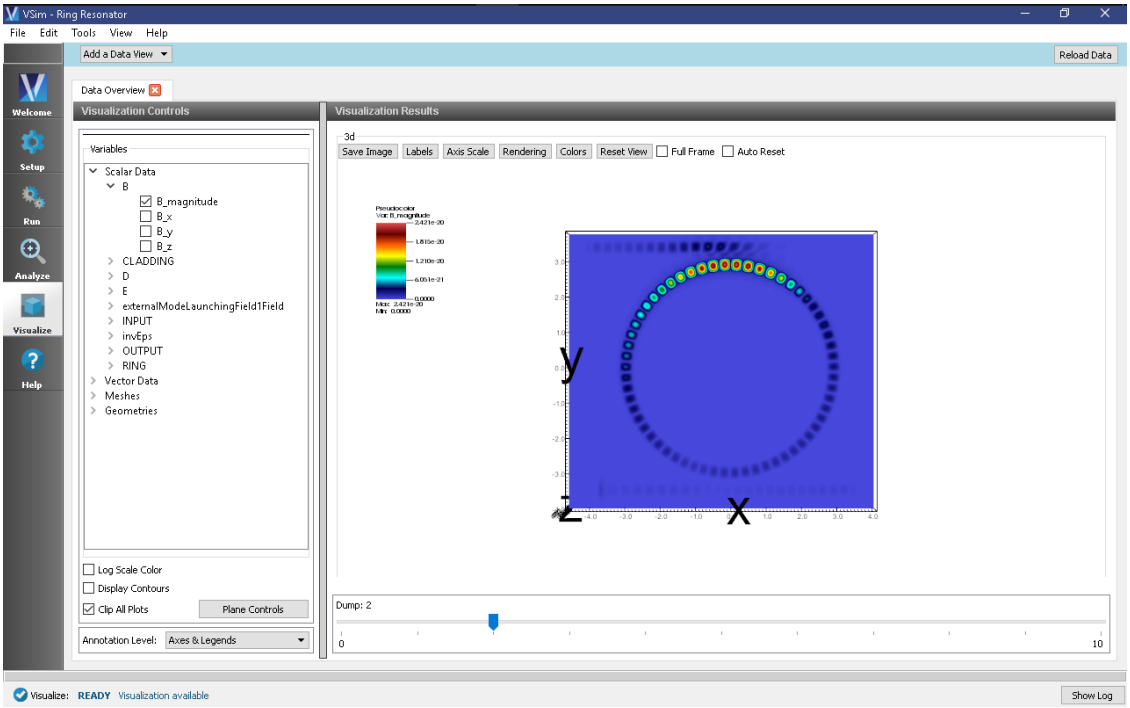


Fig. 3.96: Visualization of magnetic field.

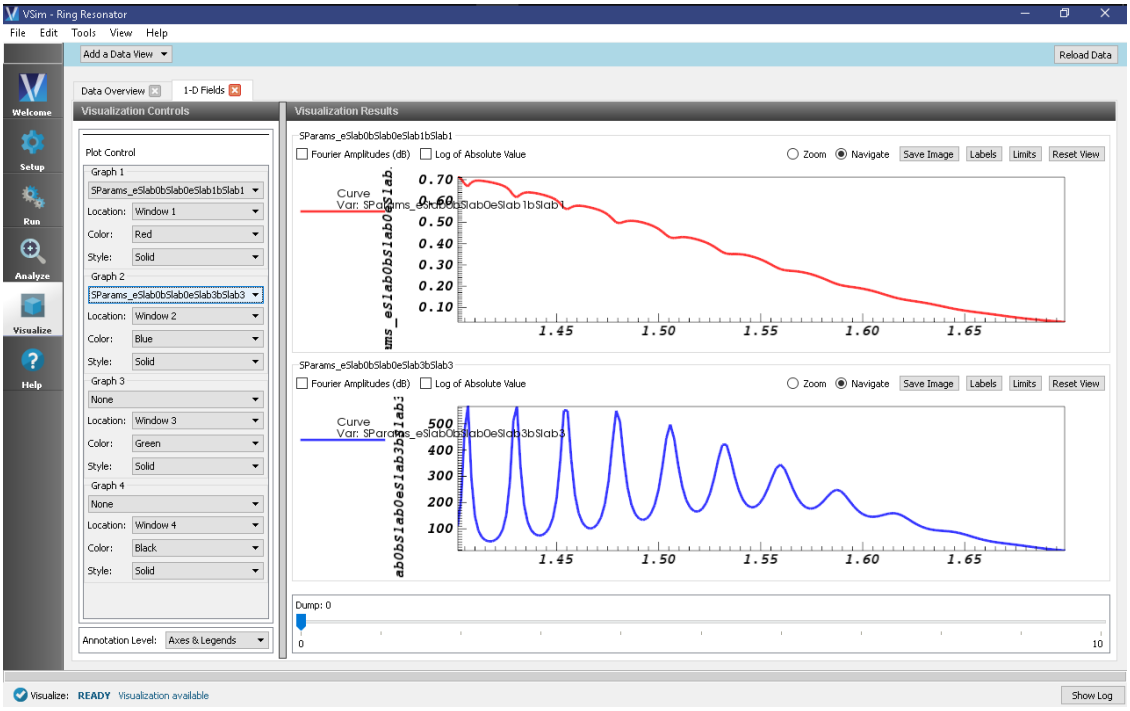


Fig. 3.97: Visualization of the s-parameters.

3.3.6 Gaussian Laser Beam and Photonic Crystal Cavity (photonicCrystalGaussSrc.sdf)

Keywords:

`Gaussian Beam source, photonic crystal, transmission efficiency`

Problem description

This example illustrates how to model a Gaussian beam source that is illuminating a cavity inside a hexagonal photonic crystal lattice. The physical setup is shown in Fig. 3.98.

A Gaussian beam is launched from above into the simulation domain, which comprises three layers: a vacuum region above and a solid dielectric below, which together sandwich a central dielectric layer that contains a lattice of holes. This example includes two possible time signals with which the Gaussian beam will have either WideBand or SingleFrequency.

This simulation can be performed with a VSimEM license.

Opening the Simulation

The Photonic Crystal example is accessed from within VSimComposer through the following steps:

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window, expand the *VSim for Electromagnetics* option.
- Expand the *Photonics* option.
- Select "Gaussian Laser Beam and Photonic Crystal Cavity" and press the *Choose* button.

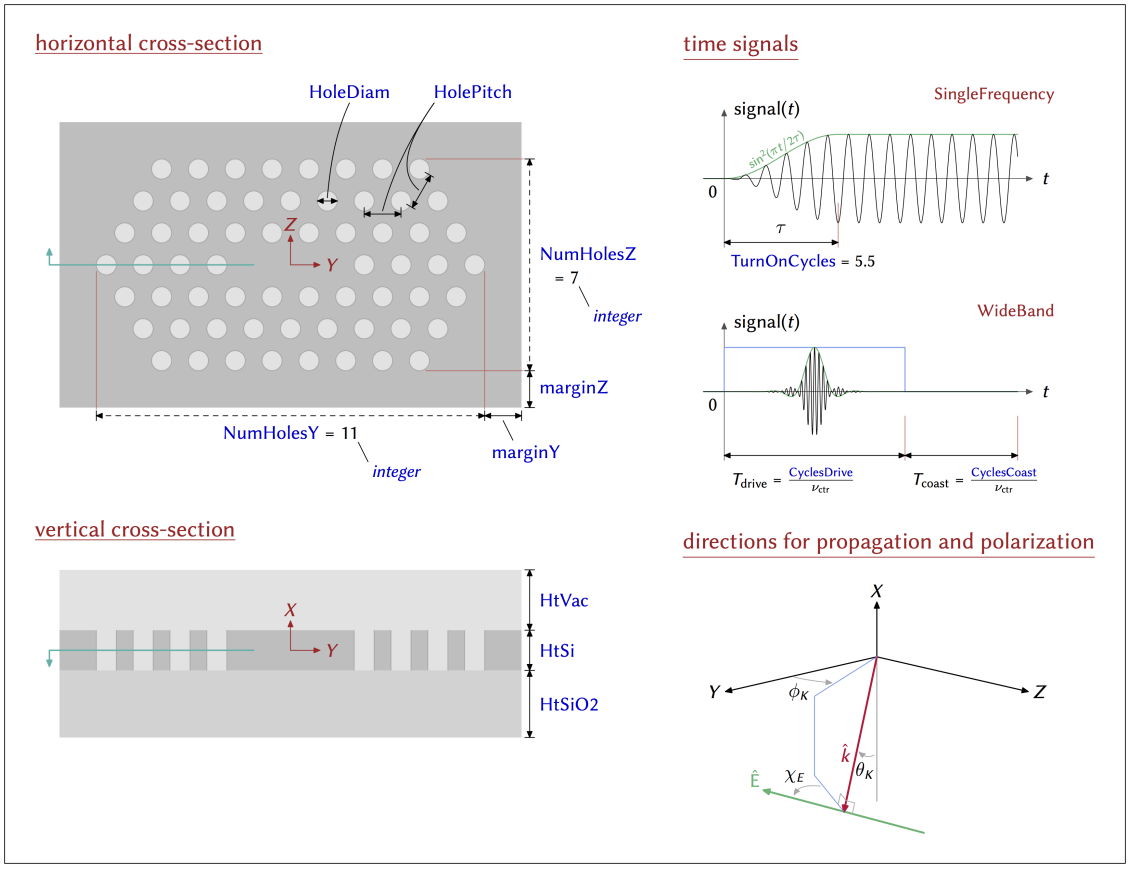


Fig. 3.98: The photonic lattice setup

- 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. 3.99. 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`.

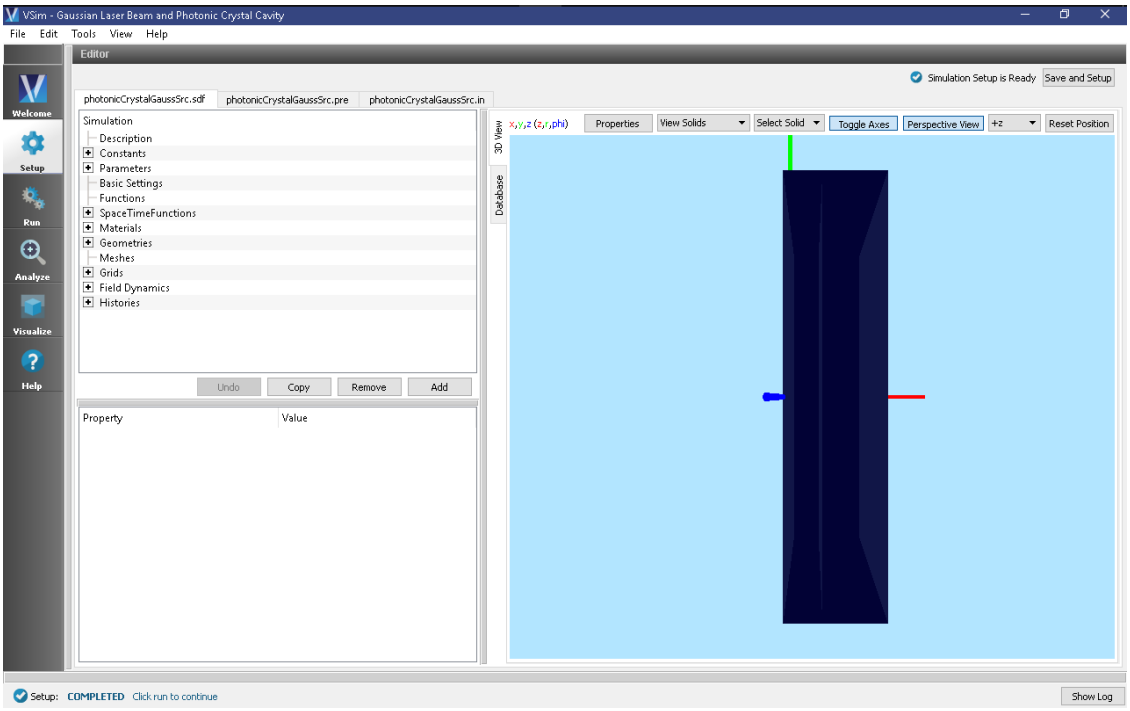


Fig. 3.99: The Setup Window for the Gaussian Laser Beam and Photonic Crystal Cavity example

Simulation Properties

This example contains a number of constants that are defined to make the simulation easily modifiable, as can be seen in Fig. 3.100.

All the following constants should be the only properties you should need to alter in order to specify your simulation domain.

General Simulation Parameters:

- `L{X,Y,Z}` = The length of your simulation domain in the {X,Y,Z} dimension.
- `HT_{VACUUM,SI,SI02}` = The height of the vacuum, SI and SI02 layers of the photonic crystal.

Source Specifications: (located in the *Parameters* section of the *Elements Tree*)

- `{K_THETA,K_PHI,E_CHI}` = The {polar angle, azimuthal, angle of polarization} respectively.
- `WAVEL_CENTER` = The central wavelength of your wideband signal. This is also the frequency used in the single frequency simulation type.
- `WAVEL_BAND` = The wavelength width of your wideband signal, only used in wideBand simType.
- `BEAM_WAIST` = The width at which your beam power falls off like 1/e.
- `BEAM_POWER` = The amplitude of your E/M wave.

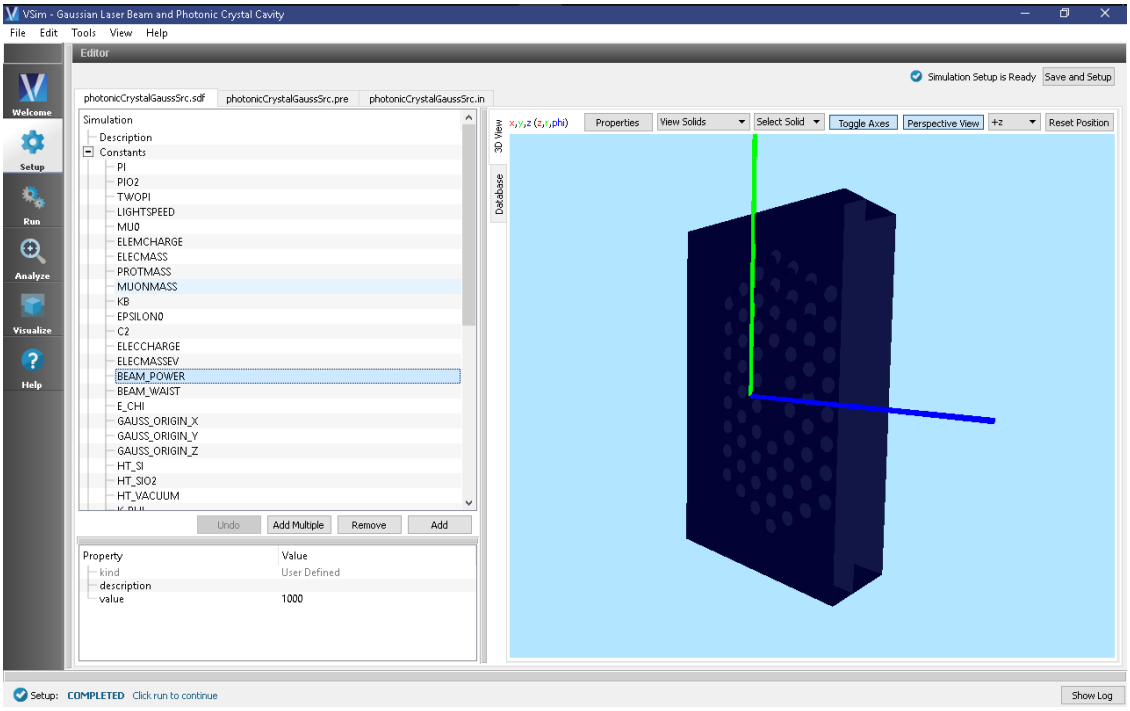


Fig. 3.100: The Setup Window showing the constants

- `GAUSS_ORIGIN_{X,Y,Z}` = The point around which your Gaussian profile is centered.
- `TURNONCYCLES` = The number of cycles you want your single frequency to reach full power.
- `SIMCYCLES` = Number of wave cycles you want your simulation to run.
- `CYCLESPERDUMP` = Number of cycles between each dump in the simulation.

The tool used to input the wave into the simulation is a port launcher. It specifies the Electric Displacement Field (D) at a boundary in this case the lower X boundary. The functions defining your D on the boundary are defined in the *SpaceTimeFunctions* element of the *Elements Tree*.

SpaceTimeFunctions:

- `dSingleFreq{Y,Z}` = This is the {x,y,z} component of the single frequency Gaussian beam source, as seen in Fig. 3.100. You put this as a parameter in the current distribution.
- `dWideBand{Y,Z}` = This is the {x,y,z} component of the wideband Gaussian beam source, as seen in Fig. 3.100. You put this as a parameter in the current distribution.

To choose which signal you want to input into this example simulation:

1. Expand the *FieldBoundaryConditions* element.
2. Left click on the *portLauncherLowerX* condition.

At this point, you should see what is shown in Fig. 3.101.

3. To add a signal to the *portLauncherLowerX* just right click on the D{x,y} and select *Assign SpaceTimeFunction*. This will expand another menu that will show you all four defined SpaceTimeFunctions. Select which one you want to input into your simulation. For this documentation, `WideBand{Y,Z}` will be used to demonstrate the functionality of this example.

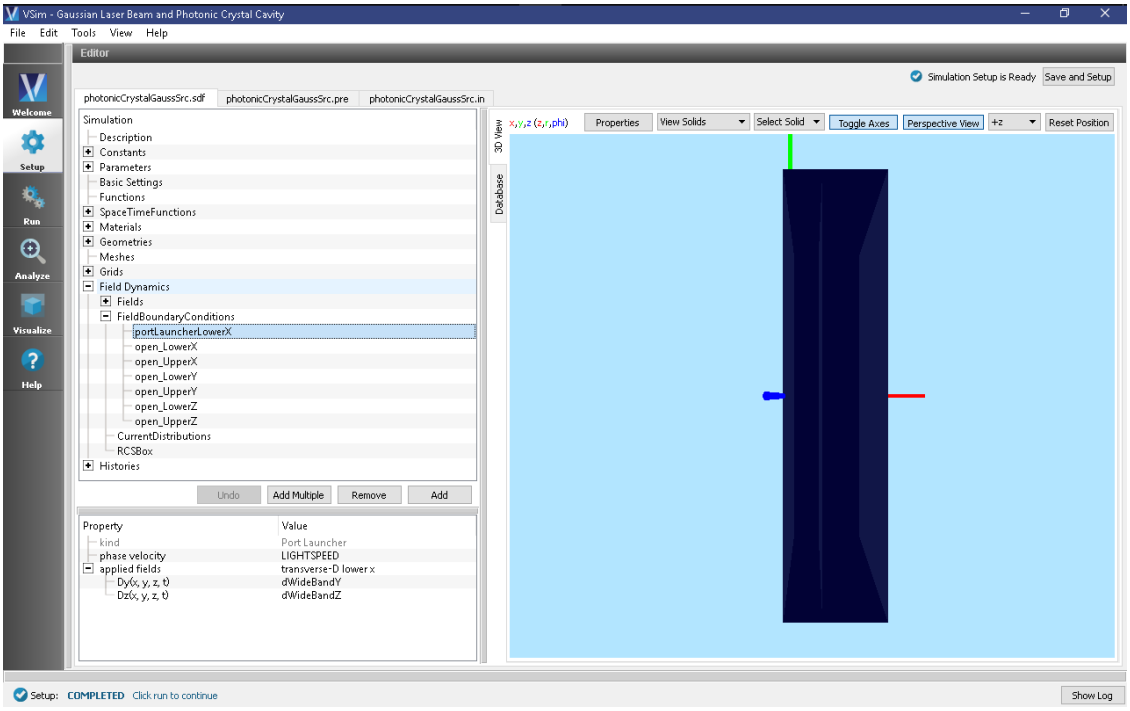


Fig. 3.101: The Setup Window for specifying the signal for the portLauncher

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.
- One can enable MPI options to utilize multi-core systems.
- The default values of `Number of Time Steps` and `Dump Periodicity` are taken from the parameters `STEPSTOTAL` and `STEPSPERDUMP`, which use the constants `SIMCYCLES` and `CYCLESPERDUMP`. The formulae for these variables can be found back in the *Setup Window*. These variables are for convenience to calculate good default values and it is important to know that the override option default values ultimately come directly from the numbers in the *Basic Settings* section.

  Number of Steps and *Dump Periodicity*. Just copy these values into the correct fields in the *Run* menu.

  - *Number of Steps* = `TOTALSTEPS` = 32220
  - *Dump Periodicity* = `STEPSPERDUMP` = 3600
- 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 the right pane. The run has completed when you see the output, "Engine completed successfully." This is shown in Fig. 3.102.

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.

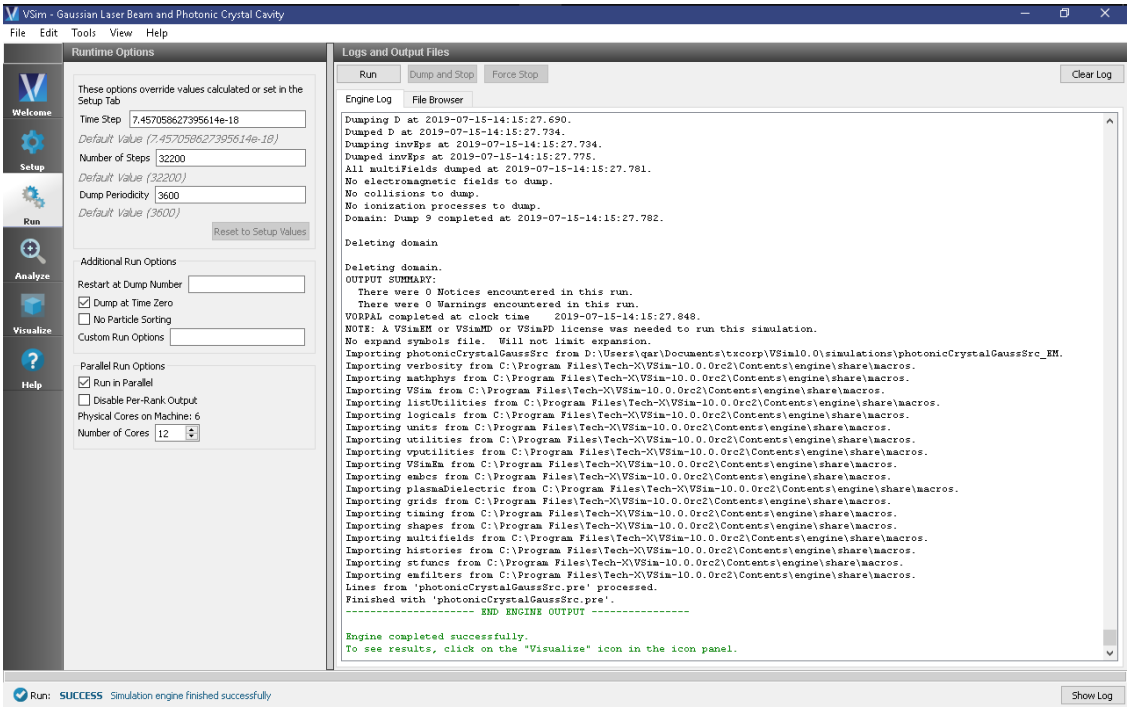


Fig. 3.102: The Run Window for the Gaussian Laser Beam and Photonic Crystal Cavity example

In the simulation, there are specific grid points which store field histories. These histories are placed in various positions of the simulation.

- Click the *Add a Data View* dropdown below the menu bar and select *History*.

In Fig. 3.103, one can see there are 4 possible graphs to view at one time in the *Visualize Window*. For each graph, one can select the following fields to analyze: (0 = x, 1 = y, 2 = z).

- {E,B}_AtDet_{0,1,2} is in the middle of (y,z) plane and 60 nm above the surface of the crystal.
- {E,B}_AtSrc_{0,1,2} is aligned with the inCav history in the (y,z) plane and is 60 nm below the Si layer, into the SiO2 layer.
- {E,B}_InCav_{0,1,2} is slightly offset from the middle of one of the cavities in the silicon layer (the layer with the lattice).

In each individual graph, one can choose the *Fourier Amplitudes (dB)* option to view the frequency domain of your field. This can enable the analysis of the frequency response of the photonic crystal cavity.

Fig. 3.103 depicts four graphs of histories. The first two graphs are amplitude vs time, and the second two are Fourier Amplitudes of the first two on a log scale.

The first and third graphs depict the history `AtSrc_2`, while the second and fourth graphs show the `AtDet_2` history.

Further Experiments

By using the wideband source and examining the field strength detected below the crystal lattice, one may study the frequency response of this photonic crystal as one changes the device geometry, the dielectric constants, and the location and polarizations of the radiation source and detector.

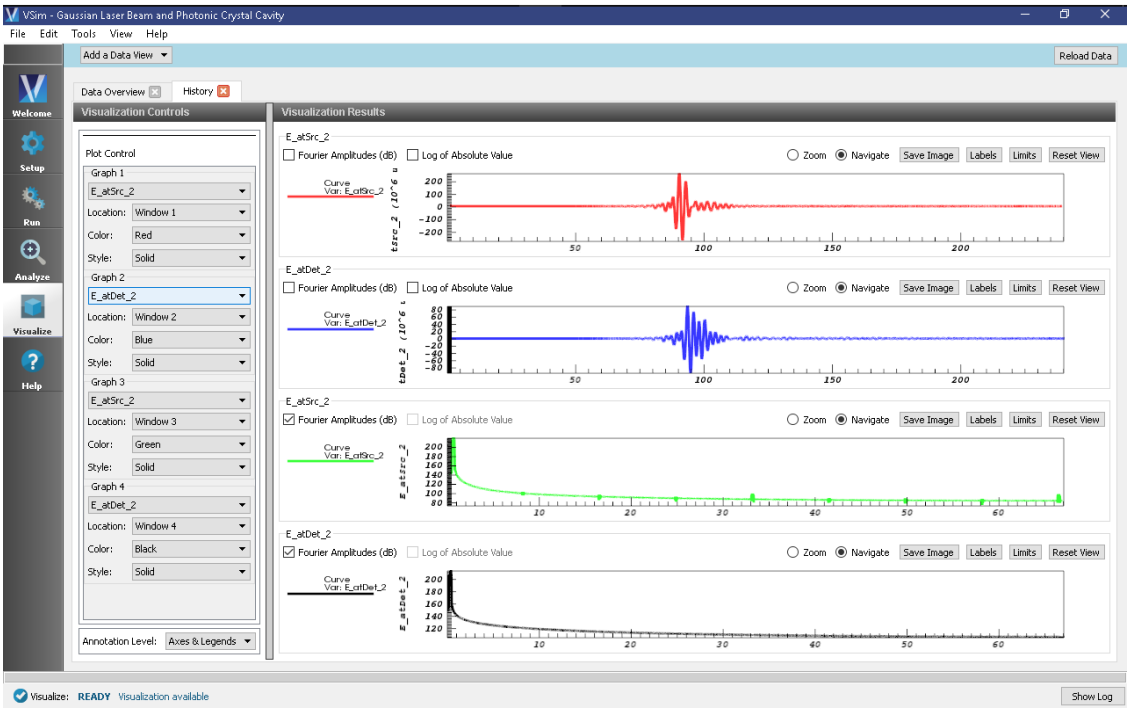


Fig. 3.103: The Visualize Window for the Gaussian Laser Beam and Photonic Crystal Cavity example

3.3.7 Dipole Source Illuminating a Photonic Crystal Cavity (photonicCrystalDipoleSrc.sdf)

Keywords:

dipole source, photonic crystal, transmission efficiency

Problem description

This example illustrates how to model a dipole source that is illuminating cavities inside a hexagonal photonic crystal lattice. The physical arrangement is shown in Fig. 3.104 and Fig. 3.105.

A point-like dipole lies above the simulation domain, which comprises three layers: a vacuum region above and a solid dielectric below, which together sandwich a central dielectric layer that contains a lattice of holes. This example includes two possible time signals with which to ring the dipole source, as shown in Fig. 3.106.

This simulation can be performed with a VSimEM license.

Opening the Simulation

This Photonic Crystal example is accessed from within VSimComposer through the following steps:

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window, expand the *VSim for Electromagnetics* option.
- Expand the *Photonics* option.

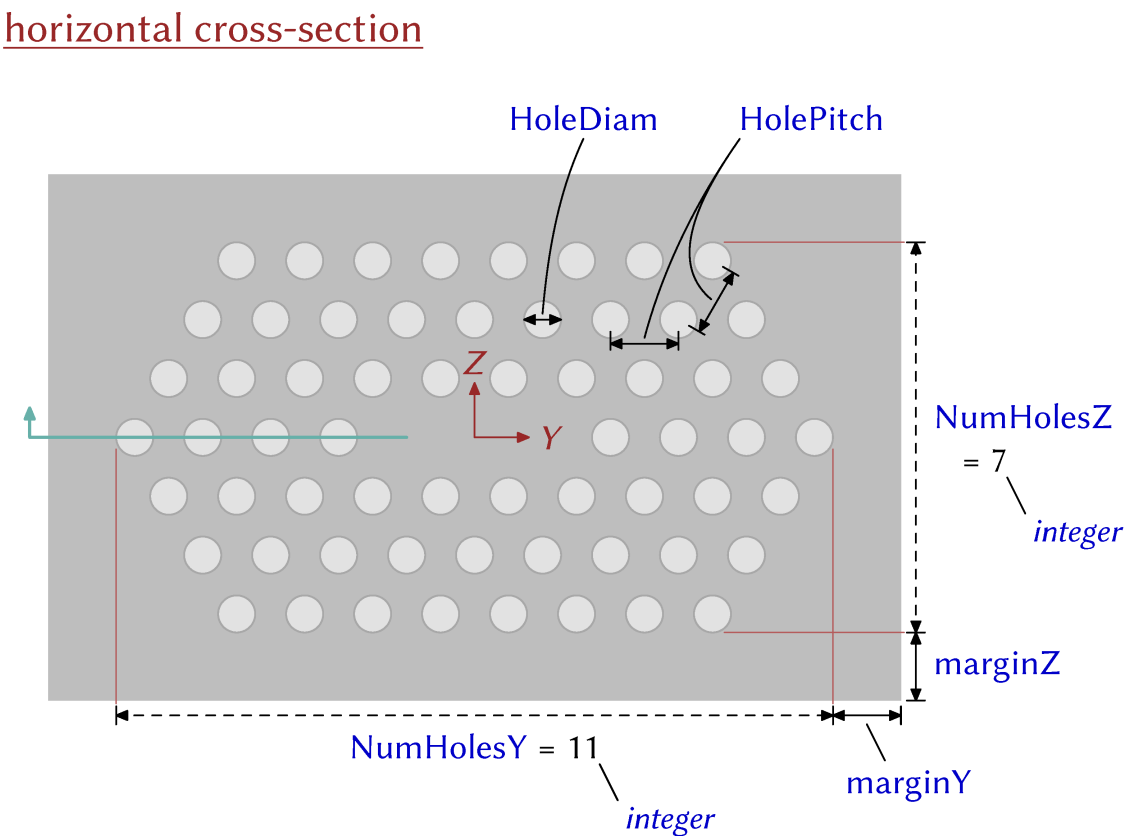


Fig. 3.104: Top view of photonic lattice.

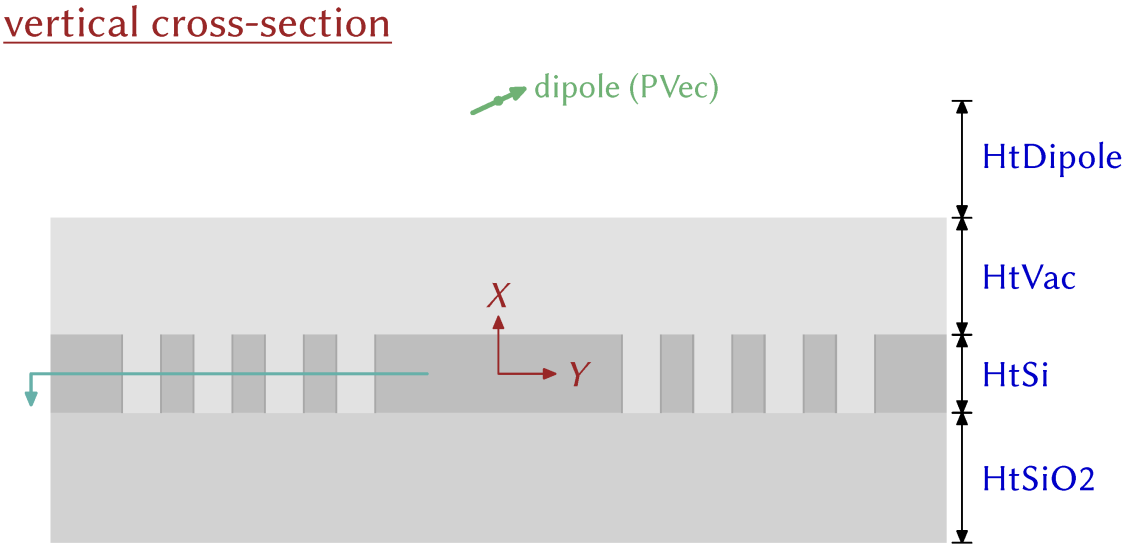


Fig. 3.105: Side view of photonic lattice.

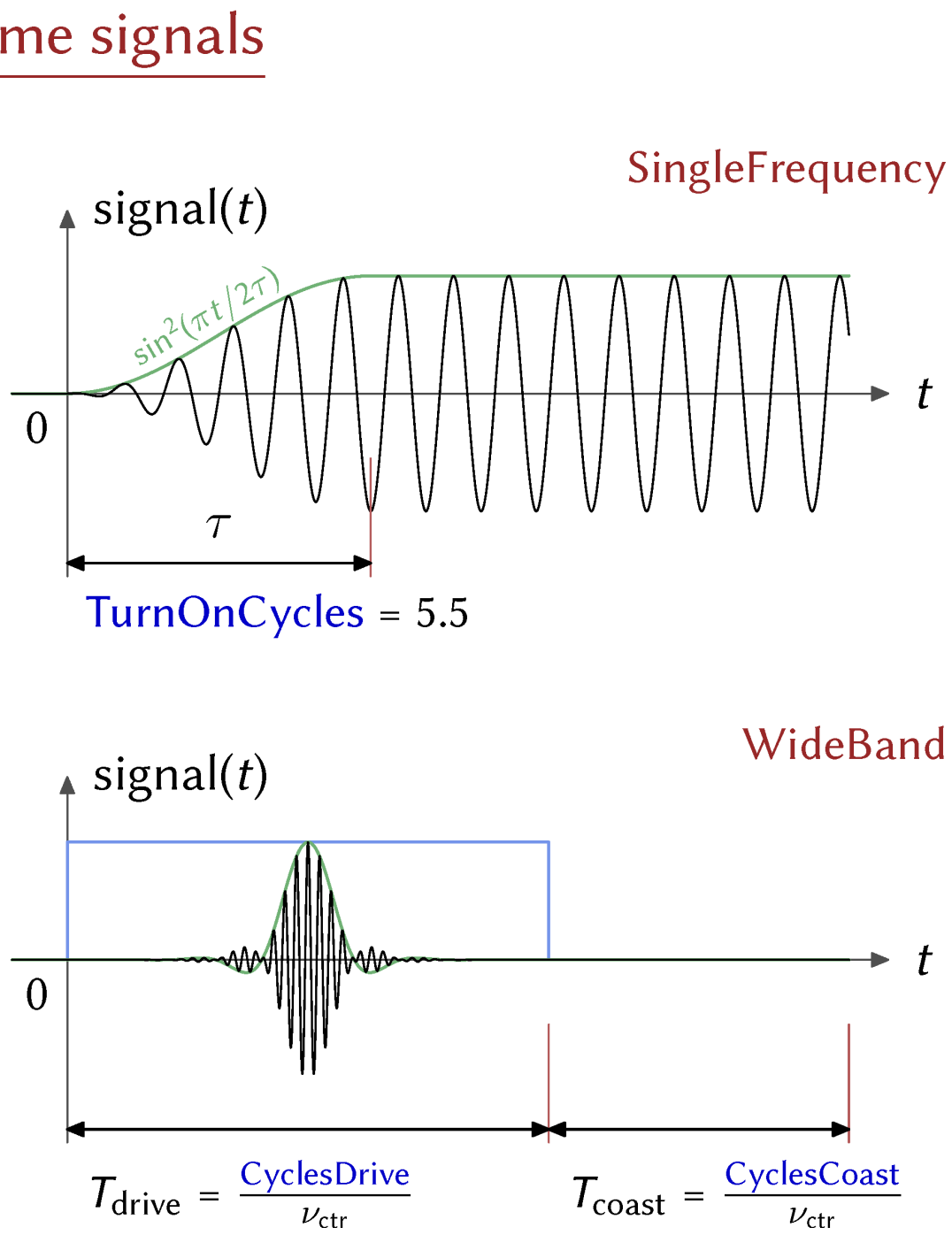


Fig. 3.106: Two possible time signals for ringing the dipole source.

- Select "Dipole Source Illuminating a Photonic Crystal 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.

All of the properties and values that create the simulation are now available in the *Setup Window* as shown in Fig. 3.107. 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 photonic crystal geometry. To show or hide the grid, expand the Grid element and select or deselect the box next to `Grid`.

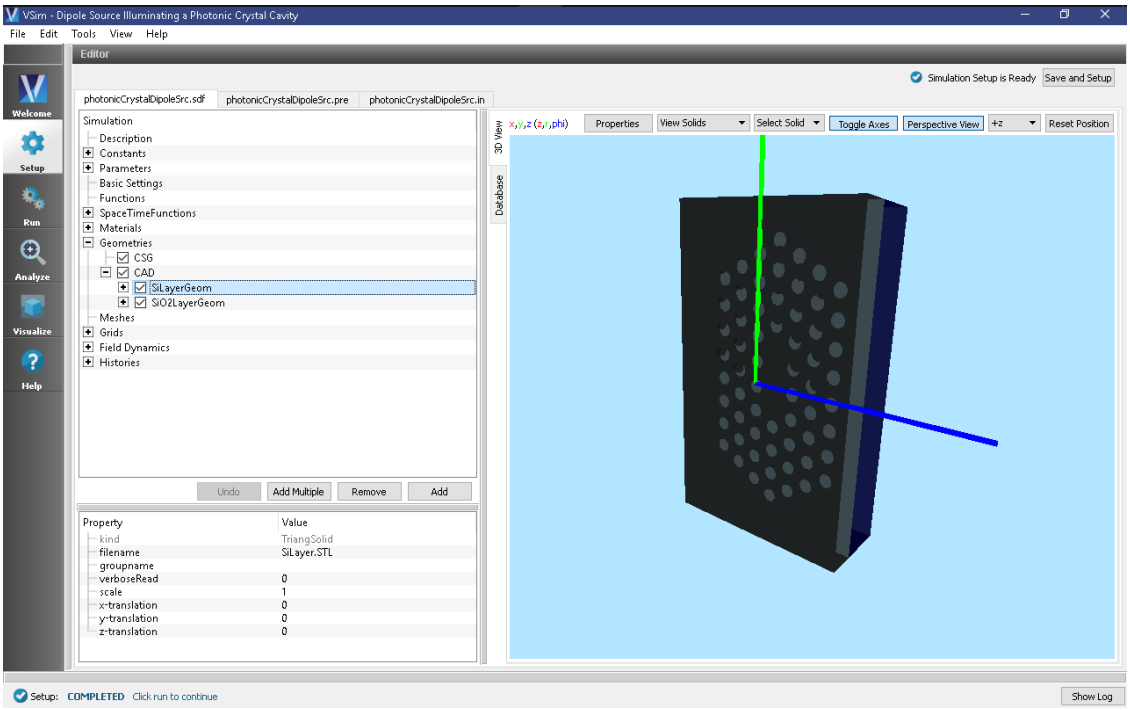


Fig. 3.107: The Setup Window for the Dipole Source Illuminating a Photonic Crystal Cavity example

Simulation Properties

This example contains a number of constants defined to make the simulation easily modifiable, as can be seen in Fig. 3.108.

The following constants should be the only properties you should need to alter in order to specify your simulation domain.

General Simulation Parameters:

- `CYCLES_DRIVE` = How many cycles at which the E/M source is driven.
- `HT_{VACUUM,SI,SI02}` = The height of the vacuum, SI and SI02 layers of the photonic crystal.
- `L{X,Y,Z}` = The length of your simulation domain in the {X,Y,Z} dimension.

Source Specifications: (located in the *Parameters* section of the *Elements Tree*)

- `TURNONCYCLES` = The number of cycles you want your single frequency to reach full power.
- `WAVEL_BAND` = The wavelength width of your wideband signal, if doing a wideband simulation.
- `WAVEL_CENTER` = The central wavelength of your wideband signal, and is the frequency used in the single frequency simulation type.

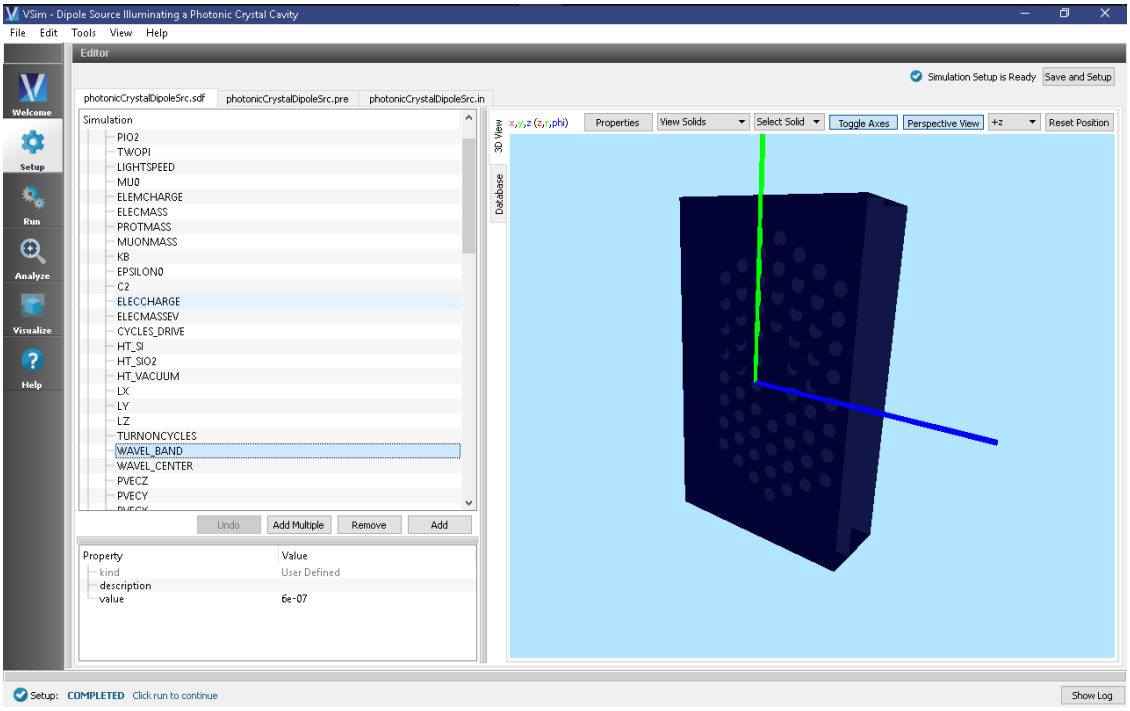


Fig. 3.108: The Setup Window showing the constants

- `PVEC{X,Y,Z}` = The {x,y,z} component of your moment vector for your dipole source.
- `HT_DIPOLE` = The height of the dipole from the lowerX boundary.
- `SIMCYCLES` = Number of wave cycles you want your simulation to run.
- `CYCLESPERDUMP` = Number of cycles between each dump in the simulation.

The tool used to input the wave into the simulation is a port launcher. It specifies the Electric Displacement Field (D) at a boundary in this case the lower X boundary. The functions defining the D on the boundary are defined in the *SpaceTimeFunctions* element of the *Elements Tree*.

SpaceTimeFunctions:

- `dSingleFreqDipole{Y,Z}` = This is the {x,y,z} component of the single frequency dipole source; as seen in Fig. 3.109, you put this as a parameter in the PortLauncher.
- `dWideBandDipole{Y,Z}` = This is the {x,y,z} component of the wideband dipole source; as seen in Fig. 3.109, you put this as a parameter in the PortLauncher.

To choose which signal you want to input into this example simulation:

1. Expand the *FieldBoundaryConditions* tab.
2. Left click on the *portLauncherLowerX* condition.

At this point, you should see what is shown in Fig. 3.109:

3. To change the signal applied to the *portLauncherLowerX*, right click on the Dx(x, y, z, t) or Dy(x, y, z, t) property and select *Assign SpaceTimeFunction*. This will expand another menu that will show you all four defined SpaceTimeFunctions. Select which one you want to input into your simulation. For this documentation, `dWideBandDipole{Y,Z}` is selected by default to demonstrate the functionality of the example.

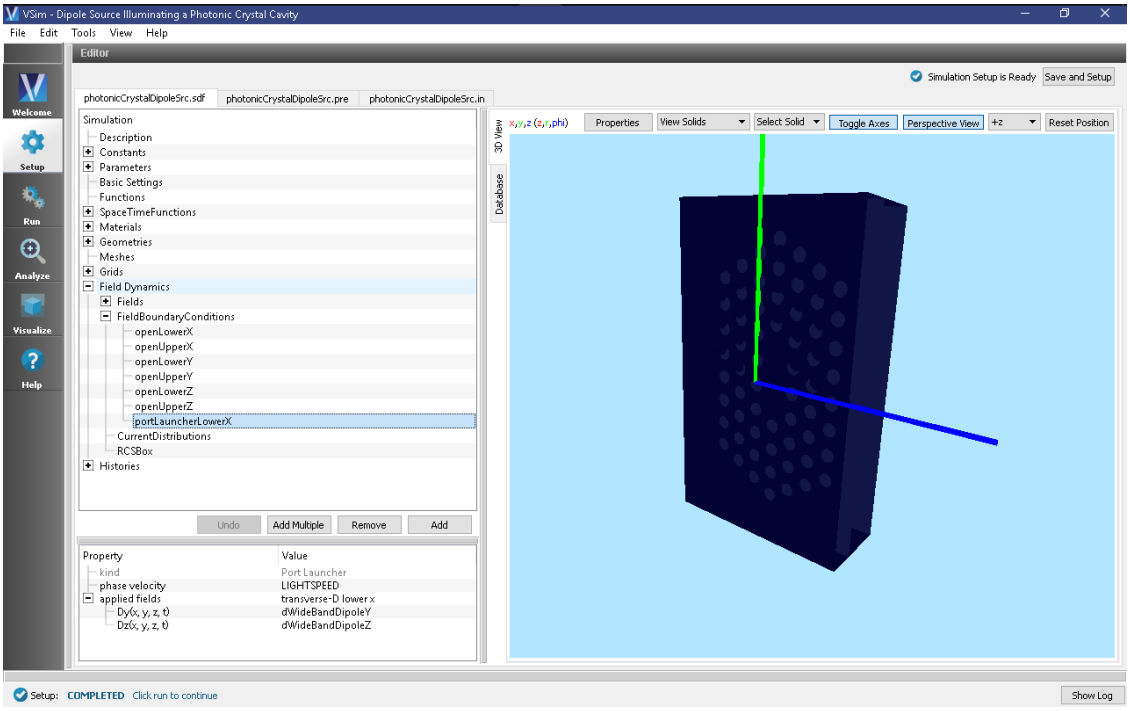


Fig. 3.109: The Setup Window showing where to specifying the applied field signal for the port launcher

Running the Simulation

Once finished with the problem setup, continue as follows:

- Proceed to the *Run Window* by pressing the *Run* button in the left column of buttons.
- One can enable MPI options to utilize multi-core systems.
- The default values of `Number of Time Steps` and `Dump Periodicity` are taken from the parameters `STEPSTOTAL` and `STEPSPERDUMP`, which use the constants `SIMCYCLES` and `CYCLESPERDUMP`. The formulae for these variables can be found back in the *Setup Window*. These variables are for convenience to calculate good default values and it is important to know that the override option default values ultimately come directly from the numbers in the *Basic Settings* section.
  - *Number of Steps* = `STEPSTOTAL` = 36250
  - *Dump Periodicity* = `STEPSPERDUMP` = 3600
- 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 when you see the output, "Engine completed successfully." This is shown in Fig. 3.110.

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.

In the simulation, there are specific grid points which store field histories. These histories are placed in various positions of the simulation.

- Click the *Add a Data View* dropdown below the menu bar and select *History*.

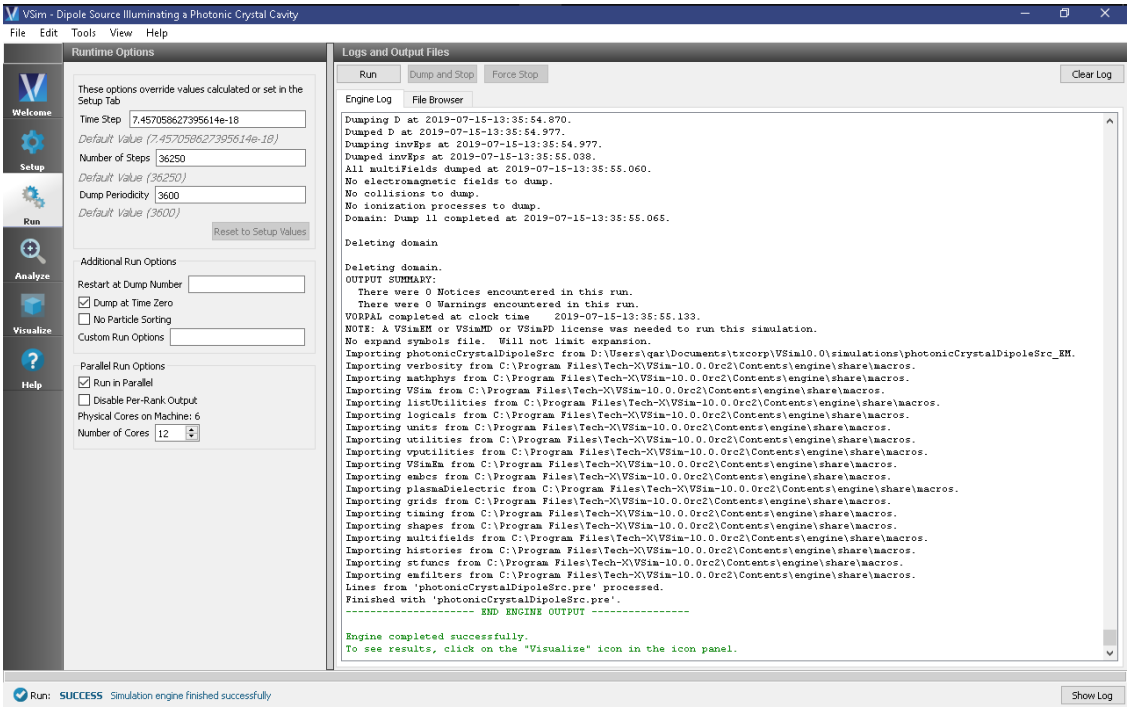


Fig. 3.110: The Run Window for the Dipole Source Illuminating a Photonic Crystal Cavity example

In Fig. 3.111, one can see there are 4 possible graphs to view at one time in the *Visualize Window*. For each graph, one can select the following fields to analyze: (0 = x, 1 = y, 2 = z).

- {E,B}_AtDet_{0,1,2} = In the middle of (y,z) plane and 60 nm above the surface of the crystal.
- {E,B}_AtSrc_{0,1,2} = Is aligned with the inCav history in the (y,z) plane and is 60 nm below the Si layer, into the SiO2 layer.
- {E,B}_InCav_{0,1,2} = Is slightly offset from the middle of one of the cavities in the Silicon layer (the layer with the lattice).

In each individual graph, one can choose the *Fourier Amplitudes (dB)* option to view the frequency domain of your field. This can enable the analysis of the frequency response of the photonic crystal cavity.

Fig. 3.111 depicts four graphs of histories. The first two graphs are amplitude vs time, and the second two are a Fourier Amplitudes of the first two on a log scale.

The first and third graphs depict the history `AtSrc_2`, while the second and fourth graphs show the *AtDet_2* history.

Further Experiments

By using the wideband source and examining the field strength detected below the crystal lattice, one may study the frequency response of this photonic crystal as one changes the device geometry, the dielectric constants, and the location and polarizations of the radiation source and detector.

3.3.8 Metal Insulator Metal (MIM) Waveguide using Drude and Lorentz Materials (drudeLorentzMIM.sdf)

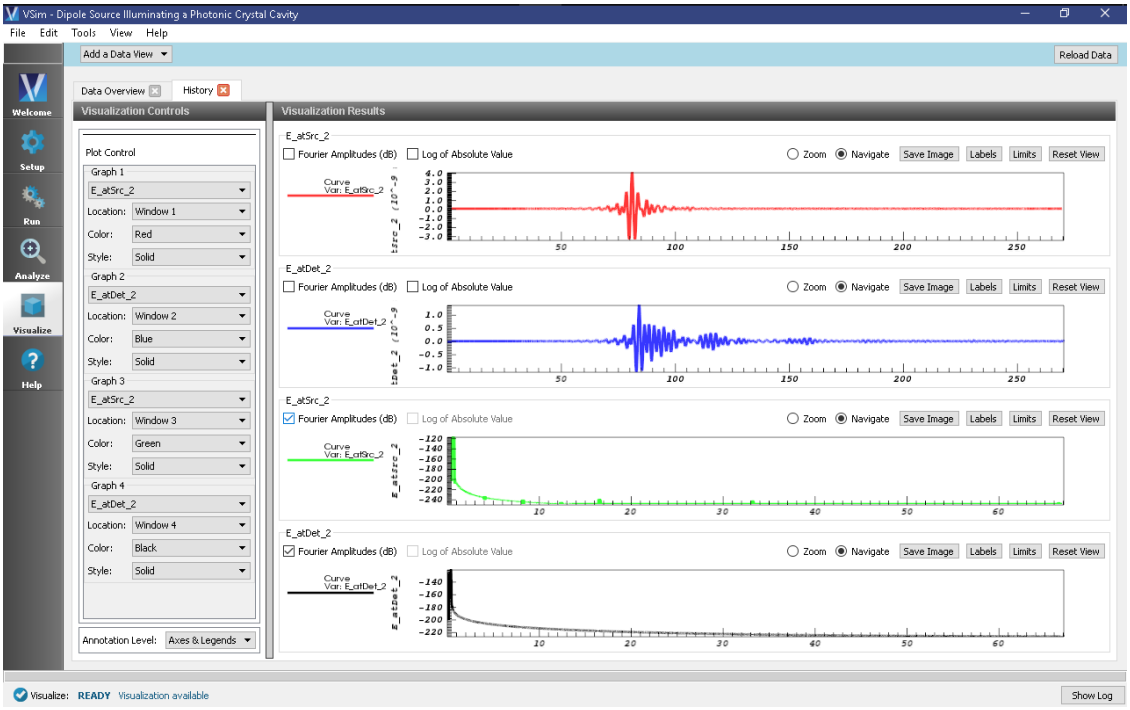
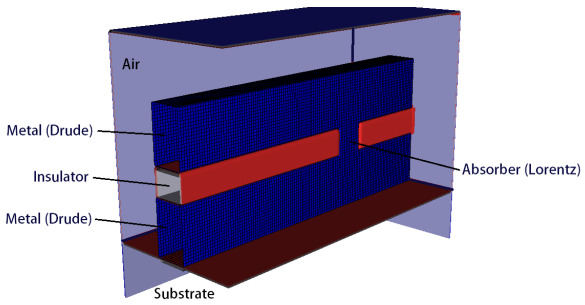


Fig. 3.111: The Visualize Window for the Dipole Source Illuminating a Photonic Crystal Cavity example



Problem Description

A metal-insulator-metal (MIM) waveguide can propagate optical frequency electromagnetic radiation due to the effective negative index material property of the metal at those frequencies. This negative index material is represented with a time-domain Drude model dielectric, which can support a wide range of frequencies and wide bandwidth.

In addition to the MIM waveguide, a section of the insulator is removed, and replaced with a resonant absorber material, using a time-domain version of the traditional Lorentz material.

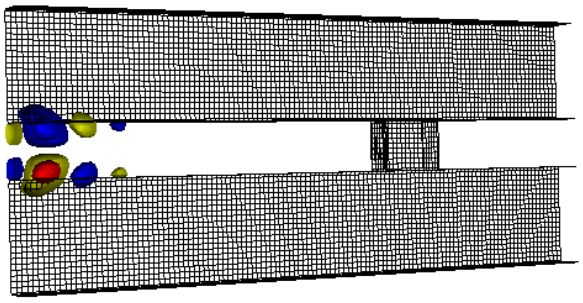


Fig. 3.112: Longitudinal electric field in the MIM waveguide.

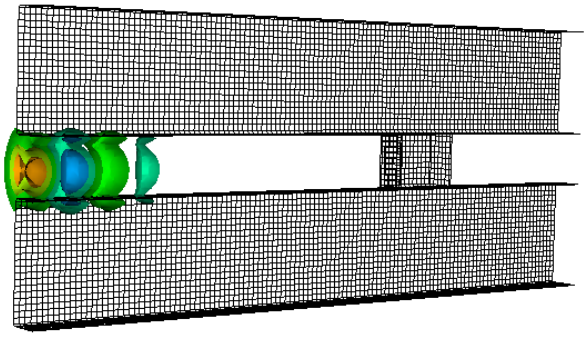


Fig. 3.113: Transverse electric field in the MIM waveguide.

A spatial gaussian waveform is incident on the edge of the MIM waveguide, coupling to it, and propagating down the length of the waveguide until it encounters the Lorentz material inclusion, where the wave is absorbed. For the incident wave to couple effectively to the MIM waveguide, the spatial size of the gaussian waveform must be a good match to the size of the waveguide, or a large portion of the incident wave will scatter off the structure, rather than coupling to it.

Also, the width, strength, and natural oscillation frequency of the Lorentz material inclusion determines whether the wave is reflected, absorbed, or transmitted when it encounters the inclusion. In this example there is only one Lorentz curve, but multiple Lorentz curves can be added to the simulation.

The length of the MIM waveguide, and the direction of wave propagation is in the x-direction. The width of the waveguide is in the z-direction, and the height of the waveguide is in the y-direction. The waveguide sits atop an insulator substrate, and is surrounded by air. The boundaries of the simulation are ports, allowing for incoming and outgoing waves.

This simulation can be performed with a VSimEM license.

Opening the Simulation

The MIM waveguide example is accessed from within VSimComposer by the following actions:

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim for Electromagnetics* option.
- Expand the *Scattering* option.
- Select "MIM 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 now shown with all the implemented physics and geometries, if applicable. See Fig. 3.114.

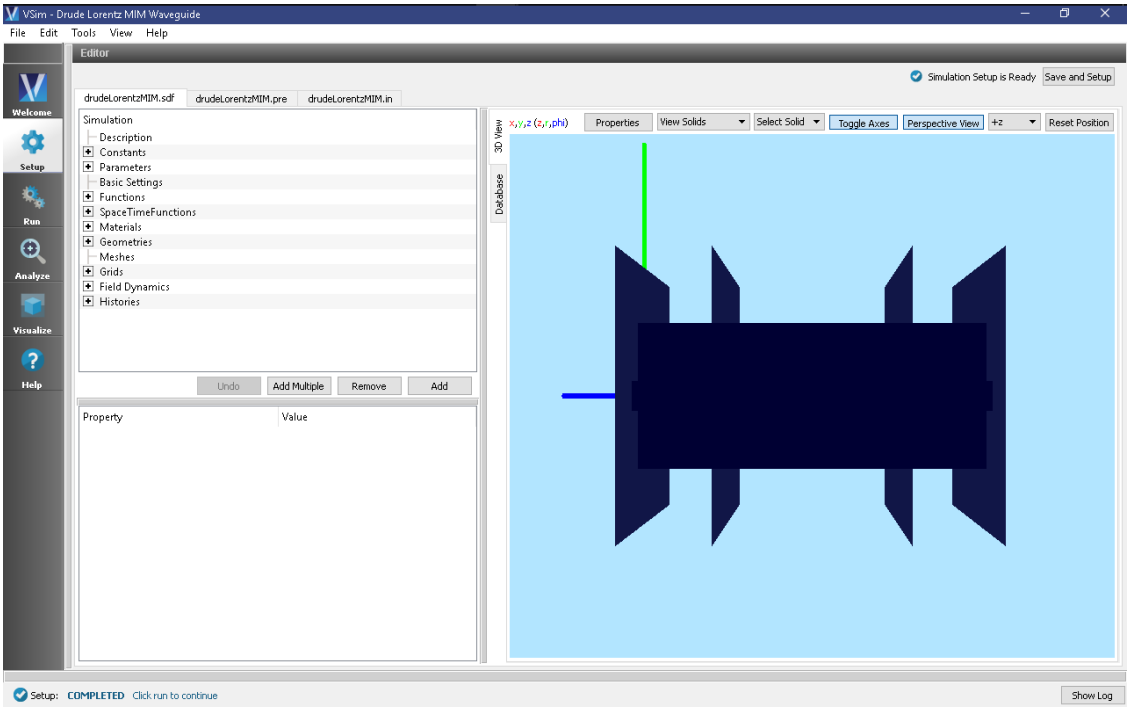


Fig. 3.114: Setup Window for the drudeLorentzMIM example.

Simulation Properties

The geometry of the waveguide is fully parameterized, allowing for easy adjustment to the waveguide.

A port launcher boundary is used at the lower X boundary to launch a Y polarized wave.

The Drude-Lorentz model dielectric allows for full specification of the Drude model collision and conductivity function, as well as a background conductivity. In this example a single Lorentz model is used with the oscillator, frequency and line width. More Lorentz's can be added by adding to the vector of these three properties.

The input file also contains a parameter to adjust the spatial resolution of the mesh.

Default parameters are selected to correspond to violet light, a Drude material corresponding to silver, SiO2 insulator (and substrate), and a Lorentz material corresponding to AlAs. The default variable values can be compared to the well known material properties of these materials to establish the exact correspondence to the well-known mathematical descriptions of the Drude and Lorentz models.

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.
- To run the file, click on the *Run* button in the upper left corner of the window. 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 is shown in Fig. 3.115.

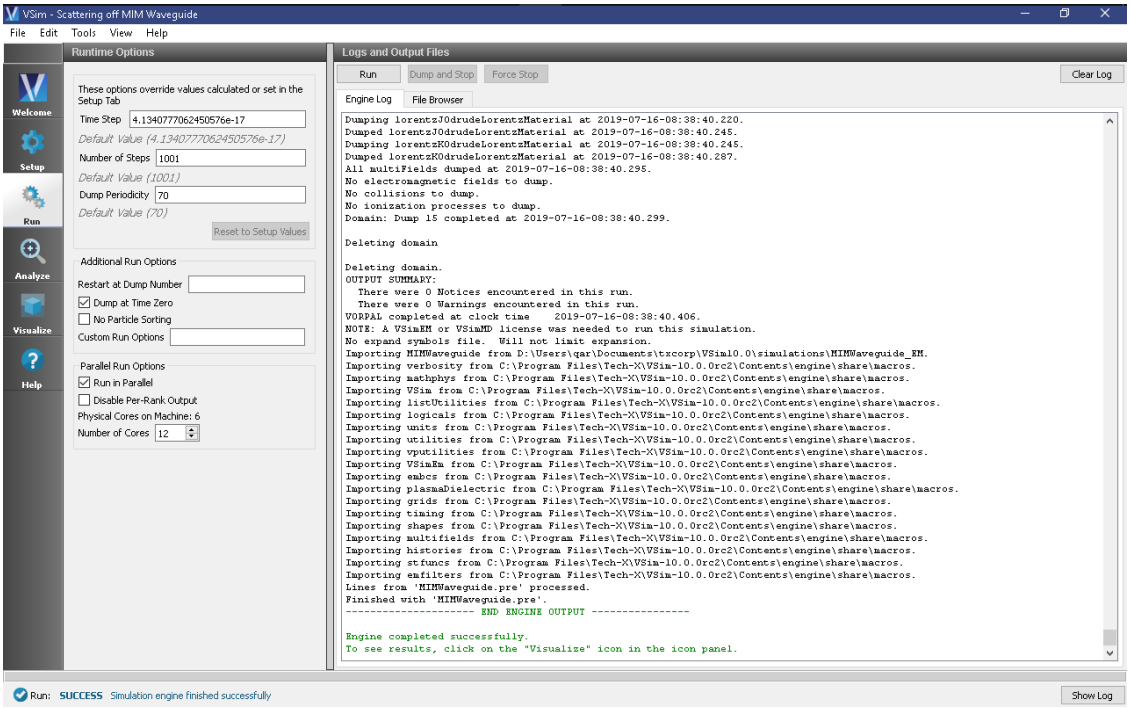


Fig. 3.115: The Run Window at the end of execution.

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.

The results are best viewed by looking at the y component of the electric field. To view the fields, select *Scalar Data* variable and check the *E_y* box. Check the *Clip All Plots* checkbox. In the *Colors* tab above the simulation, set the minimum value to -0.075 and the maximum value to 0.075. The field is shown in Fig. 3.116.

We can see that fields are well coupled between the two metal layers of the waveguide, with only some small leakage, and transient behavior at the entrance. The fields then diminish abruptly at the inclusion, where the wave is mostly absorbed.

3.4 Photonics (text-based setup)

3.4.1 Multimode Fiber with Mode Launcher (multiModeFiberModeLaunchT.pre)

Keywords:

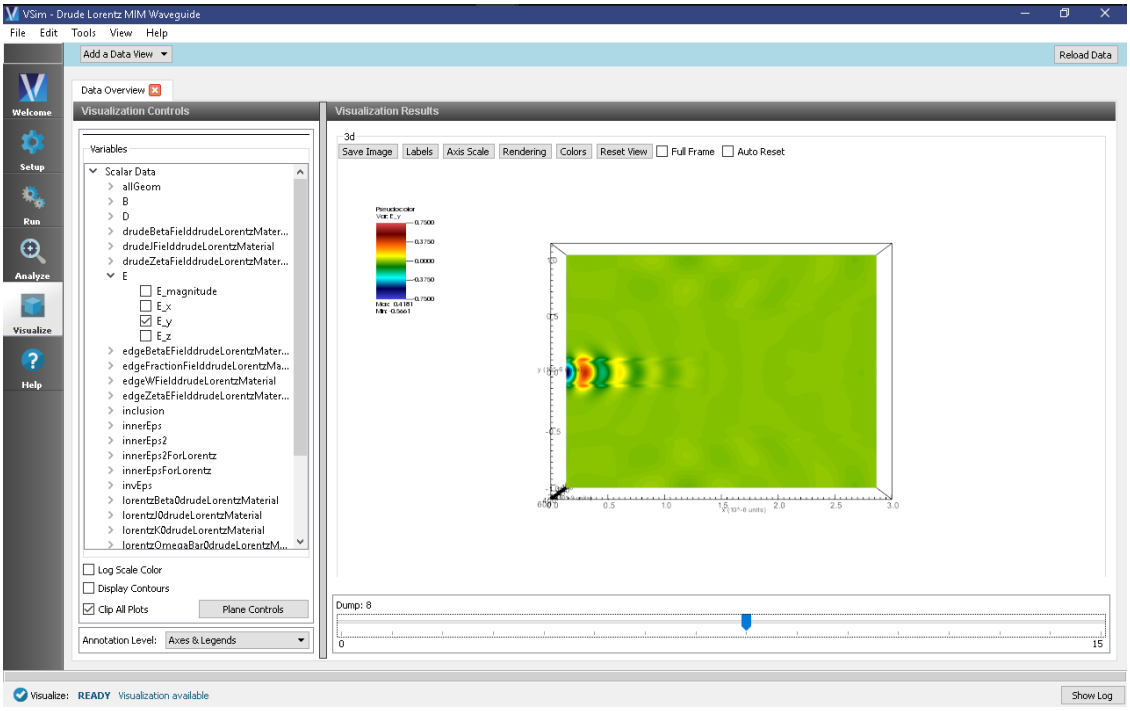


Fig. 3.116: Visualization of the E_y field component.

Mode Loading, Photonic Waveguide, Unidirectional Mode Launcher, MAL, Guided Mode, Semicondu

Problem Description

This example consists of a single, straight cylindrical fiber waveguide that is parallel to the x-axis and centered at the origin. The waveguide is surrounded by a background material with a greater permittivity. Matched Absorbing Layers (MALs) are used to dampen the E and B fields near the boundary of the simulation. This is a way to dampen reflected fields from the simulation boundaries.

The fundamental guided mode is launched in the waveguide. The fundamental mode was extracted in the "Multimode Fiber Mode Calculation" example using the computeDielectricModes.py analyzer. A sample mode comes with this example saved in the file save4Launch_EigenD_0.vsh5, but this can be replaced another mode from the mode calculation example if the user desires. We take the field profile from the save4Launch_EigenD_0.vsh5 file and then apply time dependence via a SpaceTimeFunction, timeCompSingleFrequency. By launching the true eigenmode into the waveguide we should minimize losses and see a constant field profile.

This simulation can be performed with a VSimEM license.

Opening the Simulation

This example can be accessed from within VSimComposer through the following steps:

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window, expand the *VSim for Electromagnetics* option.
- Expand the *Photonics (text-based setup)* option.
- Select *Multimode Fiber with Mode Launcher (text-based setup)* 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.

Some relevant parameters should now be visible as seen in Fig. 3.117.

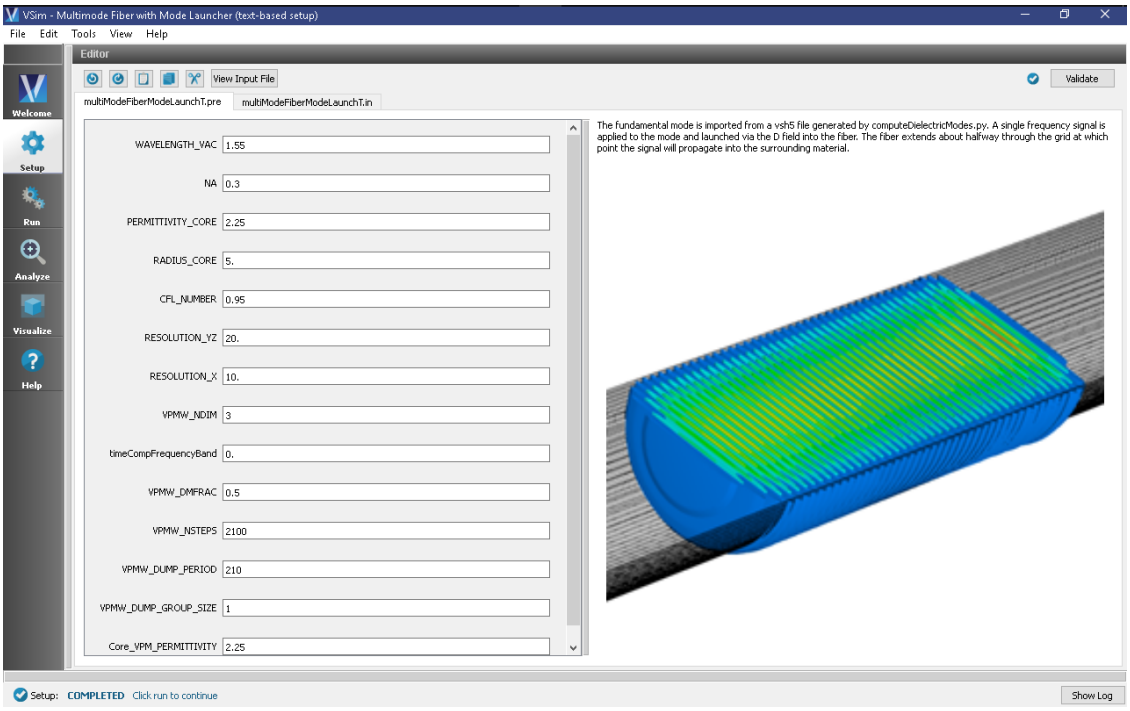


Fig. 3.117: The Setup window for the showing some relevant parameters.

Simulation Properties

This example contains a number of constants defined in the Multimode Fiber Mode Calculation example. These constants should not be modified if you wish to launch a true eigenmode. Some relevant constants that could be modified are listed below.

RESOLUTION_X: The number of cells per estimated wavelength in the propagation (x) direction.

CFL_NUMBER: The time step, DT, will be this value times the limit for numerical stability.

To expose more variables and see the geometries, boundary conditions, and fields, select *View Input File*. From here you can see the modifications made to import the eigenmode from the .vsh5 file. The modifications are clearly set apart with rows of equal signs as seen in Fig. 3.118.

The variables NBGNX_SOURCE, NENDX_SOURCE, NBGNYZ_SOURCE, and NENDYZ_SOURCE define where the source is located in integer grid cells. This should likely correspond to the location you specified when running the computeDielectricModes.py analyzer. To ensure the source is aligned with the fiber, we recommend using the same grid for mode extraction and mode launching and then defining the aforementioned variables as is seen in the input file.

The D field in the specified source location depends spatially on the imported mode and temporally on the expression timeCompSingleFrequency, which drives the mode at its respective frequency for a length of time, TIME_EXCITE. In photonics simulations, Matched Absorbing Layers (MALs) are the most stable boundary conditions for limiting reflections.

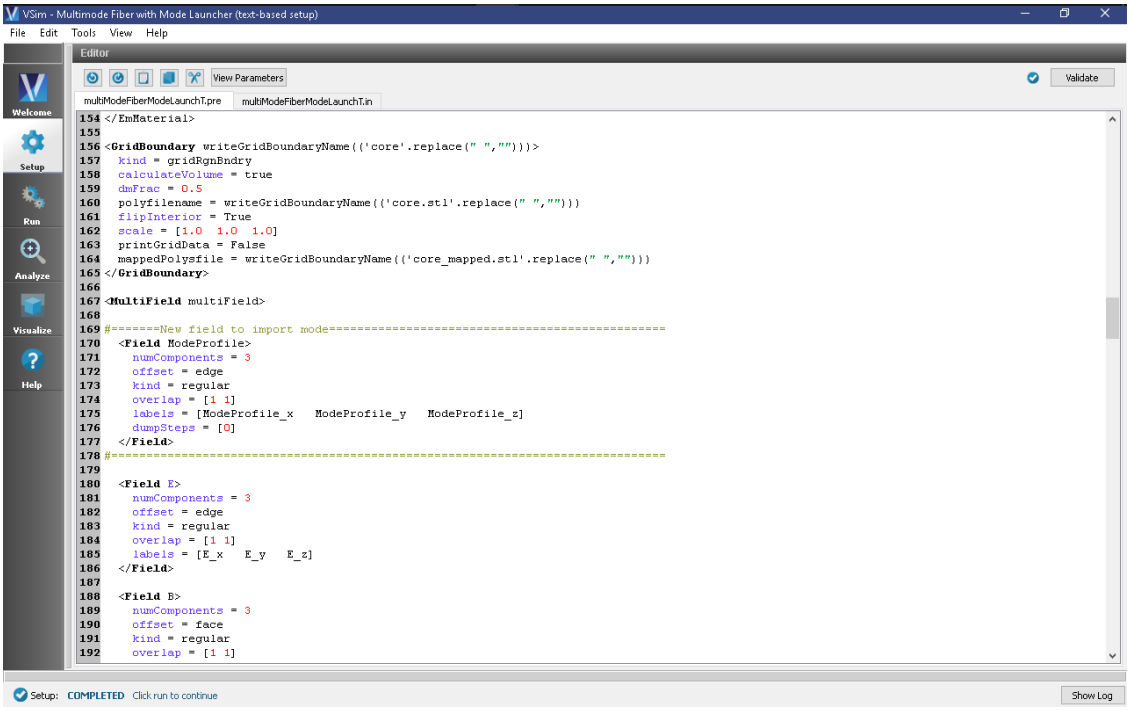


Fig. 3.118: The input multiModeFiberModeLaunchT.pre file showing the newly defined field.

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. You will be asked to Save. Click *Save*.
- To run the file, 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." The result is shown in Fig. 3.119.

Visualizing the Results

Then proceed to the Visualize window by pressing the *Visualize* button in the left column.

A useful visualization of the dielectric waveguide would be to view the magnitude of the D field to qualitatively see the mode propagate down the waveguide.

- Near the top left corner of the window, make sure *Data View* is set to *Data Overview*.
- Expand *Scalar Data*, expand *D*, and select *D_magnitude*
- Expand *Geometries* and select *poly* to show how far the waveguide extends.
- In the controls below the variables frame, select *Clip All Plots*.
- Slide the Dump Slider (beneath the Visualization Results) to Dump 3

Your screen should resemble Fig. 3.120. Indeed, the mode launch is quite clean! To see what happens at the end of the fiber slide the dump slider further.

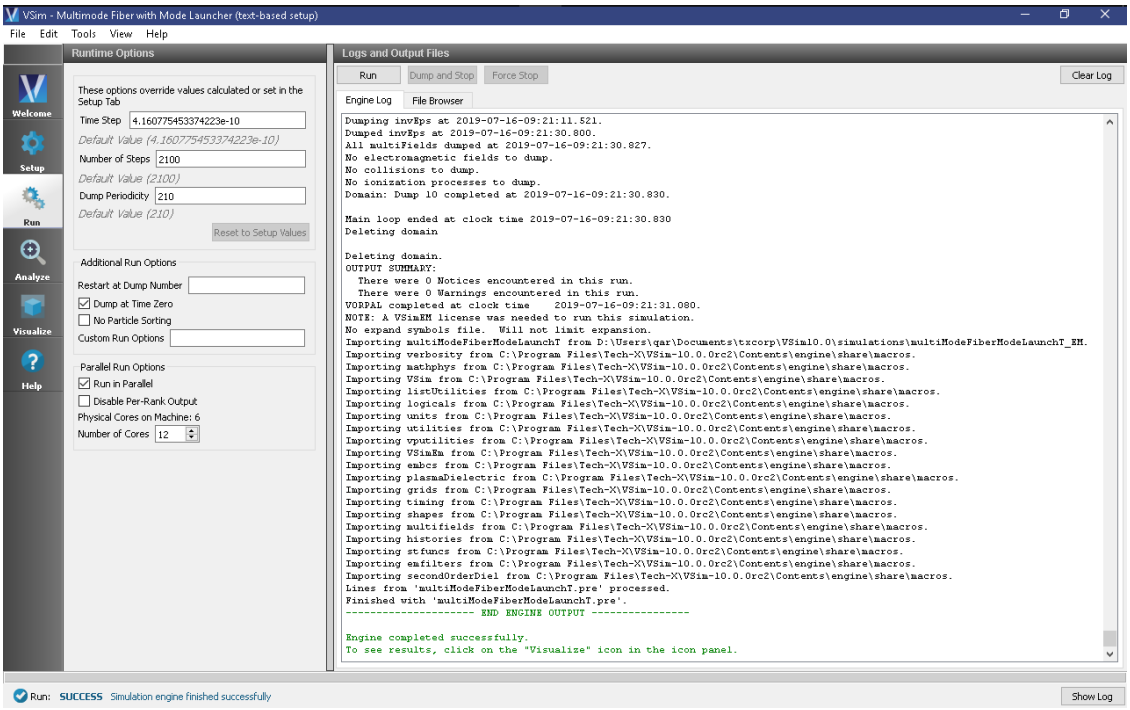


Fig. 3.119: The output after a successful run.

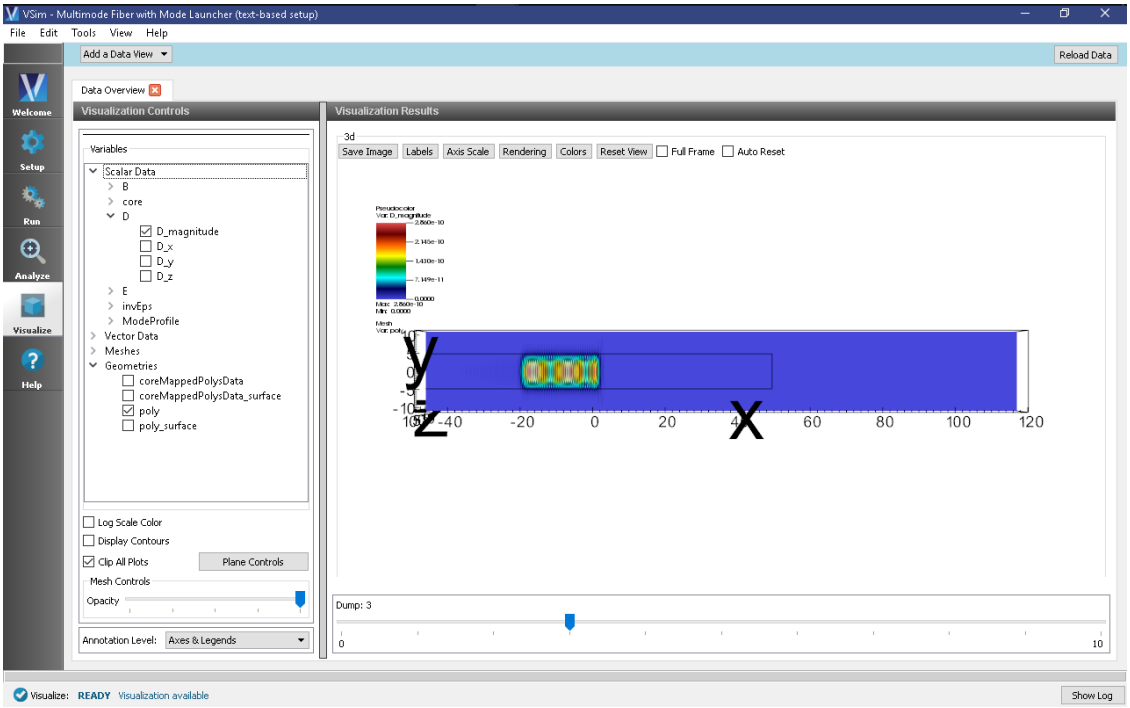


Fig. 3.120: Visualization of the D field.

Further Experiments

One can experiment by changing constants or introducing a different signal to drive the waveguide and note the effects on loss or propagation.

One could also choose a different mode generated by the Multimode Fiber Mode Calculation example and launch that.

3.5 Scattering

3.5.1 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.

This simulation can be performed with a VSimEM, VSimMD or VSimPD license.

Opening the Simulation

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

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim 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. 3.121. 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`.

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 the how the electromagnetic wave can scatter off of different materials.

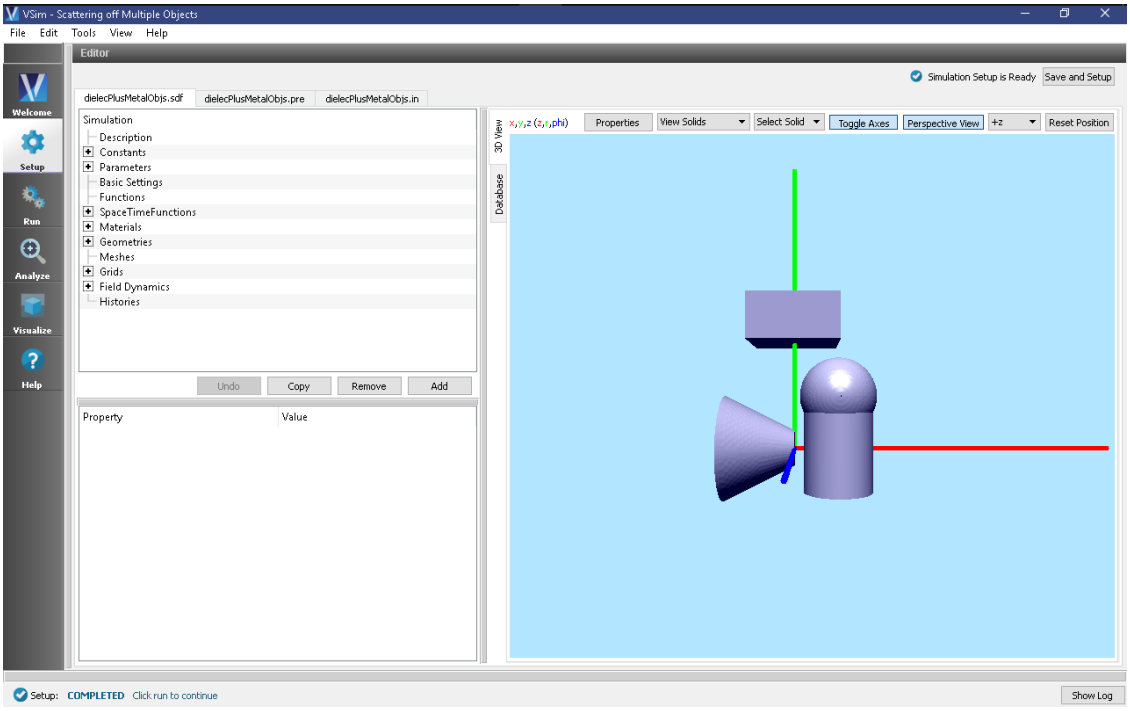


Fig. 3.121: Setup Window for the Scattering off Multiple Objects example.

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.
- Here you can set run parameters, including how many cores to run with (under the MPI tab).
- When you are finished setting run parameters, 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 the right pane. The run has completed when you see the output, "Engine completed successfully." This is shown in Fig. 3.122.

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. 3.123, do the following:

- Expand *Scalar Data*
- Expand *E*
- Select *E_z*
- Expand *Geometries*
- Select *poly (AluminaObject) poly (PECObject) poly (sapphireObject)*
- Select *Clip All Plots*
- Above the image, select *Colors*

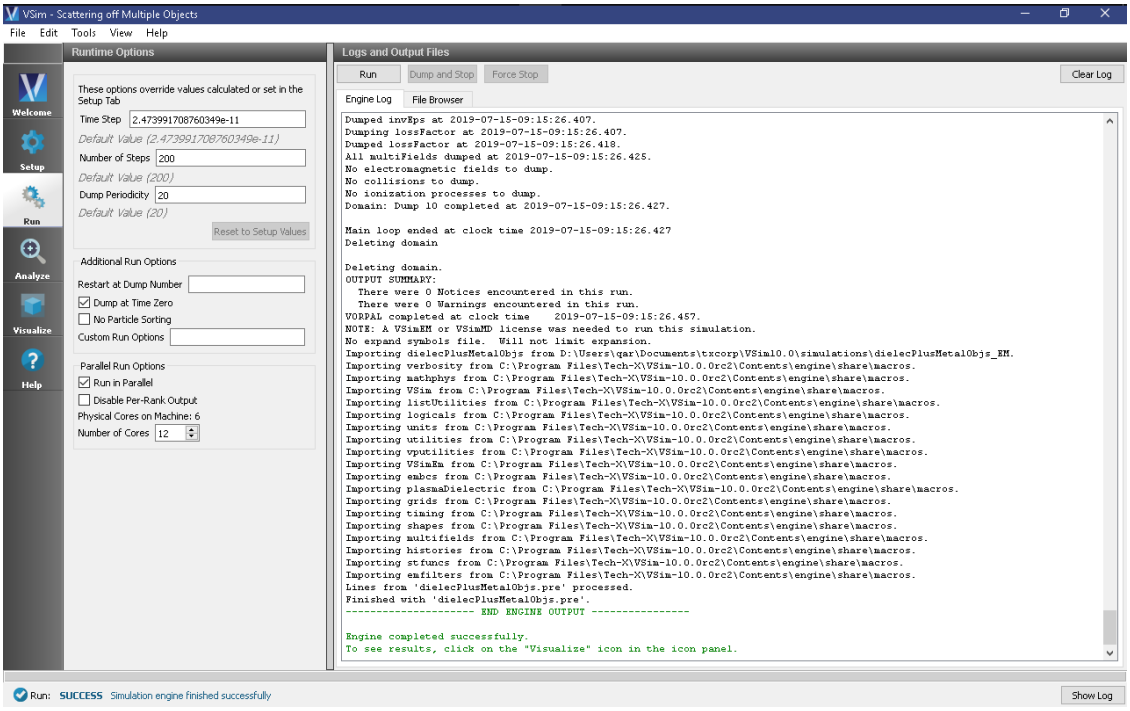


Fig. 3.122: The Run Window at the end of execution.

- In the *Color Options* window fix the minimum and maximum values to -2 and 2, respectively.
- Click *Apply*, followed by *OK*.

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.

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.

3.6 Scattering (text-based setup)

3.6.1 Ground Penetrating Radar (groundPenetratingRadarT.pre)

Keywords:

`GPR, ground penetrating radar, lossy dielectric`

Problem description

This simulation launches a plane wave, polarized in the Z-direction into a lossy dielectric. Embedded within the lossy dielectric is a mine, modelled as a pure electric conductor. The return wave at the surface can be monitored with histories. The simulation is 2D, but with minor effort can be expanded to 3 dimensions.

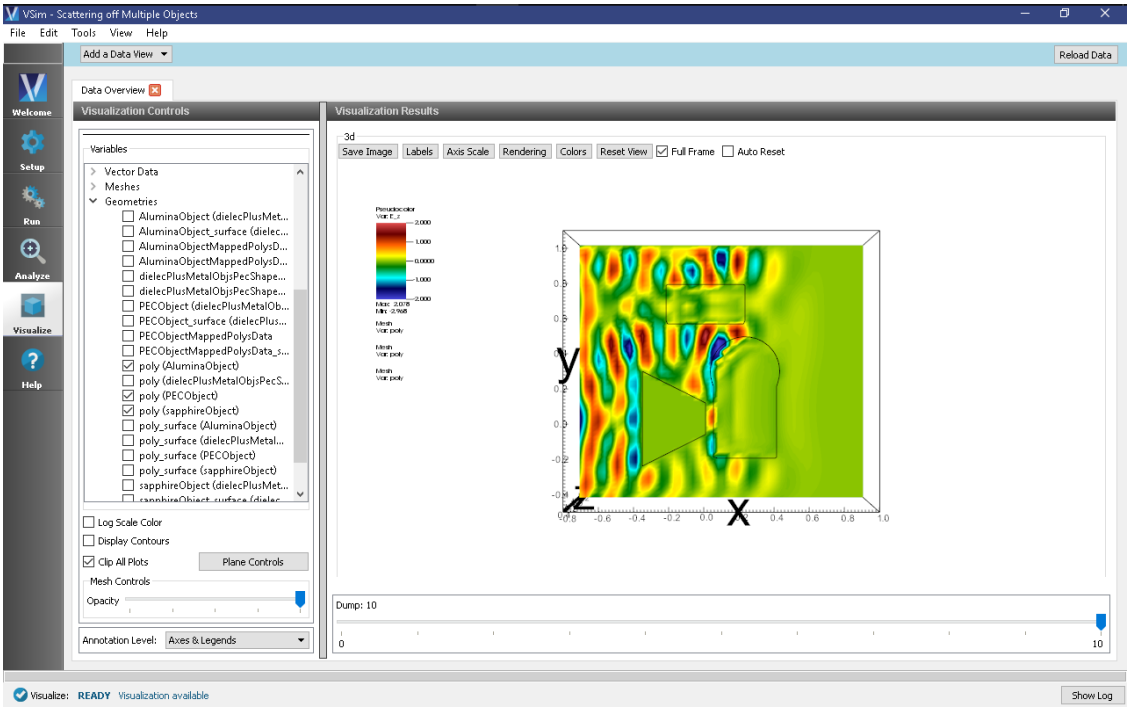


Fig. 3.123: Visualization of wave as it hits the objects.

This simulation can be performed with a VSimEM license.

Opening the Simulation

The Ground Penetrating Radar example is accessed from within VSimComposer by the following actions:

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim for Electromagnetics* option.
- Expand the *Scattering (text-based setup)* option.
- Select "Ground Penetrating Radar (text-based setup)" 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 key parameters of this problem should now be alterable via the text boxes in the left pane of the *Setup Window*, as shown in Fig. 3.124.

Input File Features

This file allows for the modification of plane wave operating frequency, simulation domain size, resolution, dielectric permittivity, size and conductivity, mine size and location as well as history location.

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.

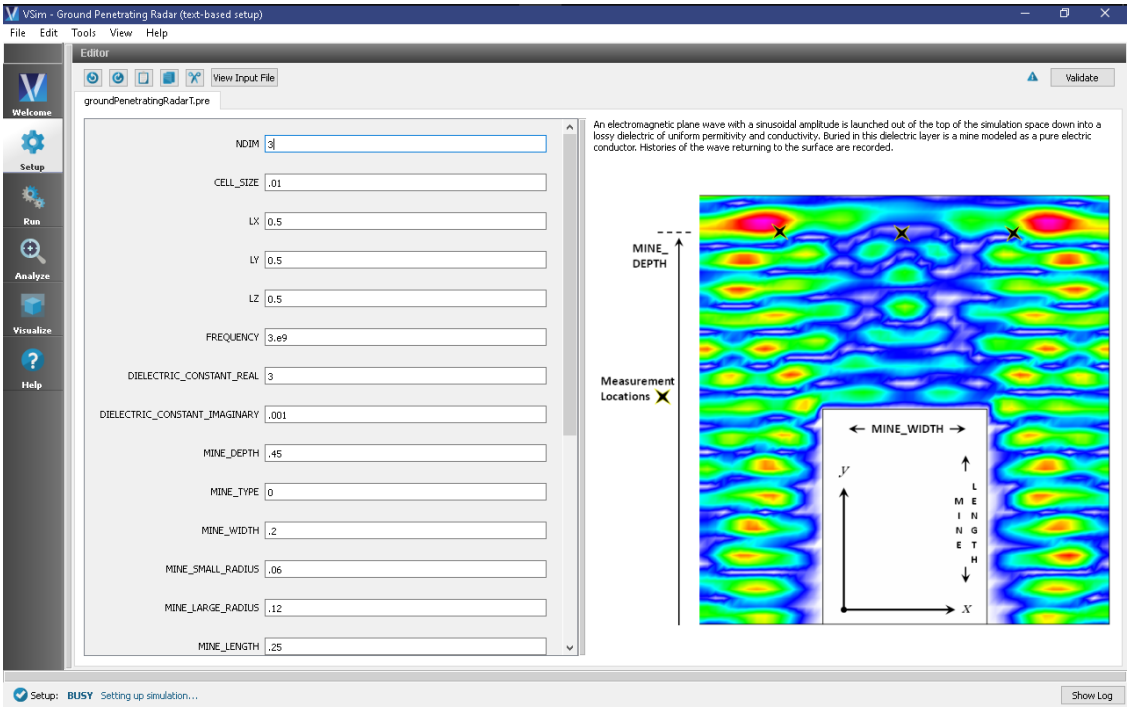


Fig. 3.124: Setup Window for the Ground Penetrating Radar example.

- To run the file, click on the *Run* button in the upper left corner of the window. 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 is shown in the window below.

Visualizing the Results

After the simulation has completed running, click on the Visualize Window.

The electric field can be viewed by:

- Expand *Scalar Data*
- Expand *E*
- Select *E_z*
- Expand *Geometries*
- Select *poly*
- Move the dump slider forward in time

The wave reflection measured at the three surface locations can be viewed under the *Histories* Data View

Further Experiments

The parameters of the dielectric can be easily modified. It would also be possible to modify the sources to be horn antennas instead of point sources; this would more accurately model a real world ground penetrating radar situation.

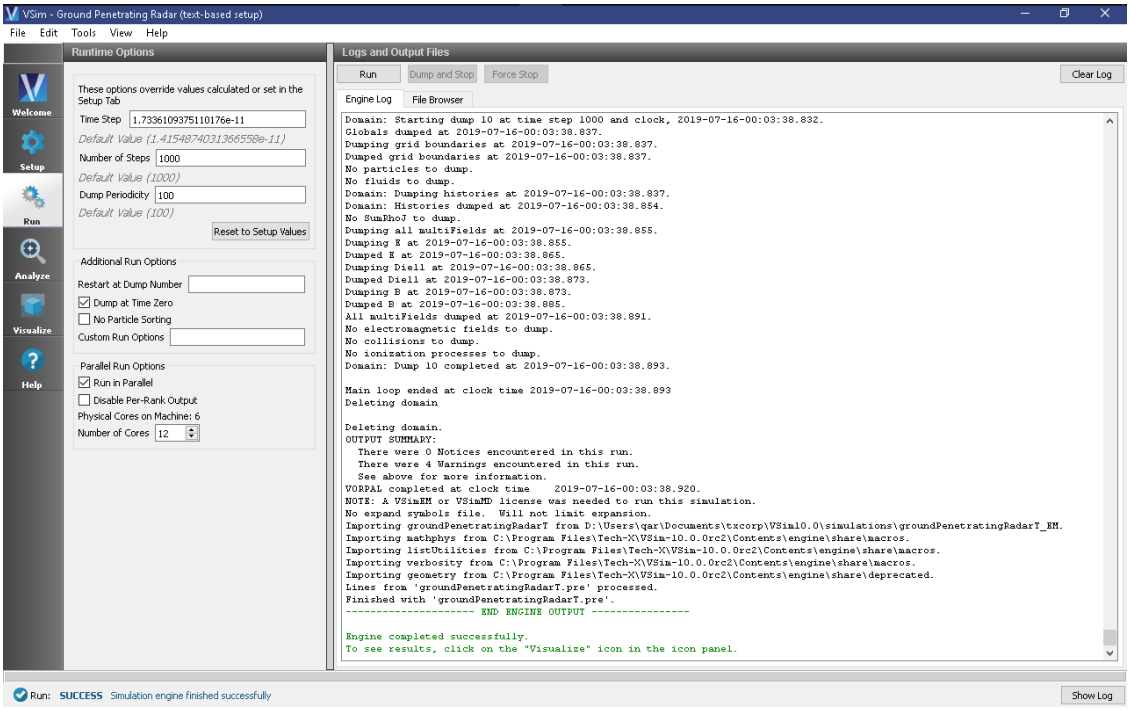


Fig. 3.125: The Run Window at the end of execution.

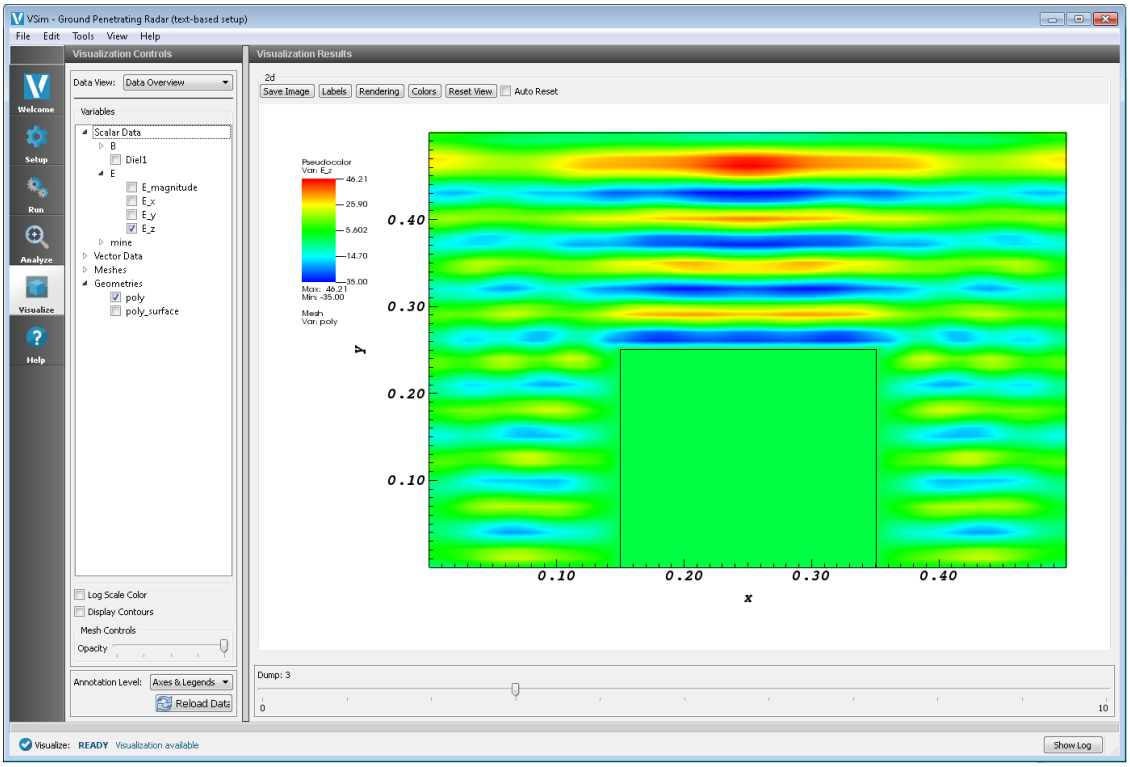


Fig. 3.126: The electric field in the simulation space. Seen here: the cylindrical cone mine-shape.

3.6.2 Radar Cross Section of a Cylinder (radarCrossSectionT.pre)

Keywords:

RCS, far field, radar cross section

Problem description

This simulation launches a plane wave, polarized in the Y-direction at a conducting cylinder in free space. After the plane wave has been launched the Radar Cross section is computed. This problem is a template to solve any Bistatic radar cross section problem.

This simulation can be performed with a VSimEM license.

Opening the Simulation

The radarCrossSectionT example is accessed from within VSimComposer by the following actions:

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim for Electromagnetics* option.
- Expand the *Advanced Examples (Text-based setup)* option.
- Select "Radar Cross Section" 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 key parameters of this problem should now be alterable via the text boxes in the left pane of the *Setup Window*, as shown in the figure Fig. 3.127.

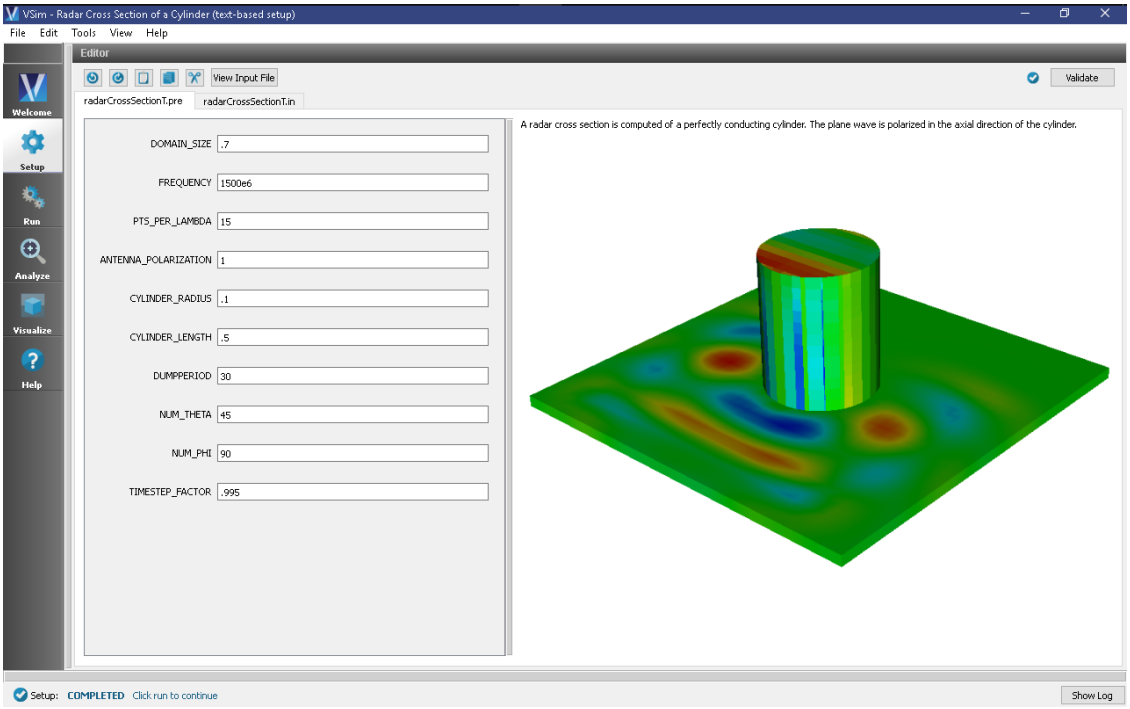


Fig. 3.127: Setup Window for the Radar Cross Section example.

Input File Features

This file allows the modification of plane wave operating frequency, orientation, simulation domain size and far field resolution. This file has had its accuracy reduced marginally in order to reduce run time. It is generally recommended that between 10 and 20 points per wavelength are used for full accuracy.

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.
- 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 when you see the output, "Engine completed successfully." This is shown in Fig. 3.128

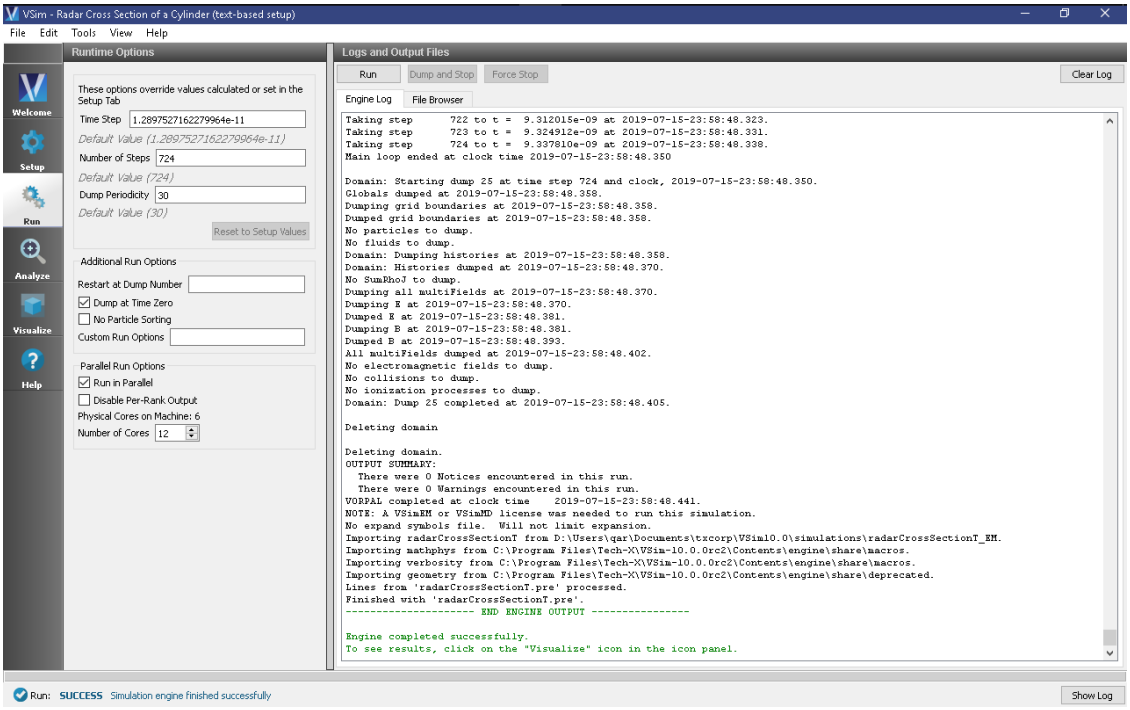


Fig. 3.128: The Run Window at the end of execution.

Analyzing the Results

To calculate the radar cross section at far field, complete the following steps:

- Proceed to the Analysis window by pressing the Analyze button in the left column of buttons.
- In the list of Available Analyzers, select computeFarFieldFromKirhhoffBox.py (Fig. 3.129) 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 - radarCrossSectionT (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)
- timeStepStride - 20 (number of timesteps between far field calculations; determines how many far fields are output; 20 steps should yield 4 far fields in this case)
- getFourierComponent - 0 (whether to fourier analyze for a particular frequency)
- frequency - the frequency to use in the fourier analysis (not needed here).
- numTheta - 60 (number of theta points in the far field, 18 for a quick calculation, 45 for finer resolution)
- numPhi - 120 (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
- varyingRadiusMesh - 0 (Set to 1 in order to make far field mesh adapt to magnitude of far field solution: the classic lobe view)
- simpsonIntegration - 0 (Set to 1 for more accurate integration)

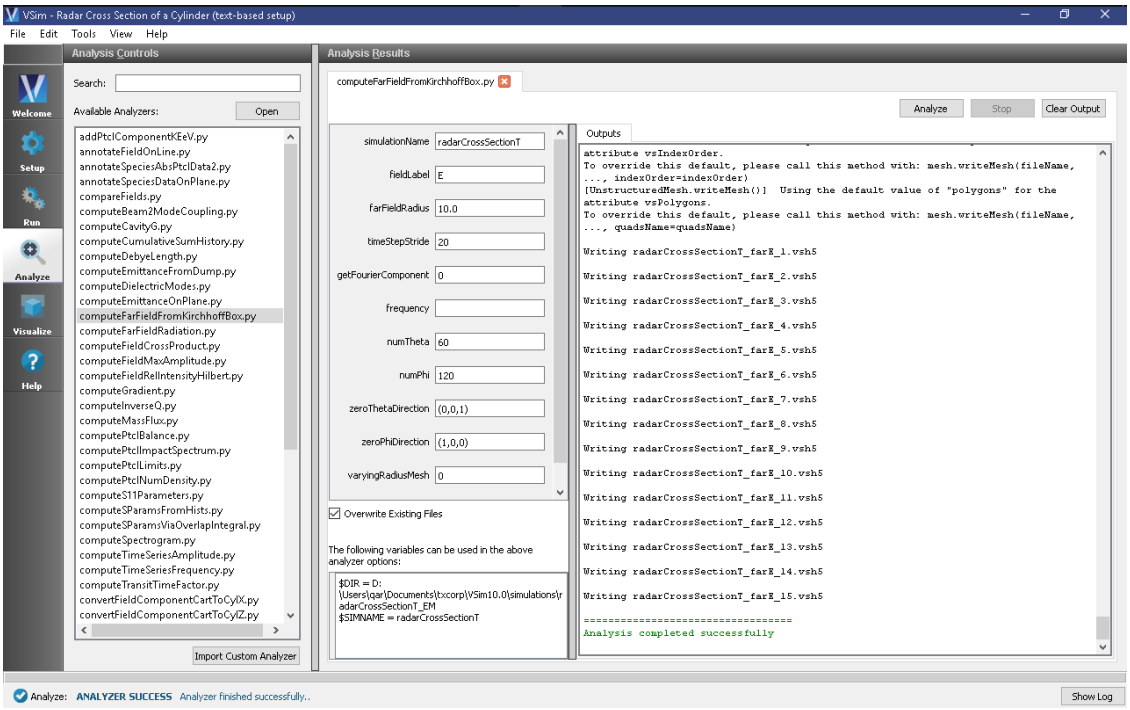


Fig. 3.129: The Analysis Window after running

Visualizing the Results

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

The Radar cross section as shown in Fig. 3.130, do the following:

- Expand *Scalar Data*
- Select *farE*
- Expand *Geometries*

- Select *farSphere*
- Select *poly*
- Select *Clip All Plots*

You should see the far field, time-dependent signal on the sphere with the cylinder in the center.

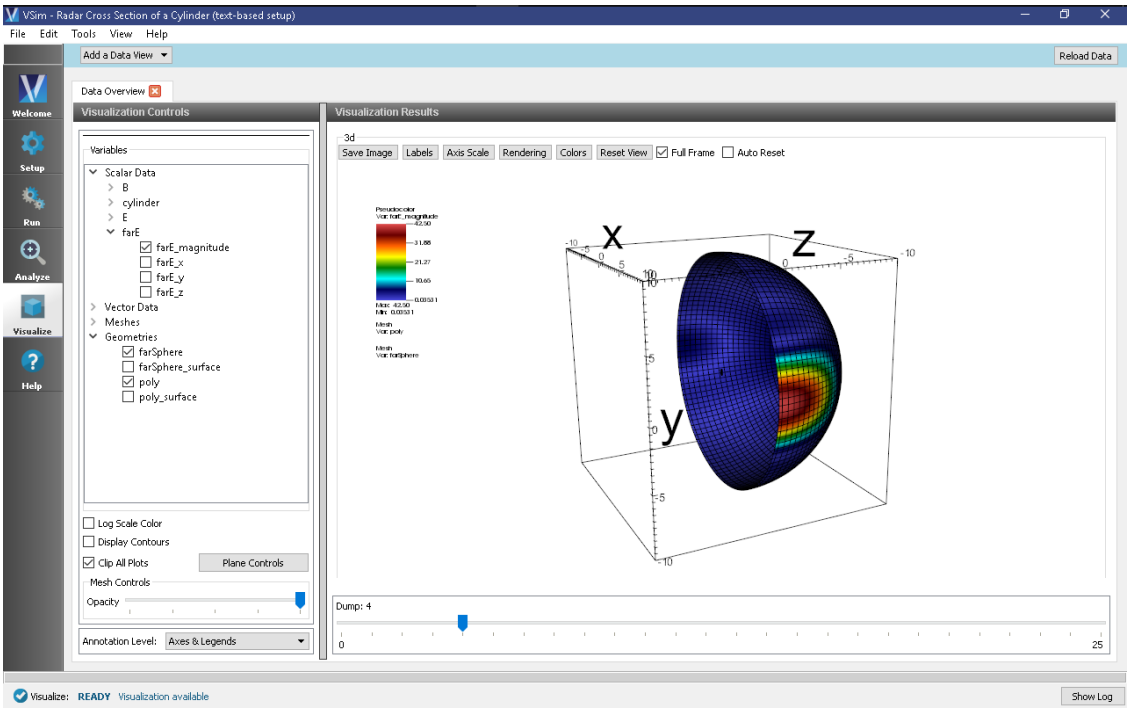


Fig. 3.130: The radar cross section

Further Experiments

The physical dimensions of the cylinder can be modified from the parameters window.

3.7 Other EM

3.7.1 Spherical Lens (sphericalLens.sdf)

Keywords:

`refraction, focusing, dielectrics`

Problem Description

The Spherical Lens is a full wave solution to a simple, thin lens with spherical surfaces. Focusing occurs because light rays farther from the center hit the surface at a more oblique angle, resulting in more bending, according to Snell's law. The focusing length of a spherical lens is given by $f = R/(2 - 2/\epsilon_r^{1/2})$, where ϵ_r is the relative permittivity of the material making up the lens.

This simulation can be performed with a VSimEM license.

Opening the Simulation

The Spherical Lens example is accessed from within VSimComposer by the following actions:

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim for Electromagnetics* option.
- Expand the *Other EM* option.
- Select "Spherical Lens" 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. 3.131. 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`.

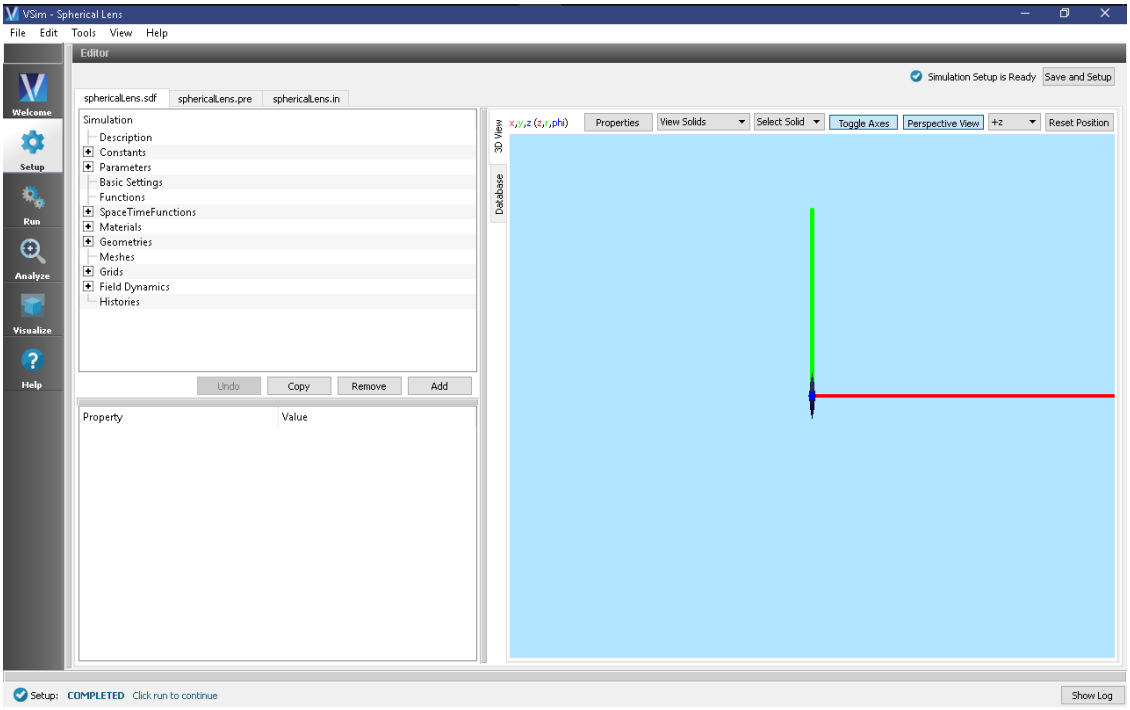


Fig. 3.131: Setup window for the Spherical Lens example.

Simulation Properties

The spherical lens is constructed in CSG using the intersection of two spheres. You can pull the spheres apart to get a taller lens, and you can change the radius of the spheres to have a lens with more curvature. The grid is set so that it will capture the focus at the right for the initial setup.

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.

- To run the file, click on the *Run* button in the upper left corner of the right panel. 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 is shown in Fig. 3.132.

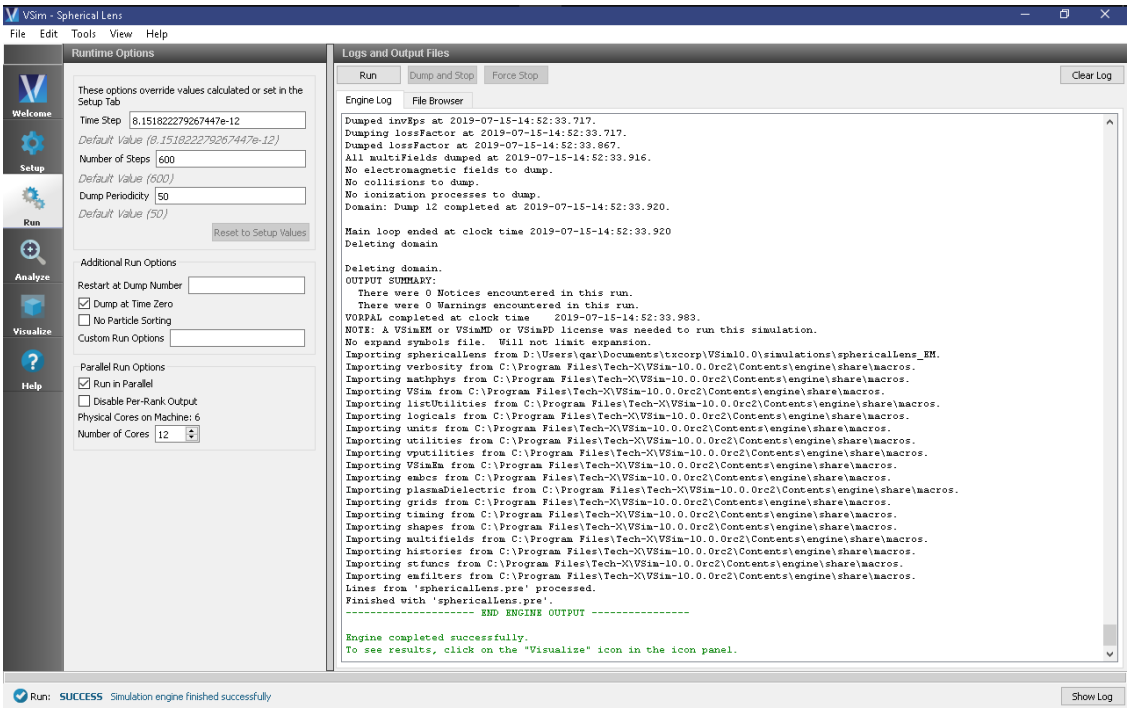


Fig. 3.132: The Run window at the end of execution.

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 see the field focus after the lens as shown in Fig. 3.133, do the following:

- Expand *Scalar Data*, expand *E*
- Select *E_z*
- Click the *Colors* button and set the *Fix Minimum* to -1 and the *Fix Maximum* to +1.
- Expand *Geometries*
- Select *poly*
- Select *Clip all plots*
- Move the dump slide to the right to see the wave come in, focus after the lens, and then diverge again after approximately x=0.4. One can see interference of the incoming wave with the reflection off the face of the lens. One can also see interference patterns within the lens.

Further Experiments

Use a material of larger dielectric constant to see more focusing.

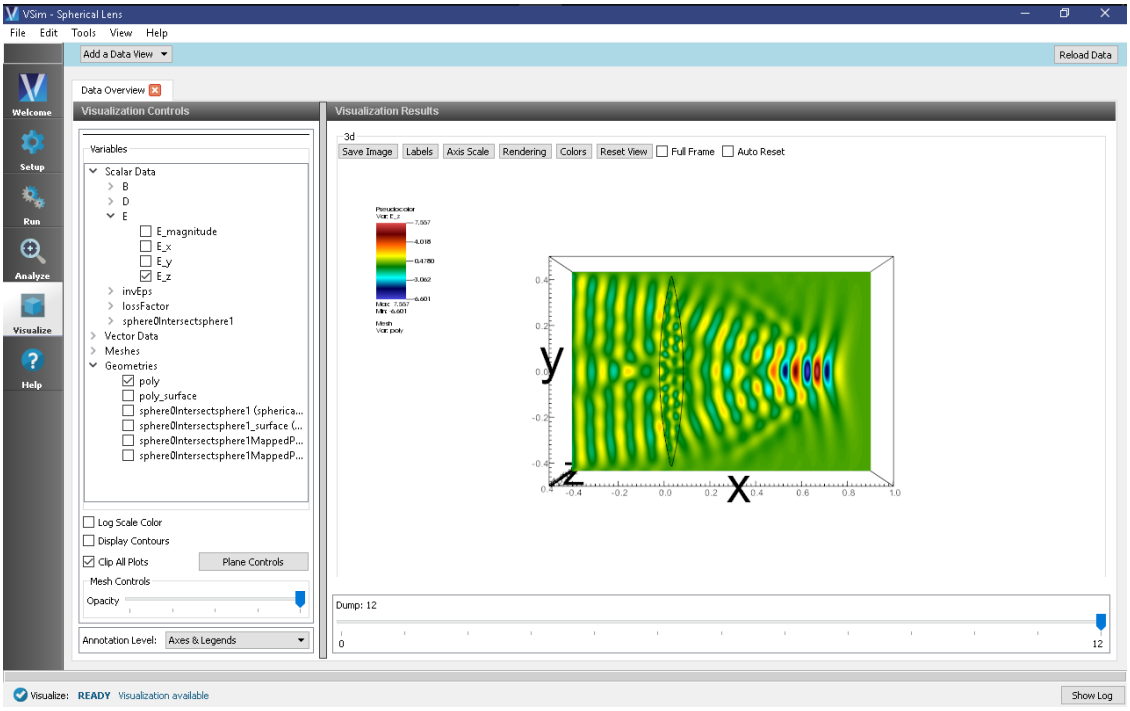


Fig. 3.133: Visualization of the lens focusing

Reduce the sphere radii to have more focusing.

3.8 Other EM (text-based setup)

3.8.1 Specific Absorption Rate (humanHeadT.pre)

Keywords:

dielectrics, power calculations, stl files

Problem Description

The Specific Absorption Rate simulation computes the power absorption in a human head where the brain tissue is approximated using a salt water model. A dipole source is included to imitate a simple antenna source from a cell phone. This example can serve as the basis for a true specific absorption rate calculation for a human head with a source coming from a cell phone antenna.

This simulation can only be performed with a VSimEM, VSimMD or VSimPD license.

Opening the Simulation

The Human Head example is accessed from within VSimComposer by the following actions:

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim for Electromagnetics* option.

- Expand the *Other EM (text-based setup)* option.
- Select "Specific Absorption Rate (text-based setup)" 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 basic variables of this problem should now be alterable via the text boxes in the right pane of the "Setup" window, as shown in Fig. 3.134.

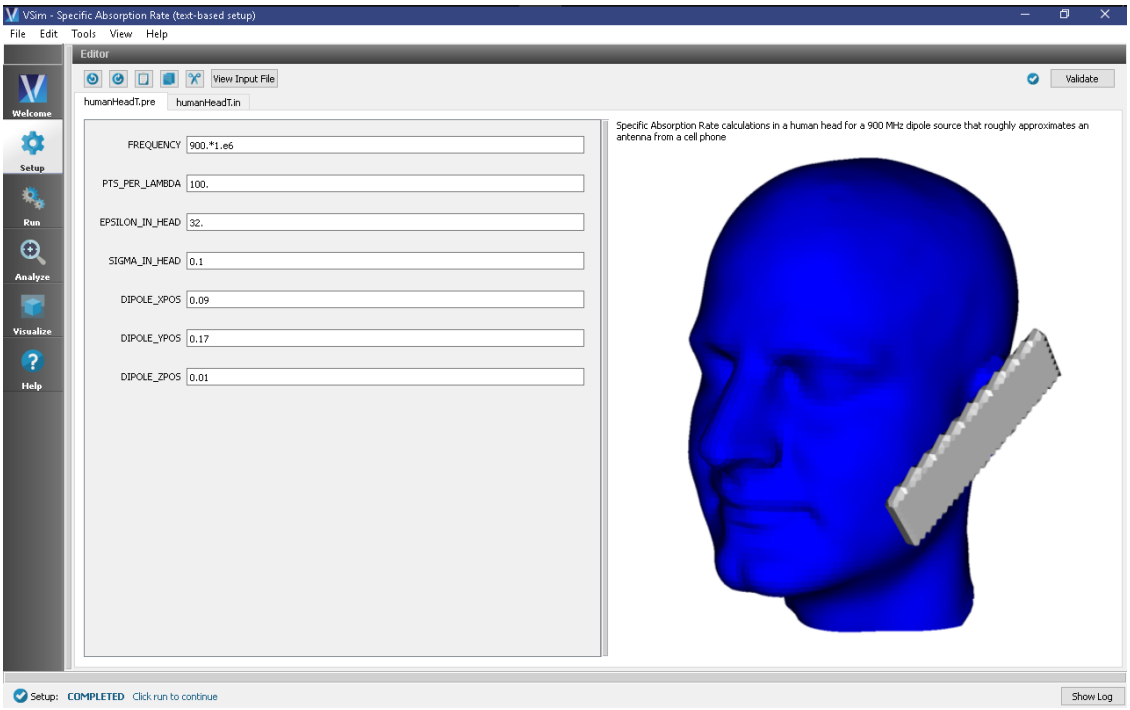


Fig. 3.134: Setup window for the Human Head example.

Input File Features

The input file allows one to select the frequency of the dipole source as well as the number of grid points to include per wavelength for the wave in a vacuum. One can also set the dielectric value and conductivity value in the human head. We have also included the ability to select the position of the dipole source approximating the cell phone antenna. A voxel representation of the human head can also be used for more specific tissue values via a voxel dat file with python.

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.
- To run the file, click on the *Run* button in the upper left corner of the right panel. 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 is shown in Fig. 3.135.

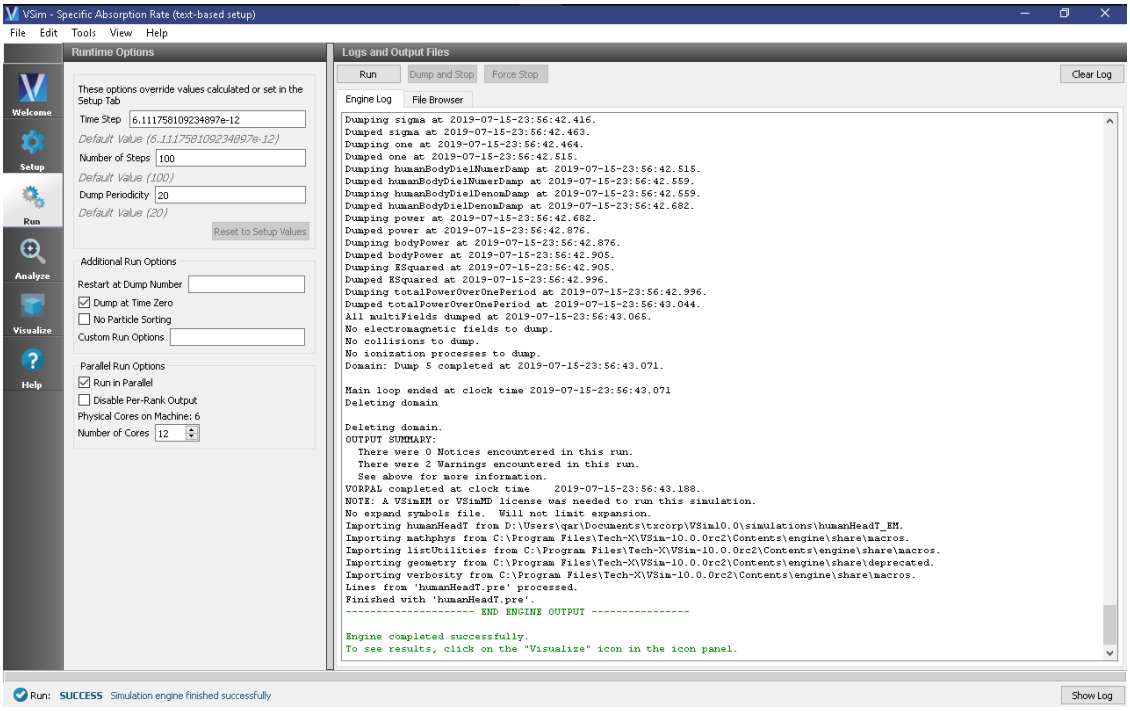


Fig. 3.135: The Run window at the end of execution.

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 create the image shown in Fig. 3.136:

- Expand *Scalar Data*
- Select *E* and *ESquared*
- Expand *Geometries*
- Select *poly*
- Select *Log Scale Color*
- Select *Display Contours* and set the # of contours to 7
- Select *Clip All Plots*
- Click the *Plane Controls* button and set the *Clip Plane Normal* to X
- Set the *Origin Of Normal Vector* to X = 0.1, Y = 0, Z = 0
- Move the dump slider forward in time
- Click and drag with the mouse to rotate the image

Further Experiments

We suggest the user change the frequency of the dipole source to imitate different cell phone models at different frequencies.

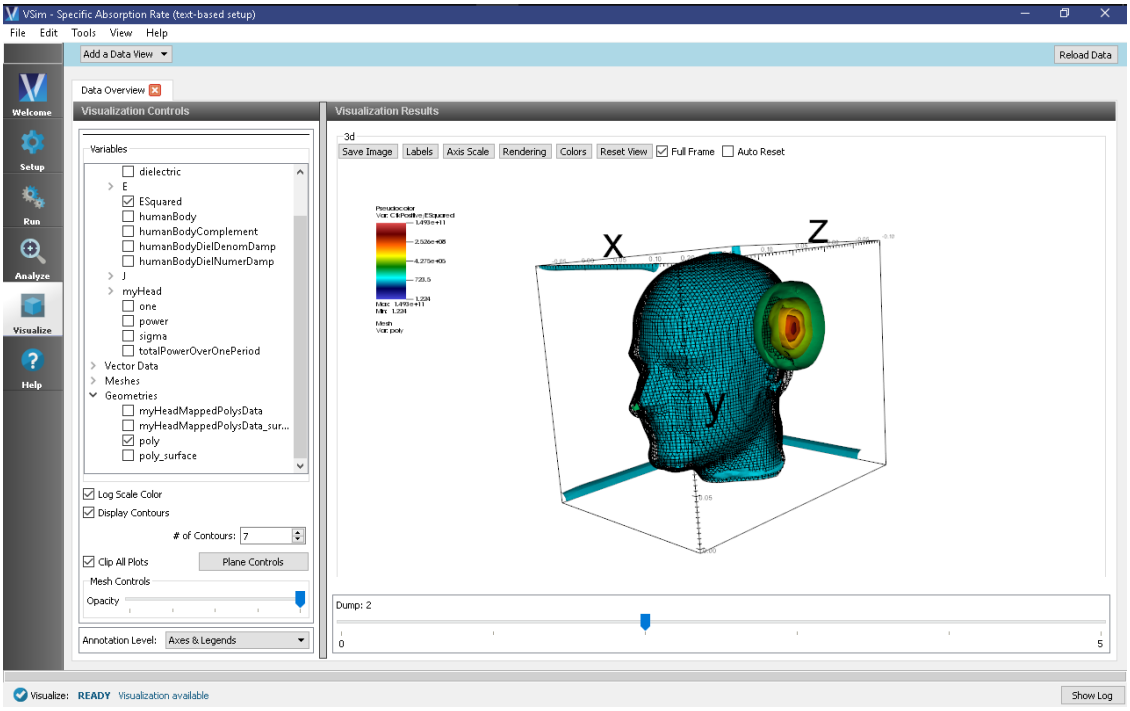


Fig. 3.136: Visualization of the absorption of power by a human head via a clip.

We also suggest the user change the position of the dipole implying a change in location of the cell phone antenna.

It would also be interesting to change the dielectric and conductivity value to model a dipole source in a vacuum.

3.8.2 Photonic Crystal in Metal Cavity (phcInMetalCavityT.pre)

Problem Description

A photonic crystal (PhC) is capable of confining electromagnetic fields in waveguides and cavities using a periodic geometry. This simulation features a dielectric photonic crystal cavity—a triangular lattice of dielectric rods, with one rod removed—inside a metal cavity. The cavity axis, and the dielectric rods, are in the z direction.

The photonic crystal structure is similar to that described in *[BWC08]*, truncated after two layers of the lattice structure. The metal cavity resembles an elliptical (or rounded pillbox) cavity, with short beam tubes.

Modeling dielectric and metal can be difficult: at dielectric corners or triple points (where dielectric, metal, and vacuum meet), the electromagnetic fields generally must diverge (to infinity) to preserve continuity dictated by Maxwell's equations *[Had02]*. However, when the interface between dielectric and vacuum is always perpendicular to the metal surface, as in this simulation, the fields remain finite.

This simulation demonstrates a method for combining dielectric and metal, as long as the metal surface is perpendicular to x, y, or z whenever it intersects dielectric (and the vacuum/dielectric interface remains perpendicular to the metal surface at those points).

When the PhC cavity mode is excited, the fields are trapped radially mainly by the dielectric rods.

This simulation can be performed with a VSimEM, VSimMD or VSimPA license.

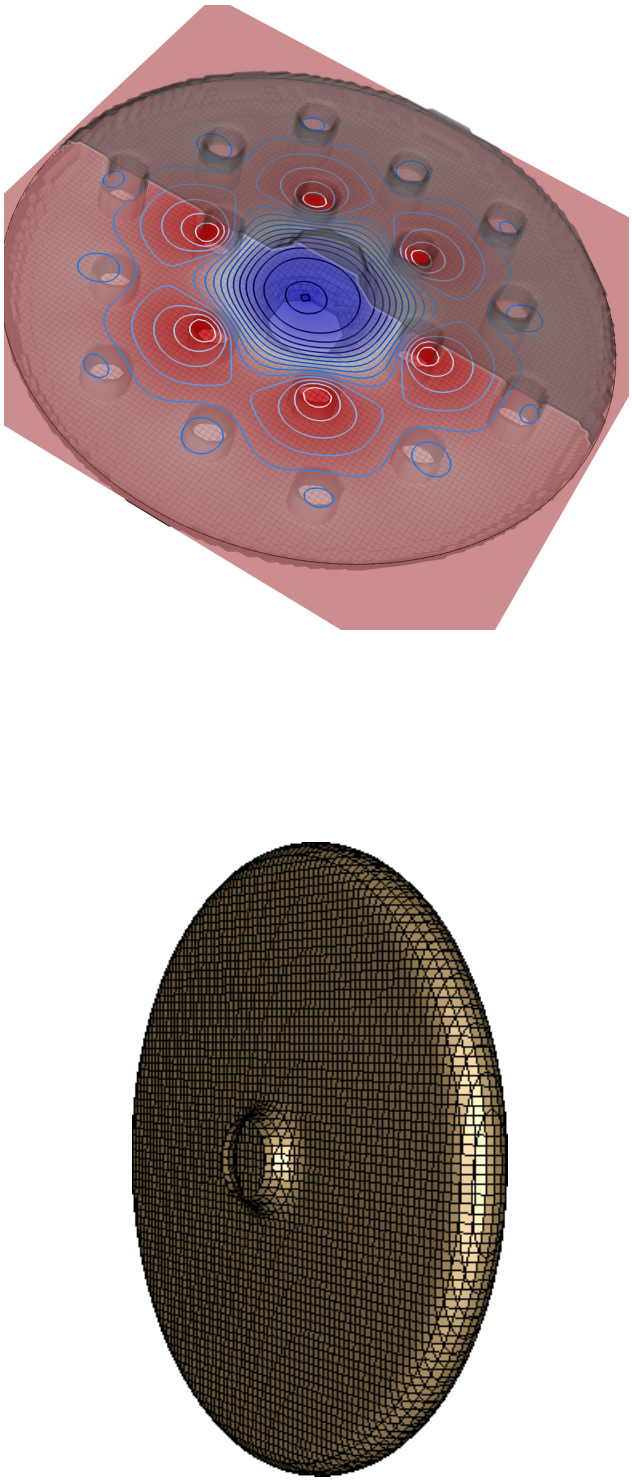


Fig. 3.137: The metal cavity surrounding the PhC structure.

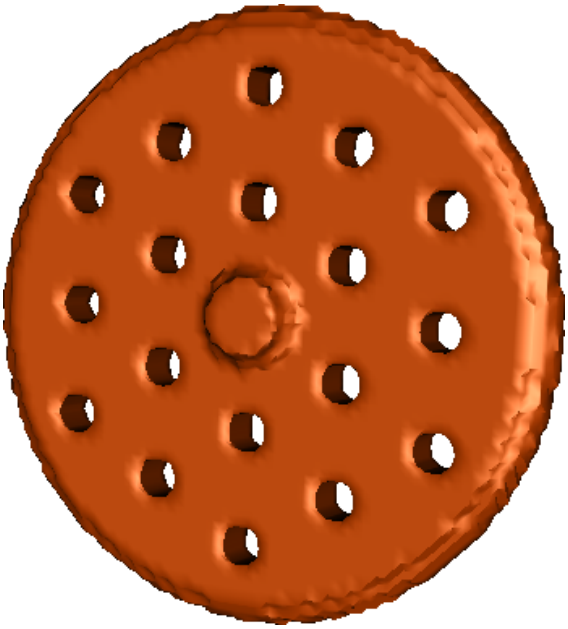


Fig. 3.138: The vacuum region (inside the metal cavity, outside the dielectric rods).

Opening the Simulation

The PhC in Metal Cavity example is accessed from within VSimComposer by the following actions

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim for Electromagnetics* option.
- Expand the *Other EM (text-based setup)* option.
- Select "Photonic Crystal in Metal Cavity (text-based setup)" 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 basic variables of this problem will now be alterable via the text boxes in the left pane of the *Setup Window*, as shown in Fig. 3.139.

Input File Features

The input file allows the user to choose the dielectric contrast and radius of the rods, the shapes and sizes of the cavity and beam tubes (to some extent), the grid resolution, and the number of oscillations to simulate after excitation. The entire simulation is scaled to the lattice constant, which is set to 1 by default.

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.
- To run the file, click on the *Run* button in the upper left corner of the right pane of the window. You will see the output of the run in that pane. The run has completed when you see the output "Engine completed successfully." This is shown in Fig. 3.140.

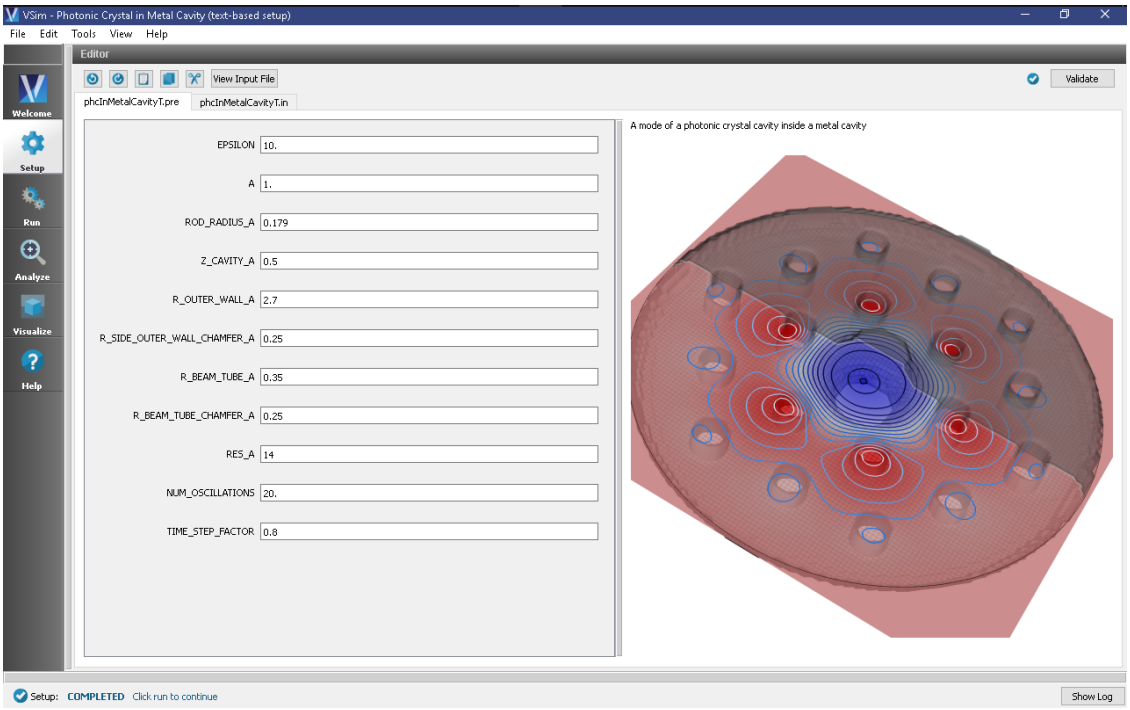


Fig. 3.139: Setup Window for the phcInMetalCavityT example.

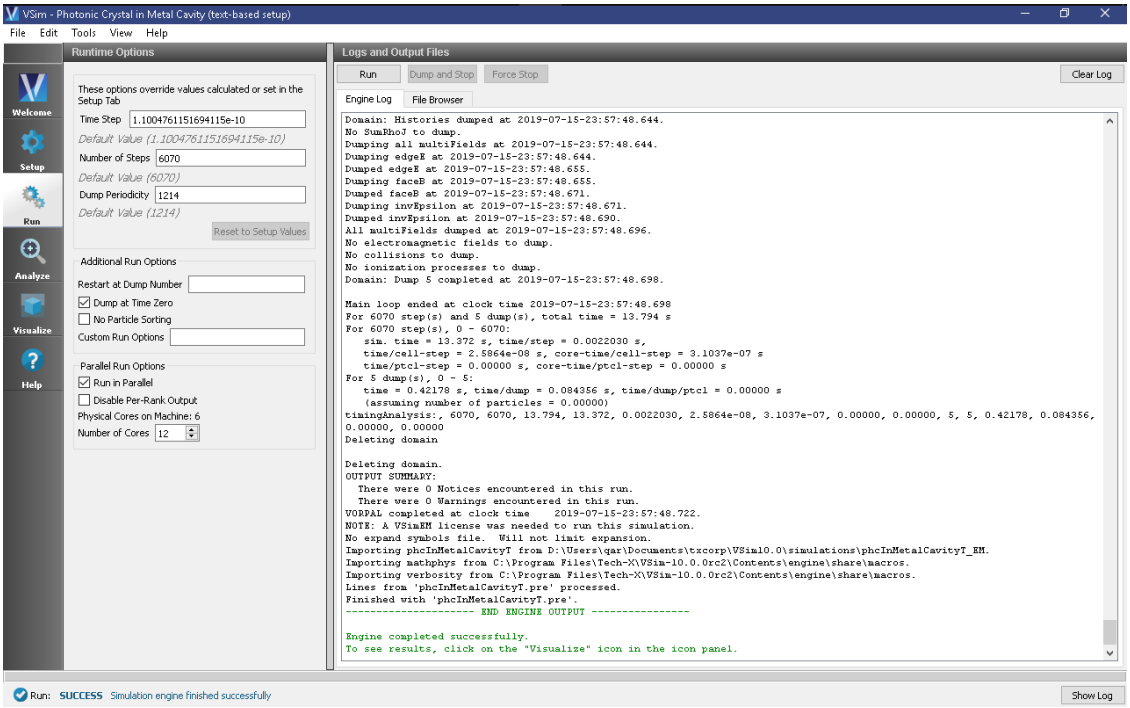


Fig. 3.140: The Run Window at the end of execution.

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.

Due to the symmetry of this system, the results are best viewed by looking at the z component of the electric field as follows:

- Expand *Scalar Data*
- Expand *edgeE*
- Select *edgeE_z*
- Select *Clip All Plots* (the default Normal is already in the z-direction, so no need to change it)
- Expand *Geometries*
- Select *poly*

The field at dump 2 is shown in Fig. 3.141.

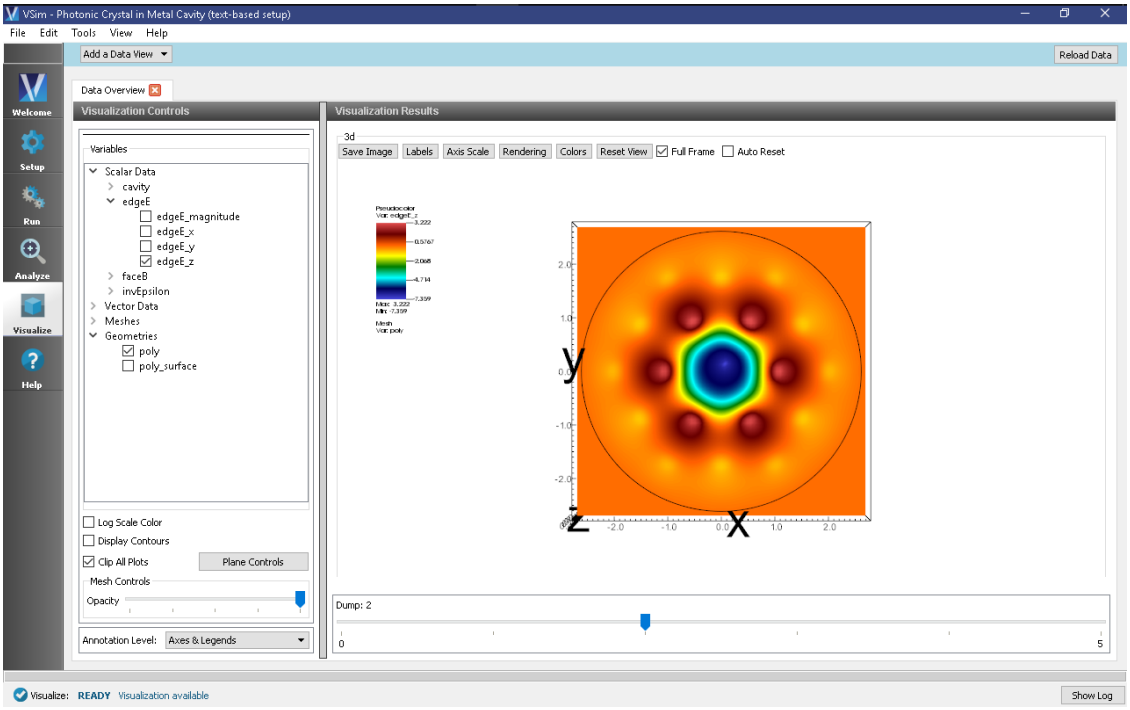


Fig. 3.141: Visualization of the E_z field component.

We can see that fields are trapped by the two layers of dielectric rods, and to a lesser (but final) extent by the surrounding metal cavity.

CHAPTER

FOUR

VSIM FOR MICROWAVE DEVICE EXAMPLES

These examples illustrate how to solve complex problems in microwave devices.

These examples can be run with a VSimMD license.

4.1 Cavities and Waveguides

4.1.1 Coaxial Cylinder (coax.sdf)

Keywords:

`coax, coaxial geometry, cylinder, current pulse, rlc circuit, step potential`

Problem description

This example probes the electromagnetic properties of a semi-infinite coaxial cylinder. One end of the cylinder lies in the simulation space. The length of the cable is large compared to its diameter. The outer radius is 8 cm, the inner radius is 2 cm, and the section considered is 20 cm long. The inner cylinder is shorter than the outer cylinder and there is an electron absorbing cap on the end of the outer cylinder. When the simulation initiates, a single EM pulse is launched into the open, continuous end of the geometry and propagates to the capped tip. Electrons are ejected from the tip of the inner cylinder when the pulse reaches it.

This computational model is equivalent to applying a step-potential to one end of a coaxial cable. The step-potential propagates at the speed of light until it reaches the tip of the inner cylinder. The RLC nature of the coax cable causes overshoot and ringing of the potential. At the inner tip, an attenuating series of oscillations occurs accompanied by electron emissions. Gradually the tip potential stabilizes at the applied potential.

This simulation can be performed with a VSimMD license.

Opening the simulation

The coax example is accessed from within VSimComposer by the following actions:

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim for Microwave Devices* option.
- Expand the *Cavities and Waveguides* option.
- Select *Coaxial Cylinder* 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.

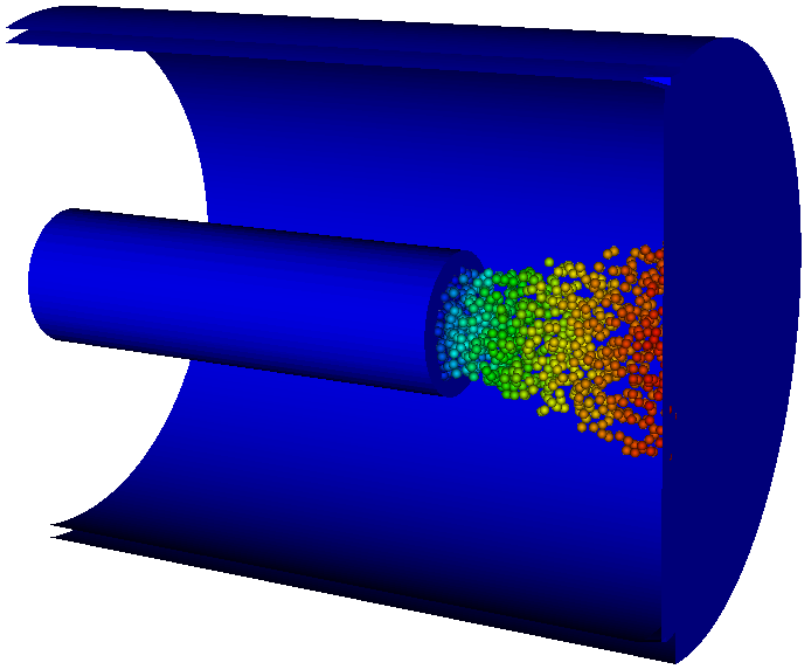


Fig. 4.1: The electrons are emitted from the tip of the inner cylinder after the pulse reaches it.

All of the properties and values that create the simulation are now available in the Setup Window as shown in Fig. 4.2. 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`.

Simulation properties

The coax example includes several Constants for easy adjustment of simulation properties. Those include:

- TFACTOR: A ramping factor of the applied field
- EFACTOR: The amplitude of the applied field
- EMITTED_CURRENT: The current emitted from the tip of the inner cylinder

There are also several SpaceTimeFunctions defined for easy application to wave launchers and particle emitters. Those include:

- edgeDy: The applied field in the y-direction
- edgeDz: The applied field in the z-direction
- nomask: This allows emission from the entire geometry of the flux emitter

Other Properties of the simulation include CSG defined geometries, a wave launcher on the lower x boundary, and a settable flux emitter on the tip of the inner cylinder.

Running the simulation

After performing the above actions, continue as follows:

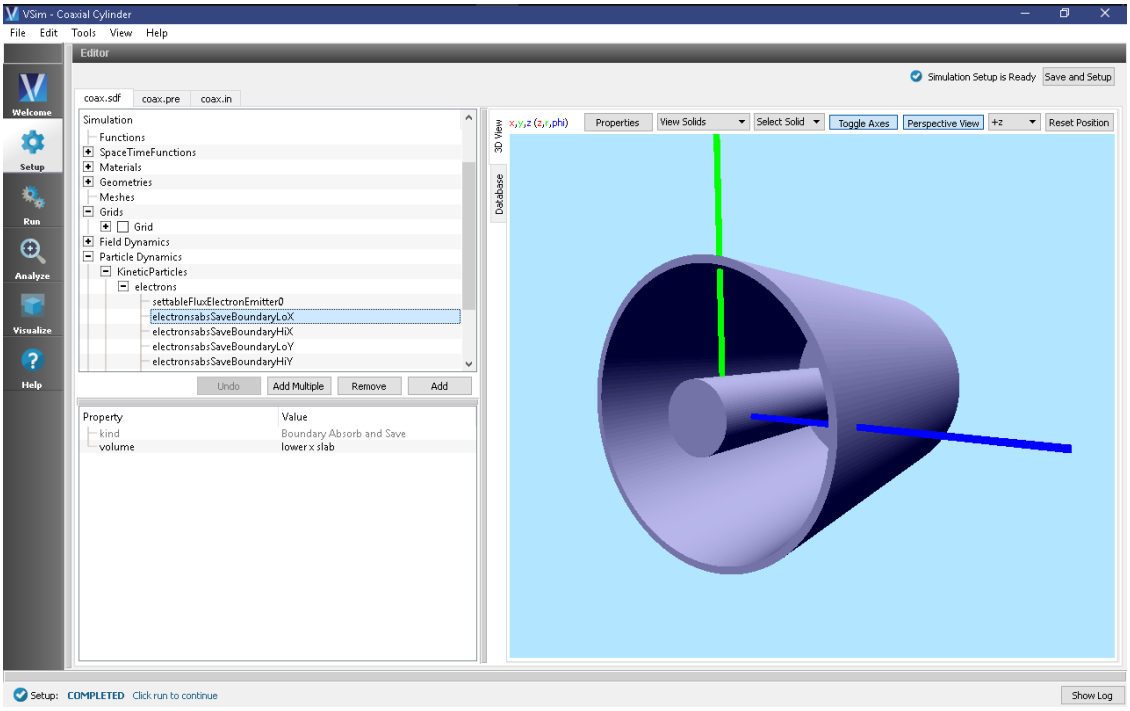


Fig. 4.2: Setup Window for the Coaxial Cylinder example.

- Proceed to the Run Window by pressing the *Run* button in the left column.
- To run the file, click 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 is shown in Fig. 4.3.

Visualizing the results

After performing the above actions, continue as follows:

- Proceed to the Visualize Window by clicking the *Visualize* button in the left column.

To create the image seen in Fig. 4.4, proceed as follows:

- In the variables tree expand *Scalar Data*
- Expand *E*
- Select *E_y*
- Expand *Geometries*
- Select *poly (coaxGeom)*
- Beneath the variables tree, select *Display Contours* and set the # of contours to 10
- Select *Clip All Plots*
- In the Particle Style section, set the size to 6 and choose Sphere for Symbol
- In the variables tree expand *Particle Data*
- Expand *electrons*
- Select *electrons_ux*

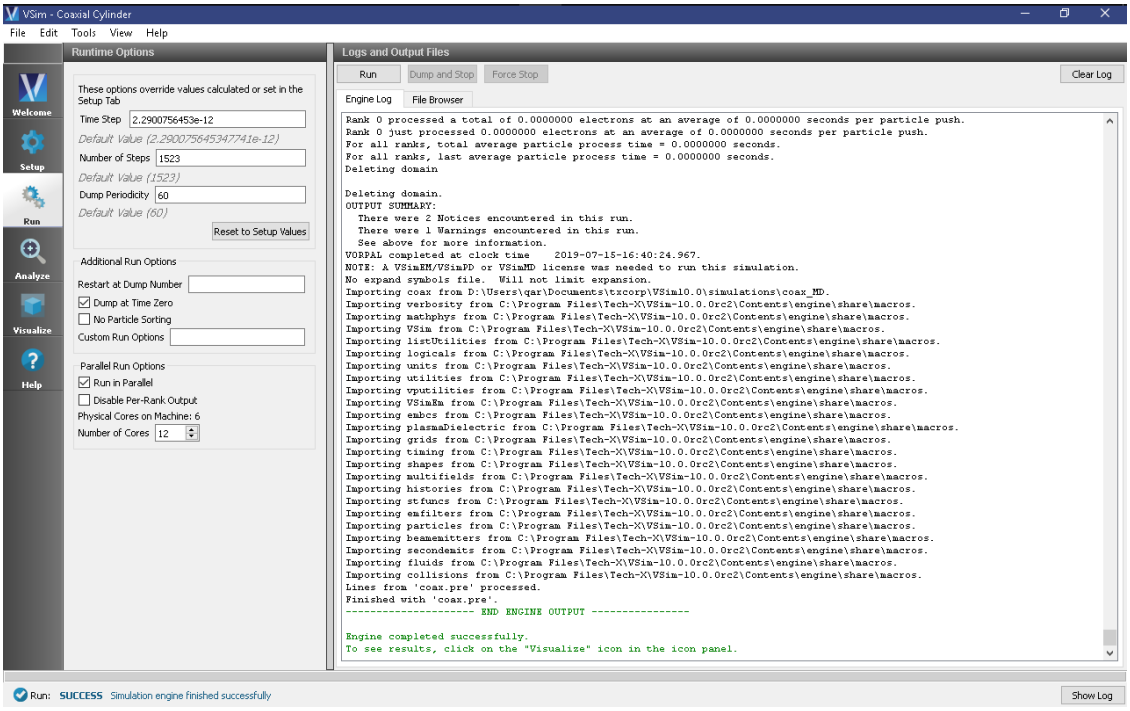


Fig. 4.3: The Run Window at the end of a successful execution.

- Now in the right pane move the dump slider forward in time
- The axis and legends can be hidden using the dropdown menu in the lower left corner of the window

To obtain a clearer picture of what is happening at the cylinder tip, switch the Data View (in the left pane) to *History*. One dimensional plots of the number of electrons (called numMacroPtcls), the electric potential (phi), and the emitted and absorbed current should come up automatically.

You can set the location of Graph 2 to Window 1 as in Fig. 4.5.

The potential is measured between the interior of the inner cylinder and the capped end of the outer cylinder. The plot of the potential is noisy due to the emission of electrons from the tip. It may be insightful to run the simulation once without electrons so you can see the ringing on the waveform of phi. A similar signal is obtained by hooking up an oscilloscope to a coaxial cable. Electrons can be suppressed by setting the EMITTED_CURRENT parameter to 0 during setup.

The coaxial cylinder behaves like an RLC circuit: the cylinders provide a series resistance along their length, they are coupled capacitively, and generate self-inductance due to the current. By default, the rise-time of the pulse is near the resonance of the circuit, resulting in an acceptable rise time, low overshoot, and quick damping. This makes it a good driver of the circuit.

Further Experiments

Try experimenting with different dimensions of coax. In particular, note how the radii and pulse profile affect the potential response on the phi History plot.

4.1.2 Cylindrical Waveguide (cylindricalWaveguide.sdf)

Keywords:

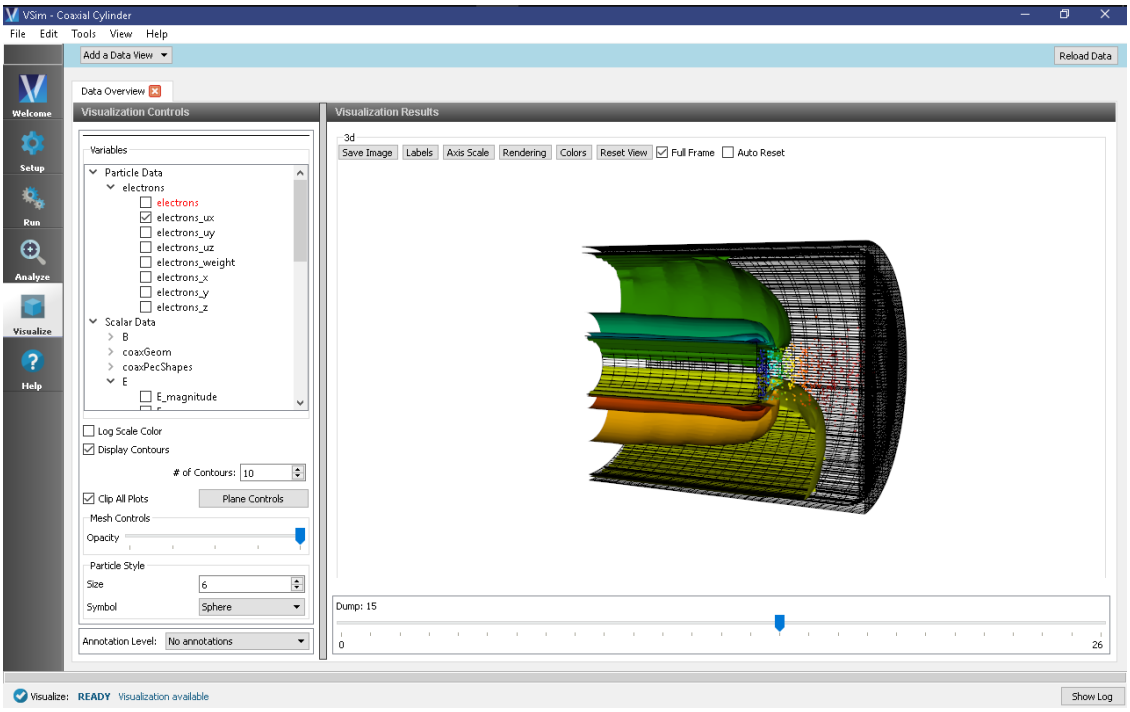


Fig. 4.4: Visualization of the coaxial cylinder as a color contour plot.

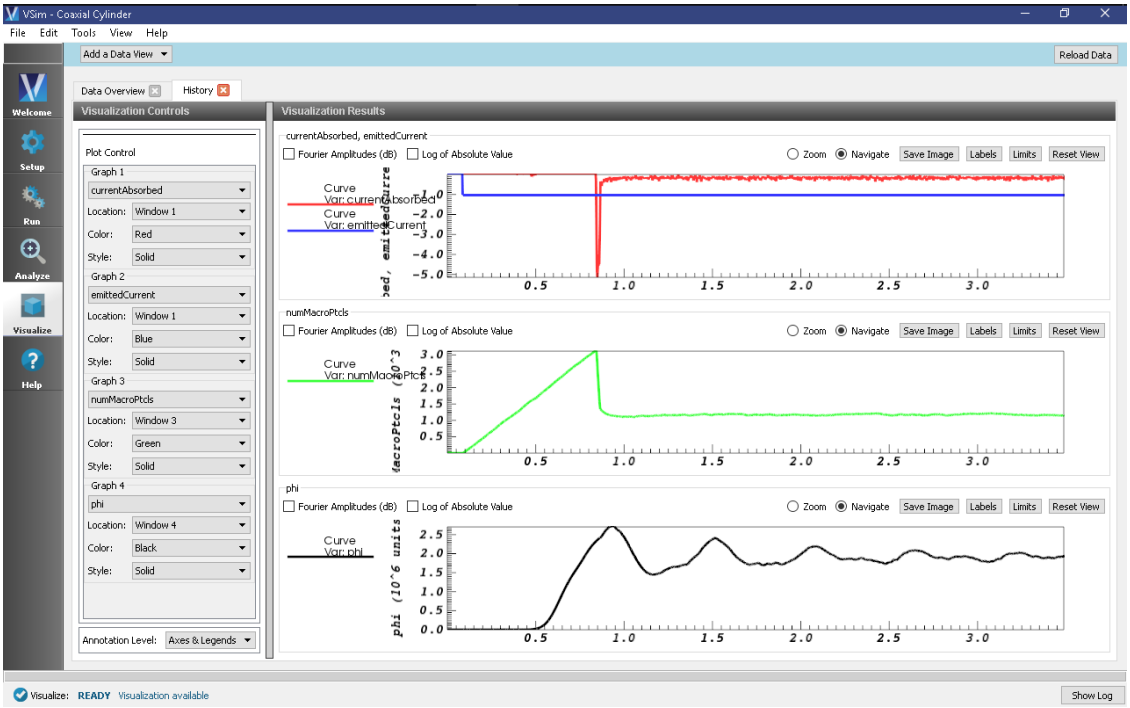


Fig. 4.5: The History visualization window with the electrons.

electromagnetics, waveguide, dispersion

Problem Description

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

This simulation can be performed with a VSimMD license.

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 $I_0 = I(\tau/2)$. 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 VSimComposer by the following actions:

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim for Microwave Devices* 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. 4.6. 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.

Opening `Functions` of the tree shows a function, freqBandPulse, which is

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

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 $f < f_h + \delta_f$. 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 icon in the left panel.
- Check the center panel that you will run for 20,000 steps, dumping every 2,000.

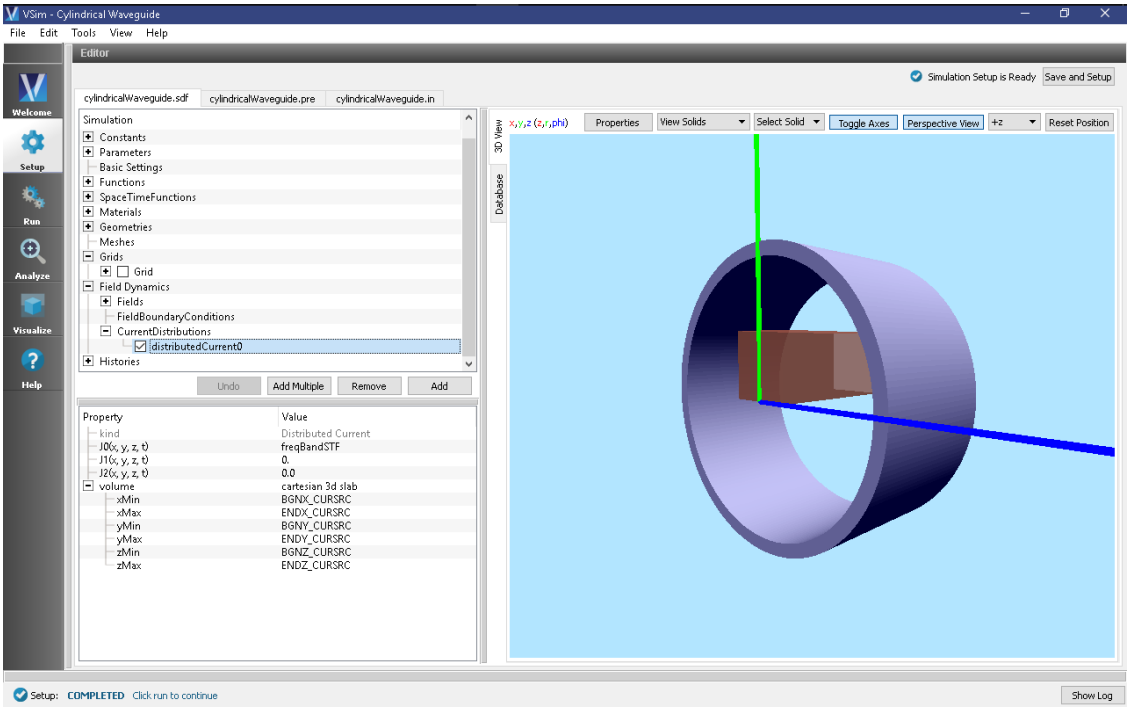


Fig. 4.6: Initial Setup Window for the Cylindrical Waveguide example.

- To run the file, click on the *Run* button in the upper left corner of the right panel. 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 is shown in Fig. 4.7.

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.
- For Graph 4, select <None> to be plotted.
- 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. 4.8.

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.

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. 4.9.

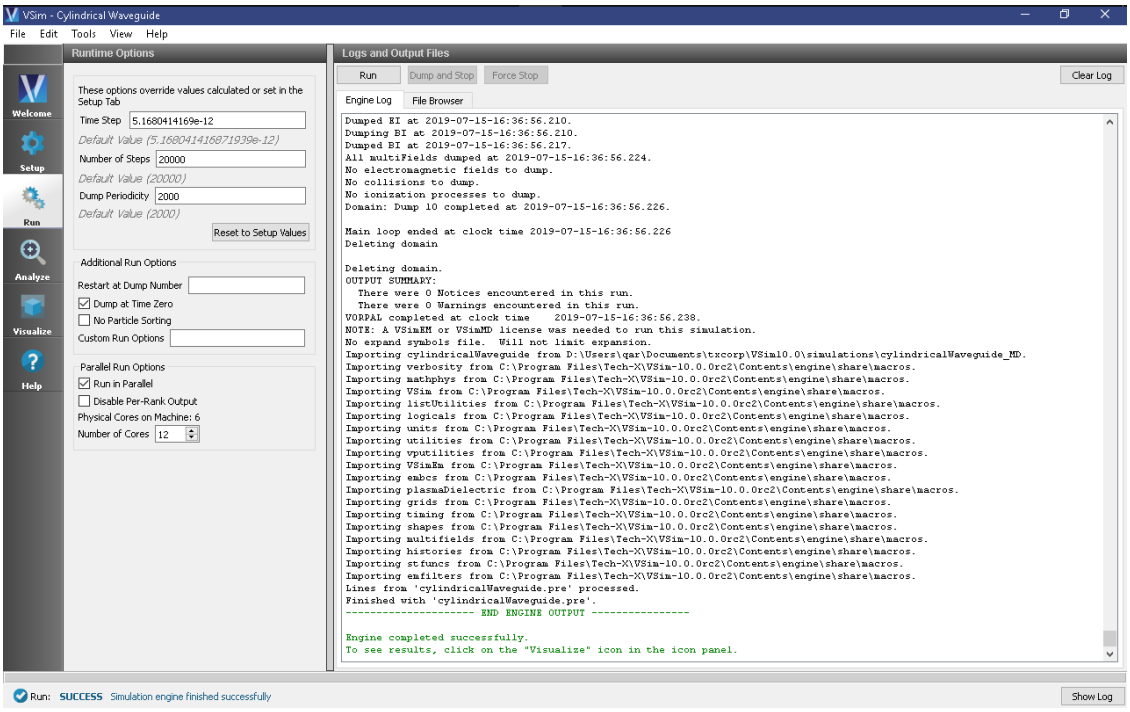


Fig. 4.7: Run Window for the Cylindrical Waveguide example after the initial run.

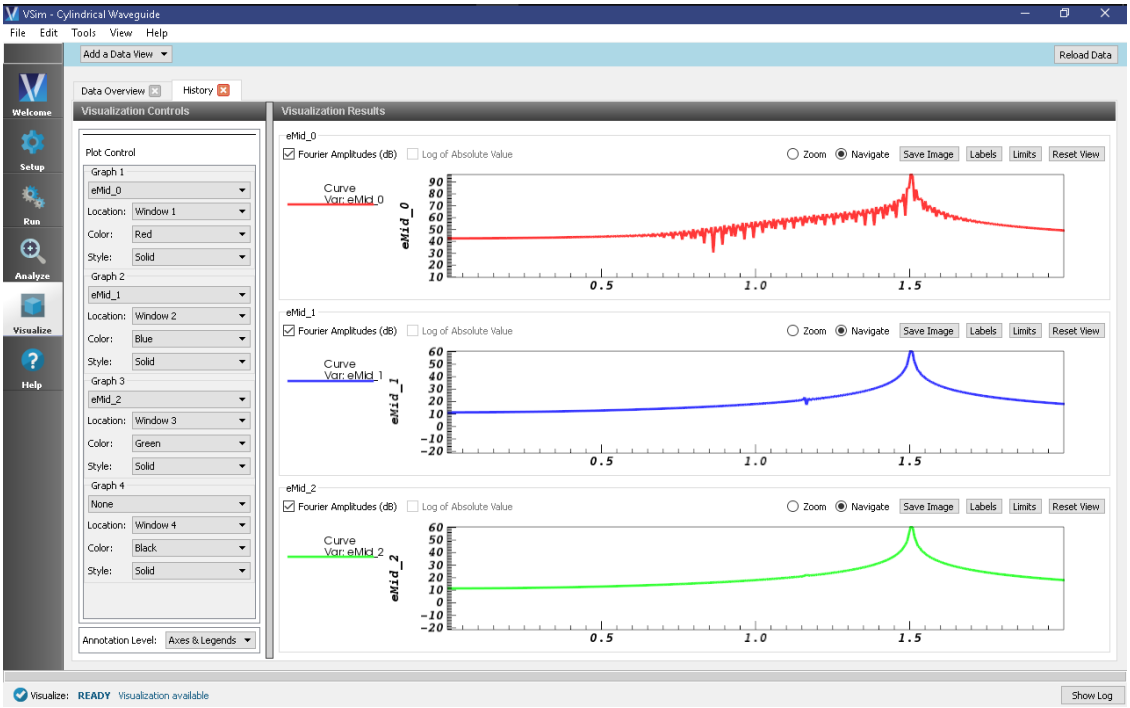


Fig. 4.8: Spectrum for the Cylindrical Waveguide example after the initial run.

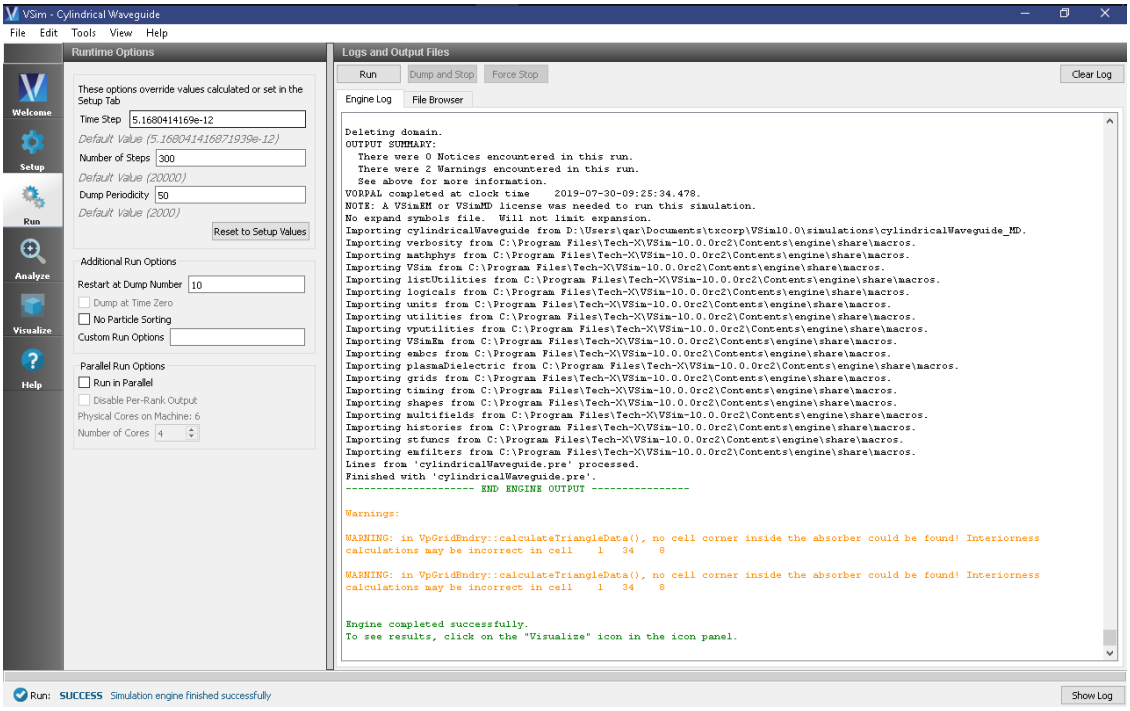


Fig. 4.9: Run Window for the Cylindrical Waveguide example for the second run.

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. 4.10.

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. 4.11. The well obtained mode occupies dumps 1-16.

4.1.3 Pillbox Cavity (pillboxCavity.sdf)

Keywords:

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

Problem description

This VSimMD example demonstrates the usage of VSim 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.

This simulation can be performed with a VSimMD or VSimEM license.

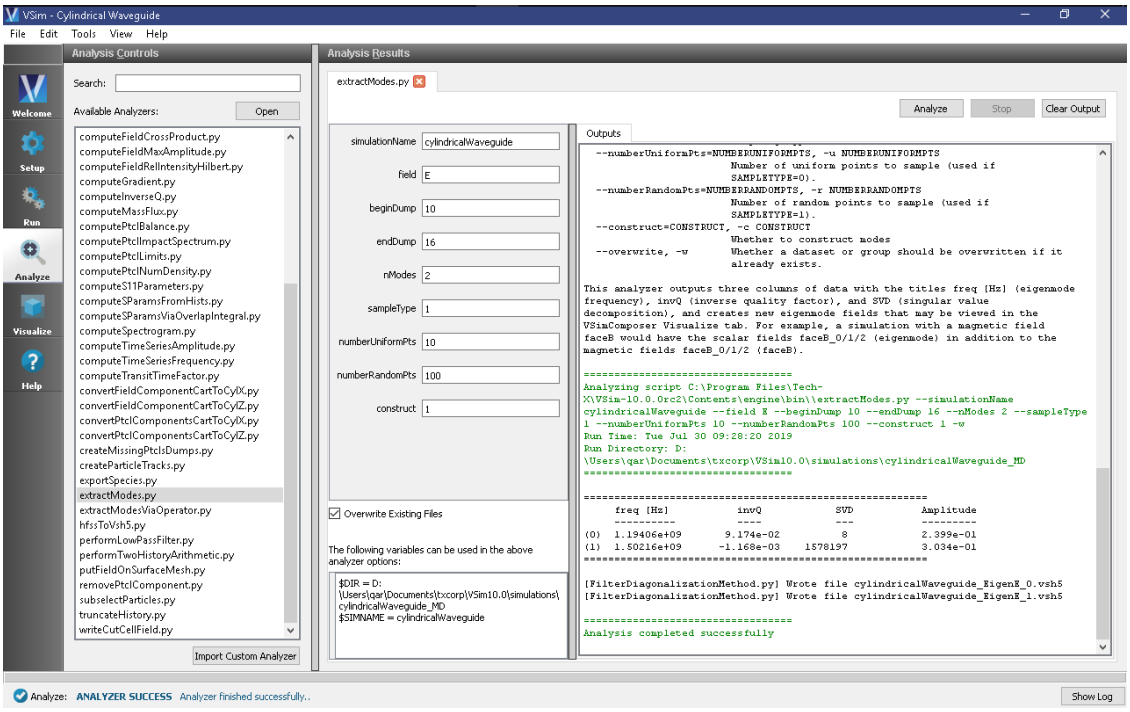


Fig. 4.10: Analysis window for the Cylindrical Waveguide example for mode extraction.

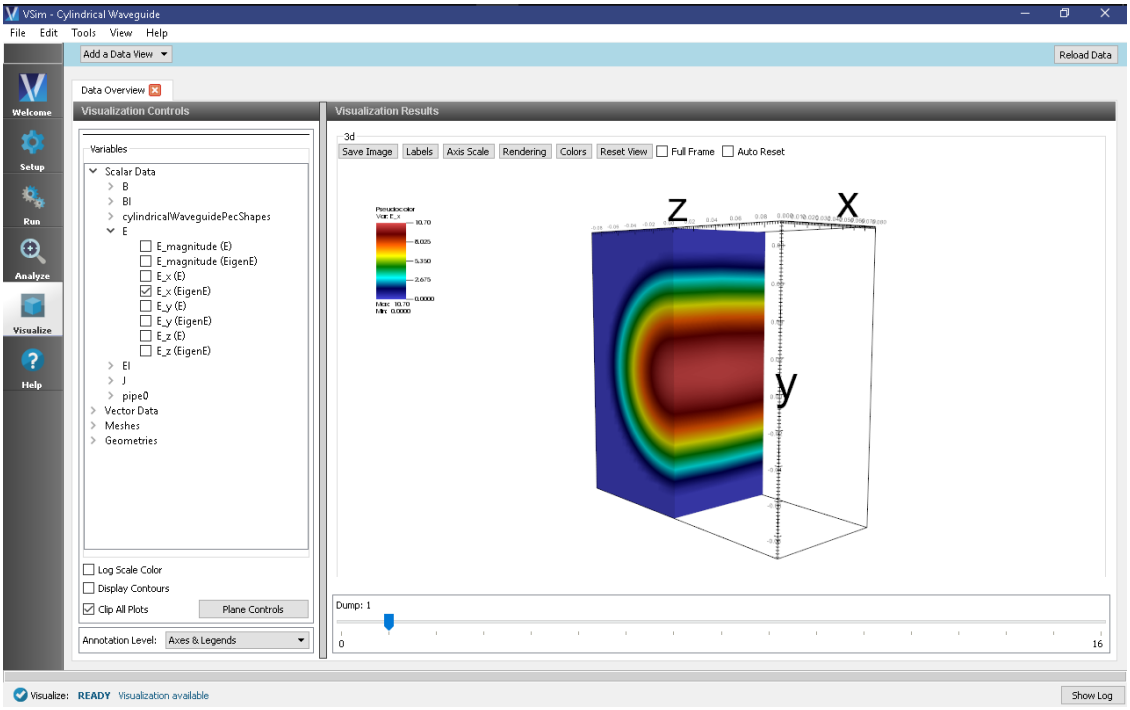


Fig. 4.11: Analysis window for the Cylindrical Waveguide example for mode extraction.

Opening the Simulation

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

- Go to *File* → *New* → *From Example...*
- In the resulting *Examples* window expand the *VSim for Microwave Devices* 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*.

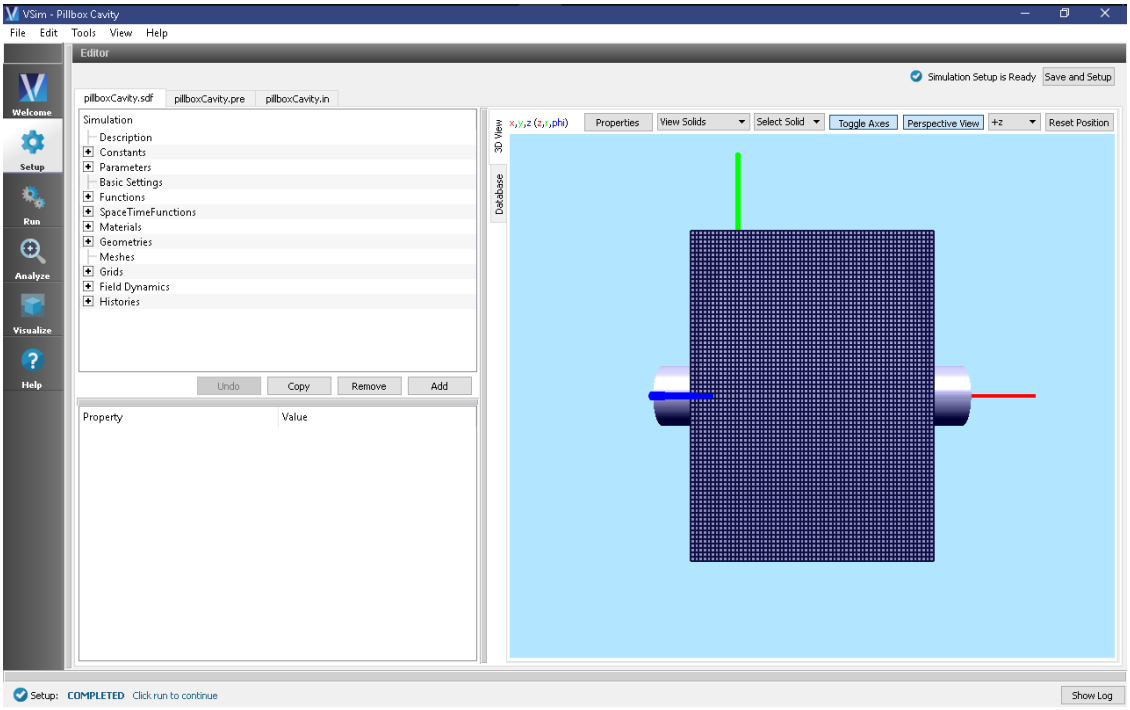


Fig. 4.12: Visualizing the periodic cavity geometry in the Setup Window.

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 *Parallel Run Options* box below *Additional Run Options* in the *Runtime Options* pane.
- 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. 4.13.

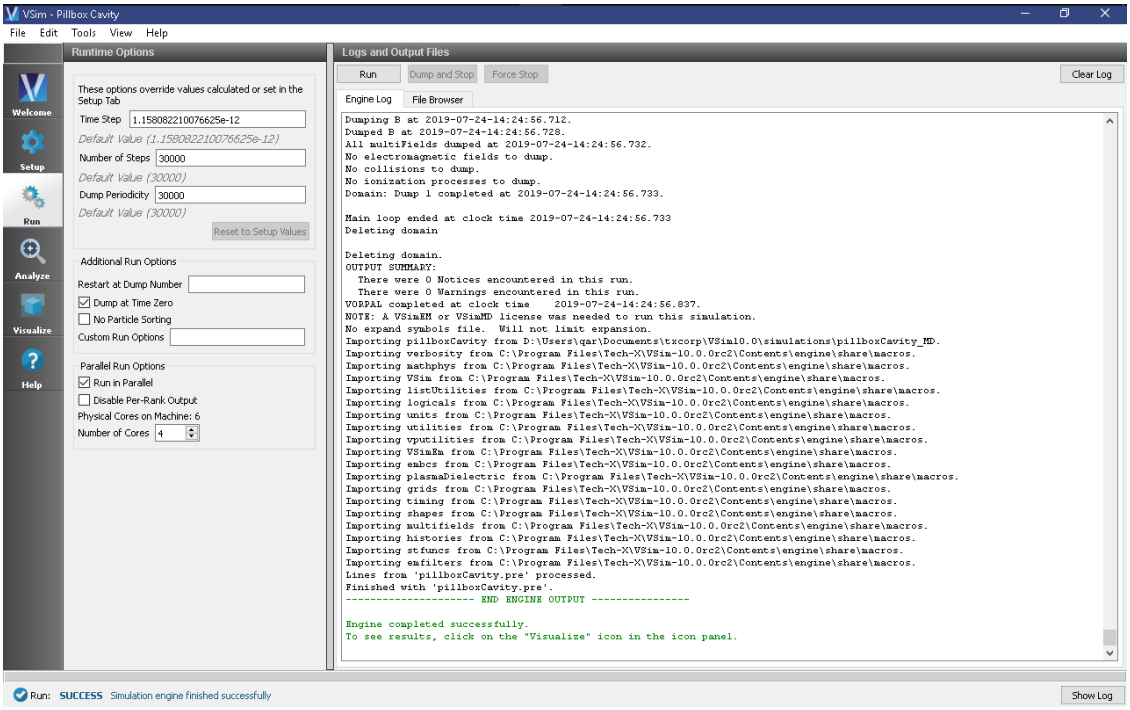


Fig. 4.13: The Run window at the end of a successful execution.

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. 4.14. 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.

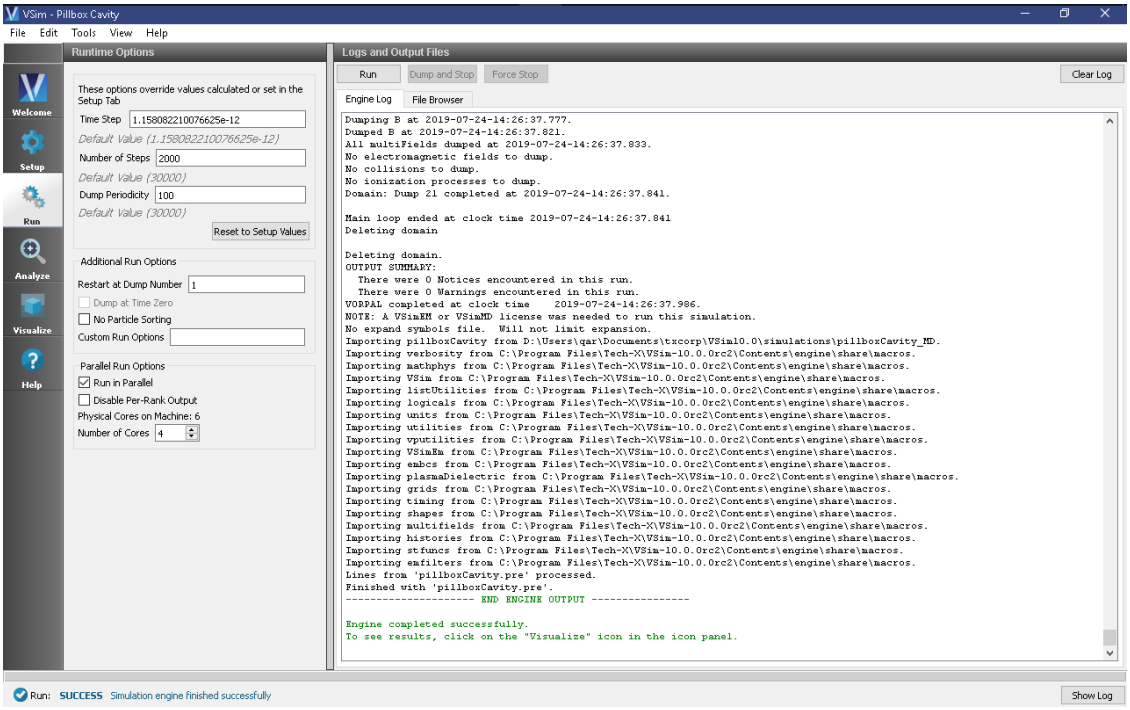


Fig. 4.14: The Run window at the end of a successful execution.

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"
  - **`sampleType`**: "0"
  - **`numberUniformPoints`**: "20"
  - **`numberRandomPoints`**: "100"
  - **`construct`**: "1"

Also, check the "Overwrite Existing Files" box. Double-check your entries against what is shown in Fig. 4.15.

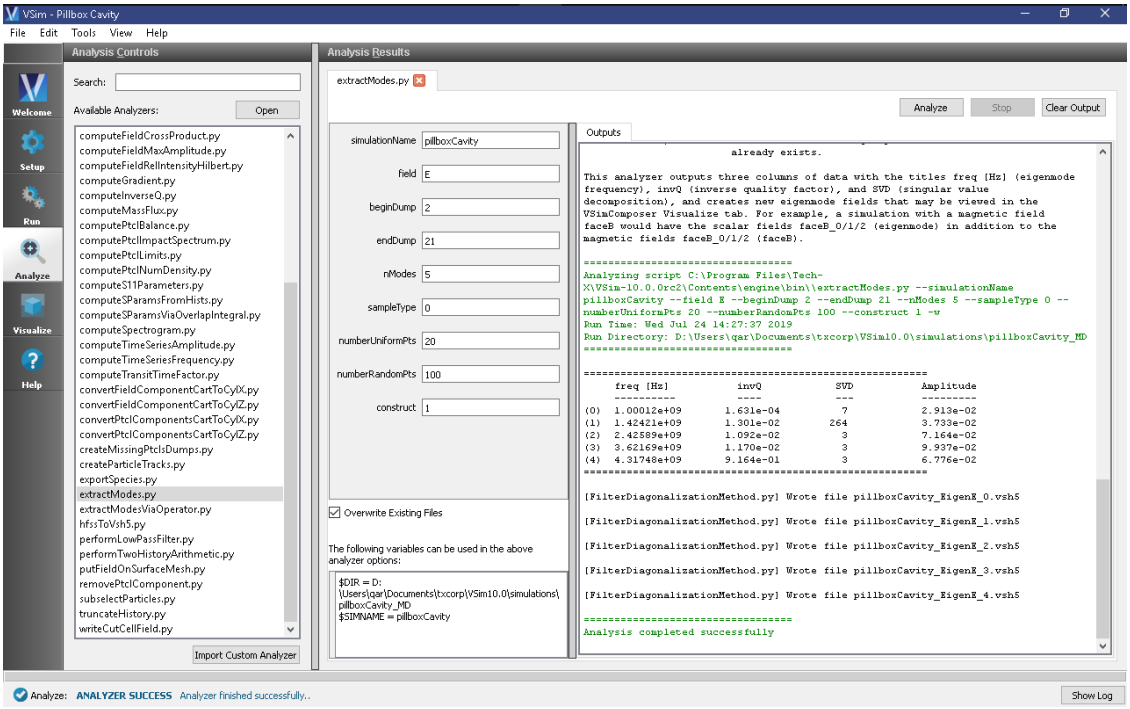


Fig. 4.15: Computing the electric field eigenfunctions and frequencies using the extractModes.py analyzer.

- Press the *Analyze* button which is located in the upper right corner.
- Compute the magnetic field eigenfunctions with the following parameters.
  - **simulationName:** "pillboxCavity"
  - **field:** "B"
  - **beginDump:** "2"
  - **endDump:** "21"
  - **nModes:** "5"
  - **sampleType:** "0"
  - **numberUniformPoints:** "20"
  - **numberRandomPoints:** "100"
  - **construct:** "1"

  After the analysis is finished, and scrolling down in the *Outputs* log pane you should see what is shown in Fig. 4.16.
- 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.

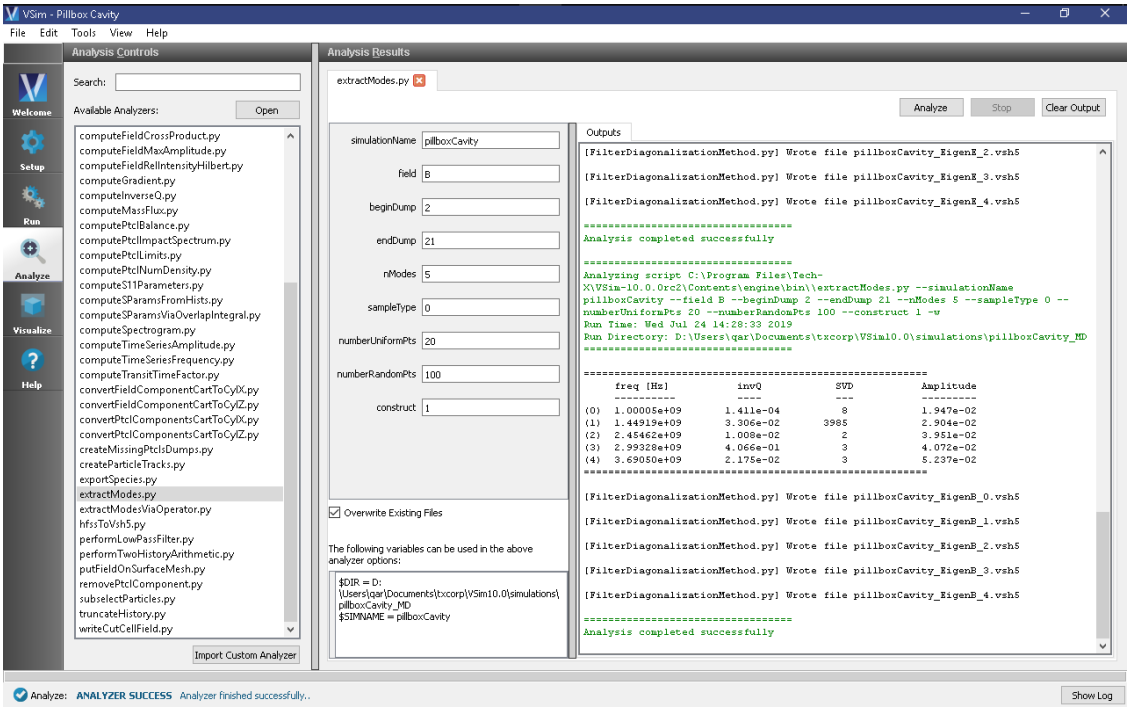


Fig. 4.16: The *Outputs* pane after Analyzing to determine the eigenmodes of the magnetic field.

- After the analyzer loads, ensure the following parameters are entered:
  - **`simulationName`**: "pillboxCavity"
  - **`beginDump`**: "0"
  - **`endDump`**: "0"
  - **`beta`**: "1"
  - **`axis`**: "0"
  - **`offsetx0`**: "0"
  - **`offsetx1`**: "0"

  And compare against what is shown in Fig. 4.17
- 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.
- If you have selected the closed cavity, the geometry factor at the mode frequency of 1000.1MHz should be very close the analytic value of 257.

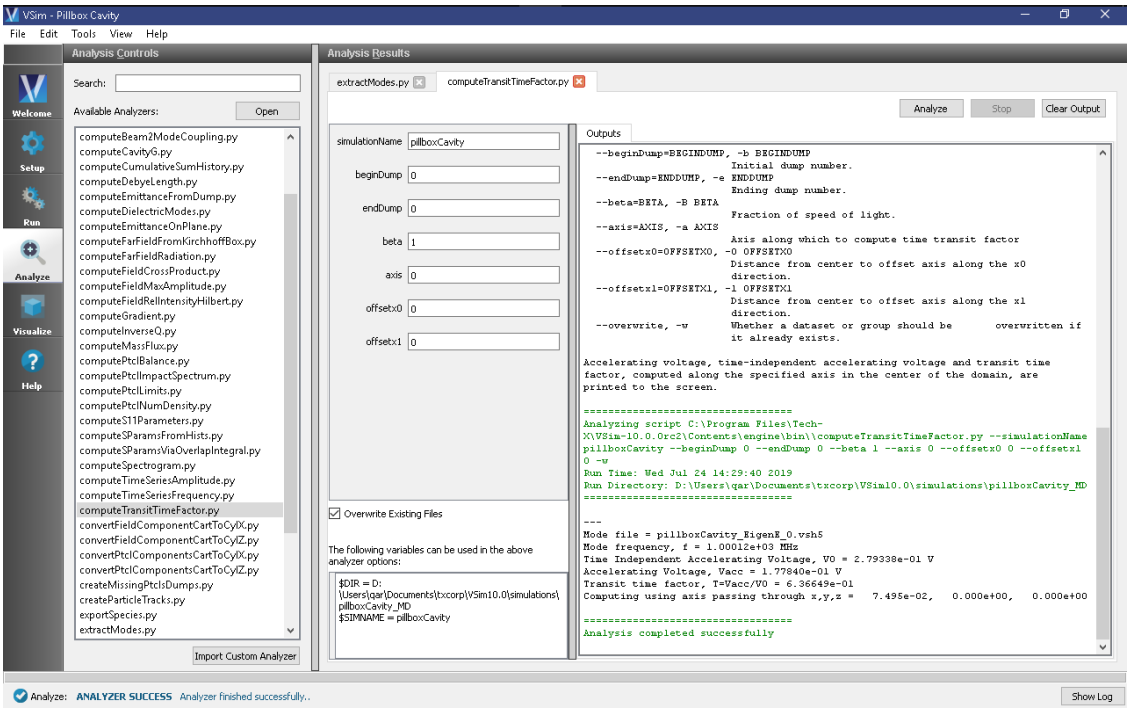


Fig. 4.17: Computing the transit time factor for the eigenmode of interest.

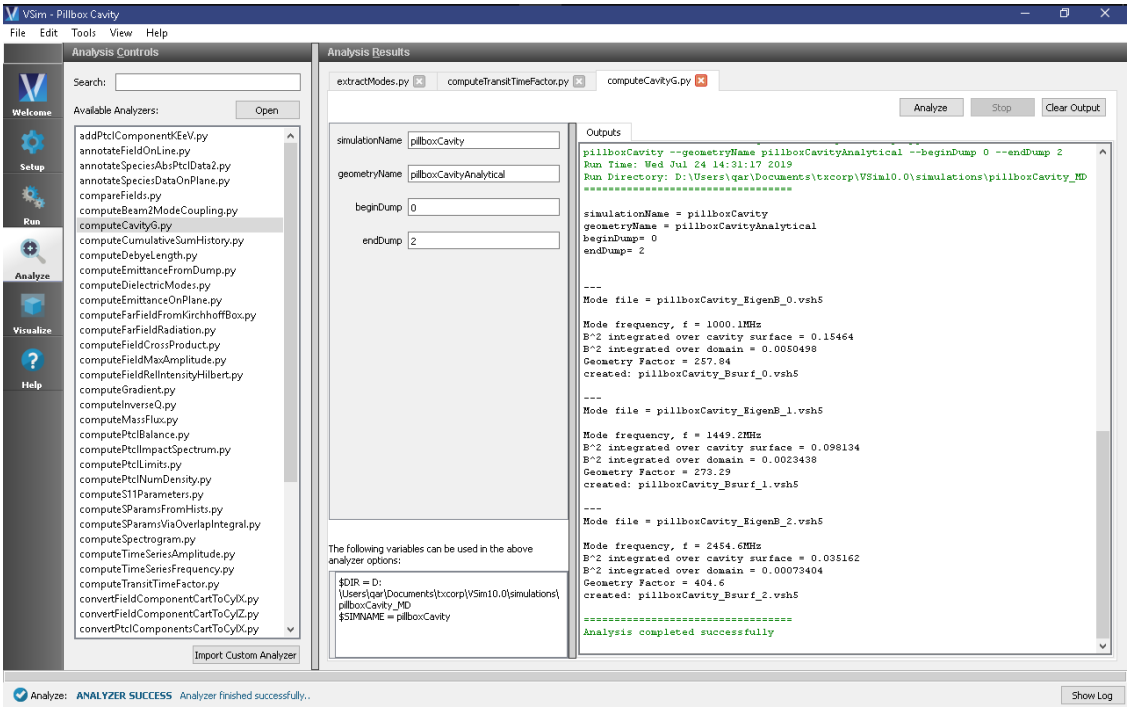


Fig. 4.18: Computing the geometry factor for the eigenmode of interest.

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. 4.19

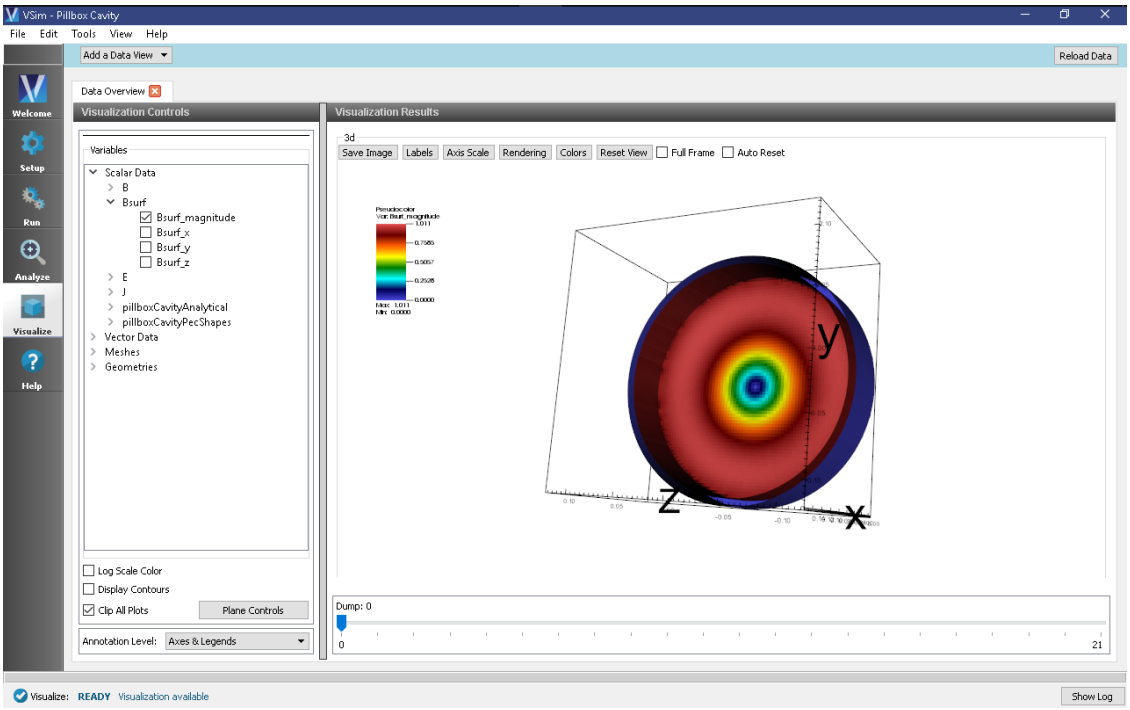


Fig. 4.19: The magnitude of the magnetic field on the wall of the cavity

To see a more quantitative visualization of the eigenmode fields, as shown in Fig. 4.20, 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.

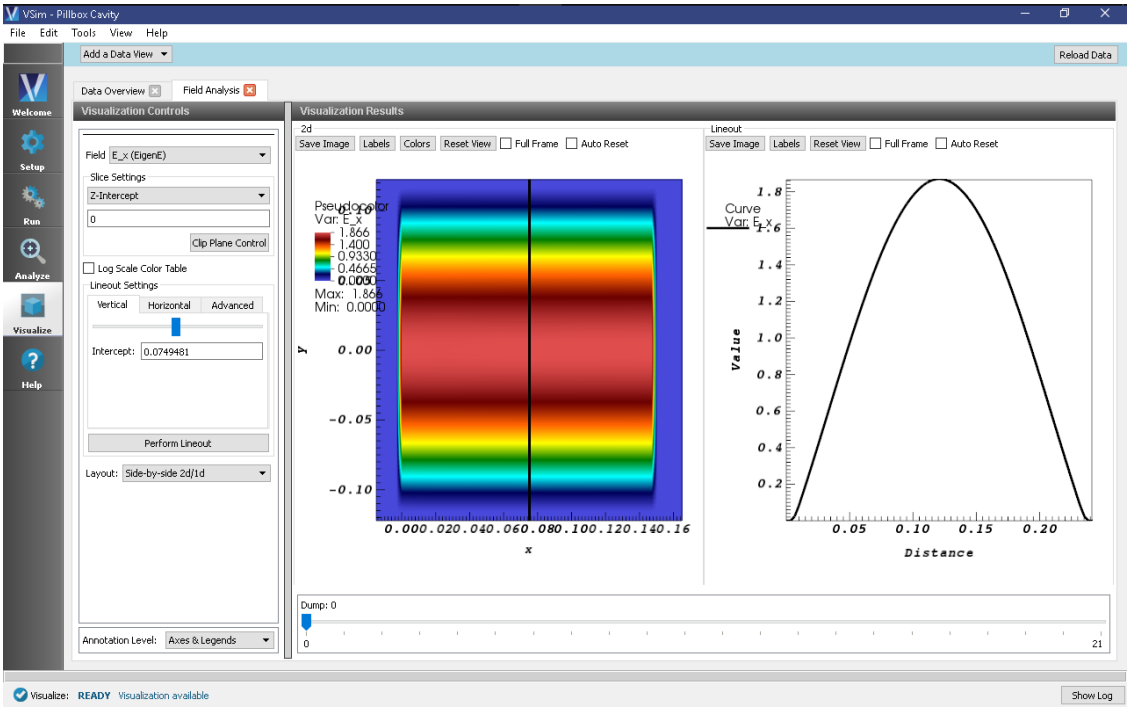


Fig. 4.20: Axial component of the electric field in the $z = 0$ plane (left) and plot of the axial electric field along $z = 0, x = 0.07495$ (right).

4.1.4 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 .

This simulation can be run with a VSimEM, VSimMD, VSimPD, or VSimPA license.

Opening the Simulation

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

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim for Microwave Devices* 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. 4.21.

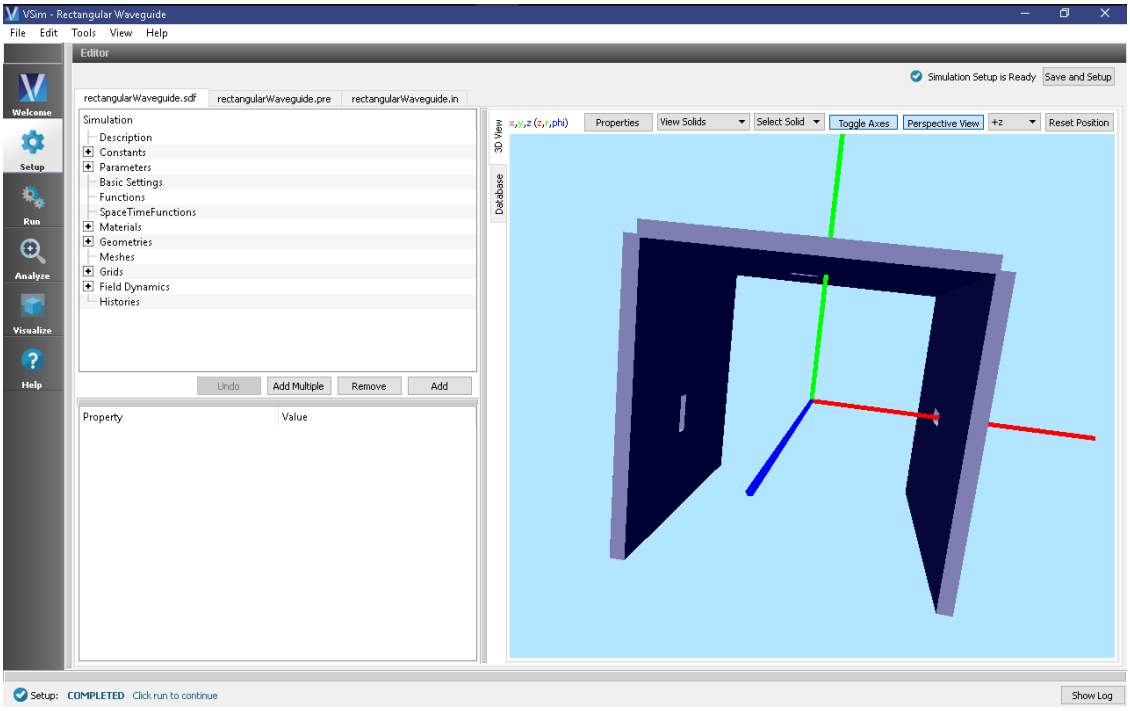


Fig. 4.21: Setup Window for the Rectangular Waveguide example.

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 extend 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

Once finished examining the problem setup, continue as follows:

- Proceed to the Run Window by pressing the *Run* button in the left column of buttons.
- Use the default values for the stepping.
- Choose parallel computing options if desired
- To run the file, 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 is shown in the window below.

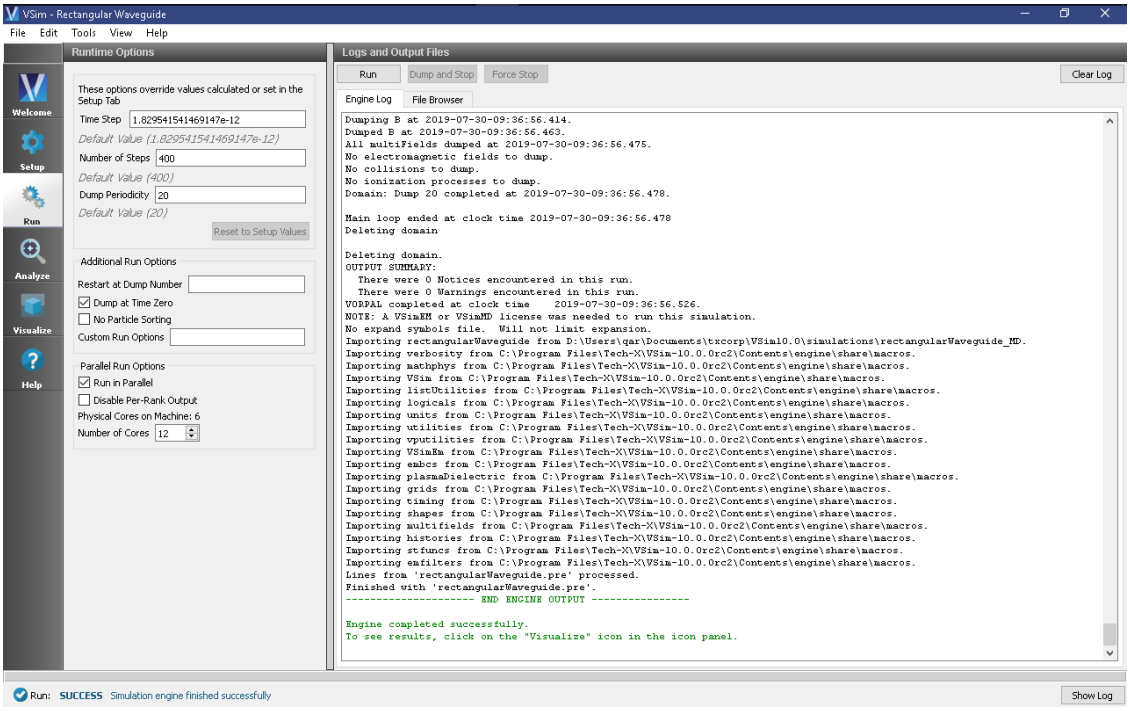


Fig. 4.22: The Run Window at the end of execution.

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, select *Clip All Plots* in the lower left hand corner. 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.

4.1.5 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. VSim 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.

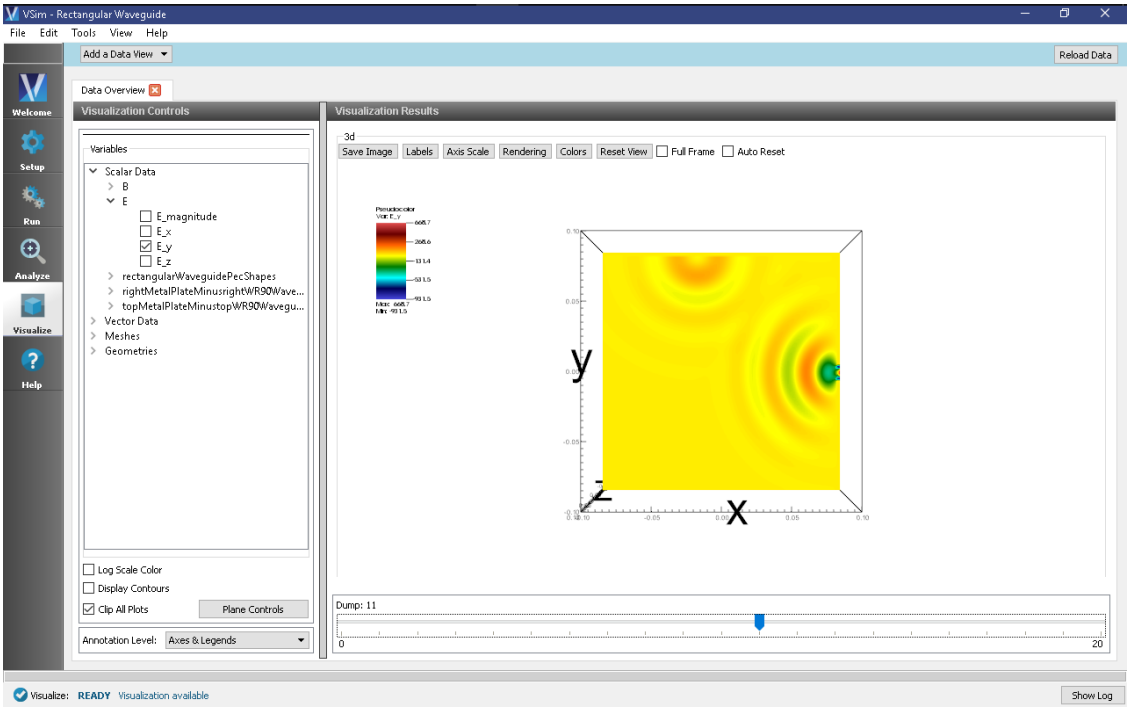


Fig. 4.23: The E_y field propagating out of the two waveguides centered on the z axis.
The effects of the third waveguide can be viewed by adjusting the "Origin of Normal Vector" parameter under the Plane Controls button.

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. 4.24.

This simulation can be performed with a VSimMD or VSimEM license.

Opening the Simulation

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

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim for Microwave Devices* 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. 4.25.

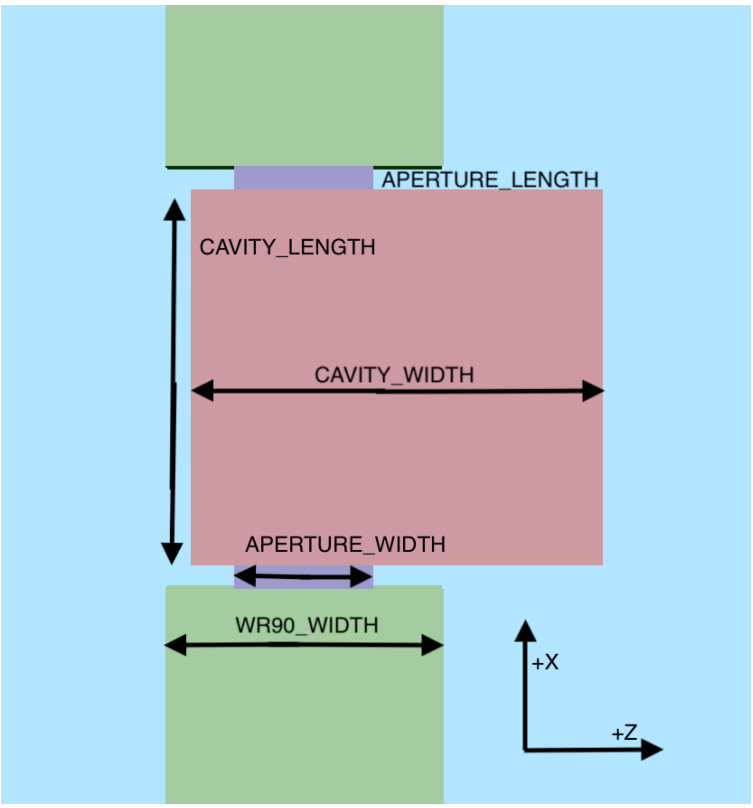


Fig. 4.24: Some relevant parameters for the S-Matrix Box Cavity example.

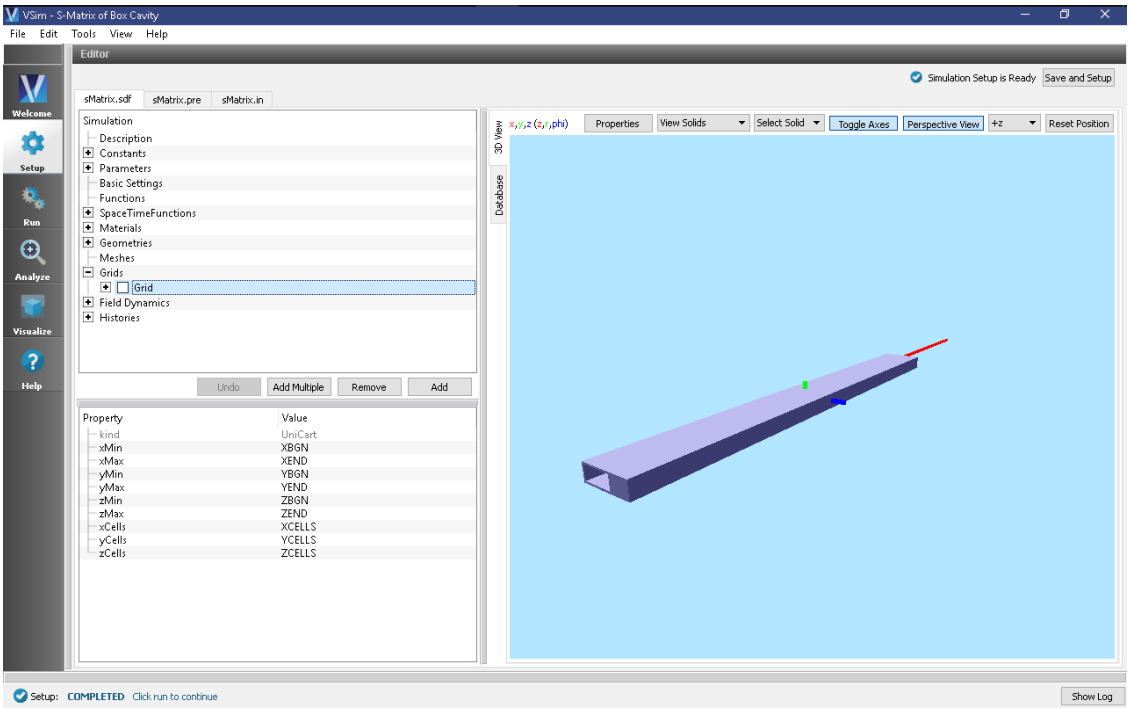


Fig. 4.25: Setup Window for the Scattering Matrix example.

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 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.

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 VSimComposer, 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.

Several space time functions are used to construct the Y and Z components of the mode's electric field pattern. 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.

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.
- To run the file, 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 is shown in the window below.

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.

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. The constant, NUMBEROFCYCLESTOCOAST,

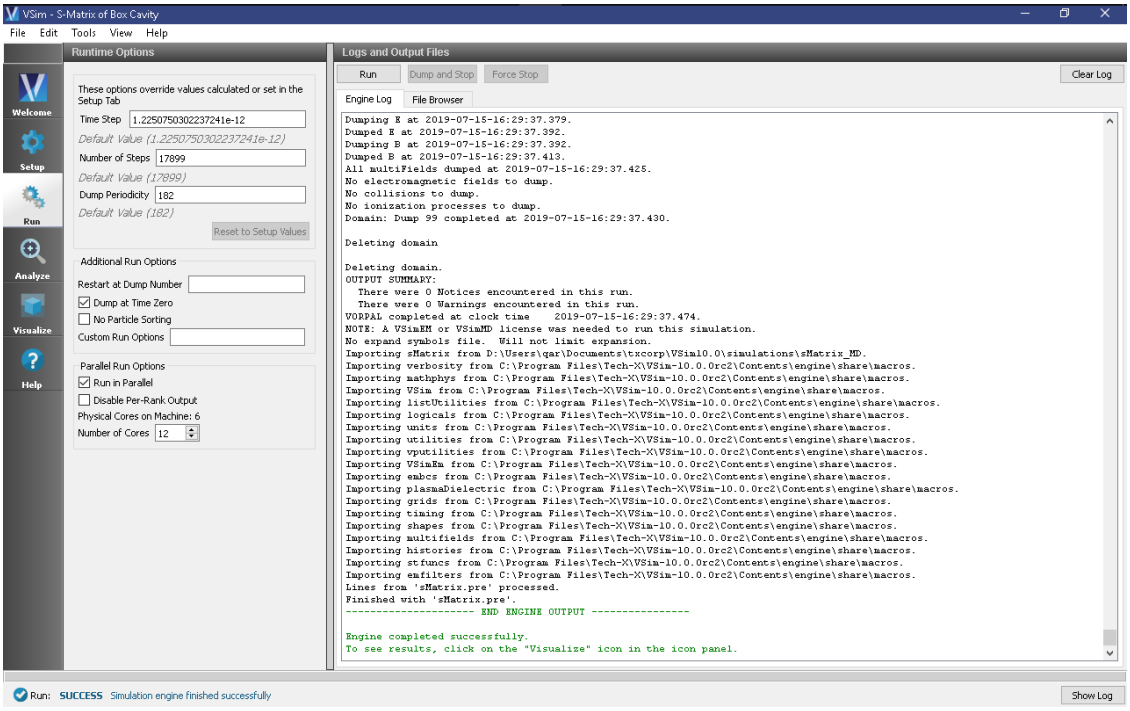


Fig. 4.26: The Run Window at the end of execution.

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 VSimComposer. 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 *metalMinuscalibrationWaveguide* 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 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 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.

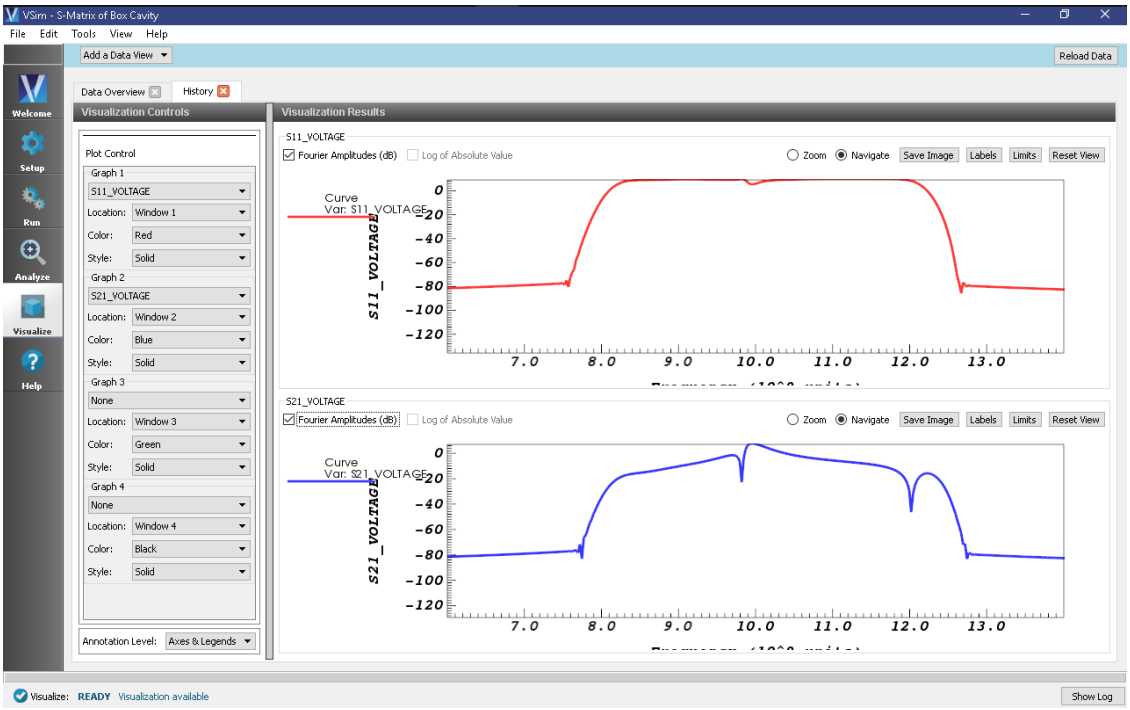


Fig. 4.27: Fourier transforms of the histories S11_Voltage and S21_Voltage as a function of frequency (in GHz).

Further Experiments

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

4.2 Cavities and Waveguides (text-based setup)

4.2.1 A15 Crab Cavity (crabCavityT.pre)

Keywords:

electromagnetic cavities, accelerators, mode frequencies

Problem Description

The Crab Cavity simulation illustrates how to extract the modes and frequencies of an accelerator cavity in a given frequency range. The range of interest here is 3.9 to 4.1 GHz. The simulation is performed by exciting the cavity with a broadly filtered pulse that excites modes in a given range. The excitation occurs through a temporally and spatially specified current source that excites the frequencies of interest. The simulation features a variable sampling frequency, allowing the cavity to first be rung up without generating excessive memory dumps. After the simulation has been rung up, sampling frequency increases, and when combined with post-processing find the modes and frequencies. The algorithm is detailed in [1].

This simulation can be performed with a VSimEM, VSimMD or VSimPD license.

Opening the Simulation

The Crab Cavity example is accessed from within VSimComposer by the following actions:

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim for Microwave Devices* option.
- Expand the *Cavities and Waveguides (text-based setup)* option.
- Select "A15 Crab Cavity (text-based setup)" 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 basic variables of this problem should now be alterable via the text boxes in the left pane of the *Setup Window*, as shown in Fig. 4.28.

Input File Features

The input file allows the user to control a number of features of the Crab Cavity simulation. The FREQ_LO_GHZ and FREQ_HI_GHZ defines the range of frequencies that we are interested in extracting whereas the DELTA_FREQ_GHZ specifies the separation in frequency between the range of interest and the next nearest mode (at 4.3 GHz).

The input file is written to run for a long time, sufficient to ring up the cavity, then dump periodically during the free oscillation period. The modes and frequencies will be extracted from those dumps. This can be seen in crabCavityT.in.

The remaining key parameter values correspond to the geometry and discretization of the cells. The focus of the Crab Cavity simulation is on a four cell cavity with end holes that were originally used for measurement purposes. See [2]. The final ZSQUASH parameter is used to squeeze each cell of the cavity to eliminate the degeneracy due to cylindrical symmetry.

Running the Simulation

After examining the inputs, do the following:

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

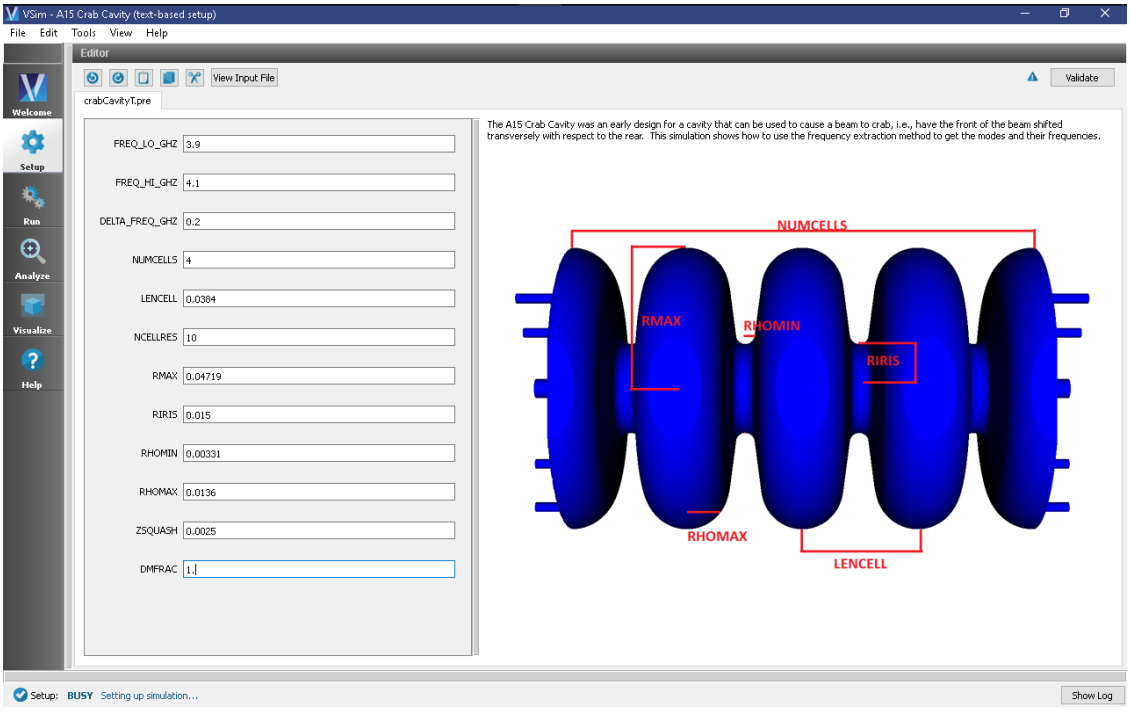


Fig. 4.28: Setup Window for the Crab Cavity example.

- Select running in parallel as desired.
- To run the file, click on the *Run* button in the upper left corner of the Logs and Output Files pane on the right. You will see the output of the run in this pane. The run has completed when you see the output, "Engine completed successfully." This is shown in Fig. 4.29.

Analyzing the Results

It is possible to extract the modes of the A15 crab cavity via post processing using the Extract Modes Analysis Script* as follows:

- Press the Analyze button in the left column of buttons.
- From the list of Available Analyzers, select *extractModes.py* and press *Open*.
- Enter the following parameters in the appropriate fields. the default simulation values are used:
  - simulationName = crabCavityT
  - field = B
  - beginDump = 2
  - endDump = 22
  - nModes = 5
  - sampleType = 0
  - numberUniformPts = 20
  - numberRandomPts = 20
  - construct = 1

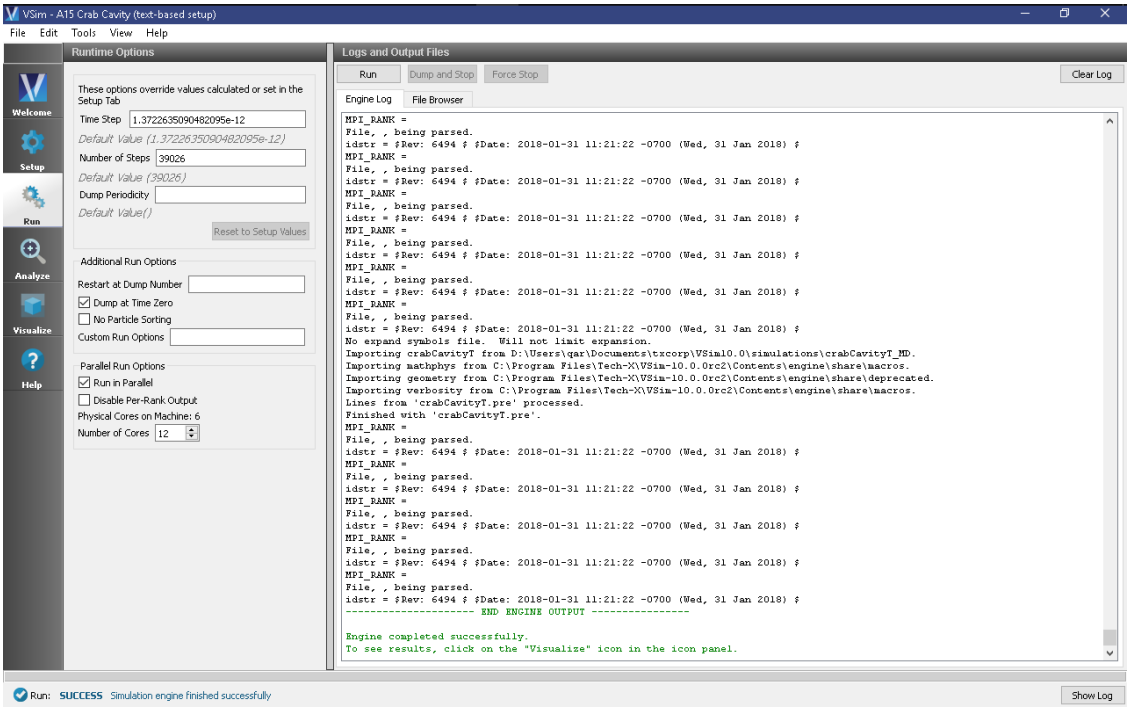


Fig. 4.29: The Run Window at the end of execution.

- Click the *Analyze* button in the upper right corner of the window.

As shown in Fig. 4.30 below, three columns of data with the titles “freq [Hz]” (eigenmode frequency), “invQ” (inverse quality factor), and “SVD” (singular value decomposition) will be output in the right pane. The analysis has completed when you see the output “Analysis completed successfully.” One can see 5 modes, but the first one is not real as one can see from its unrealistic value of invQ, which should in fact be zero for this ideal (non-lossy) cavity, and the fact that it has zero frequency.

The magnetic fields at each of the eigenmode frequencies will be available to view in the Visualize Window under the *B* Field.

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:

- Expand *Scalar Data*
- Expand *E*
- Select *E_z*
- Expand *Geometries*
- Select *poly*
- Check *Clip All Plots*
- Move the slider at the bottom of the right pane to see the electric field at different times.

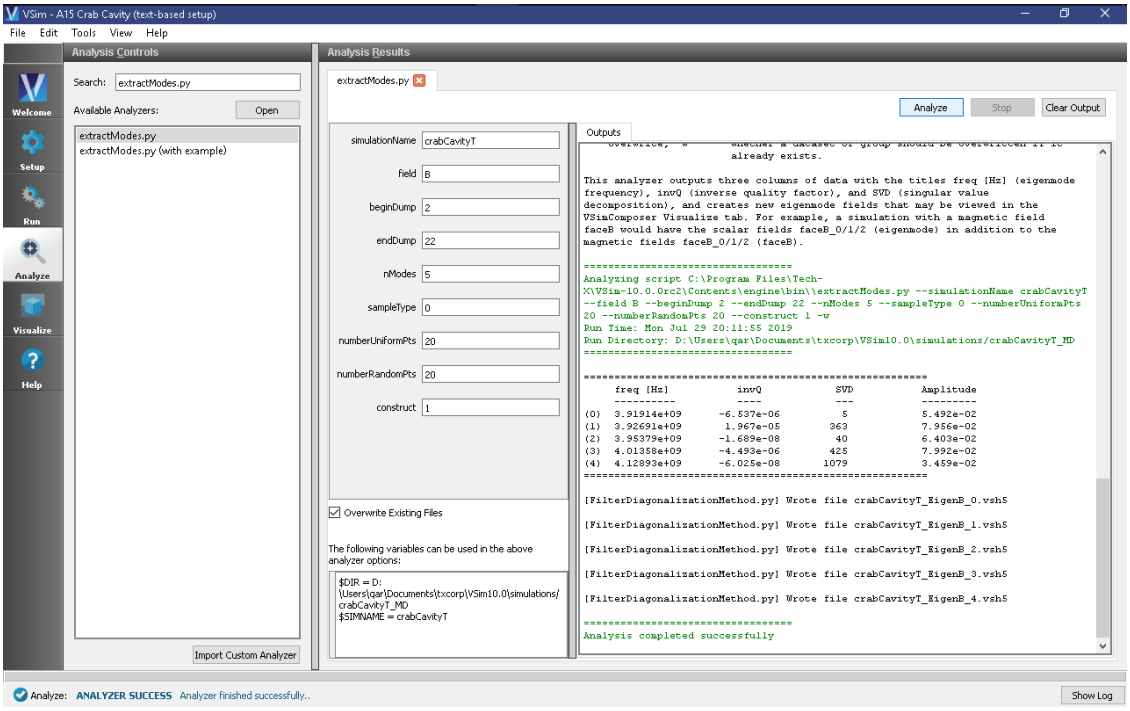


Fig. 4.30: The Analysis window at the end of execution of the extractModes.py script.

One can instead view the eigenmodes, which are so labeled under *B*. E.g., Unclick *E_z* click *B_y (Eigenmode)*. For these plots you may want to select the *Colors* control, set *Fix Maximum* and *Fix Minimum* and change the values to +-0.005 for a sharper contrast. The slider can be moved to see the eigenmodes. Slider positions past the last eigenmode will display only the last eigenmode.

Further Experiments

Additional experiments worth investigating are:

- Use Histories to record the power flow, to compute the coupling efficiency.
- Simulate one period of the waveguide with periodic boundary conditions and a user-defined phase shift, and use the frequency extraction feature to compute the waveguide modes and dispersion curves.

References

[1] G. R. Werner and J.R. Cary, "Extracting modes and frequencies from time-domain simulations with filter-diagonalization", J. Comp. Phys., 227 (10), 5200-5214, 2008.

[2] T. M. Austin et al., "Validation of frequency extraction calculations from time-domain simulations of accelerator cavities", Comput. Sci. Disc., 4, 015004, 2011.

4.2.2 Stairstep Cavity in coordinateGrid (emCavityCoordProdT.pre)

Keywords:

`stairstep boundary, coordinateGrid, Klystron cavity`

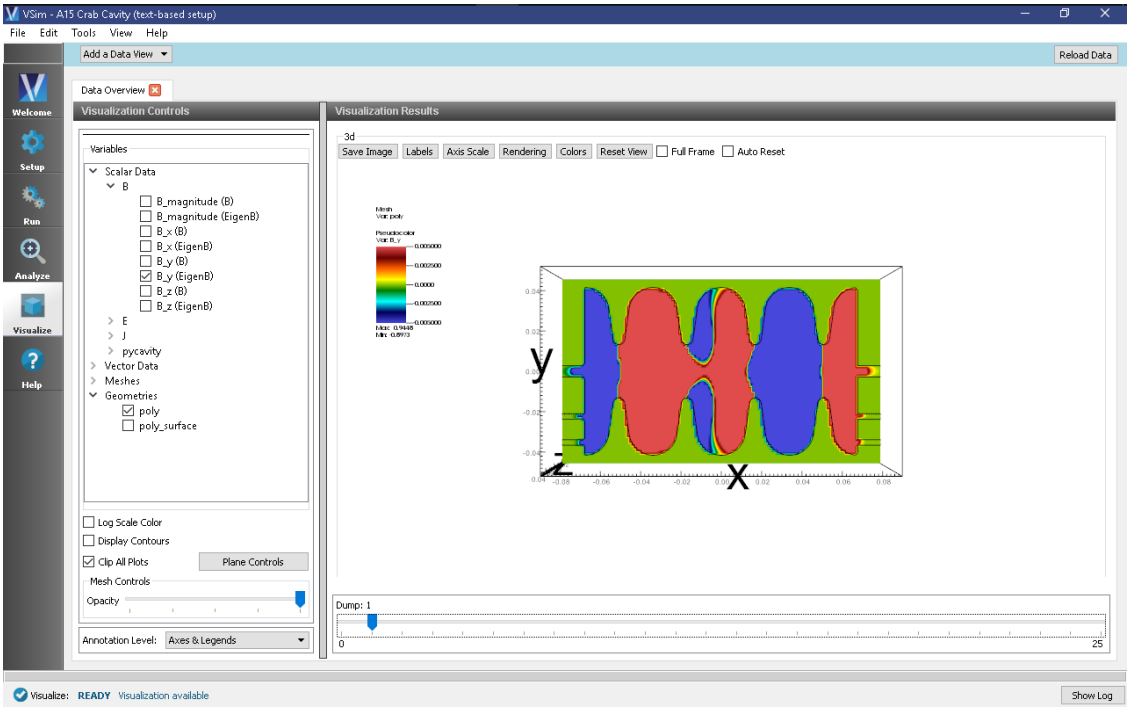


Fig. 4.31: Visualization showing an eigenmode

Problem description

This example demonstrates how to set up a complex geometry structure in VSim that uses the *coordinateGrid* system for a varying mesh size. There are two benefits of constructing a grid of kind `coordProdGrid` via `coordinateGrid` blocks. The first is a flexible choice of either Cartesian (x, y, z) or cylindrical (z, r, ϕ) coordinate systems to construct the grid. The second is that it enables one to vary the cell size along each axis of the grid. For example, a fine grid resolution can be used in a region of the domain consisting of complicated geometry, while a coarse resolution can be used in a different region of the domain where the geometry is simple. This method reduces the memory requirement for large multiscale simulations. The `gridBoundary` block (implemented in this example through the `geometry` macro) is used by VSim to represent complex geometrical surfaces with boundary conditions.

This example simulates a klystron cavity using a non-uniform Cartesian mesh generated by VSim's *coordinateGrid* system. Klystron cavities have wide applications as RF power sources by amplifying an RF input with electron beams. The simulated cavity is defined by a set of VSim geometry macros. Grid cell size is varied in the longitudinal direction so that a fine mesh exists at the round nose surface connecting the center drifting tube and outer ring cavity. Larger cell sizes are used at both ends of the drifting tube. The fundamental transverse magnetic (TM) mode is excited by a Gaussian current pulse.

This simulation can be performed with either a VSimEM, VSimMD, or VSimPD license.

Opening the Simulation

The emCavityCoordProdT example is accessed from within VSimComposer by the following actions:

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim for Microwave Devices* option.
- Expand the *Cavities and Waveguides (text-based setup)* option.

- Select “Stairstep Cavity in Coordinate Grid (text-based setup)” 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 basic variables of this problem should now be alterable via the text boxes in the left pane of the Setup Window, as shown in Fig. 4.32.

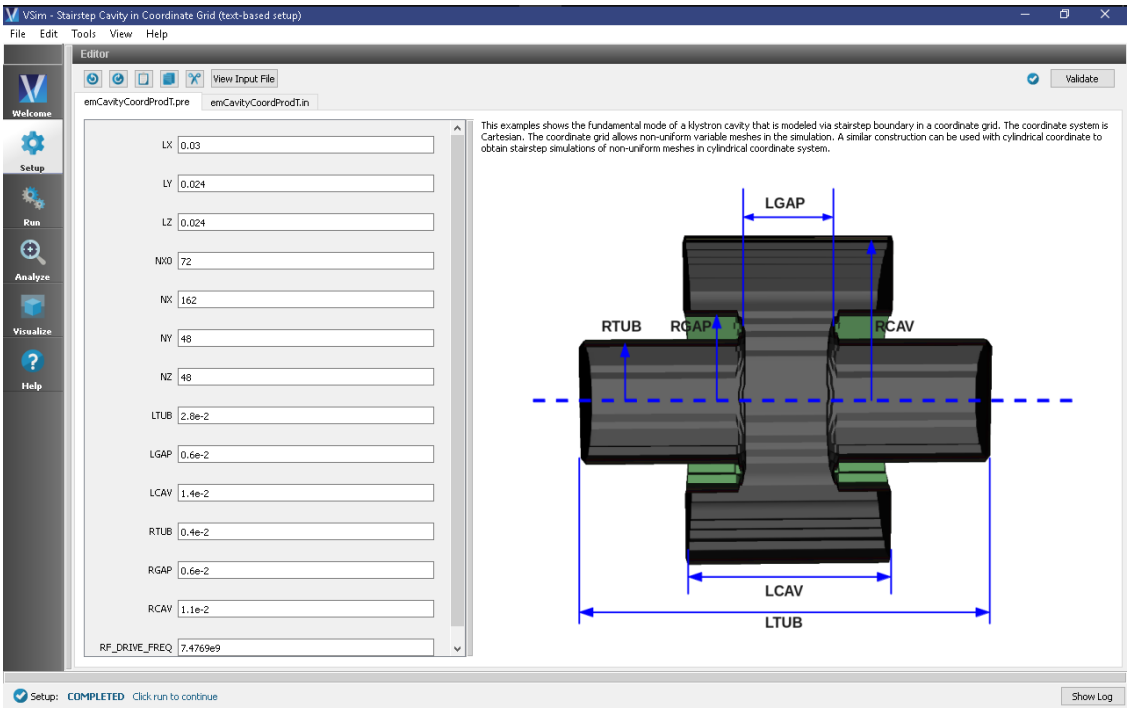


Fig. 4.32: Setup Window for the emCavityCoordProdT example.

Input File Features

- At the top of the Editor pane, click *View Input File*.

The first important feature of this input file is the setup of the non-uniform simulation grid. Scroll to the `Grid` block on line `262` (Fig. 4.33). The variable spacing in x in the non-uniform grid is specified in the definitions of `sectionBreaks` and `deltaAtBreaks` in the `coordinateGrid dir0` block. The `deltaAtBreaks` field specifies the grid cell spacing at each of the `sectionBreaks` positions, and the grid is generated such that the cell spacing transitions gradually between these positions. In this specific example, on lines `266` and `267`, from $x =$ `XBGN` to $x =$ `CAV_START` the cell spacing transitions from $\Delta x =$ `DX` to $\Delta x =$ `DX/3.0`, then from $x =$ `CAV_START` to $x =$ `CAV_END` the cell spacing stays at a constant value of $\Delta x =$ `DX/3.0`, and finally from $x =$ `CAV_END` to $x =$ `XEND` the cell spacing transitions from $\Delta x =$ `DX/3.0` back to $\Delta x =$ `DX`.

The second important feature of this input file is the electromagnetic solver for this type of grid. Both the Faraday and Ampere updaters are set to kind `curlUpdaterCoordProd`. By setting `interiorness = cellcenter` in both updaters, the curl operation is performed with a stair-stepped `gridBoundary`.

Running the simulation

Because the cells are not uniformly spaced, the number of cells in the simulation is unknown until calculated by VSim’s Vorpal engine. However, the number of cells in each dimension is required for VSim to preprocess the input file. To correctly set the number of cells `NX` in the input file, take the following steps:

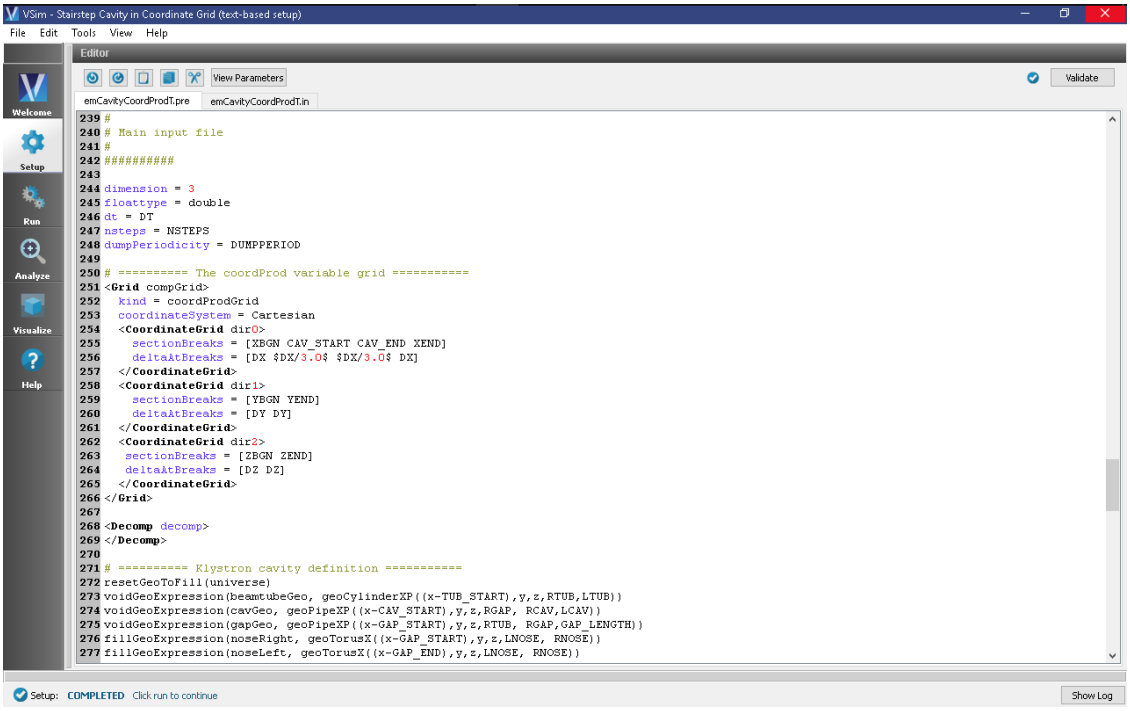


Fig. 4.33: Input file for the emCavityCoordProdT example, showing the setup of the *coordinateGrid* system.

- Set the parameter `NX0` (the default is 72). This specifies the grid spacing (`DX`) at the ends of the simulation domain as `NX0/LX`.
- Run the simulation for one time-step by clicking the **Run** button in the left column of buttons, and entering "1" in both the *Number of Steps* and *Dump Periodicity* fields.
- After the simulation completes, scroll through the log file to find the value of `numPhys` in the first row of `Global grid`, as circled in Fig. 4.34.
- Go back to the Setup Window by clicking **Setup** in the left column of buttons, and enter this value into the field for `NX`. For the default values of this example, this number should be 162.

The simulation is now ready to run. Return to the Run Window, enter the desired values for *Number of Steps* and *Dump Periodicity* and click *Run* once again. The run has completed when you see the output, "Engine completed successfully" as shown in Fig. 4.35. This will require approximately two hours of computation time when run in parallel on four processors on a modern CPU.

Visualizing the results

After performing the above actions, continue as follows:

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

To create the plot as shown in Fig. 4.36:

- In the Variables section of the Visualization Controls pane, Expand *Geometries*
- Select *poly_surface*
- Expand *Meshes*
- Expand *compGridGlobal*

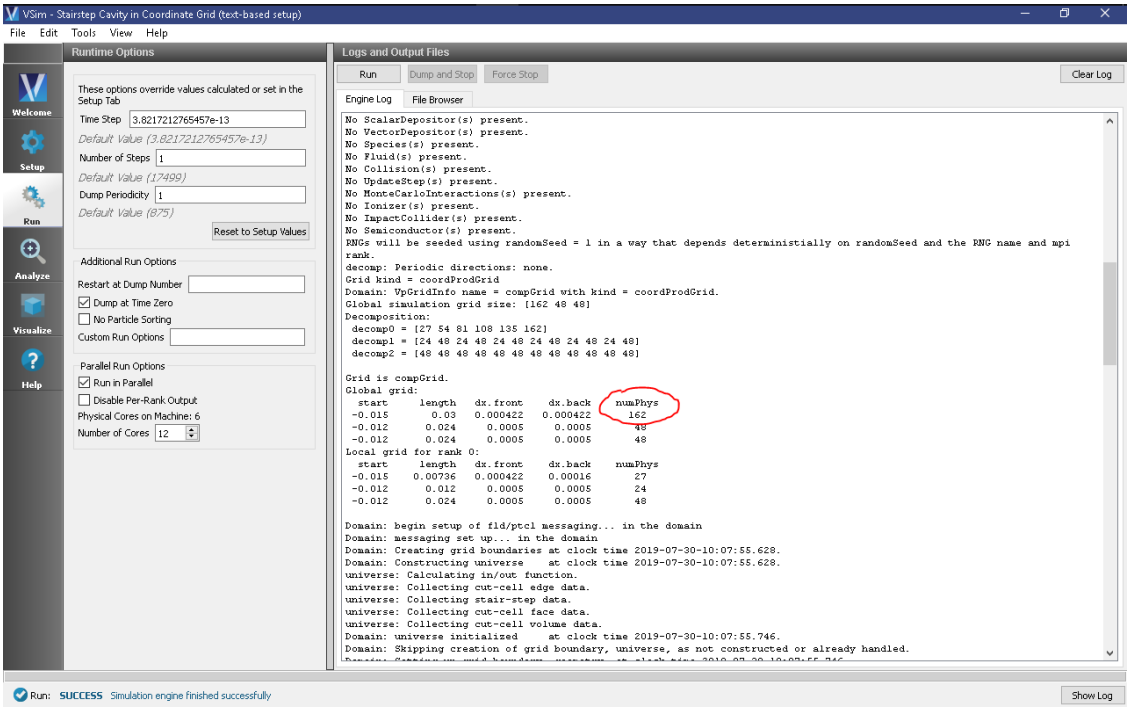


Fig. 4.34: Location of the `numPhys` output to be entered into the `NX` field by the user.

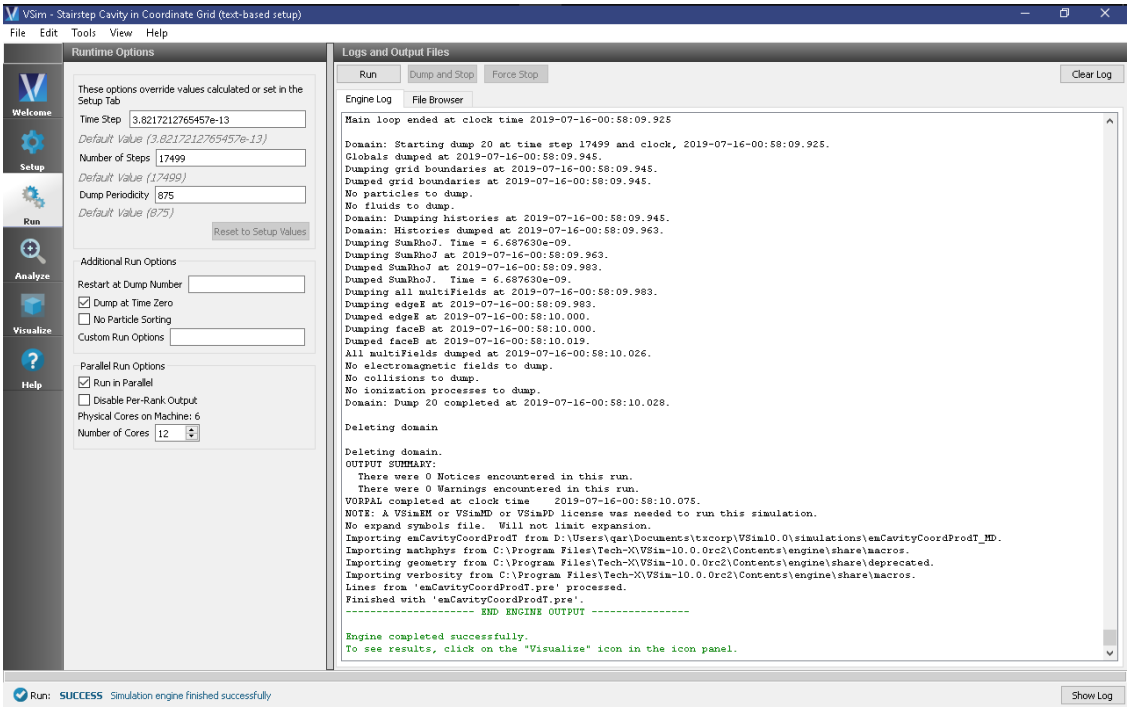


Fig. 4.35: The Run Window during the execution.

- Select any of the grid choices, for they are all identical for this simulation
- Expand *Scalar Data*
- Expand *edgeE*
- Select *edgeE_x*
- Near the bottom of the Visualization Controls pane, select *Clip All Plots*
- Click the *Plane Controls* button and set the normal in the Z-direction (which should already selected by default)
- Select *Display Contours*
- Decrease the Opacity by moving the Opacity slider the left to better see the fields through the grid
- At the bottom of the Visualization Results pane, move dump slider forward in time

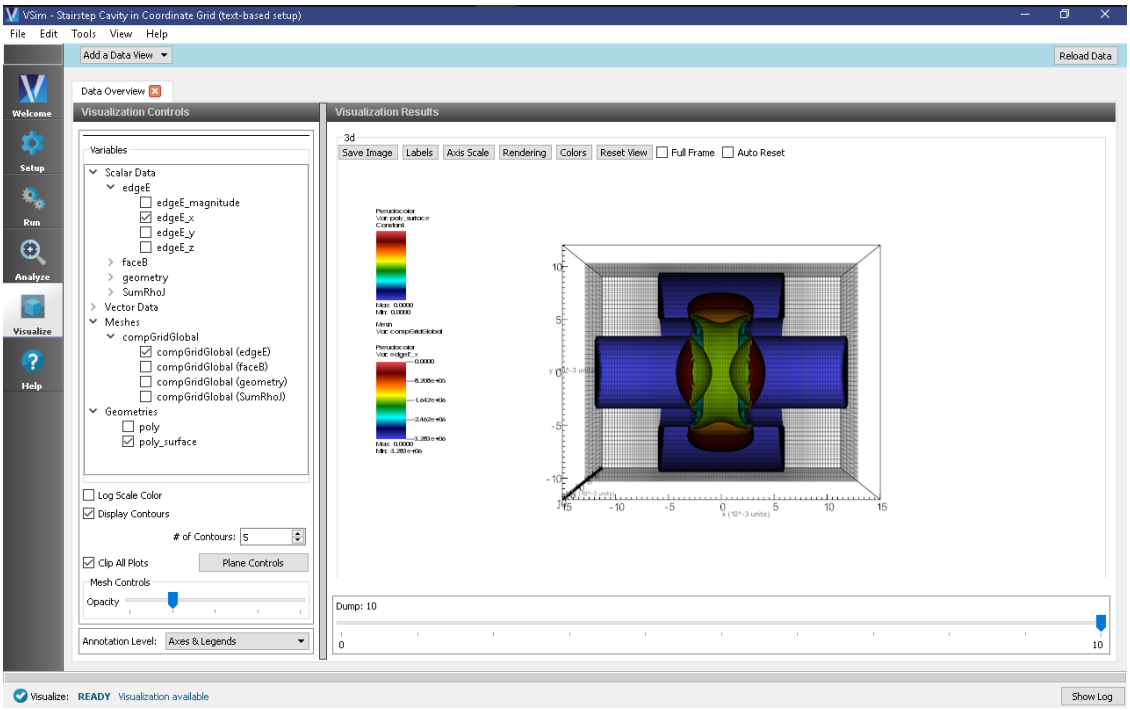


Fig. 4.36: Visualization of the x-component of the electric field in the simulation of a klystron cavity with a non-uniform mesh.

Further Experiments

The *coordinateGrid* system is also capable of creating non-uniform grids in cylindrical coordinates, by setting `coordinateSystem = Cylindrical` in the `Grid` block. The `curlUpdaterCoordProd` updater also works with the same settings in cylindrical coordinates.

4.3 Radiation Generation

4.3.1 Smith Purcell Radiation (SmithPurcellRadiation.sdf)

Keywords:

diffraction grating, radiation generation, coherent mode, Smith-Purcell

> **Warning:** This example used more than 8GB of RAM as configured, so if your machine does not have a lot of memory, then you can reduce the number of cells in the Grid (say by a factor of 2 to the size 950x800x5) to avoid running out of memory.

Problem Description

This VSim for Microwave Devices example illustrates how to setup a device that emits coherent Smith-Purcell Radiation (SPR). This phenomenon occurs when charged particles pass over a periodically graded surface in very close proximity resulting in the emission of a form of Cherenkov radiation. In recent years, engineers have been building SPR-emitting devices that can generate frequencies in the terahertz range, otherwise difficult to obtain via other methods. This paper presents how the design of an SPR-emitting device can be optimized with simulations.

This simulation can be run with a VSimMD license.

Opening the Simulation

The SmithPurcellRadiation example is accessed from within VSimComposer by the following actions:

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim for Microwave Devices* option.
- Expand the *Radiation Generation* option.
- Select *Smith Purcell Radiation* 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. 4.37.

Simulation Properties

The simulation setup was based on Donohue & Gardelle 2005 *[DG05]* who used 2D simulations for their study: a grating structure that is perfectly conducting, a cathode from which the electron beam is emitted, and a vacuum enclosure in which radiation propagates. The walls of the vacuum box are matched absorbing layers (MALs) which absorb the electromagnetic fields and eliminate any reflection. This is a quasi-3D simulation: the thickness to the gating structure of 5 mm and the simulation is periodic in z (the direction normal to the grating). The grid resolution was set high enough to resolve the small structures of the grating: 1890, 1600, and 5 cells in the x, y, and z directions, respectively. The 5 mm electron beam was generated with a 125 A/m current. The incident electron energy is 100 keV. There is an external magnetic field of 2 T in the x-direction for beam confinement.

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." This is shown in Fig. 4.38.

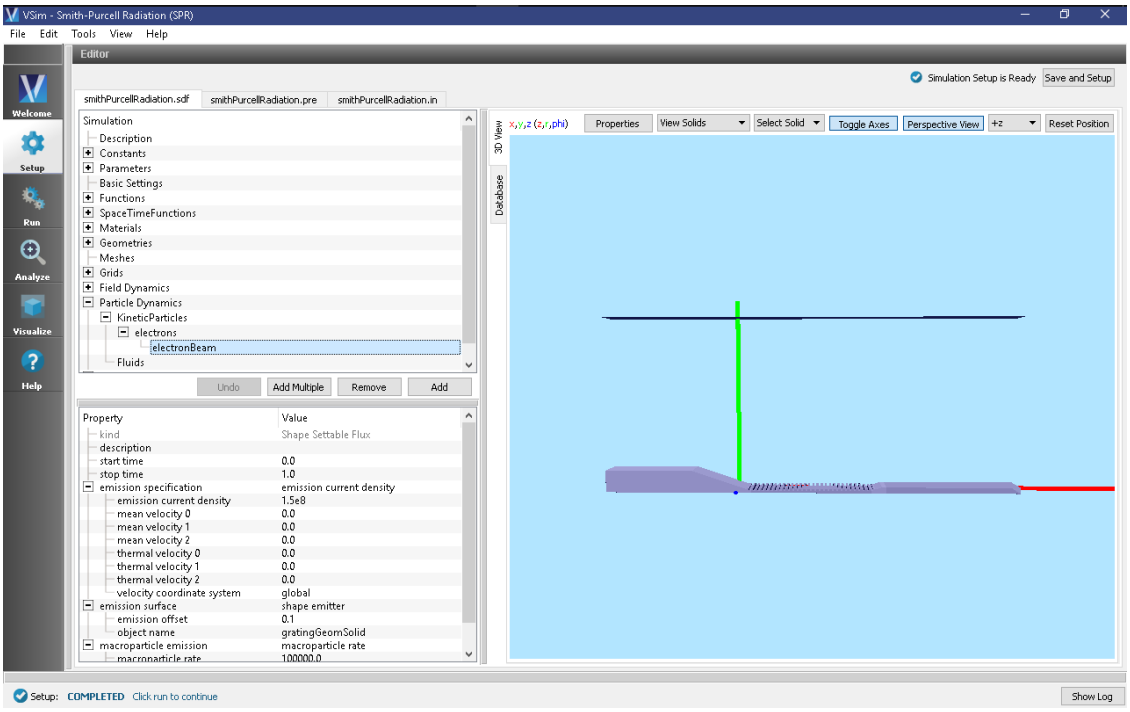


Fig. 4.37: Setup Window for the Smith-Purcell Radiation example.

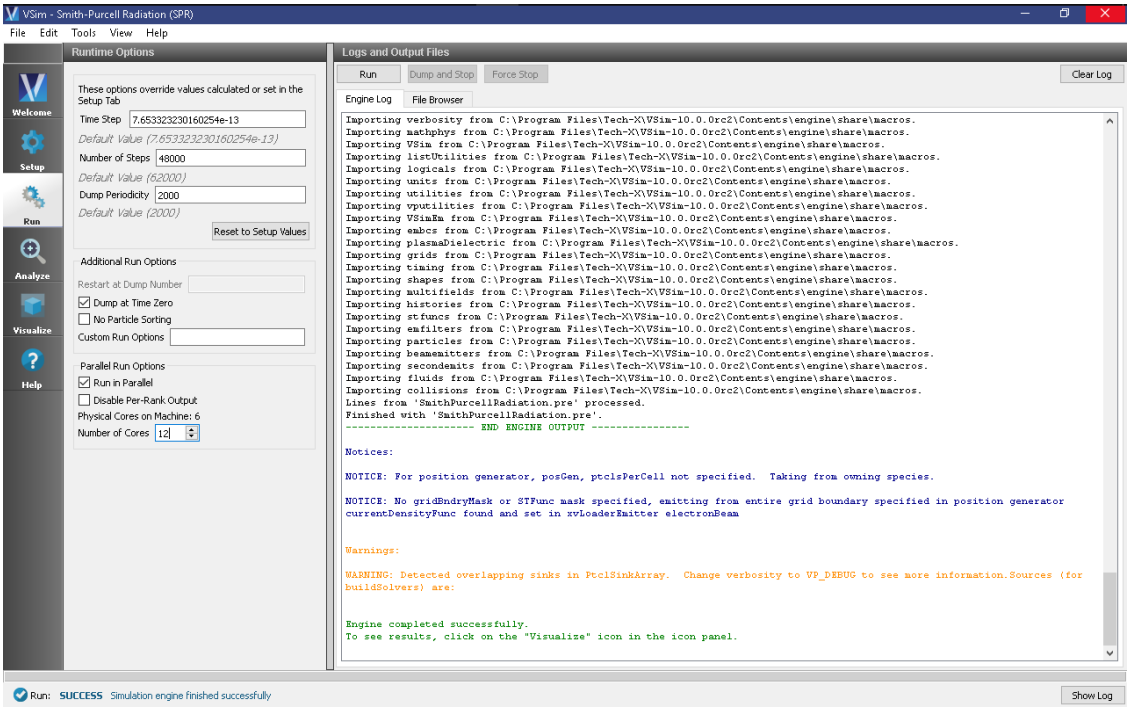


Fig. 4.38: The Run Window at the end of execution.

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.
- From the *Data View* dropdown menu, select *Data Overview*.
- In the variables tree, expand *Scalar Data*.
- Expand *B*.
- Select *B_z*.
- Check Fix Minimum and Fix Maximum and set to -1e-5 and 1e-5, respectively
- In the bottom of the right pane, mode the dump slide forward in time.
- The resulting visualization is shown in Fig. 4.39.

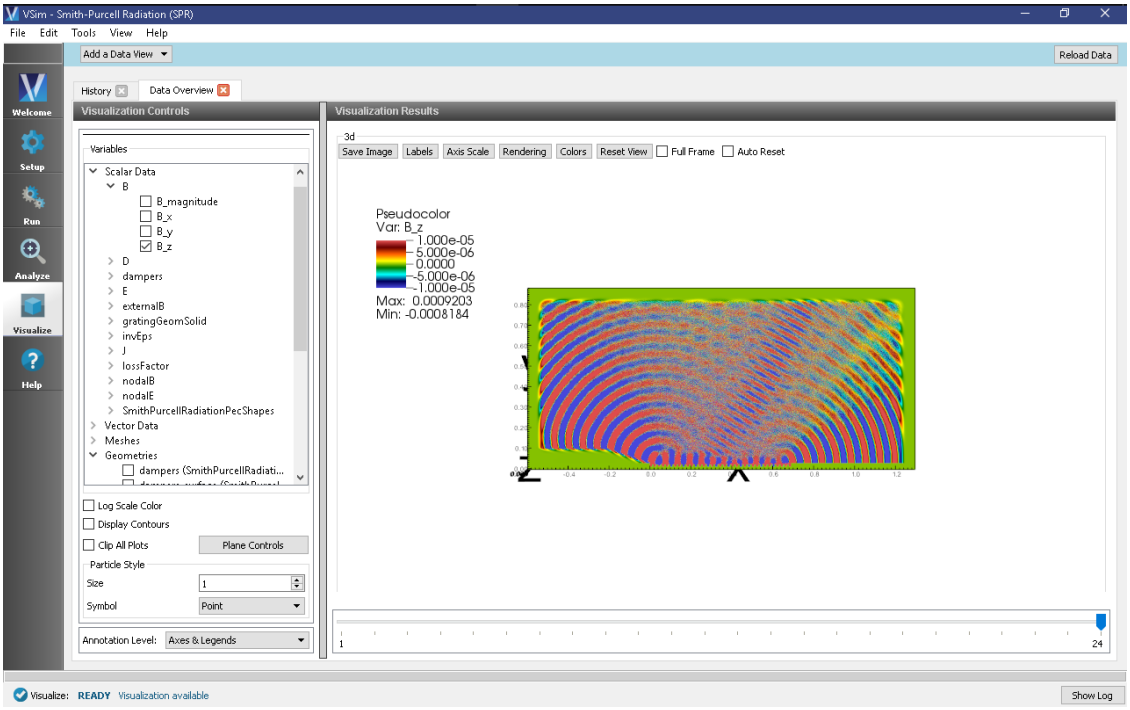


Fig. 4.39: Bz at the end at 25 ns (at dump 24).

The magnetic field in the z direction (Bz) plotted in Fig. 4.39 shows that at 85 cm and 64 degrees from the center of the grading the SPR emission was the strongest (this is known as the SPR propagating mode). This is consisted with the results found by Donohoe and Gardelle. The strong emission seen on the left side is known as the evanescent mode, but this is considered non-SPR emission.

To measure the frequencies of these modes proceed as follows:

- Proceed to the Visualize Window by pressing the *Visualize* button in the navigation column.
- From the *Data View* dropdown menu, select *History*.
- Select *Bz64deg_2* from the drop-down menu in Graph 1.
- Select *None* for Graphs 2,3, and 4.
- In the top left corner of the right pane, check the *Fourier Amplitude (dB)* option.

- In the upper right corner of each plot, select Limits and set X-Axis max to 1.2e10 and click *OK*.
- The resulting visualization is shown in Fig. 4.40.

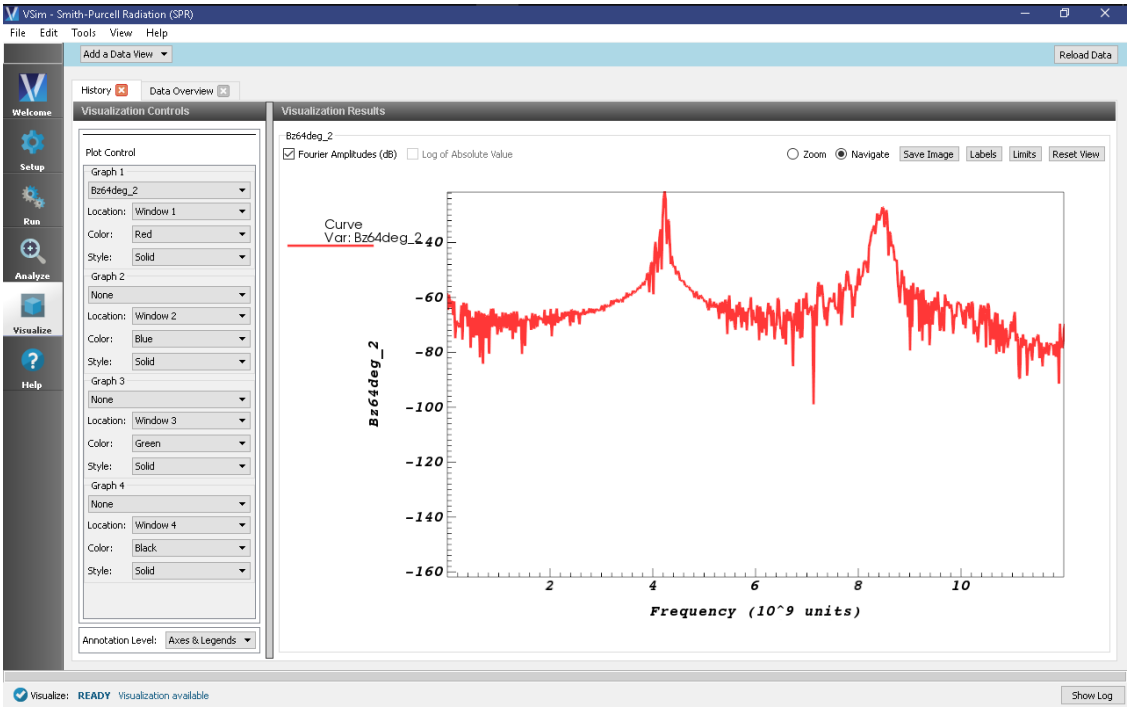


Fig. 4.40: FFT of the Bz measured at 85 cm and 64 degrees from the center of the grading where the SPR emission was the strongest. The frequencies of the evanescent and the propagating modes can be seen at around 4.5 and 9 GHz.

The FFT of this signal is shown in Fig. 4.40 where the frequencies corresponding to the evanescent mode and the propagating mode can be seen at around 4.5 and 9 GHz. The evanescent mode becomes dominant over the propagating mode when the dampers are not present. Note: it is expected for the propagating mode frequency to be an integer multiple of the evanescent mode (see Donohue and Gardelle 2005 *[DG05]* for more details).

Another signature of SPR emission is electron bunching inside the particle beam. To visualize the electron bunching, proceed as follows:

- Proceed to the Visualize Window by pressing the *Visualize* button in the navigation column.
- From the *Data View* dropdown menu, select *Phase Space*.
- For the *X-axis*, select *electrons_x*.
- For the *Y-axis*, select *electrons_ux*.
- Click *Draw*.
- Move the dump slider further in time.
- The resulting visualization is shown in Fig. 4.41.

Fig. 4.41 shows a phase-space of the electron speed in the propagation direction vs. their position. The very strong bunching effect can be observed and this effect becomes more defined and increases with time.

Further Experiments

When running the simulation without the dampers, the evanescent mode is dominant and strong fields can be seen at the beginning and end of the electron beam. Adding wedge-like dielectric structures can help damp the strong

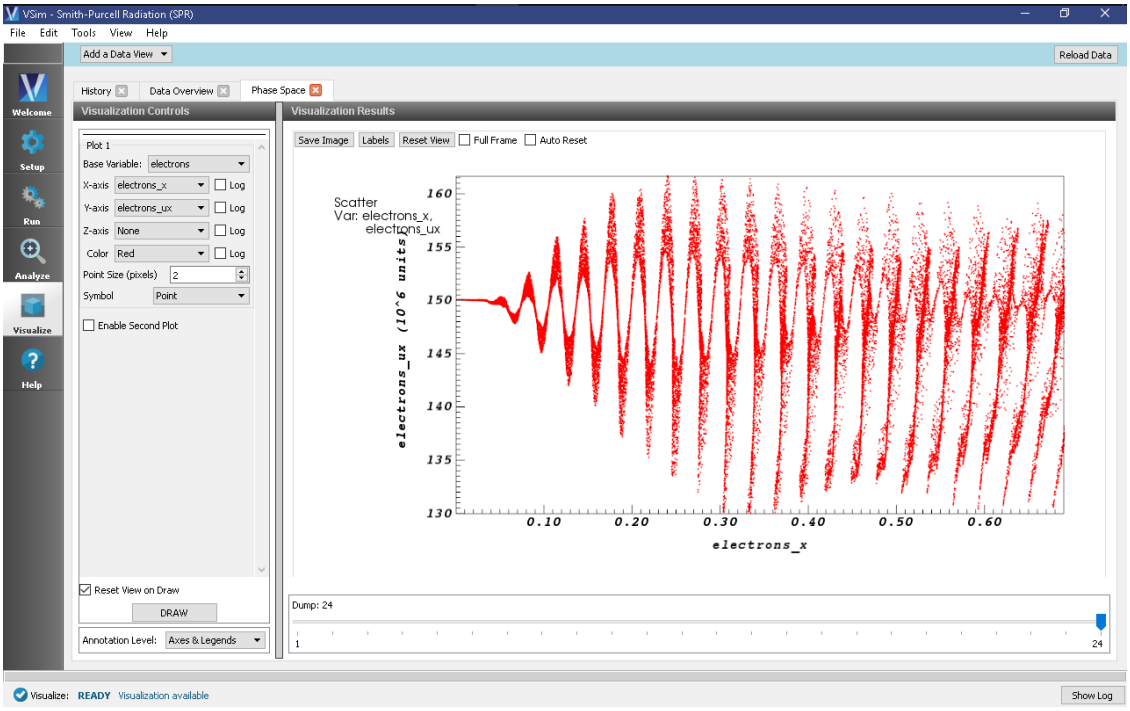


Fig. 4.41: The phase-space visualization for the Smith Purcell Radiation example.

non-SPR beam. Further extension of the cathode damper and adjusting the dielectric constant makes the SPR beam became dominant as show in this simulation. The dielectric dampers are critical in obtaining a strong SPR emission.

Using this basic setup, one can develop a simulation for special SPR emission which is generally obtained by narrowing the grooves inside the grating.

4.3.2 A6 Magnetron 1: Modes (a6Magnetron1Modes.sdf)

Keywords:

`magnetron, cavity modes, A6`

Problem Description

This VSimMD example simulates MIT's cylindrical A6 magnetron cavity with no outlets in three dimensions. The structure is generated using shape primitives within the VSim composer. The cavity is excited by a sinc pulse ping using a distributed current source within one of the resonant cavities. The spectrum of the cavity is used to find the modes, and FDM is used to extract the exact mode profile of the cold cavity.

This simulation can be performed with a VSimMD license.

Opening the Simulation

The A6 Magnetron example is accessed from within VSimComposer by the following actions:

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim for Microwave Devices* option.

- Expand the *Radiation Generation* option.
- Select "A6 Magnetron 1:Modes" 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. 4.42. 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`.

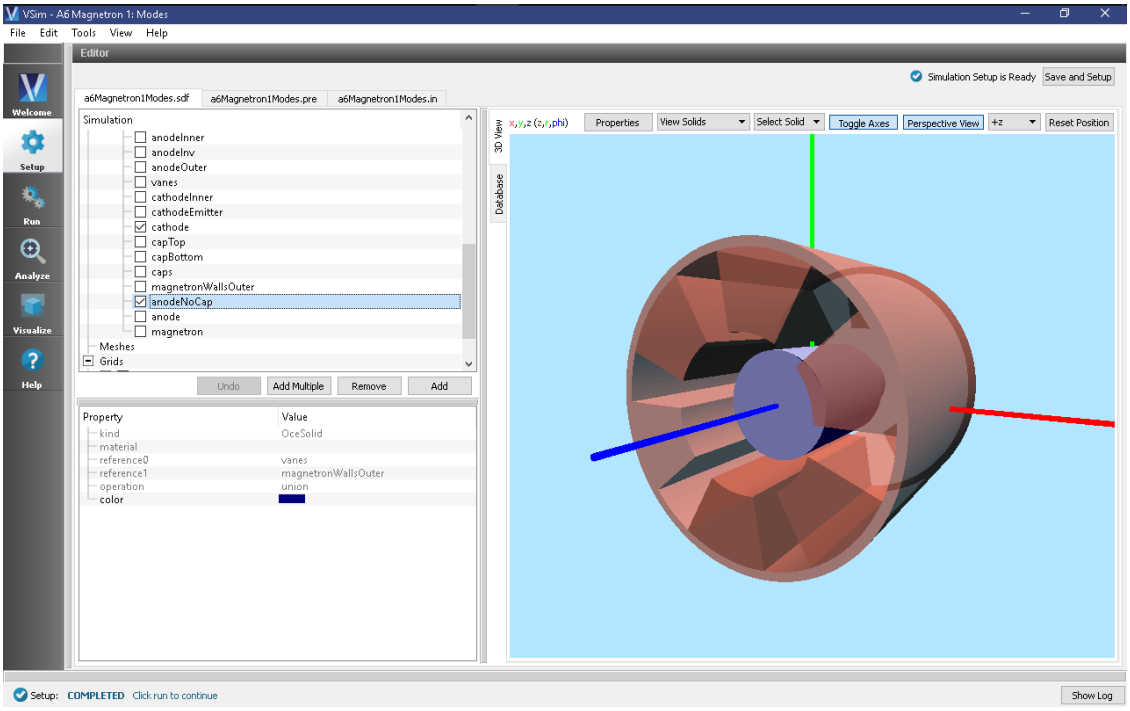


Fig. 4.42: Setup window for the A6 Magnetron example showing cathode and anode vanes only.

Simulation Properties

The A6 Magnetron example includes several constants for easy adjustment of simulation properties. User changeable parameters include:

- **RADIUS_ANODE** → Inner radius of anode.
- **RADIUS_ANODE_OUTER** → Outer radius of anode.
- **ANGLE_CAVITY** → Angle of resonant cavity openings, in degrees.
- **THICKNESS_WALL_OUTER** → Thickness of all walls.
- **WIDTH_VANES** → Total width of anode vanes in z-direction.
- **RADIUS_CATHODE** → Radius of the emitting section of the cathode.
- **RADIUS_CATHODE_INNER** → Radius of the inner section of the cathode.
- **WIDTH_CATHODE** → Width (in z-direction) of the emitting section of the cathode.
- **FREQ_LOW** → Lower frequency of excitation source range.

- **FREQ_HIGH** → Upper frequency of excitation source range.
- **(X,Y,Z)POS_CURR** → Position of excitation distributed current source.
- **(XY,Z)SIZE_CURR** → Size of excitation distributed current source region.
- **(X,Y,Z)POS_HIST** → Position of electric field history.

The axis of the cavity coincides with the z-axis and the center is at z = 0. The emitting cathode region is 4.0 cm long. All surfaces are perfect electric conductors. Histories of the electric and magnetic fields are taken at the inside of one of the cavities to find the modes. The FFT of the history shows the mode frequencies, and the exact value and profile is found using *extractModes.py - Extract Modes Analysis Scripts*.

The excitation frequency range can be set using the constants FREQ_LOW and FREQ_HIGH. The total excitation time is calculated in TIME_EXCITE. The simulation should be run for longer than TIME_EXCITE to allow the excitation source to complete.

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.
- To run the file, click on the *Run* button in the upper left corner of the right panel. 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 is shown in Fig. 4.43.

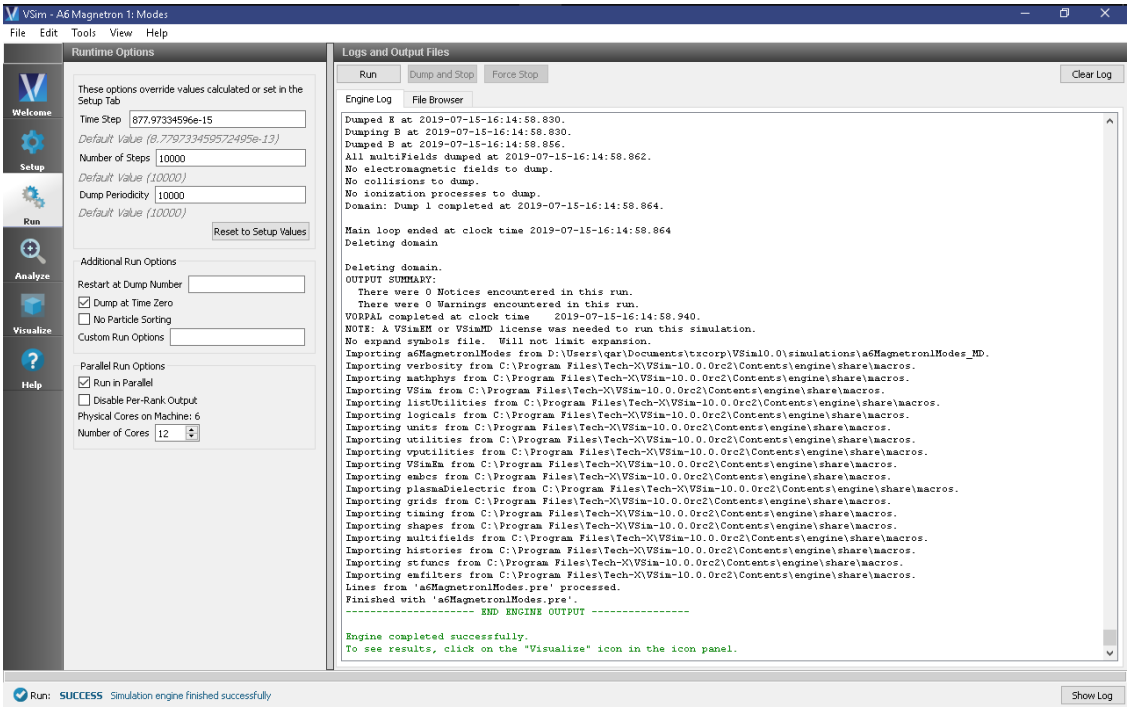


Fig. 4.43: Run window for the A6 Magnetron mode extraction example after the initial run..

The simulation is best run in two steps: The first with course dump periodicity to excite the modes, and the second with fine dump periodicity to observe the modes.

- Initially, the simulation is run for 10,000 time steps, writing one dump file at the end, to allow the current source excitation to finish.

- After the initial run, change the Number of Time Steps to 2,000, the Dump Periodicity to 50, and enter 1 into Restart at Dump Number. This will record details of the excited field after the source has finished.

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 visualize a run to determine the resonant frequency, select *History* from the *Data View* pull-down menu. Select outE_0 for *Graph 1* and *Graph 2*, and click *Fourier Amplitudes (dB)* to the left of one of the outE_0 plot in the *Visualization Results* pane. .

- Proceed to the Visualize window by pressing the Visualize icon in the left panel.
- Select History under Data View.
- For Graph 3 and Graph 4, select <None> to be plotted.
- Then for Graph 2 select the *Fourier Amplitudes (dB)* checkbox * In the upper right corner of each plot, select Limits and set X-Axis max to 1e10.
- The result should be that shown in Fig. 4.44.

Note that running the simulation longer will more sharply resolve the mode frequencies in the FFT.

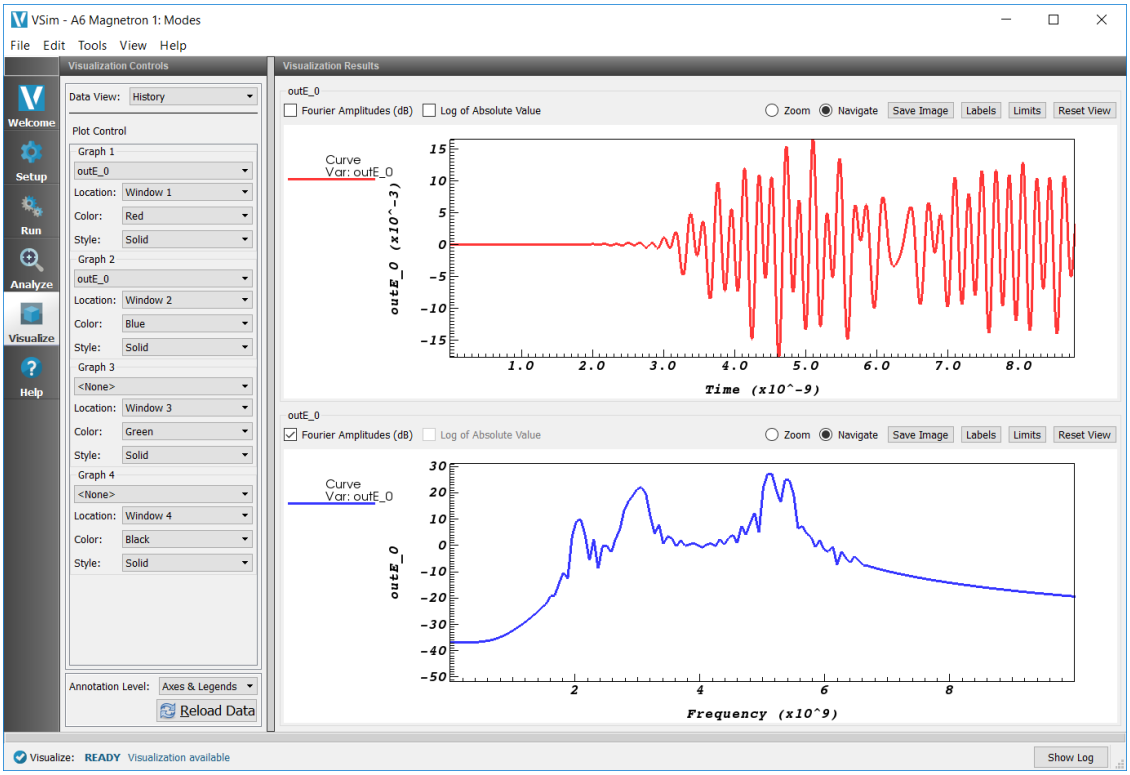


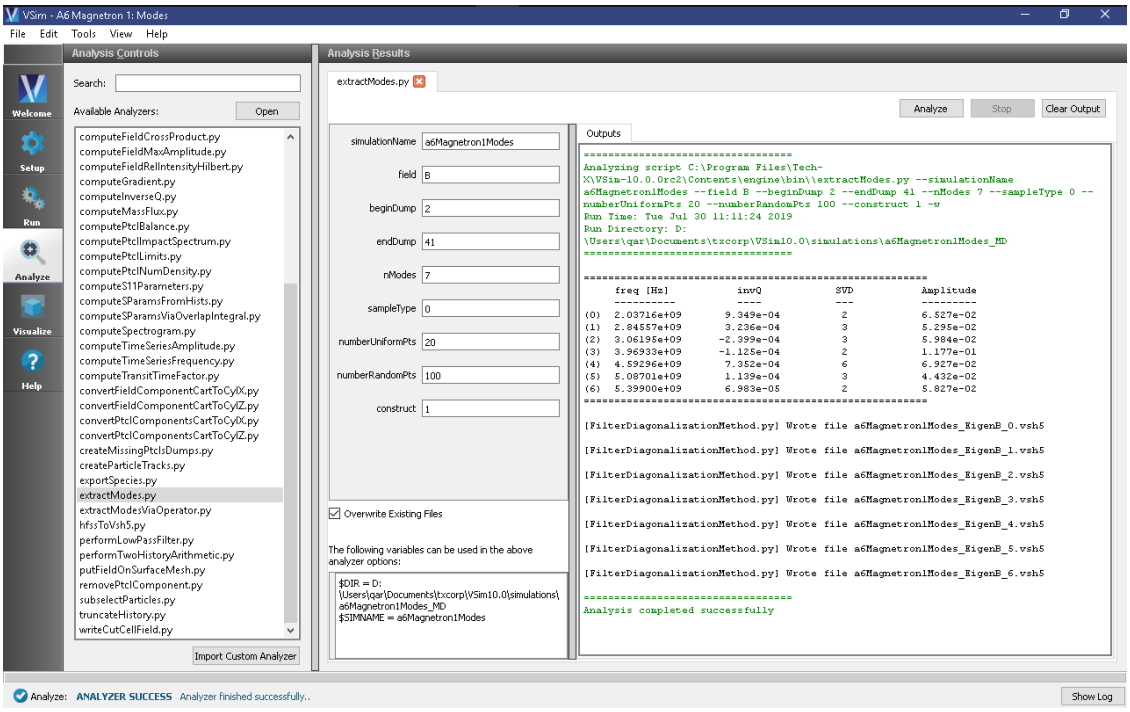
Fig. 4.44: Fourier transform of outE_0 versus time (in Hertz).

Analyzing the Results

It is possible to extract the modes of the A6 magnetron cavity via post processing using the *extractModes.py - Extract Modes Analysis Script* as follows:

- Press the Analyze button in the left column of buttons.
- Select *extractModes.py(default)*.
- Enter the following parameters in the appropriate fields:
  - simulationName = magnetronA6Modes
  - field = B
  - beginDump = 2
  - endDump = 41
  - nModes = 7
  - sampleType = 0
  - numberUniformPts = 20
  - numberRandomPts = 100
  - construct = 1
- Click the *Analyze* button in the upper right corner of the window.

Three columns of data with the titles "freq [Hz]" (Eigenmode frequency), "invQ" (inverse quality factor), and "SVD" (singular value decomposition) will be output in the right pane. The analysis has completed when you see the output "Analysis completed successfully." One can see 7 modes in Fig. 4.3.2.



- Proceed to the Visualize window by pressing the Visualize icon in the left panel.
- Select *Data Overview* under *Data View*.

- Expand *Scalar Data*
- Expand *B*
- Select *B_z (EigenB)*.
- Select *Clip All Plots* in the bottom left corner.
- Move the dump slider first to dump 2 and then to dump 5.

The axial magnetic field inside the cavity during the π and 2π mode operations are shown in Fig. 4.45 and Fig. 4.46, respectively:

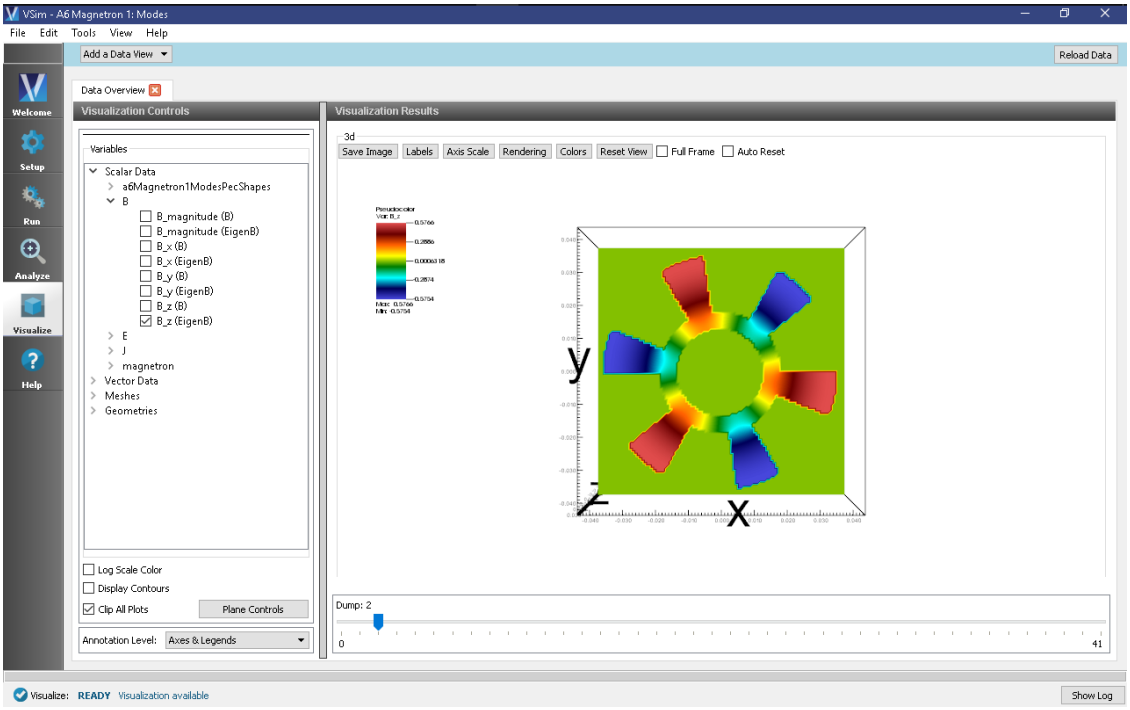


Fig. 4.45: Visualization of the axial B-field in the π Eigenmode.

Further Experiments

The values of FREQ_LOW and FREQ_HIGH can be adjusted to find additional modes, or to focus in on a specific mode. Narrowing the excitation range to fewer modes will produce a mode-accurate frequency and Q-factor extraction for the mode.

4.3.3 A6 Magnetron 2: Power (a6Magnetron2Power.sdf)

Keywords:

`magnetron, bunching, space charge, A6`

> **Warning:** Due to the randomness of the particle generation, the results of this example will differ slightly with MPI ranks. To reproduce the images exactly described in this documentation, we recommend using an MPI setting of 40 cores.

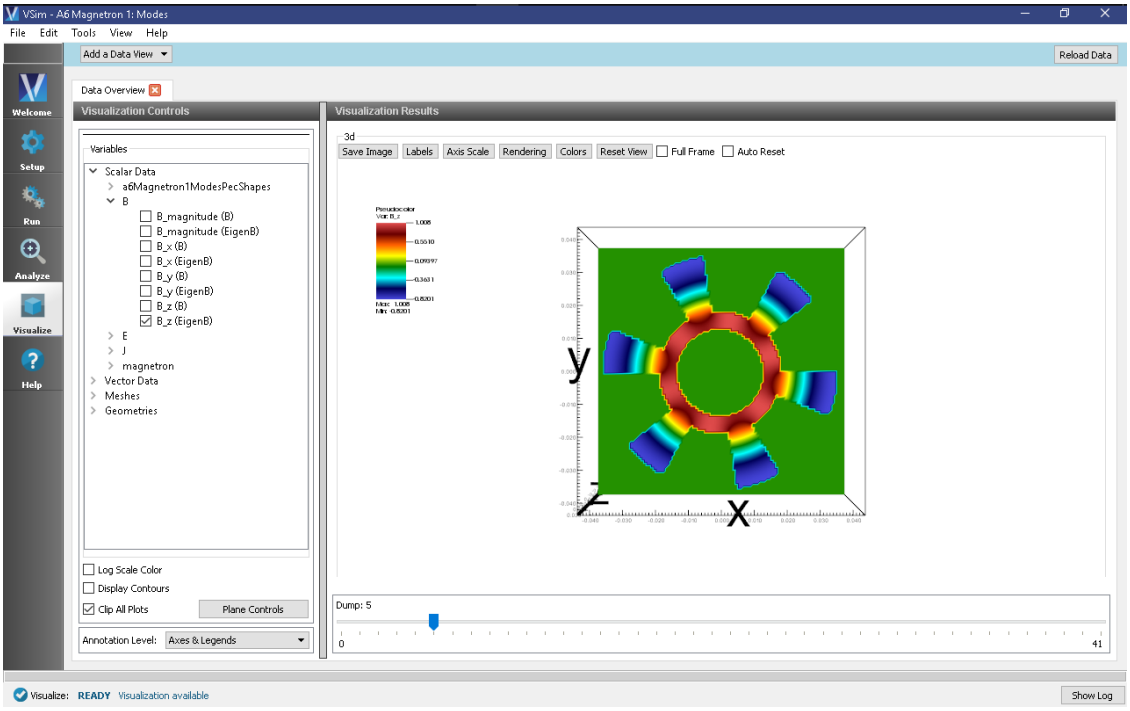


Fig. 4.46: Visualization of the axial B-field in the 2π Eigenmode.

Problem Description

This VSimMD example simulates MIT's cylindrical A6 magnetron cavity with a slot outlet in three dimensions. The geometry was defined by using VSimComposer's constructive solid geometry (CSG) capabilities. The cathode-anode voltage is ramped up from zero to around 360 kV by a current distribution source. Electrons are emitted from the emitter section of the cathode, and undergo $E \times B$ drift. Bunching of the space-charge distribution occurs and kinetic energy from the electrons is transferred to the electromagnetic modes of the cavity. If the simulation is run long enough, it will be seen that the 2π mode dominates.

This simulation can be performed with a VSimMD license.

Opening the Simulation

The A6 Magnetron example is accessed from within VSimComposer by the following actions:

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim for Microwave Devices* option.
- Expand the *Radiation Generation* option.
- Select *A6 Magnetron 2: Power* 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. 4.47. 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 de-select the box next to `Grid`.

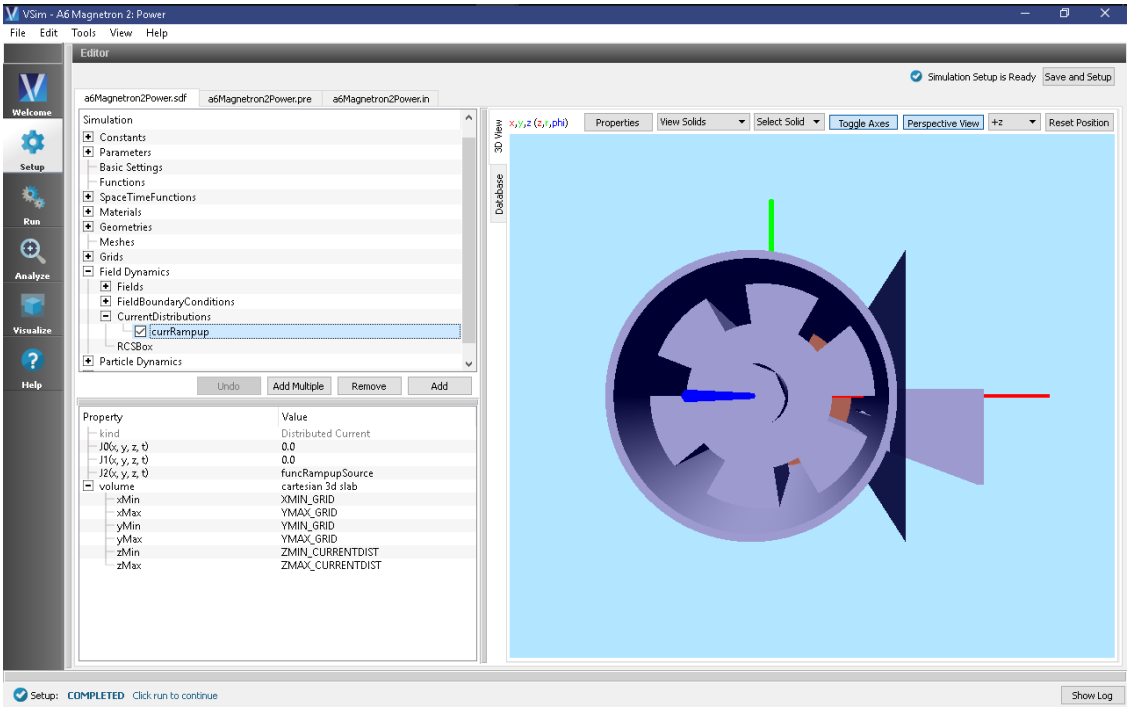


Fig. 4.47: Setup Window for the A6 Magnetron example.

Simulation Properties

The A6 Magnetron example includes several constants for easy adjustment of simulation properties. User changeable parameters include:

- **RADIUS_ANODE** → Inner radius of anode.
- **RADIUS_ANODE_OUTER** → Outer radius of anode.
- **ANGLE_CAVITY** → Angle of resonant cavity openings, in degrees.
- **THICKNESS_WALL_OUTER** → Thickness of all walls.
- **WIDTH_MAGNETRON** → Total width of magnetron in z-direction.
- **RADIUS_OUTLET** → Radius of outlet horn in x-direction.
- **WIDTH_IRIS** → Width of outlet slit opening.
- **WIDTH_VANES** → Total width of anode vanes in z-direction.
- **RADIUS_CATHODE** → Radius of the emitting section of the cathode.
- **RADIUS_CATHODE_INNER** → Radius of the inner section of the cathode.
- **WIDTH_CATHODE** → Width (in z-direction) of the emitting section of the cathode.
- **(X,Y,Z)POS_HIST** → Position of electric field history.

The axial magnetic field is uniform with a constant value of $B_z = 0.6$ T. There is an opening at the back of one of the cavities that allows microwave energy to leave the magnetron through a horn antenna and into a matched absorbing layer (MAL) boundary. A Poynting Flux history records the power output through the antenna. The emitting region of the cathode has a slightly larger radius than the rest of the cathode. There are also end caps, on either side of the vanes, in electrical contact with the cathode.

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.
- Here you can set run parameters, including how many cores to run with (under the MPI tab).
- When you are finished setting run parameters, 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."

This simulation is setup so that the cathode emission current and anode-cathode (AK) voltage ramp up relatively slowly (over many RF periods). Once the AK voltage is high enough, the bunching of the electrons will occur. If the simulation is run for long enough, around 100000 time steps, the 2π mode will eventually dominate as has been seen experimentally for this magnetron.

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.

After the simulation has been run for a sufficient amount of time, spokes will form in the electron distribution (bunching). Since the A6 is a six-cavity magnetron, operation in the 2π mode will correspond to six spokes. To visualize the spokes:

- Select *Phase Space* from the *Data View* pull-down menu
- Select *electrons_x* for the *X-axis*
- Select *electrons_y* for the *Y-axis*
- Use the *Dump* bar at the bottom of the screen to advance through the solution and visualize each dump file

Eventually, a steady state will be reached in which the electron distribution has six spokes similar to Fig. 4.48 taken at Dump 27.

To visualize the axial magnetic field during the 2π mode operation, proceed as follows:

- Select *Data Overview* from the *Data View* pull-down menu
- Expand *B*
- Select *B_z*
- Expand *Geometries*
- Select *poly (a6Magnetron2PowerPecShapes)*
- Select *Clip All Plots*
- Select *Colors*
- Check *Fix Minimum* and *Fix Maximum* and set to -0.06 and 0.06 respectively
- Move the dump slider forward in time to see the evolution.

Image Fig. 4.49 was taken at Dump 27.

To determine the operating frequency:

- Select *History* from the *Data View* pull-down menu
- Select *outB_2* for *Graph 1* and *Graph 2*

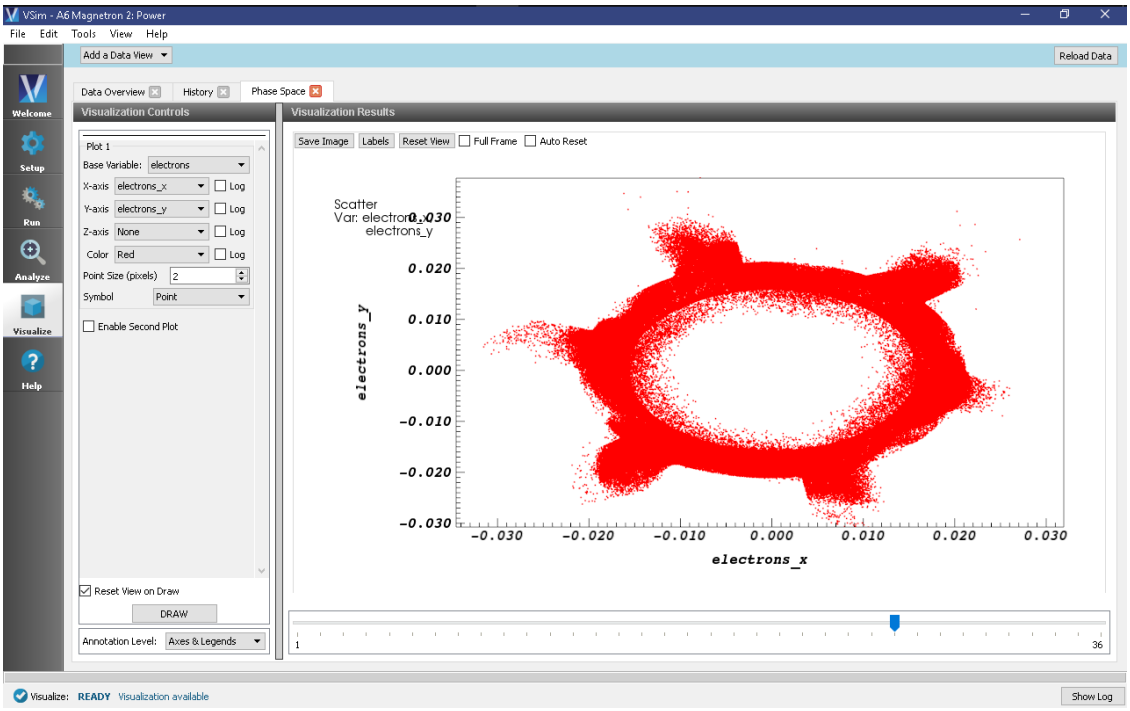


Fig. 4.48: Phase-space plot of the electron distribution showing the formation of six spokes corresponding to operation in the 2π mode.

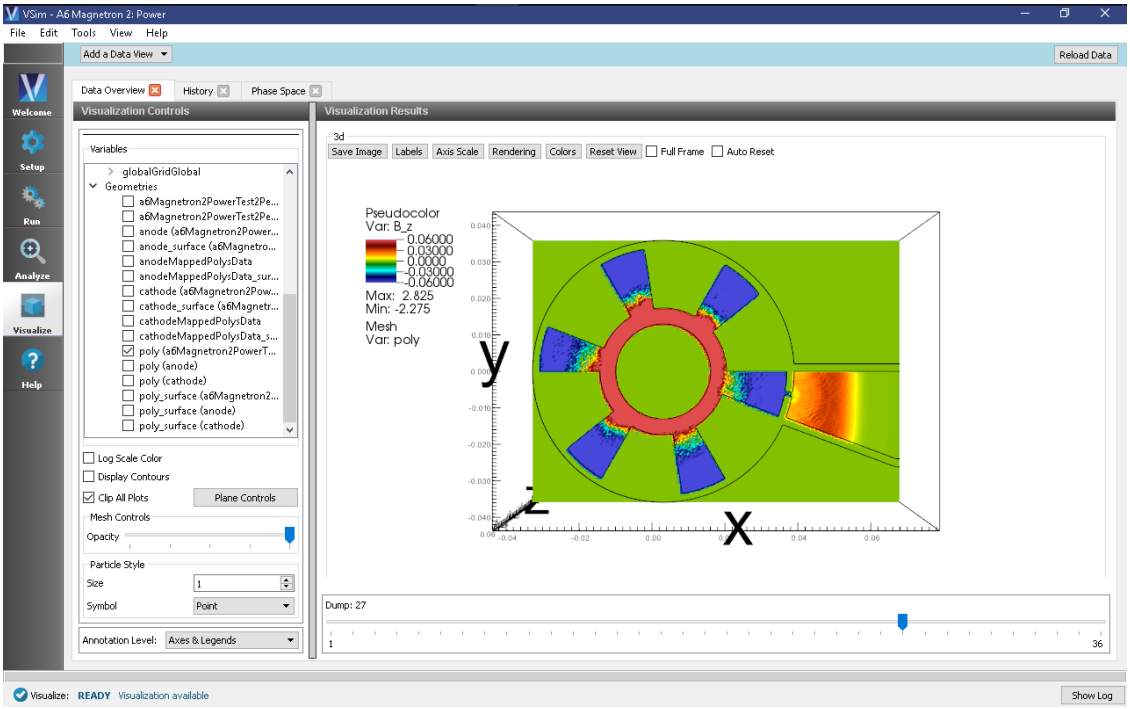


Fig. 4.49: Axial magnetic field of the magnetron operating in the 2π mode.

- Click *Fourier Amplitudes (dB)* to the left of one of the plots in the *Visualization Results* pane
- Zoom in on the maximum of this plot to determine the approximate resonance mode frequencies

The resulting plot will resemble Fig. 4.50. If the simulation has been run for long enough, the peak at 4.6 GHz, which corresponds to the 2π mode, should be the most prominent.

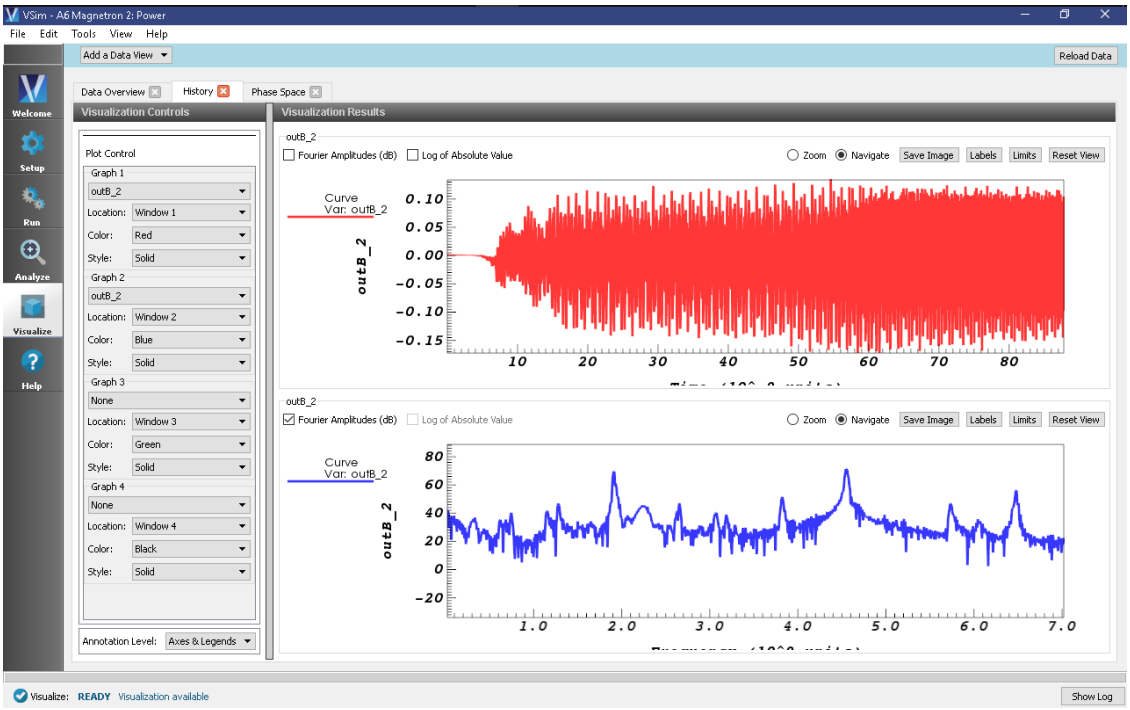


Fig. 4.50: Plot of outB_2 vs. time and vs. frequency (Fourier transform).

Further Experiments

The power output and operating mode is affected by the E/B ratio and the geometry. The user could try adjusting the strength of the magnetic field and the geometry of the cathode, including the radius of the emitting region and the configuration of the end-caps, to see how this affects magnetron operation.

4.3.4 Field Emitter Array (fieldEmitterArray.sdf)

Keywords:

`field-induced emission, particle beam, FEA, emitter array, Fouler-Nordheim, space charge`

Problem Description

Obtaining high currents and high current densities via electron emission from cold cathodes is a high demand for scientists and engineers. Field emitter arrays (FEAs) operate via field-induced particle emission from very thin cathodes and are highly efficient. For this reason, they have been widely studied via experimental methods.

This VSim for Microwave Devices example illustrates how to setup a 3x3 FEA. VSim uses a cut-cell field emitter following a space-charge corrected Fowler-Nordheim emission model [1]. VSim has the capability of managing geometry structures at the micron and even nanometer range, effectively meshing single emitters and emitter arrays.

In addition, VSim also models dielectric to second-order accuracy, making it possible to include dielectrics in the FEA design.

This simulation can be run with a VSimMD license.

Opening the Simulation

The SmithPurcellRadiation example is accessed from within VSimComposer by the following actions:

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim for Microwave Devices* option.
- Expand the *Radiation Generation* option.
- Select *Field Emitter Array* 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. 4.51.

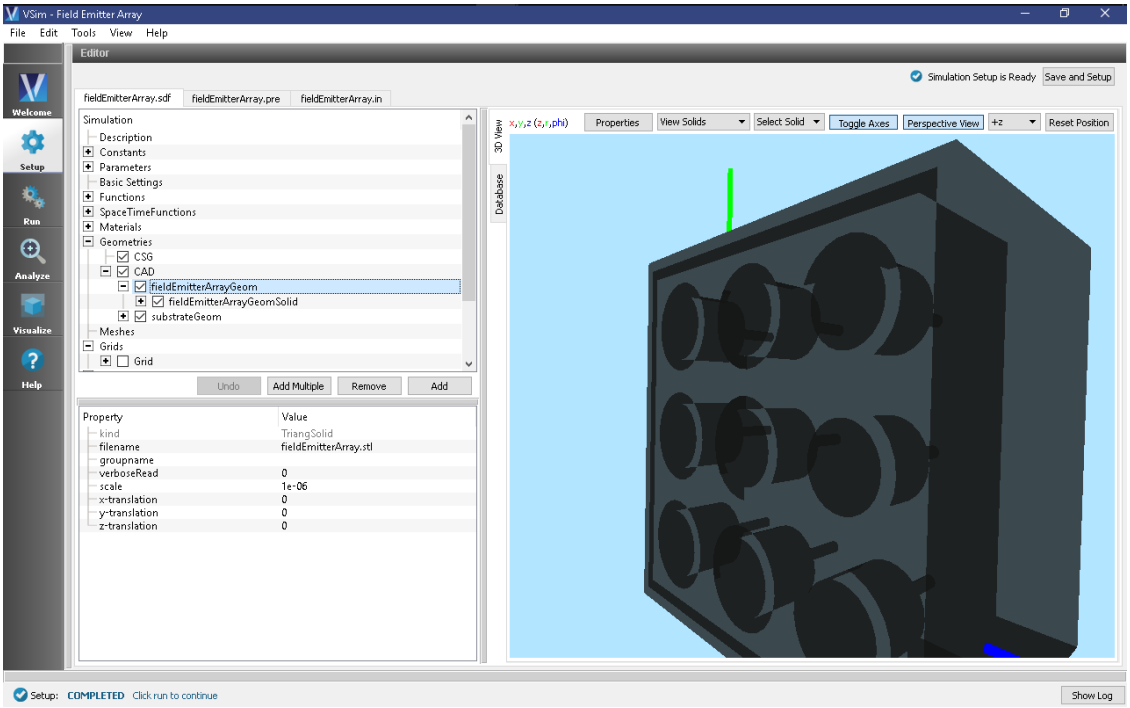


Fig. 4.51: Setup Window for the Field Emitter Array example.

Simulation Properties

The simulated device is a 3x3 field emission array with a dielectric substrate. In order to simplify the setup of the geometry, each emitter tip was set up as a thin cylinder of length of 0.55 microns and a radius of 0.05 microns. The emitter tips are 0.05 microns deep in inside the gate openings which are 0.15 microns in radius. The metal gate thickness is 0.1 microns. The distance between the emitter and the cavity wall is 1.15 microns. The distance between the centers of adjacent emitters is 1.00 microns. The metal gate was topped with a dielectric layer with a thickness of 0.1 microns. Alumina was set for the dielectric material. The voltage between the cathode and the gate was set to 100 V, while the gate to anode voltage was set to 4000 V. These voltages were set through a feedback algorithm.

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." This is shown in Fig. 4.52.

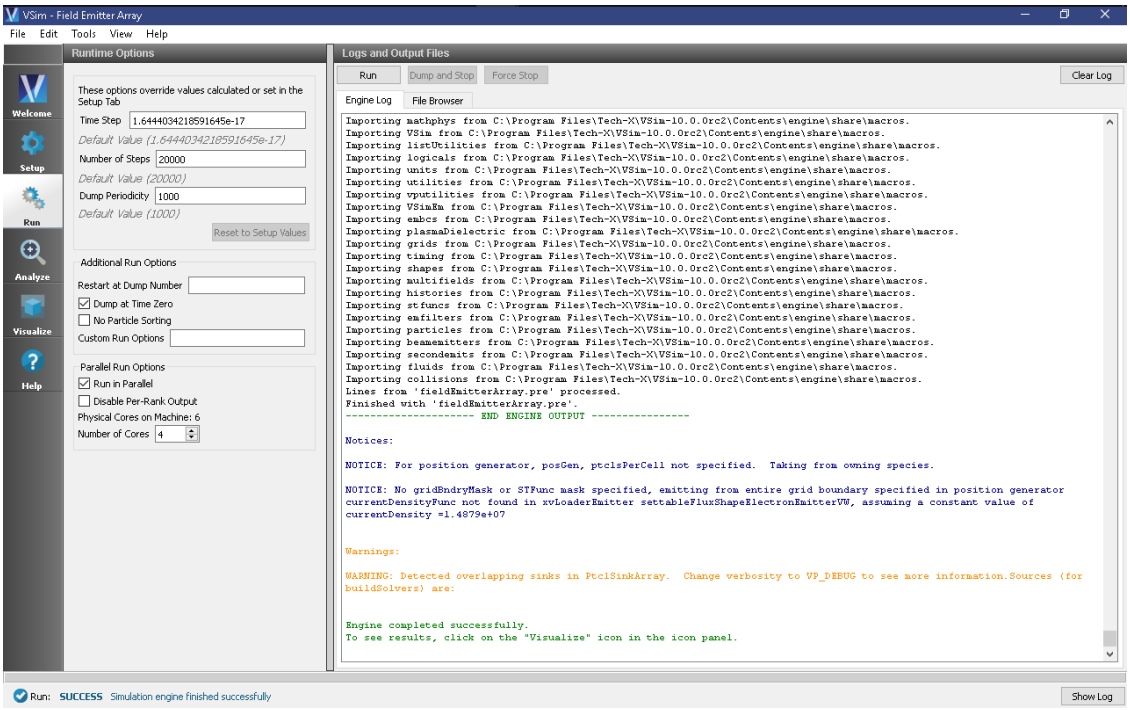


Fig. 4.52: The Run Window at the end of execution.

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.
- From the *Data View* dropdown menu, select *Data Overview*.
- In the variables tree, expand *Particle Data*.
- Select *electrons*.
- In the variables tree, expand *Geometries*.
- Select *poly_surface(fieldEmissionArrayPecShapes)*.
- Select *poly_surface(substrateGeomSolid)*.
- Select *Clip All Plots* in the bottom left corner.
- Click on *Plane Controls*.
- Under *Clip Plane Normal* select *X (plane normal to x-axis)*.

- Under *Origin of Normal Vector* type *1.7e-6* and click *Ok*.
- In the bottom of the right pane, move the dump slide forward in time.
- The resulting visualization is shown in Fig. 4.53.

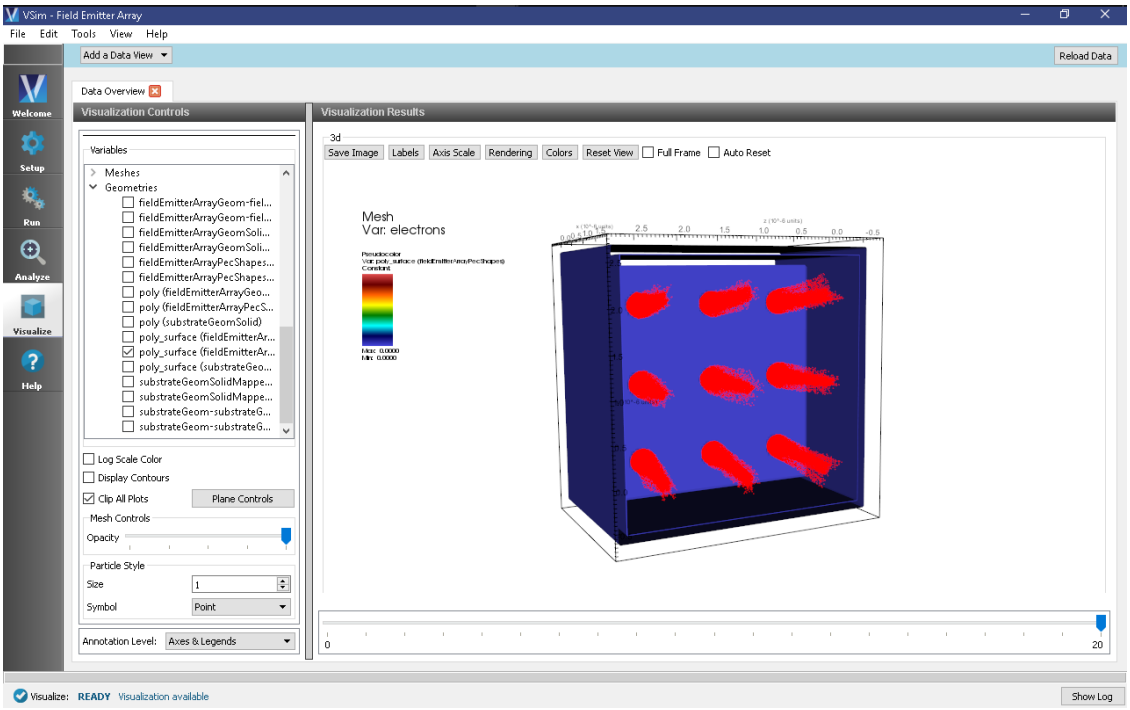


Fig. 4.53: Electron beams originating from the emitter tips behind the dielectric substrate.

To perform an analysis of the axial electric field, proceed as follows:

- Proceed to the Visualize Window by pressing the *Visualize* button in the navigation column.
- From the *Data View* dropdown menu, select *Field Analysis*.
- Under *Field* select *E_x* from the drop-down menu.
- Select *Log Scale Color Table*.
- Under *Intercept* input *6.5e-7*.
- Under *Layout* select *Side-by-side 2d/1d*.
- Click *Perform Lineout*
- Click on *Colors* and select *Set Maximum* with the value *1e+10*.
- In the bottom of the right pane, move the dump slide forward in time.
- The resulting visualization is shown in Fig. 4.54.

To visualize the electron energy and current histories, proceed as follows:

- Proceed to the Visualize Window by pressing the *Visualize* button in the navigation column.
- From the *Data View* dropdown menu, select *History*.
- The resulting visualization is shown in Fig. 4.55.

To visualize the electron Px-x phase space, proceed as follows:

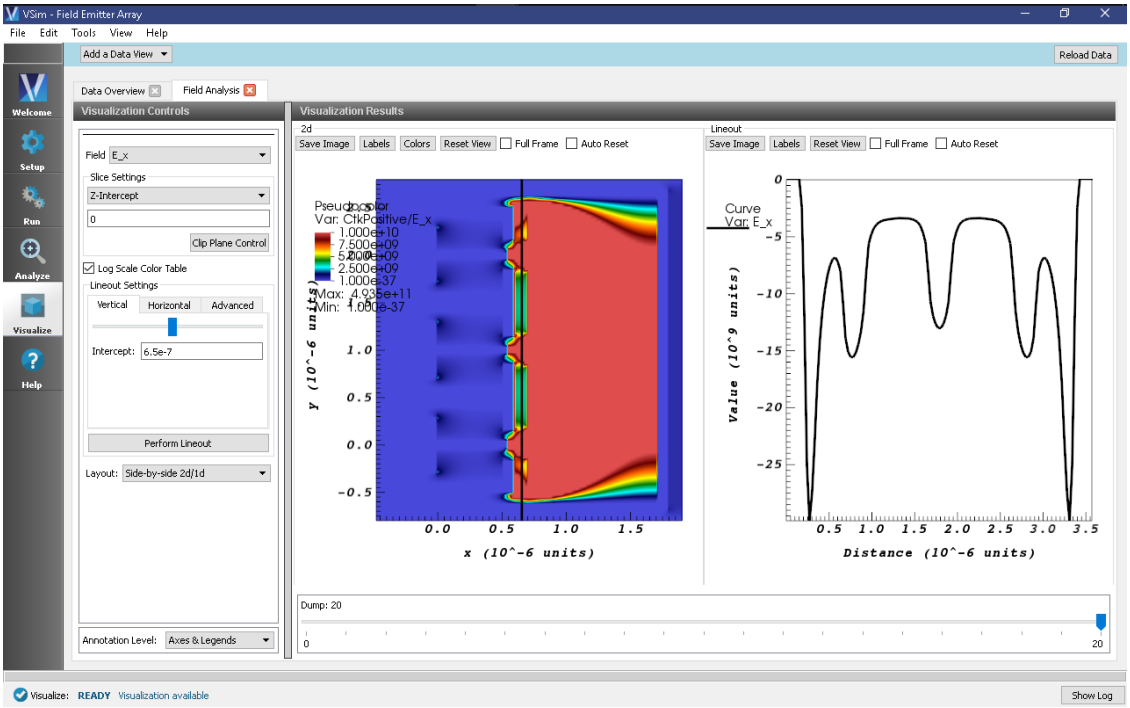


Fig. 4.54: 2D and 1D studies of the electric field which is highly space-charge dominated. The 1D lineout was performed along the dielectric substrate.

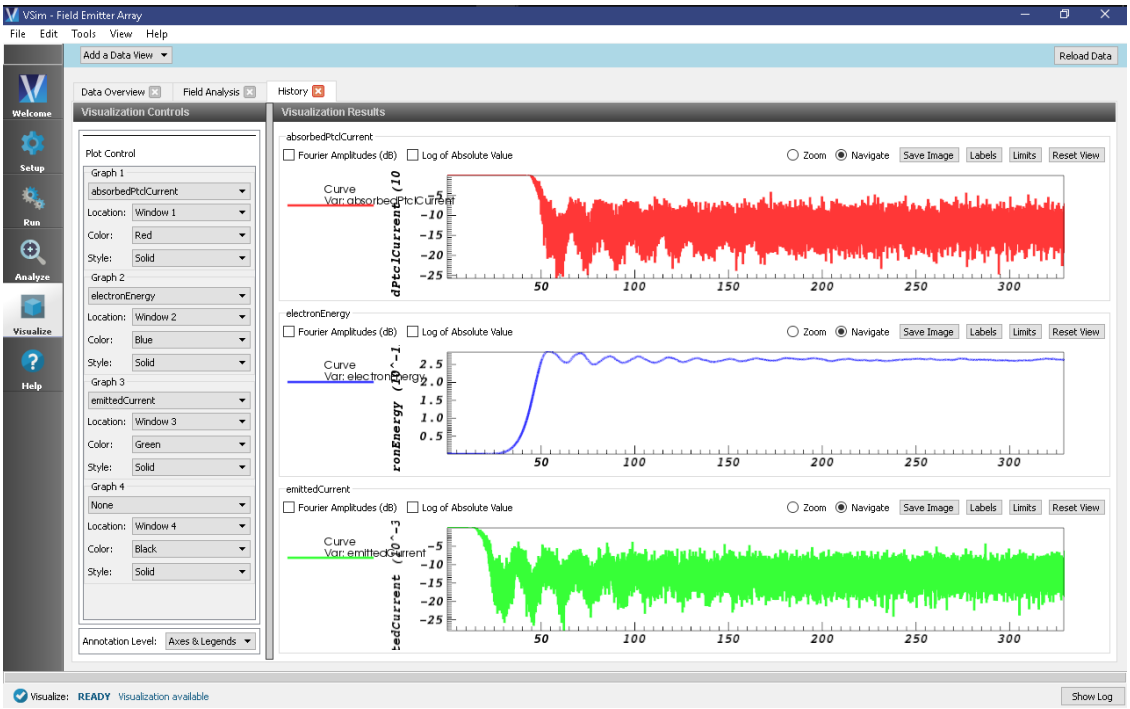


Fig. 4.55: The histories of the emitted and absorbed particle current, and the electron beam energies.

- Proceed to the Visualize Window by pressing the *Visualize* button in the navigation column.
- From the *Data View* dropdown menu, select *Phase Space*.
- For the *X-axis*, select *electrons_x*.
- For the *Y-axis*, select *electrons_ux*.
- Click *Draw*.
- Move the dump slider further in time.
- The resulting visualization is shown in Fig. 4.56.

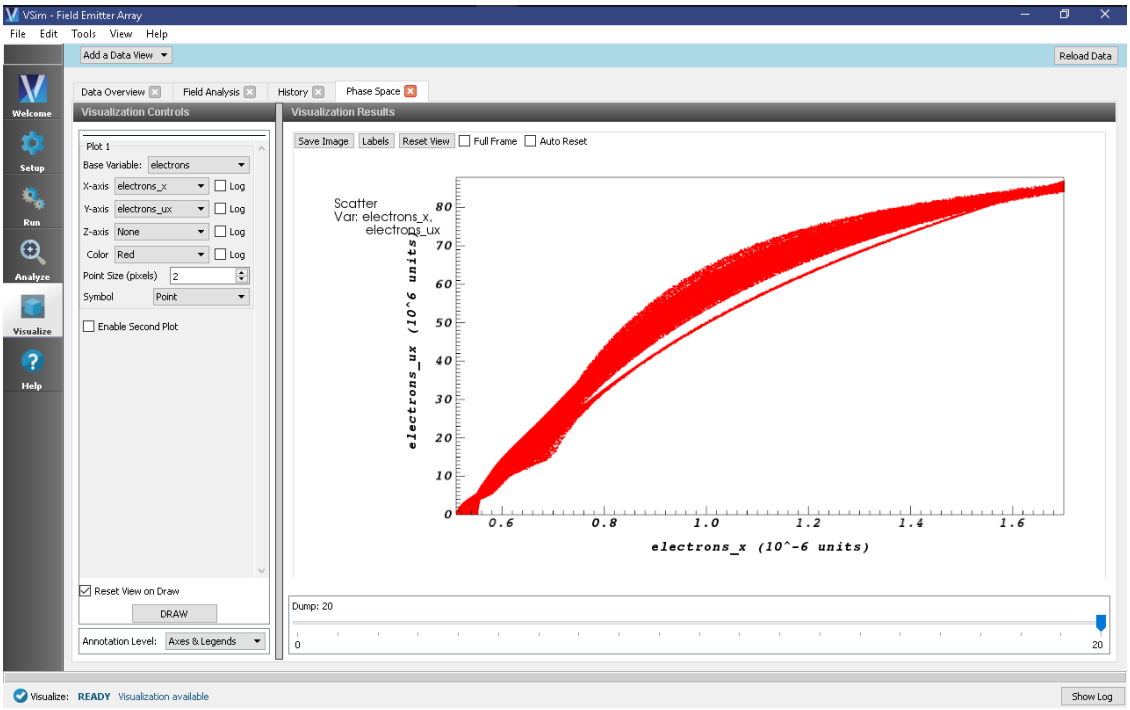


Fig. 4.56: The distribution of field-emitted electrons on the Px-x phase space.

Fig. 4.53 shows the first results: narrow strong beams of particles emitted from the tips of the 9 emitters. In addition, Fig. 4.54 shows the axial component of the electric field which is highly space-charge dominated. A lineout measurement is performed at the location of the dielectric substrate (right-hand image). The histories of the emitted and absorbed particle currents and the electron energy are shown in Fig. 4.55. Lastly, the distribution of field-emitted electron on the Px-x phase space is shown in Fig. 4.56.

Further Experiments

One of the first tests that can be perfomred easily using this simulation is to investigave how the electron emission behaves when changing the anode and cathode voltages. These voltages are defined in the setup element tree under ANODE_VOLTAGE and DC_BIAS, respectively.

Another study can be performed by changing the properties of the dielectric substrate and investigating the resulting effects. The user can edit the dielectic material properties directly in the setup window under “Materials” by selecting the material and then manually changing the property values (conductivity, permitivity, etc.) in the pane below.

4.3.5 Gyrotron Mode (gyrotronMode.sdf)

Keywords:

gyrotron

Problem description

This VSimMD example illustrates a very high order mode, TE-22-6, propagating in a cylindrical waveguide, very near to the cutoff frequency, which is a common situation in a gyrotron. The example is intended to allow investigation of the axial phase and group velocity of such a mode, as a function of frequency, and to highlight the intricacies of simulating a mode that is propagating within a percent or two of its cutoff frequency.

This simulation can be performed with a VSimMD license.

Opening the Simulation

The Gyrotron Mode example is accessed from within VSimComposer by the following actions:

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim for Microwave Devices* option.
- Expand the *Radiation Generation* option.
- Select "Gyrotron Mode" 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, if applicable. See Fig. 4.57.

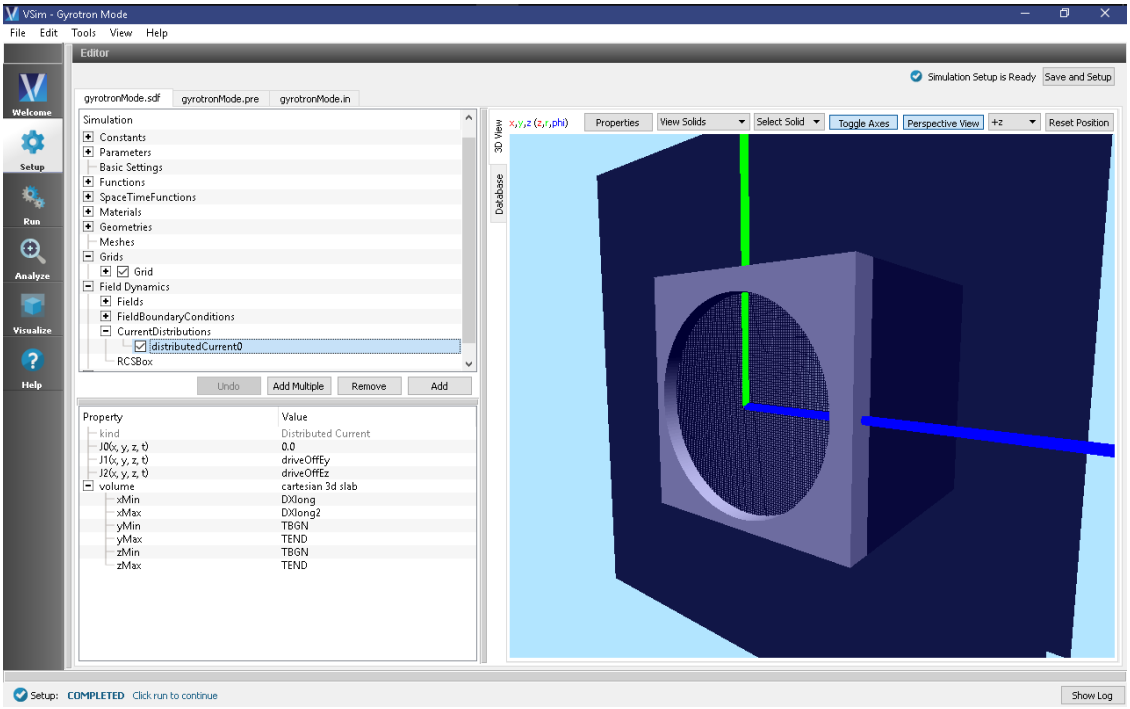


Fig. 4.57: Setup Window for the Gyrotron Mode example.

The basic variables of this problem should now be alterable via the text boxes in the left pane of the *Setup Window*, as shown in Fig. 4.57.

Simulation Properties

There are only two geometrical input parameters; the waveguide radius and length. The user may also control the excitation frequency, the duration of the simulation, and the nature of the excitation, specifically whether it is pulsed or continuous-wave. Additional exposed parameters include the grid sizes, and the tuning of the exiting wave boundary condition, which allows for more in-depth study with this example.

The excitation may be pulsed or continuous-wave, depending on the current input values in the element tree under *Field Dynamics* → *Current Distributions* → *distributedCurrent0*. The default *J_y* and *J_z* values are *driveOffEy* and *driveOffEz*, respectively, for a pulsed driver. The user can change these values to *driveOnEy* and *driveOnEz* for a continuous driver. The two methos of driving the cavity are described in the following.

Pulsed Simulation (driveOffE)

In this case, the wave is driven for half of the simulation duration, with a smooth turn-on / turn-off time window. Then, for the remaining half of the periods, the excitation propagates freely. The axial profile of the pulse will be very short, typically just one or two axial wavelengths. It will propagate slowly down the waveguide, as expected from the group velocity which is very small near cutoff. In the center of the pulse the TE-22-6 mode is preserved, but because this is a pulse, nearby modes in frequency are also present. One can observe a rich set of other mode patterns just a few grid planes away from the center of the pulse.

Continuous-Wave Simulation (driveOnE)

The drive may be kept on, instead of having it turn off halfway through the simulation. After an initial transient, this sets up a single TE-22-6 traveling wave mode pattern throughout the waveguide. This allows for accurate measurement of the axial wavenumber, beta, for the 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.
- To run the file, click on the *Run* button in the upper left corner of the right panel. 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 is shown in Fig. 4.58.

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 B_x field is the best component for looking at in this simulation, as shown in Fig. 4.59.

- Expand *Scalar Data*.
- Expand *B*
- Select *B_x*
- Using the cursor, grab the image and rotate it from right to left by 90 degrees.
- Move the dump slider to dump 11.

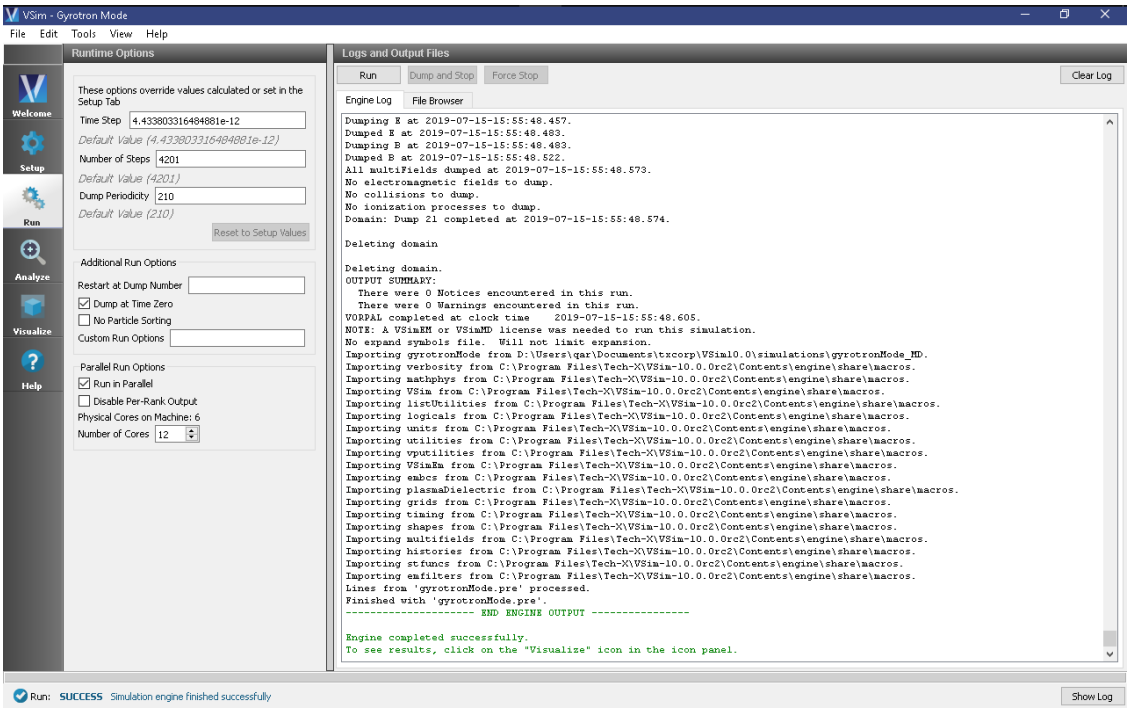


Fig. 4.58: The Run Window at the end of execution.

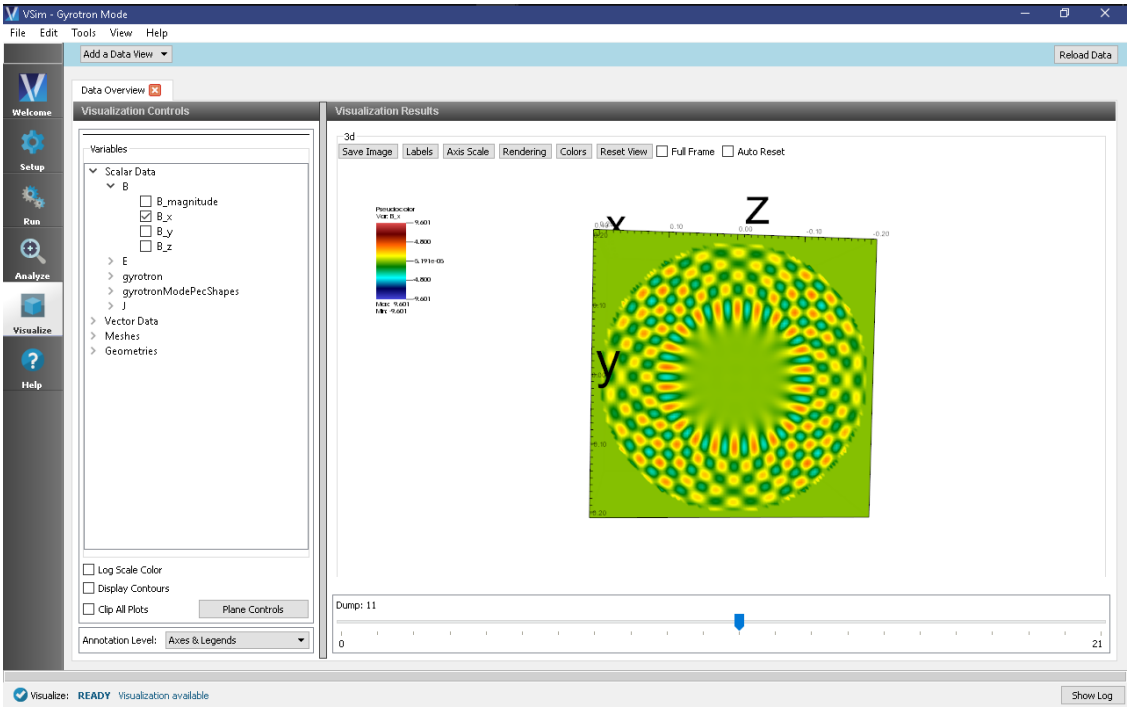


Fig. 4.59: Illustration of the mode pattern, and propagation of the mode down the length of the tube.

The initial parameters are selected so that the excitation frequency is just barely above cutoff. While the axial phase velocity is high in this case, the group velocity is quite low, and the simulation shows a narrow wavepacket slowly moving down the length of the tube, while remarkably still maintaining the very high order TE-22-6 pattern. Contamination of the pattern increases as the duration of the excitation is reduced, since more frequencies are brought into the transient. The user is encouraged to look at the mode pattern and contamination properties as frequency and duration are varied.

The TE-22-6 mode's cutoff frequency, for the suggested initial radius of 20 cm, is known analytically to be 10.8845 GHz, which derives from the value of the 6th root of the J_{22} bessel function, which is 45.624312. However, the user will note that the suggested initial drive frequency is below this, at 10.74 GHz, and yet the wave appears to propagate! This illustrates an important property of finite-difference dispersion, that in fact the speed of light is ever-so-slightly slower in the finite-difference-time-domain simulation than in reality. In most cases, this is hardly noticed, however, when operating this close to the cutoff frequency of a waveguide, this difference can be readily seen, as this example illustrates. The discrepancy between the discrete FDTD cutoff frequency and the analytic cutoff frequency, depends on the grid resolution of the wave, and in general decreases as δx^2, where δx is the grid size.

A very useful piece of information is the FDTD cutoff frequency. This may be found with a series of simulations, each at different drive frequencies, ω. The KEEP_DRIVE_ON parameter should be set to 1, so that the axial wavelength, β, can be measured from the field plots. A plot of ω^2 vs. β^2 should be essentially linear, with the intercept on the ω^2 axis being the FDTD cutoff frequency, ω^2_{cutoff} ($\omega^2 = \omega^2_{cutoff} + c^2\beta^2$), and with slope being the FDTD speed-of-light-squared. A spreadsheet showing this exercise for the suggested initial values of the example is shown below. The result of this study is that the FDTD cutoff frequency is actually 10.675 GHz, or 2% below the known analytical result, for the initial suggested grid resolution.

Further Experiments

The user is encouraged to repeat the simulations discussed in the previous section with a finer resolution, to see how the FDTD cutoff frequency approaches the analytic result as resolution improves.

The detailed TE-22-6 mode pattern is very carefully crafted using polynomial fitting functions, and is introduced into the axial magnetic field, B_x, at the left side of the simulation. There is no direct option to use a different mode, although the user may attempt to edit the detail of the input to do so.

Finally, a boundary condition tuning parameter, VPHASE_PORT, is offered to allow the user to experiment with tuning of the outgoing wave boundary condition in this near cutoff scenario. In this circumstance, the optimal phase velocity may be 5 to 10 times the speed of light.

An additional exposed user parameter, FREQ_CUTOFF, is offered, and may be used to store the value derived from the simulations discussed in the previous section. By default, this parameter is not used. However the user may look into the detail of input file, and notice a comment line that indicates how this parameter might be used to set the value of VPHASE_PORT more accurately.

4.3.6 Helix Traveling Wave Tube 1: Dispersion (helixTwt1Dispersion.sdf)

Keywords:

`Helix TWT Dispersion Analysis Run`

Problem description

This VSimMD example is one of a set of simulations showing different calculations to aid the design of a helix traveling wave tube (TWT) in three dimensions. The 100-turn helix with end feeds is imported from a CAD file, but all other geometrical parts are created with the Constructive Solid Geometry (CSG) capabilities within VSimComposer. The dependence of the geometries on the constants and parameters will be discussed in *Helix Traveling Wave Tube 2: Impedance and Attenuation (helixTwt2ImpedAtten.sdf)*.

Frequency	Wavelength		beta^2	omega^2
10.54	*cutoff*			4385.721
10.64	*very long*			4469.336
10.74	0.20		986.9604	4553.741
10.84	0.14		2014.205	4638.935
10.94	0.11		3262.679	4724.919
11.04	0.09		4873.879	4811.693
Cutoff Frequency		<==		**intercept**
10.675				4498.7

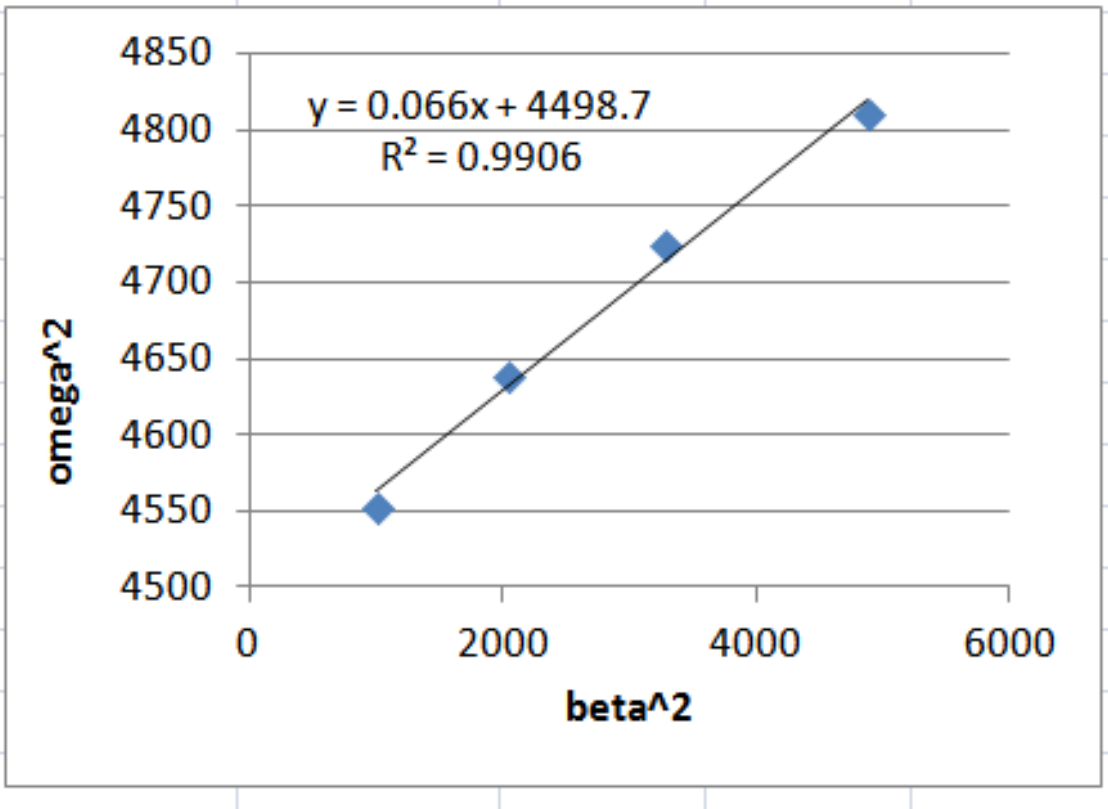


Fig. 4.60: Computing the FDTD cutoff frequency of the TE-22-6 mode.

This simulation addresses the dispersion analysis of the tube and as such runs with a grid covering a reduced number of helix turns. An impulse signal is excited between the helix and the body tube (which would be the vacuum interface in a real device) and periodic boundary conditions are enforced at the two ends. The tube is allowed to run for sufficiently long to observe the behavior at relatively low frequencies. We are able to recover the phase velocity of the wave on the helix (and so the structure/RF curve on an omega beta diagram) from this simulation. It differs from the other simulations of helix TWT in that the simulated region contains no coaxial coupler. As well the attenuator is outside the simulated region. For other studies of the TWT, see

- *Helix Traveling Wave Tube 2: Impedance and Attenuation (helixTwt2ImpedAtten.sdf)*
- *Helix Traveling Wave Tube 3: Power Run (helixTwt3PowerRun.sdf)*

This simulation can be performed with a VSimMD or VSimPD license.

Opening the Simulation

The Helix TWT example is accessed from within VSimComposer by the following actions:

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim for Microwave Devices* option.
- Expand the *Radiation Generation* option.
- Select "Helix Traveling Wave Tube 1: 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.

All of the properties and values that create the simulation are now available in the Setup Window as shown in Fig. 4.61. 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`.

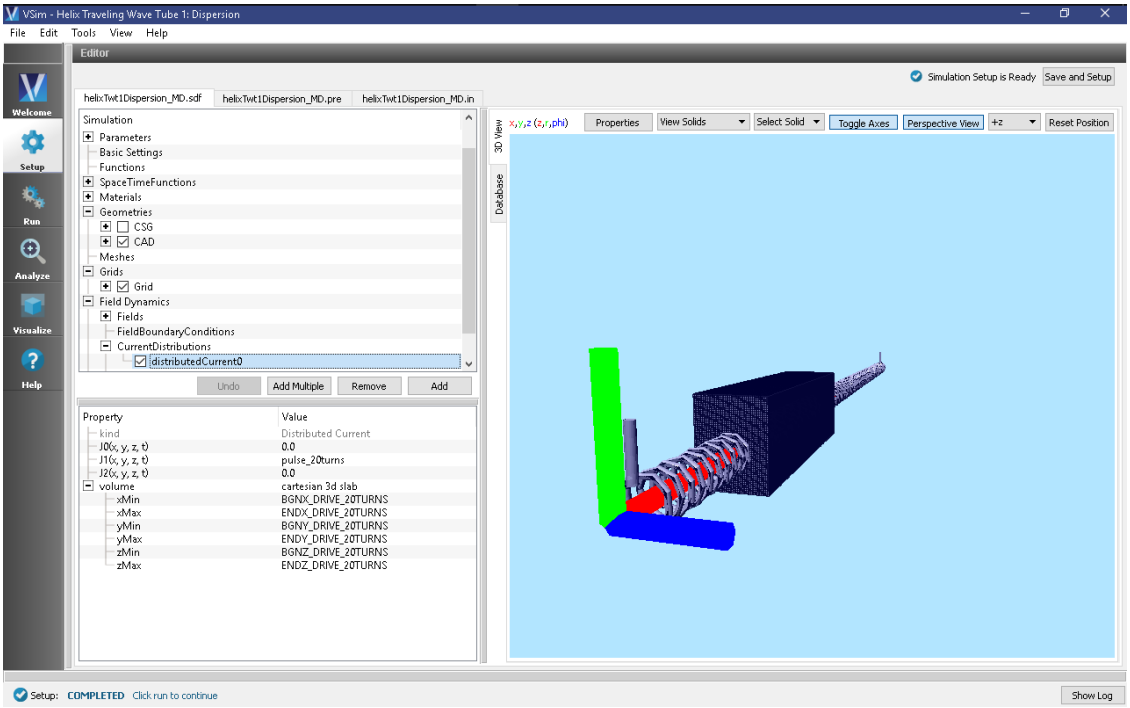


Fig. 4.61: Setup Window for the Helix TWT example.

The geometry of the helix can be made more visible by unselecting the tube and emitter disk parts under Geometries/CSG in the tree (e.g. driftTubeSolid, tubeSolid, driftTubeVoid, tubeVoid, tube, and emitterDisk) or by changing their color property and selecting a low alpha on Windows or Linux (or opacity on Mac). See color property setting in the User Guide.

setting is made available by clicking on the "Properties" button.

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.
- To run the file, click on the *Run* button in the upper left corner of the right panel. 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 is shown in Fig. 4.62.

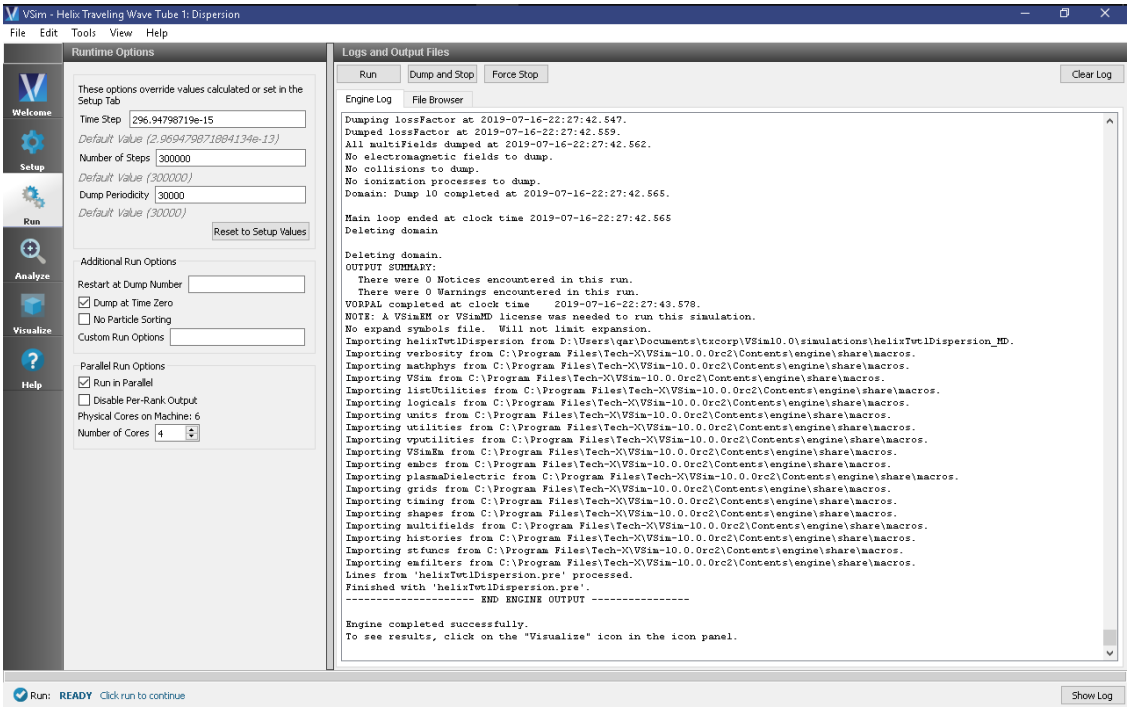


Fig. 4.62: The Run Window at the end of execution.

The simulation allows a frequency domain analysis to be performed. In order to resolve low frequency signals, a large number of steps is required. Increasing the number of steps further may help to improve the frequency domain resolution, especially at the low frequency end of the spectrum. The Dump Periodicity may be increased to reduce the number of data dumps and save space at the expense of being able to view up to date simulation data while it runs.

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 fields inside the tube as shown in Fig. 4.63, do the following:

- Expand *E*

- Select *E_x*
- Expand *Geometries*
- Select *poly (helixTwtWithFeeds)*
- Select *Clip All Plots*
- Select *Colors*
- Check *Fix Minimum* and *Fix Maximum* and set to -.05 and 0.05 respectively
- Move the dump slider forward in time to see the evolution
- Click and drag to rotate the image
- Select *Display Contours* and set # of contours to 4.

The wave travels along the helix, and the strongest fields occur between turns. The individual modes are not separated out in this case.

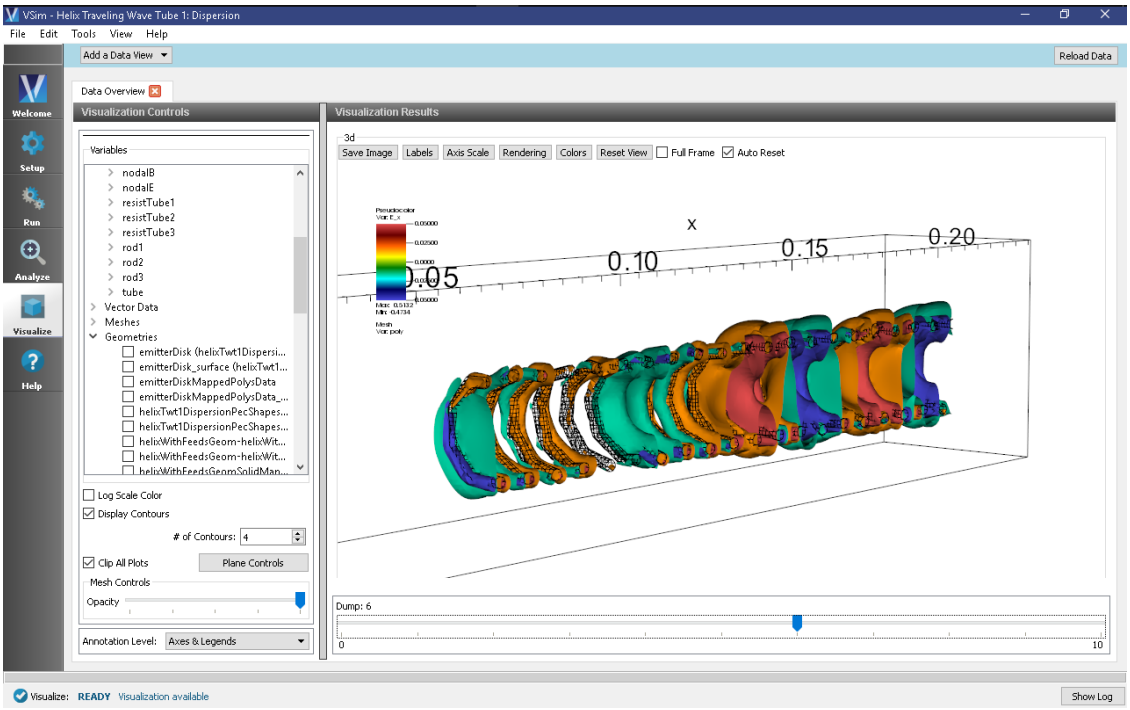


Fig. 4.63: Longitudinal field, E_x, for the dispersion run at time step 120,000.

The *History* records can be used to calculate the dispersion curve:

- Set EonAxisA_0 under Graph 1
- Set EonAxisA_1 under Graph 2
- Under Graph 3 and Graph 4 change the dataset to <*None*>
- Check “Fourier Amplitudes (dB)” button at top left of plot
- Select Zoom radio button
- Using the *Limits* button above the plot set both to have limits of 0 to 2e9.

The result is shown in Fig. 4.64 (to which we have added vertical measuring lines using an external image editing software). A series of peaks can be seen. The mode frequencies correspond to the maxima of this plot. Having more

than one history is important as due to mode variation in space, one history may pick up modes another misses and vice-versa.

Record the frequencies at which these peaks occur, e.g. in a spreadsheet. With the view mode switched from *Zoom* to *Navigate*, a wheel mouse can be scrolled up and down to zoom in and out. This may expedite the process of collecting the data. Or, as we have done, one can add vertical measuring lines after opening the image in some external software.

The first seven frequencies from the 300000 step simulation are listed in the table below. To determine the phase velocity v_n for each mode frequency, first note that the wave number for the n-th mode is given by

$$k_n = \frac{2\pi}{L} n$$

where L is the length of the helix TWT. The phase velocity for the n-th mode is then

$$\frac{v_n}{c} = \frac{2\pi f_n}{c k_n} = \frac{L f_n}{c n}$$

where f_n is the frequency of the n-th mode and c is the speed of light. The first 10 mode phase velocities are listed in the table below for L = 15 cm. For large frequencies, we should expect the phase velocity to approach the ratio of the helix pitch (0.75 cm) to the circumference (6.28 cm), or 0.119.

n	frequency (GHz)	phase velocity (c)
1	0.285	0.1425
2	0.555	0.1387
3	0.810	0.1350
4	1.048	0.1310
5	1.285	0.1285
6	1.520	0.1267
7	1.770	0.1264

Higher resolution and longer duration simulation could be used to better measure the frequencies and, hence, determine the phase velocity. Even more precise frequencies could be obtained by the Filter Diagonalization Method.

Further tests

The axial phase velocity is chosen to be synchronous with the beam. Adjust the helix parameters (in an external CAD editor) and observe the changes to the phase velocity.

Restarting after the default 300000 steps allows more accurate definition of the frequencies.

Use the Filter Diagonalization Method to get the frequencies more precisely.

4.3.7 Helix Traveling Wave Tube 2: Impedance and Attenuation (helix-Twt2ImpedAtten.sdf)

Keywords:

`Helix TWT Impedance and Attenuation Run`

Problem description

This VSimMD example is one of a set of simulations showing different calculations to aid the design of a helix traveling wave tube (TWT) in three dimensions. The 100-turn helix with end feeds is imported from a CAD file, but all other geometrical parts are created with the Constructive Solid Geometry (CSG) capabilities within VSimComposer.

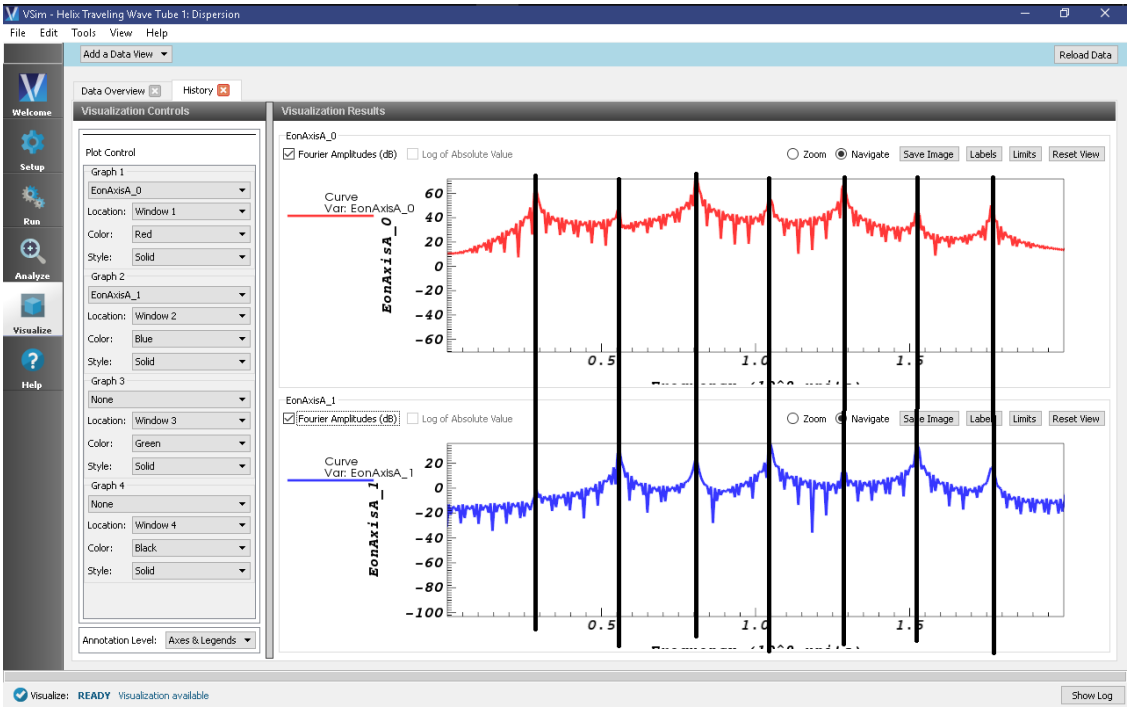


Fig. 4.64: Fourier transform of various histories after 300000 timesteps.

An input signal is sent into a short section of coaxial input waveguide and a similar section of coaxial waveguide at the opposite end of the tube provides an output power port. The geometry includes three dielectric support rods, each clad by a section of resistive tubing for attenuation. The Impediance and Attenuation simulation enables the user to calculate the transverse impedance and Pierce interaction impedance of the helix TWT. The transverse impedance is relevant for impedance matching at the input and output coaxial ports, and the Pierce interaction impedance is related to the interaction of the particles with the signal, and thus the signal gain.

The user may wish to run this simulation type multiple times, varying parameters such as the coax radius and dielectric permittivity, in order to result in a design with the correct impedance parameters.

Related simulations:

- *Helix Traveling Wave Tube 1: Dispersion (helixTwt1Dispersion.sdf)*
- *Helix Traveling Wave Tube 3: Power Run (helixTwt3PowerRun.sdf)*

This simulation can be performed with a VSimMD or VSimPD license.

Opening the Simulation

The Helix TWT Impedance and Attenuation example is accessed from within VSimComposer by the following actions:

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim for Microwave Devices* option.
- Expand the *Radiation Generation* option.
- Select "Helix Traveling Wave Tube 2: Impedance and Attenuation" 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. 4.65, with the *Elements Tree* in the upper center, and the *Property Editor* in the lower center. The right pane shows a 3D view of the geometry as well as the grid, if its visibility has been activated (which it was not when this image was captured).

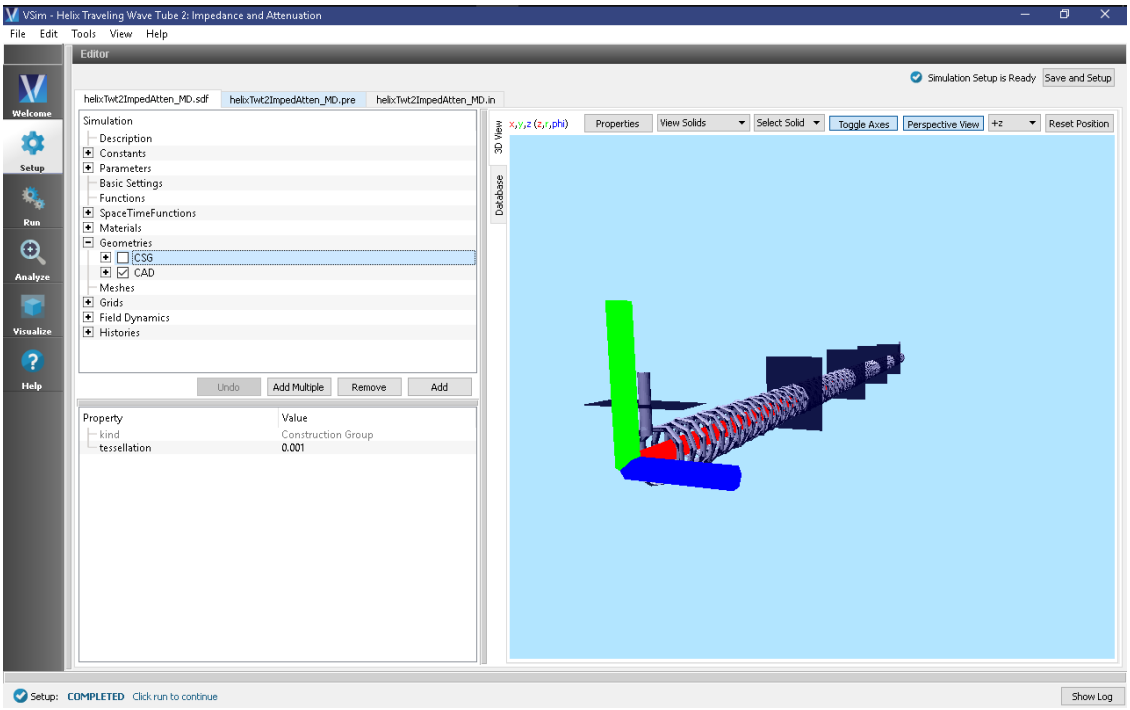


Fig. 4.65: Setup Window for the Helix TWT impedance and attenuation example.

Geometry details

The various geometrical objects can all be seen in the tree by pulling down the bar separating the Elements Tree from the Property Editor and then expanding *Geometries*, *CSG*, and *Grids*. Make sure *tube* and *Grid* are unclicked, *helixWithFeedsGeom* is clicked, and that the *Toggle Axes* button is set to remove the axes from the view. This allows one to see the interior geometrical objects, including the incoming feed, the dielectric support rods, the resistive tubes in the center, the particle emission disk at the left, and various planes where measurements are taken.

Constants and Parameters

Pulling the separator bar between the Elements Tree and the Property Editor and opening the Constants part of the tree gives the view shown in Fig. 4.67.

There are three types of constants. The first set of constants, from `PI` through `ELECMASSEV` are not changeable by the user. These are the various mathematical and physical constants that the simulation will use. The second set of constants are those with `HELIX` in the name. These must correspond to the helix geometry, the beginning, mid-radius, wire radius, pitch, and number of turns of the helix. These cannot be set arbitrarily, as the helix was imported as an STL file. Instead they must be set to match the imported helix parameters. The remaining constants define fundamental geometry quantities, such as where the tube begins and its radius, other physical simulation parameters, such as the wave frequency, and numerical parameters, such as the number of cells in each direction.

Moving the scroll bar and opening the Parameters part of the tree shows the Parameters, values that come from arithmetically combining constants and other parameters. This is shown in Fig. 4.68.

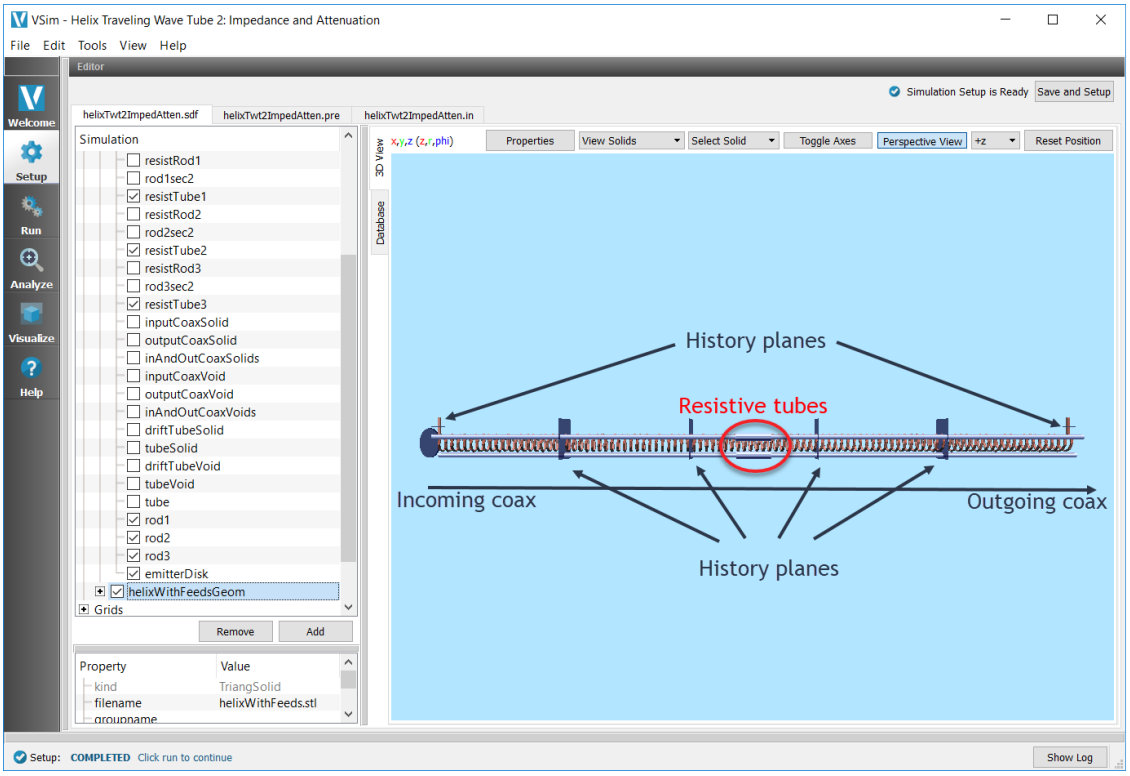


Fig. 4.66: Interior geometry for the Helix TWT impedance and attenuation example.

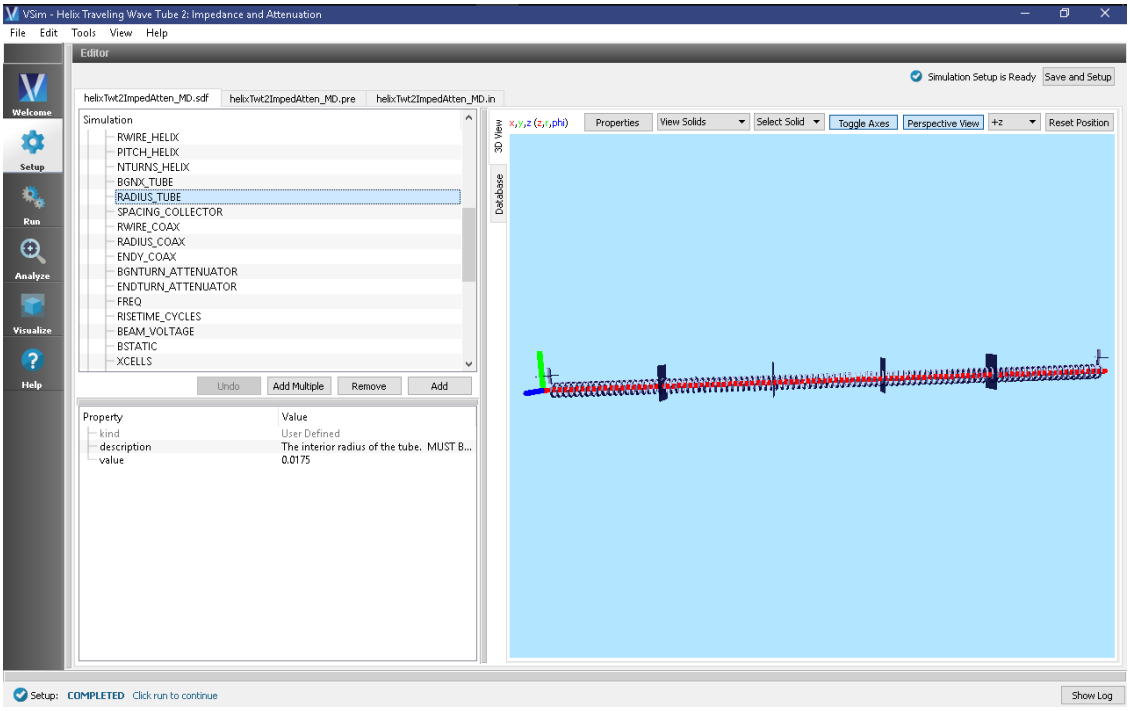


Fig. 4.67: Constants for the Helix TWT impedance and attenuation example.

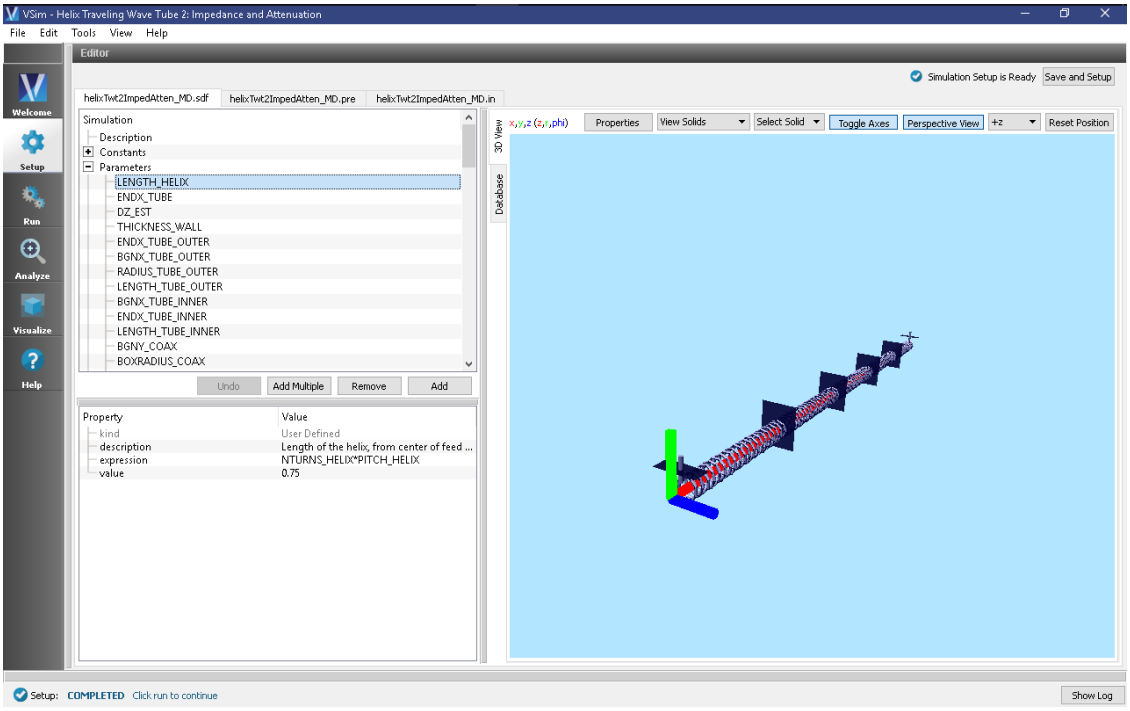


Fig. 4.68: Parameters for the Helix TWT impedance and attenuation example.

As an example, `LENGTH_HELIX` is shown. The expression shows that this is the number of helix turns times the helix pitch. It also shows the value. Of course, the expression is editable, while the value is not.

Both constants and parameters have a *description* field that allows the user to document the quantity.

Materials

To bring materials in the simulation, in the right pane, select the *Database* tab, select one or more materials (with ctrl-click) and hit the button *Add To Simulation*. The materials will then appear under *Materials* in the tree view. At this point one can change the properties of the materials including the name. In this example we imported *resistive damper*, changed its name to *LossyMaterial*, and changed its value of conductivity to *0.55*. This is shown in Fig. 4.69.

Once one has materials in the simulation, one can set the materials of any of the geometries. Click on the geometry, then in the Property Editor pane, double click on the material value. A context menu will allow you to set the material of the geometry to any material in 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.
- To run the file, click on the *Run* button in the upper left corner of the right panel. 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 is shown in Fig. 4.70.

The simulation to determine the impedance should run long for any mismatch at the outgoing boundary to stabilize. That is, the simulation must be run long enough for the electromagnetic wave to reach the far end of the tube and for any reflections to return some distance to the last history in the tube. This will take about 100,000 steps. On a 4-core

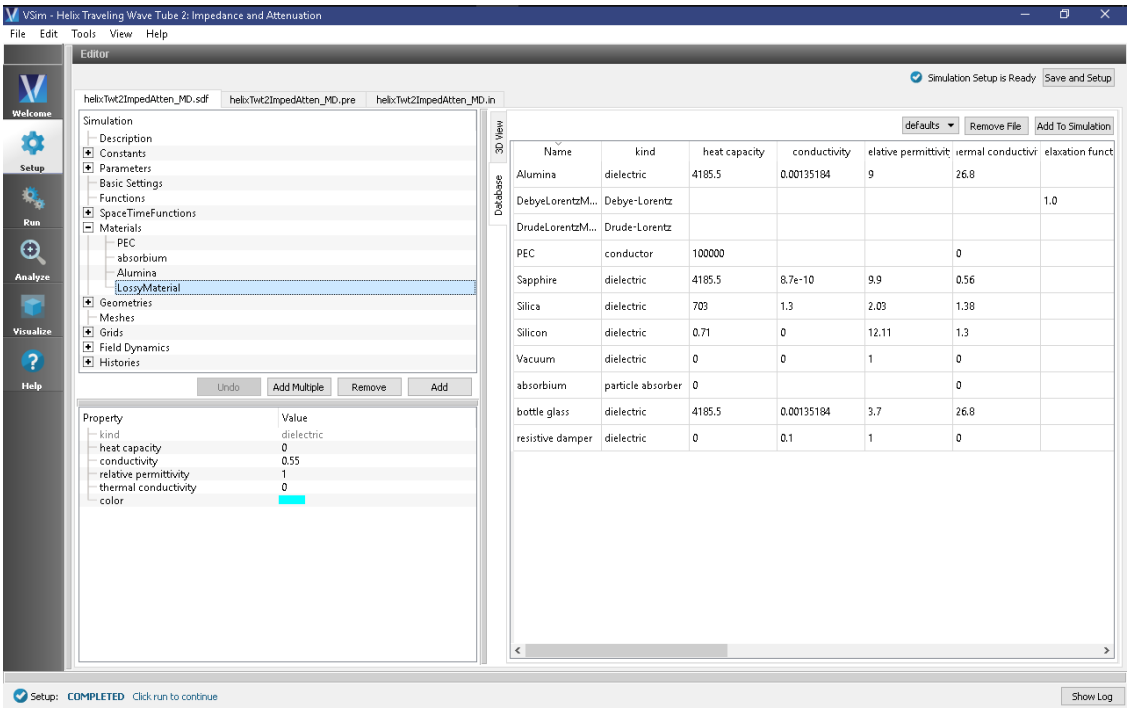


Fig. 4.69: Materials for the Helix TWT impedance and attenuation example.

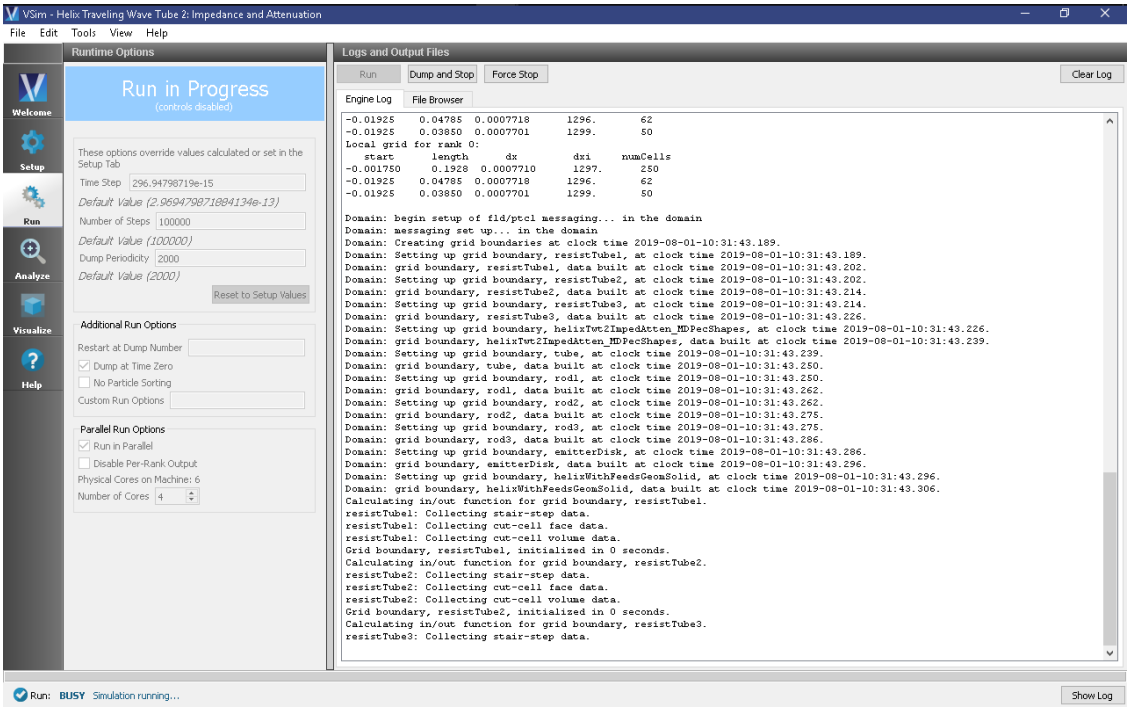


Fig. 4.70: The Run Window at the end of execution.

machine, we have observed 0.23s/step, so this simulation will take about 7 hours. This simulation parallelizes well up to 16 cores, so with a sufficient license and workstation or cluster, one can bring this simulation time down to about 2 hours.

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 fields inside the tube as shown in Fig. 4.71, continue as follows:

- Select Data View: *Data Overview*
- Expand *Scalar Data* then *D*, then select field *D_y*.
- Expand *Geometries* then select *poly (rod1)*.
- Check *Clip All Plots*, which cuts through the data at the $z = 0$ plane.
- Select the *Colors* option from the upper menu bar in the *Visualization Results* pane.
- In the *Color Options* window: check *Fix Minimum* and set it to -400, then check *Fix Maximum*, set it to 400, and click *OK*.
- Move the dump slider forward in time to see the evolution.

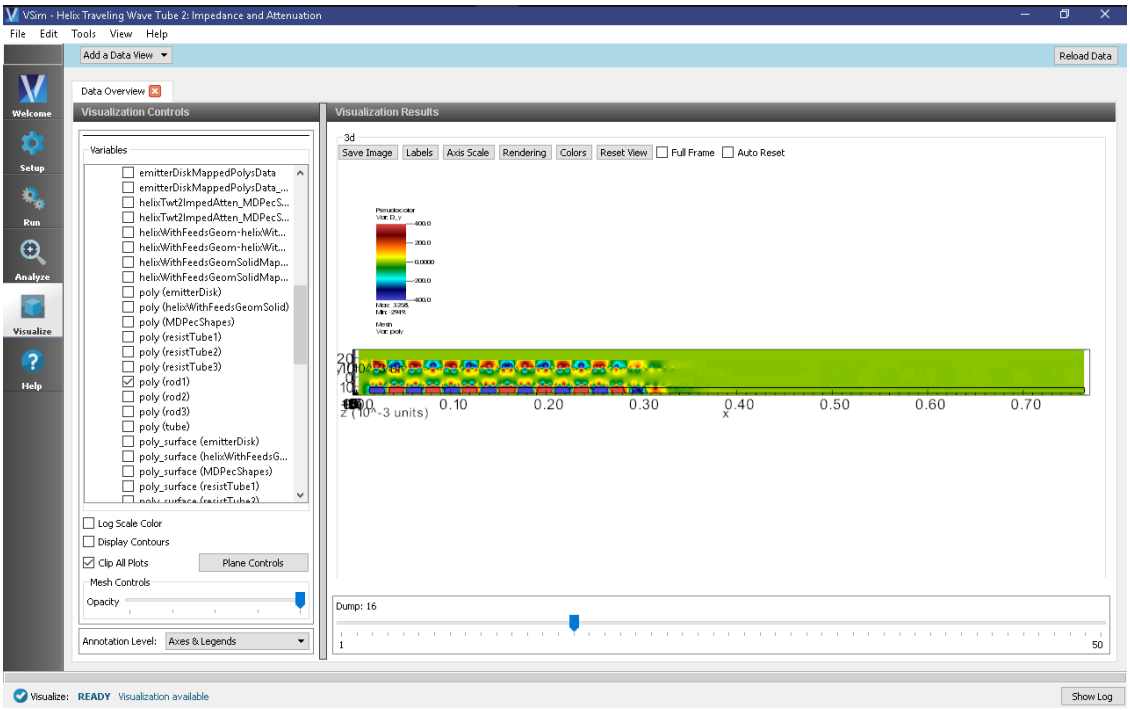


Fig. 4.71: The transverse displacement field, Dy, on the central x-y plane at dump 31.

This plot shows the transverse displacement field. Once can see that it is confined inside the tube (sanity check), and that it is most intense inside the dielectric rod at the bottom. The field is larger at the left, as it is just entering and propagating down the tube.

At any time one can leave this visualization pane to move back to the run pane to see how the simulation is progressing.

The longitudinal field inside the tube is shown in Fig. 4.72, which can be obtained by the steps:

- Select Data View: *Field Analysis*
- Select Field *E_x*.
- Select *Horizontal* under *Lineout Settings*, set the Intercept to `0`, and click *Perform Lineout*.
- Under *Layout* select *Stacked 2d/1d*
- Move the dump slider forward in time to see the evolution.

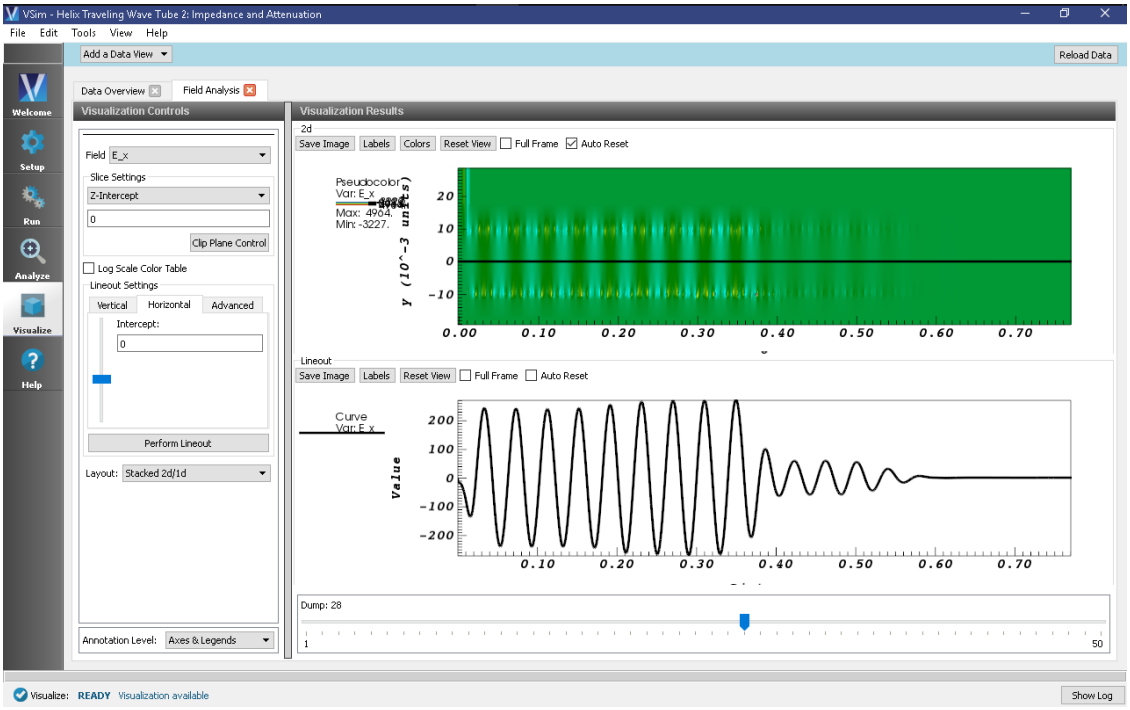


Fig. 4.72: The longitudinal field, Ex, on axis at dump 27, which is time step 54,000.

As seen in Fig. 4.72, the longitudinal field has dropped from about `240 V/m` to about `68 V/m` in the center of the simulation where the resistive tubes are. This corresponds to about `14 dB` of attenuation. The purpose of this attenuation is to have sufficient damping so that reflections coming back from the end to the beginning and then reflection again do not grow, as that would change the device into an oscillator, with energy growth that could destroy the system. If the *Helix Traveling Wave Tube 3: Power Run (helixTwt3PowerRun.sdf)* shows that this is happening, one can return to this run and increase the conductivity of the LossyMaterial or the length of the resistive tube (through `BGNTURN_ATTENUATOR` and `ENDTURN_ATTENUATOR`) to provide more attenuation.

Histories contain the time evolution of quantities defined in the input file. These can be seen by selecting the *Data View*, *History*. To determine various impedances we want particular histories obtained by the process:

- Select Data View: *History*
- Under Graph 1 select *poyntingA*
- Under Graph 2 select *transverseVoltageA*
- Under Graph 3 select *EonAxisA_0*
- Under Graph 4 select *<None>*

The result is shown in Fig. 4.73. The power through the plane was defined such that incoming is negative.

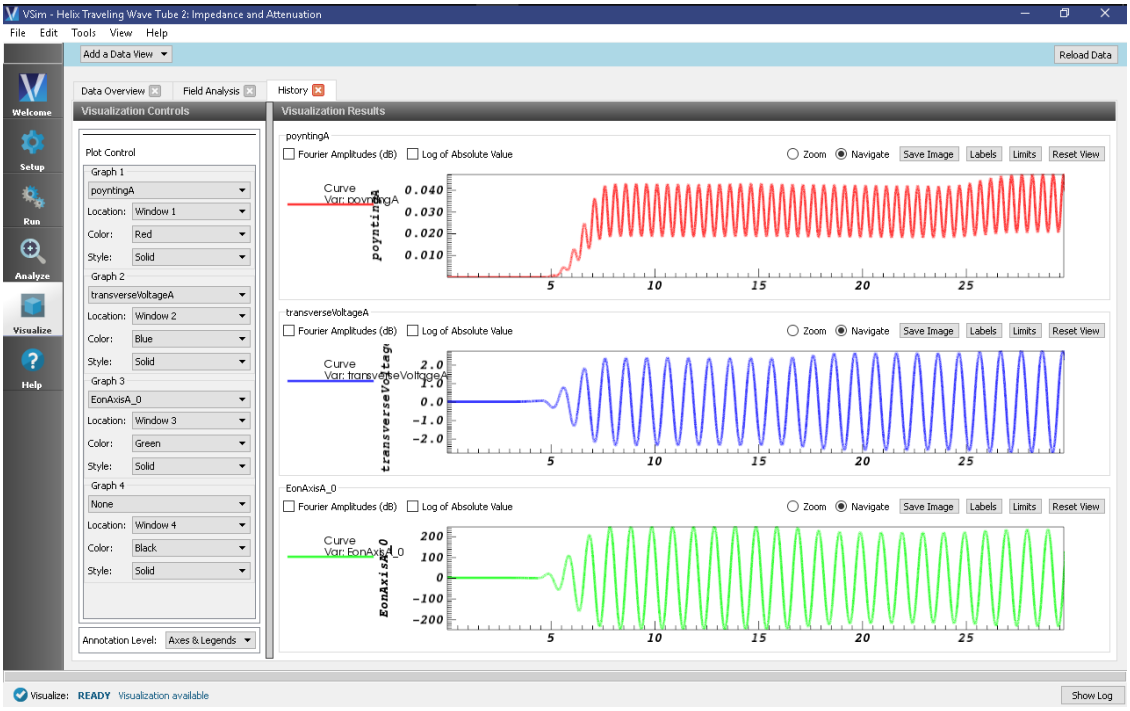


Fig. 4.73: Poynting power (W), transverse voltage (V), and electric field (V/m), at Plane A, along the helix TWT as a function of time (s).

Impedance parameters of interest are the transverse impedance

$$Z_t = \frac{V_t^2}{2P}$$

and the Pierce interaction impedance

$$Z_p = \frac{E_x^2 \lambda^2}{8\pi^2 P}$$

where P is the poynting power (recorded in the history poyntingA), V_t is the transverse Voltage amplitude (recorded in the histories transverseVoltage), E_x is the electric field amplitude (recorded in the history EonAxis), and λ is the wavelength of the field along the helix TWT axis.

The histories show the graphs of these quantities. To get precise values for any of these, one can press the *Limits* button, which will pop up a window with the precise values. First, the *X-Axis* limits show that the units are ns. Secondly, one needs to choose a consistent time period, where all amplitudes are roughly constants. The period $28ns < t < 32ns$ is chosen. One can now adjust the limits until the peaks line up with the limits. We want average poynting power. We find $P_{min} = 18.\ mW$ and $P_{max} = 44.\ mW$. Hence, $P_{av} = 31.\ mW$. During that same time interval we find $V_t = 2.4\ V$ and $E_x = 240\ V/m$.

Fig. 4.72 can be used to obtain the wavelength. One can see four wavelengths between $0.20m$ and $0.357m$. Therefore the wavelength is $(.357m - .20m)/4 = 0.039m$

We now compute

$$Z_t = \frac{2.4 * 2.4}{2 * .031} = 92.9\ \Omega$$

and the Pierce interaction impedance is

$$Z_p = \frac{240^2 \cdot 0.039^2}{8\pi 0.031} = 35.8\ \Omega.$$

Further Experiments

As noted above, one can change the attenuation by varying the conductivity of the resistive tubs or their length. For any given length, there is a maximum attainable attenuation, as there is no attenuation at zero conductivity (infinite resistance, i.e., vacuum) and none as well at infinite conductivity (metallic shielding). So if more than 14 dB attenuation is needed one can vary the conductivity, but a maximum will be observed, and if that is insufficient one will have to vary the rod length.

With additional computing resources, one could increase the grid resolution so that the resistive tube could be made thinner. As it is made thinner, one can go to greater conductivity without having the skin depth less than the resistive tube thickness.

The coupling is determined by the transverse impedance of the structure, which in turn depends on the capacitance provided by the rods. Varying the relative permittivity changes the transverse impedance.

4.3.8 Helix Traveling Wave Tube 3: Power Run (helixTwt3PowerRun.sdf)

Keywords:

`Helix TWT Power Run`

Problem description

This VSimMD example is the last of a set of simulations showing different calculations to aid the design of a helix traveling wave tube (TWT) in three dimensions. The 100-turn helix with end feeds is imported from a CAD file, but all other geometrical parts are created with the Constructive Solid Geometry (CSG) capabilities within VSimComposer.

An input signal is sent into a short section of coaxial input waveguide and a similar section of coaxial waveguide provides an output power port. The geometries in and constant parameters of this simulation are described in more detail in *Helix Traveling Wave Tube 2: Impedance and Attenuation (helixTwt2ImpedAtten.sdf)*. Electrons are injected at the left end of the tube. Gain can be computed from the ratio of voltages in the input and output waveguides.

The user may wish to run this simulation type multiple times, varying parameters such as the input signal power and the electron energy, in order to result in a design with maximum output power.

Related simulations:

This simulation can be performed with a VSimMD or VSimPD license.

Opening the Simulation

The Helix TWT Power Run example is accessed from within VSimComposer by the following actions:

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim for Microwave Devices* option.
- Expand the *Radiation Generation* option.
- Select "Helix Travelling Wave Tube 3: Power Run" 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. 4.74. You can expand the tree elements and navigate through the various properties. Some of these changes will affect the geometry, and so one should review the look of the geometry in the right pane as one changed geometrical variables. To show or hide the grid, expand the Grid element and select or deselect the box next to `Grid`. One can, e.g., hide the tube to see inside it.

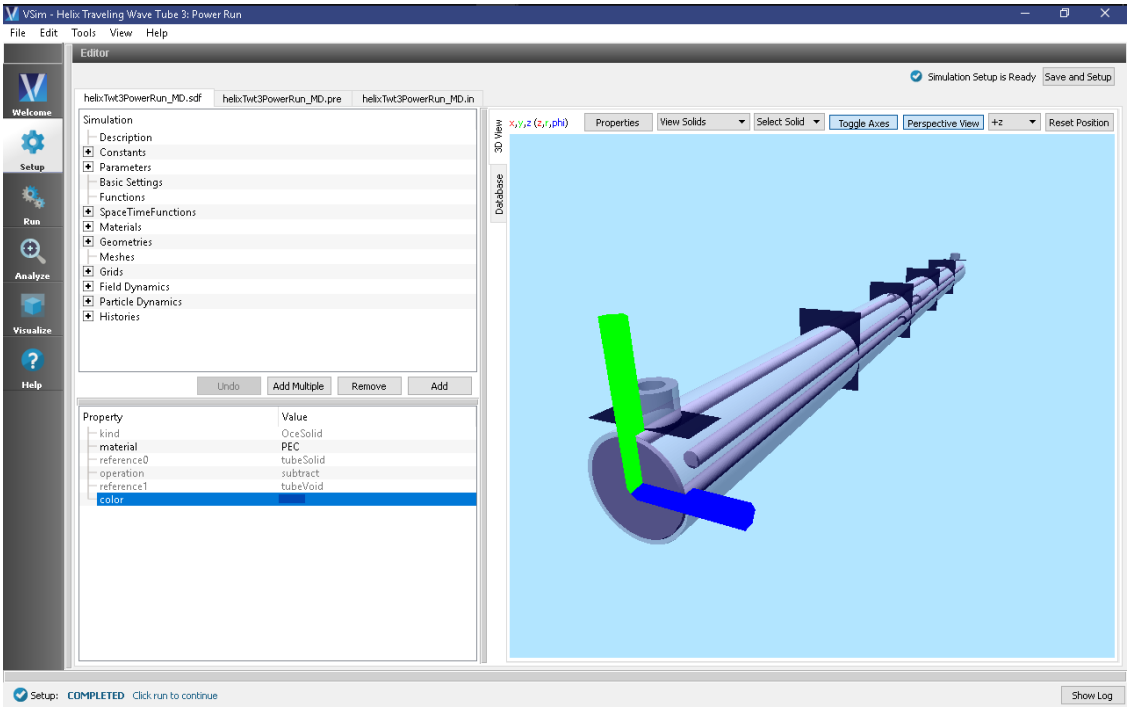


Fig. 4.74: Setup Window for the Helix TWT example.

The geometry of the helix can be made more visible by unselecting the tube and emitter disk parts under Geometries/CSG in the tree (e.g. driftTubeSolid, tubeSolid, driftTubeVoid, tubeVoid, tube, and emitterDisk) or by changing their color property and selecting a low alpha on Windows or Linux (or opacity on Mac). See color property setting in the User Guide.

Additional detail of the geometry is shown in figure Fig. 4.75. Fig. 4.74 shows the dielectric rods and Fig. 4.75 shows how the coaxial waveguide connects to the helical wire.

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.
- To run the file, click on the *Run* button in the upper left corner of the right panel. 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 is shown in Fig. 4.76.

The default number of time steps will run the simulation long enough to verify that the electron beam is traveling down the tube, that the input signal has entering the simulation and propagated down the tube, that the amplified signal is leaving the system, and that the amplification has reached a steady state. However, the simulation has not been run long enough to ensure that there are no deleterious, backward wave oscillations. To determine that, one should run the simulation twice as long (ensuring a backward and forward traversal) or more, depending on the growth rate of the oscillation.

Visualizing the results

After performing the above actions, continue as follows:

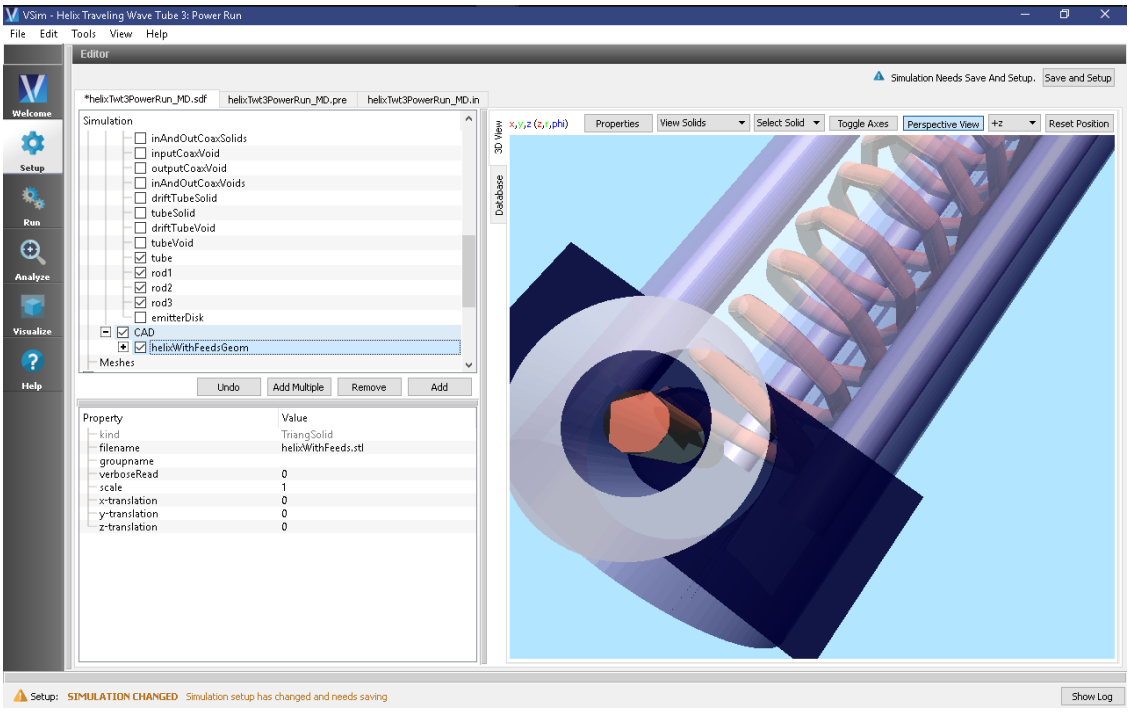


Fig. 4.75: A view in through the input coax.

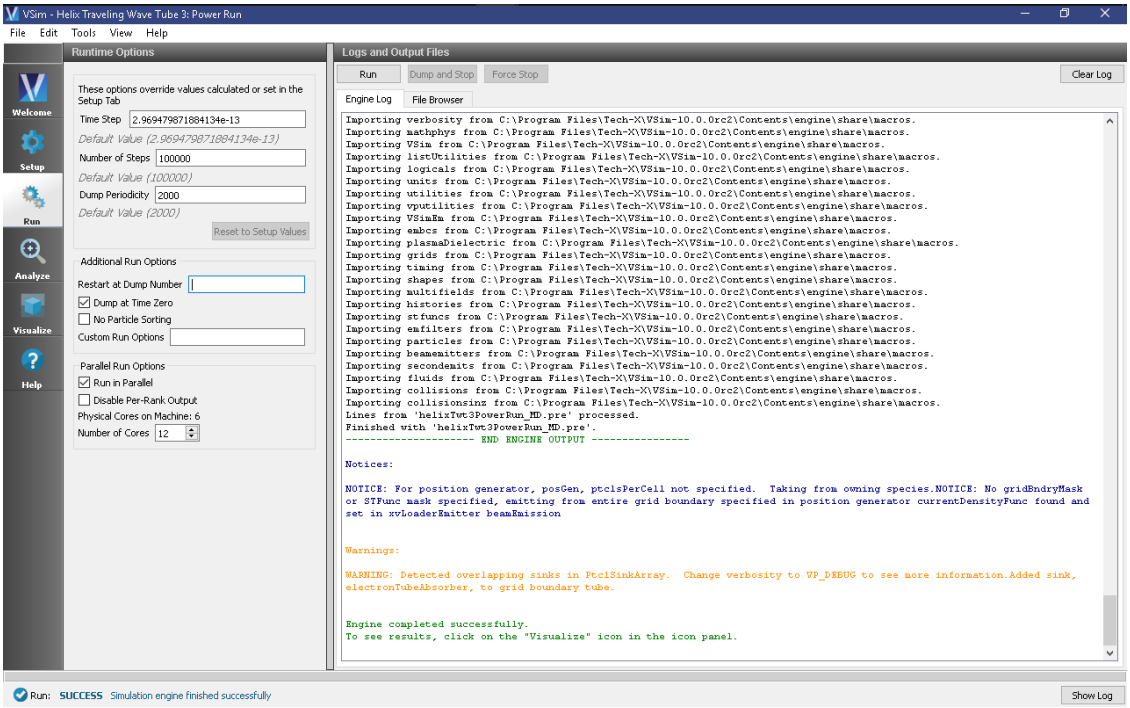


Fig. 4.76: The Run Window at the end of execution.

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

The particle phase space, Fig. 4.77, shows how the energy is being extracted from electron beam. To generate this image:

- For *Data View* select *Phase Space*.
- Set *X-axis* to electron_x.
- Set *Y-axis* to electron_ux.
- Press *Draw*.
- Move the dump slider forward in time to see the evolution
- The image is at dump 37.

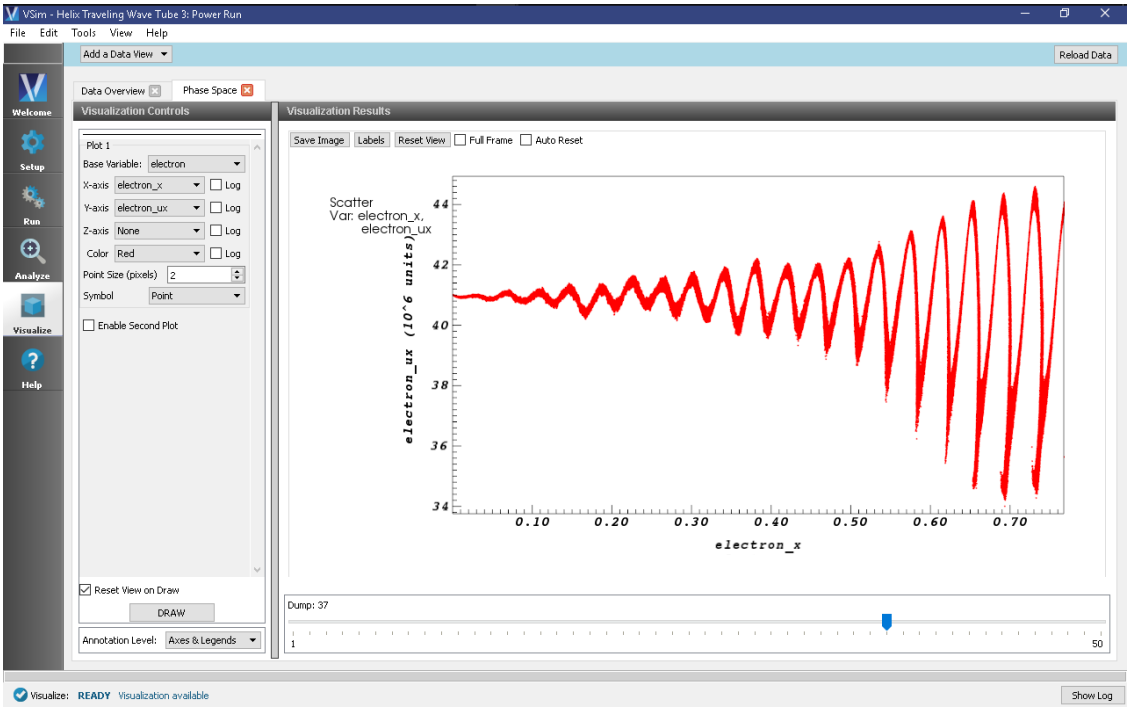


Fig. 4.77: Longitudinal phase space of the electron beam.

One can see in Fig. 4.77 that the beam has been overdriven, such that trapping is beginning to occur. Hence, one must either reduce the input power or one must reduce the gain. In the middle of the tube one can see that the beam oscillation for one cycle decays a bit before taking off again. This is where the attenuator is located.

The effect of overdriving the tube can also be seen in the longitudinal field, Fig. 4.78. This image is obtained by

- For *Data View* select *Field Analysis*.
- Set *Field* to E_x.
- For *Lineout Settings*, choose *Horizontal* with Intercept of 0.
- For *Layout* select Stacked 2d/1d.
- Press *Perform Lineout*.
- Click the *Colors* button and set the limits to Minimum = -2e3 and Maximum = 2e3.
- Move the dump slider forward in time to see the evolution.

- The image is at dump 37.

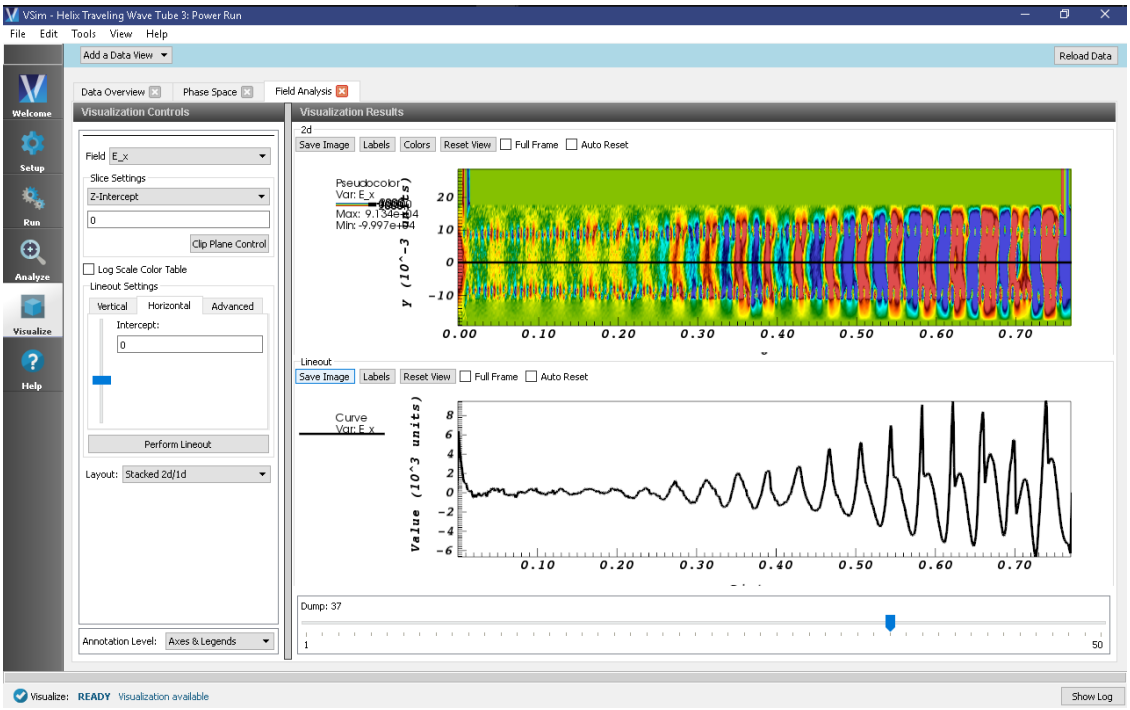


Fig. 4.78: Longitudinal electric field in the center of the tube.

As expected, the longitudinal field is largely confined within the helix. Again, at around x=0.4, one sees the field being damped out by the attenuator. Because the tube has been overdriven, harmonics are appearing in the field at the right. This image shows that the harmonics occur at about 1/5 of the current output power, indicating the amount by which one should decrease the input power or the gain to obtain linear operation.

The gain can be seen in the History records, which are available under the *History* Data View. A sample of these is shown in Fig. 4.79.

To obtain this history image:

- For *Data View* select *History*.
- For *Graph 1* select inputVoltage.
- For *Graph 2* select outputVoltage.
- For *Graph 3* and *Graph 4* select <None>

This image shows that the voltage gains is about a factor of 10 or 20 dB. The voltage history also shows the harmonics in the output that come from overdriving the tube.

Further Experiments

As noted at the beginning, this run could be run for many more time steps to determine whether backward oscillations are present. Additionally, one can experiment with the beam energy to determine what energy gives the most gain. Varying the input power can determine the maximum output power, which happens when the beam begins trapping at the end of the tube, or the input power at which one obtains large gain while remaining in the linear regime.

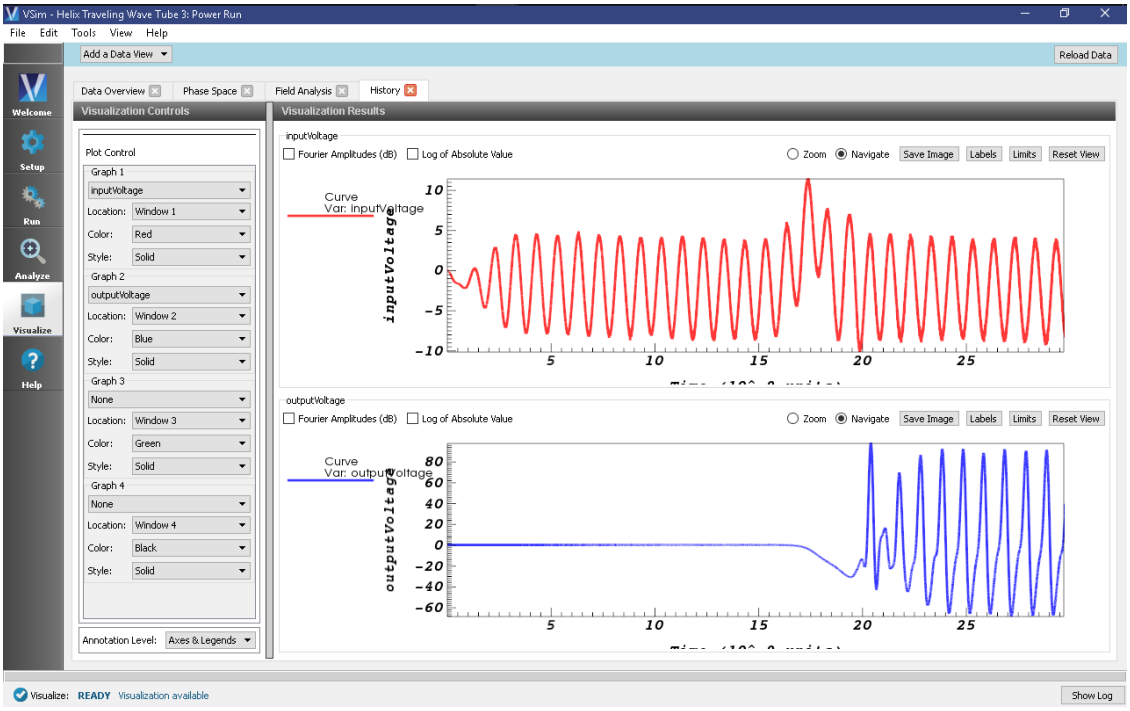


Fig. 4.79: Input and output voltage histories.

4.3.9 Klystron (klystron.sdf)

Keywords:

klystron

Problem description

This VSimMD example simulates a two cavity klystron in three dimensions. The first cavity is driven at its lowest resonant frequency. The resultant cavity voltage creates a velocity modulation in the electron beam which translates to charge modulation as the beam travels in the tube between cavities. The charge modulation then drives the second cavity. The cavities are loaded to give them finite Q.

This simulation can be performed with a VSimMD or VSimPD license.

Opening the Simulation

The Klystron example is accessed from within VSimComposer by the following actions:

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim for Microwave Devices* option.
- Expand the *Radiation Generation* option.
- Select *Klystron* 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. 4.80. 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*.

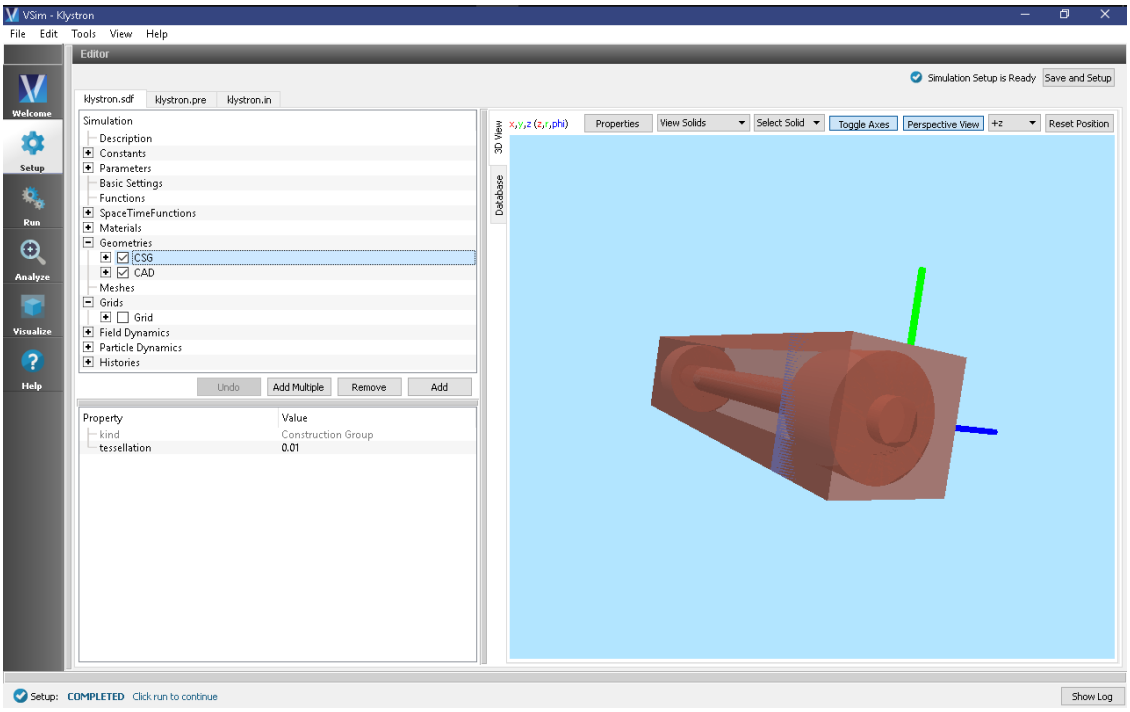


Fig. 4.80: Setup Window for the Klystron example.

This example illustrates two methods for generating geometries. Under *Geometries* in the elements tree there are two paths, *CSG* and *load1Geom*. The CSG components are constructed from primitives within VSim. The *load1Geom* was imported as an STL file. Highlighting any geometry under the CSG group shows how it was created, either as a primitive with parameters or by operations on other shapes.

Simulation Properties

This simulation is set up to do a Power Run with full capabilities. After completing the Power Run and visualizing the results, you may wish to refine the performance of the klystron by adjusting the setup. Some useful tuning procedures are described in the **Further Experiments** section. These include the **Resonant Frequency Run** and the **Attenuation Calibration Run**.

Some constants that you may wish to modify include:

FREQUENCY:The frequency (in Hz) at which the signal across cavity is driven. This can be tuned to the resonant frequency once determined.

BEAMRADIUS: The radius of the emitted beam of electrons into the klystron.

BEAMCURRENT: The current of the electron beam.

Running the Simulation

With the default setup, complete the Power Run with the following steps:

- Proceed to the Run window by pressing the *Run* button in the left column.

- To run the file, 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 is shown in Fig. 4.81 below.

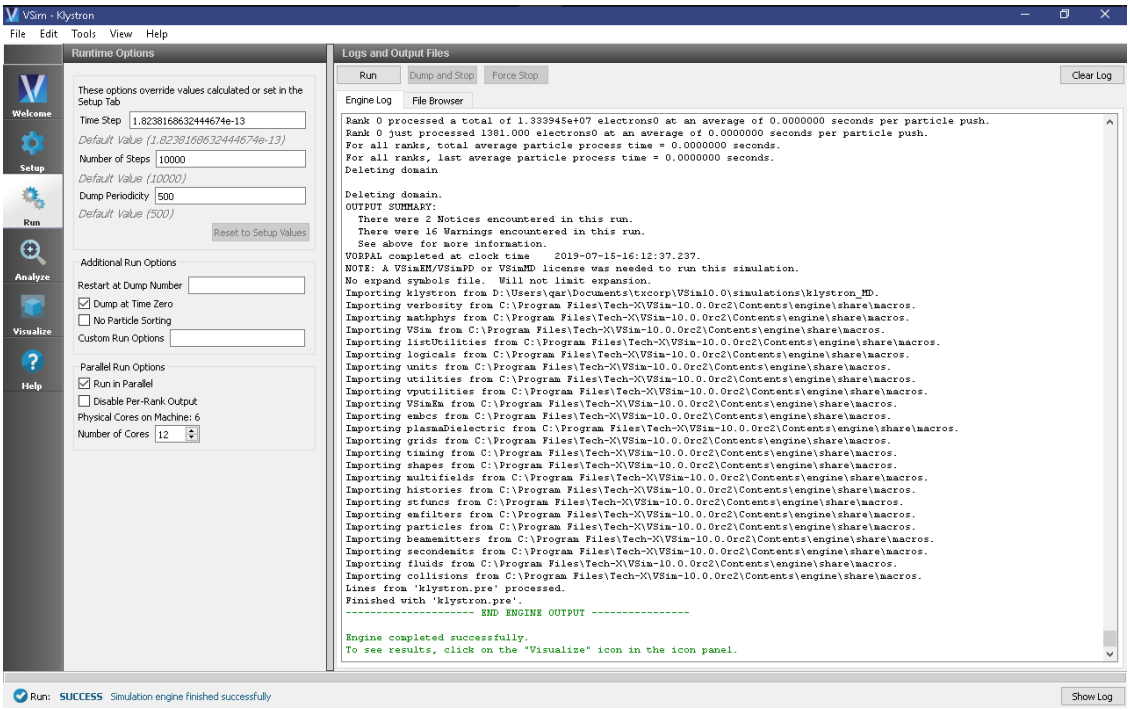


Fig. 4.81: The Run Window at the end of execution.

Visualizing the Results

After the simulation run has completed successfully, you may proceed to the Visualize Window by pressing the *Visualize* button in the left column. To reproduce Fig. 4.82 follow these steps:

- Select *Data Overview* from the *Data View* pull-down menu.
- Expand *Particle Data*
- Expand *electrons0*
- Select *electrons0_ux*
- Expand *Geometries*
- Select *poly_surface (klystronPecShapes)*
- Select *Clip All Plots*
- Step through time using the dump slider on the bottom of the right pane.

Further Experiments

The **Attenuation Calibration Run** and **Resonant Frequency Run** are outlined below. These experiments will allow you to tune the klystron. You may want to iterate through these experiments to get the desired performance. Once the cavity performance is satisfactory you can repeat the Power Run to see the effects on the electrons. To see the full

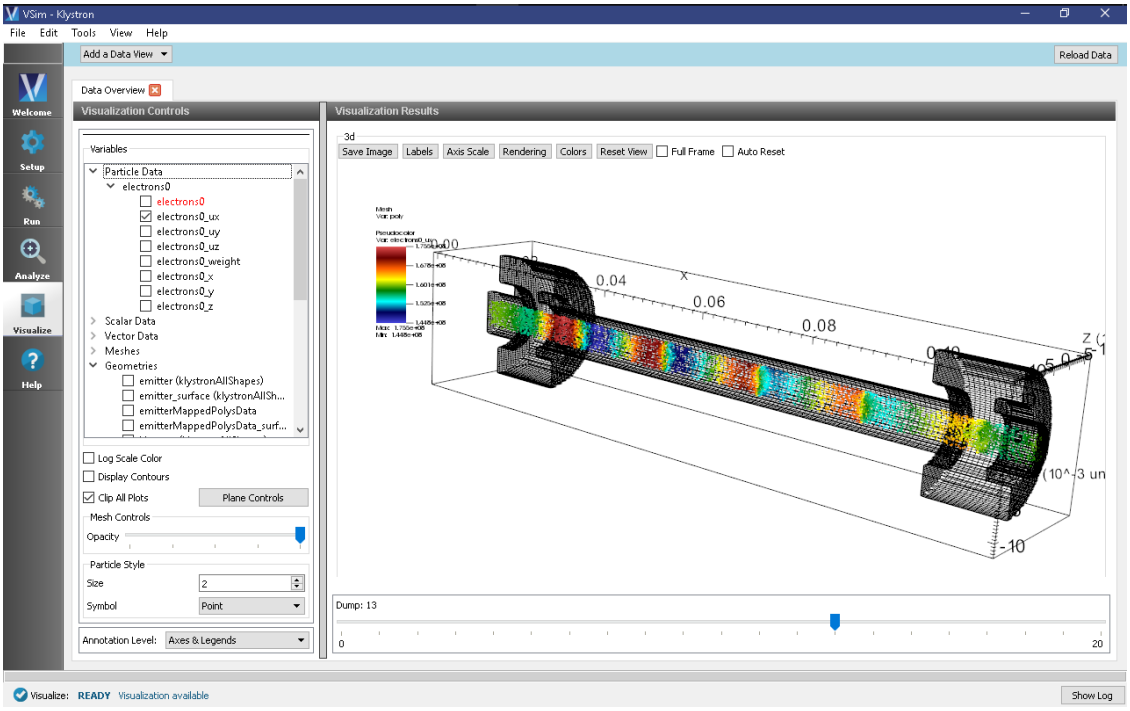


Fig. 4.82: A power run with an electron beam.

behavior of the device, increase the number of steps to (5 x Default). This will allow you to see the saturation of the second cavity.

Resonant Frequency Run

To determine the resonant frequency of the first cavity we will analyze the fourier transform of its voltage. In the Setup window, under *Basic Settings*, set *particles* to *no particles*. Then, under *SpaceTimeFunctions*, in *ring1J* change "rampUp" to "rampUpAndDown". This will ping Cavity 1 and allow us to observe the ringing signal. Run the simulation with this setup.

To determine the resonant frequency proceed to the Visualize window. Select *History* from the *Data View* pull-down menu. Click FFT to the left of the Cavity1Voltage plot in the *Visualization Results* pane. The resulting plot will resemble Fig. 4.83. Zoom in on the maximum of this plot to determine the resonant frequency. You can now update the FREQUENCY under *Constants* in the Setup window with this new value and use it to drive future simulations.

Attenuation Calibration Run

The user can integrate this run type in order to calibrate the observed attenuation to the desired loss. The attenuation can be tuned by modifying the *conductivity* of the material, *resistive damper*. The Q of the cavity can be computed using a feature of the Analysis Tab, as described below.

For the **Attenuation Calibration Run**, use the same Setup as the **Resonant Frequency Run**. After running the simulation, the quality factors, Q_1 and Q_2, for cavities 1 and 2 can be calculated using the *computeInverseQ.py* script in the Analyze window.

- Press the Analyze button in the left column of buttons.
- Select the *computeInverseQ.py* analyzer, then *Open*.
- Enter Cavity1Voltage or Cavity2Voltage in the *historyName* field to designate the history to analyze.
- Enter the value of the FREQUENCY constant as defined in the Setup window in the *frequency* field to designate the frequency at which the history will be analyzed.

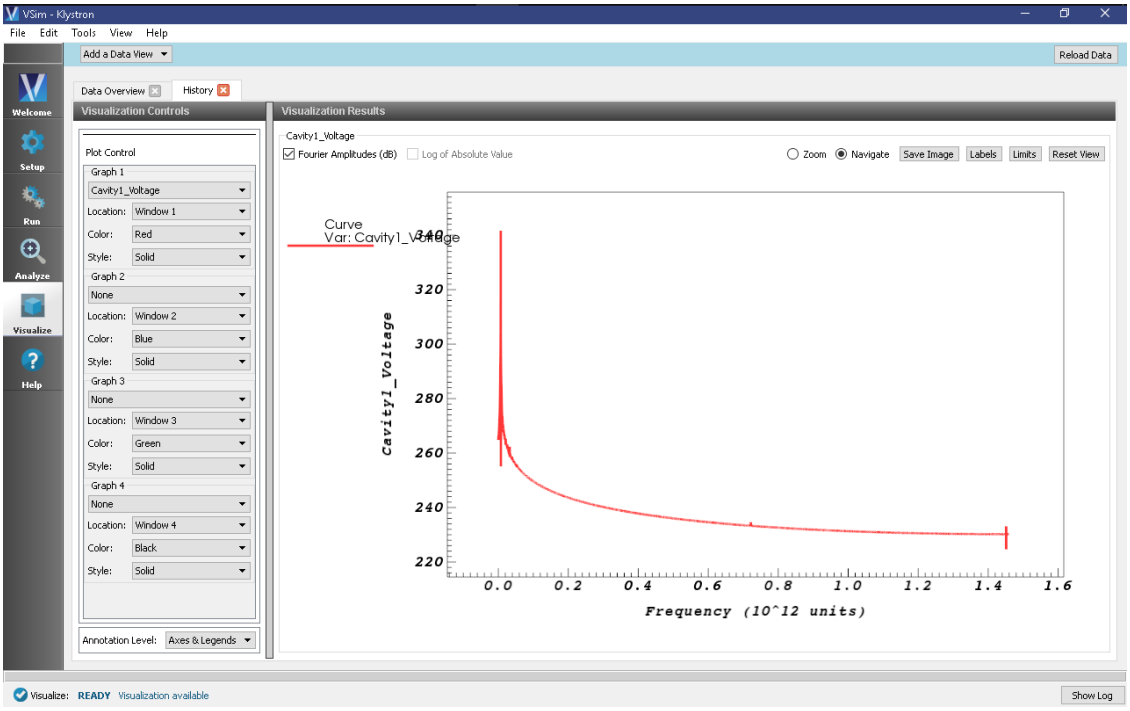


Fig. 4.83: Fourier transform of Cavity1_Voltage versus time (in GHz).

- Update the *outputFileName* field if desired
- Click the *Analyze* button in the top right corner of the window. As shown in Fig. 4.84 below. Two columns of data with the titles "Time (s)" and "Inverse Q" will be output in the right pane. The analysis has completed when you see the output "Analysis completed successfully."

Scrolling through or plotting the output data in the Visualize window enables the user to understand the Klystron's performance. The user may iterate this run type to achieve the desired attenuation.

4.3.10 2D Magnetron (magnetron2D.sdf)

Keywords:

`magnetron`

Problem description

This VSimMD example simulates a rising sun magnetron in two dimensions. A load is added to one cavity, representing a coupler to the magnetron through the quality factor, Q. Upon configuring an electrostatic voltage across the anode and cathode, particles are introduced to the simulation, exhibiting a five spoke pi-mode.

This simulation can be performed with a VSimMD or VSimPD license.

Opening the Simulation

The 2D Magnetron example is accessed from within VSimComposer by the following actions:

- Select the *New* → *From Example...* menu item in the *File* menu.

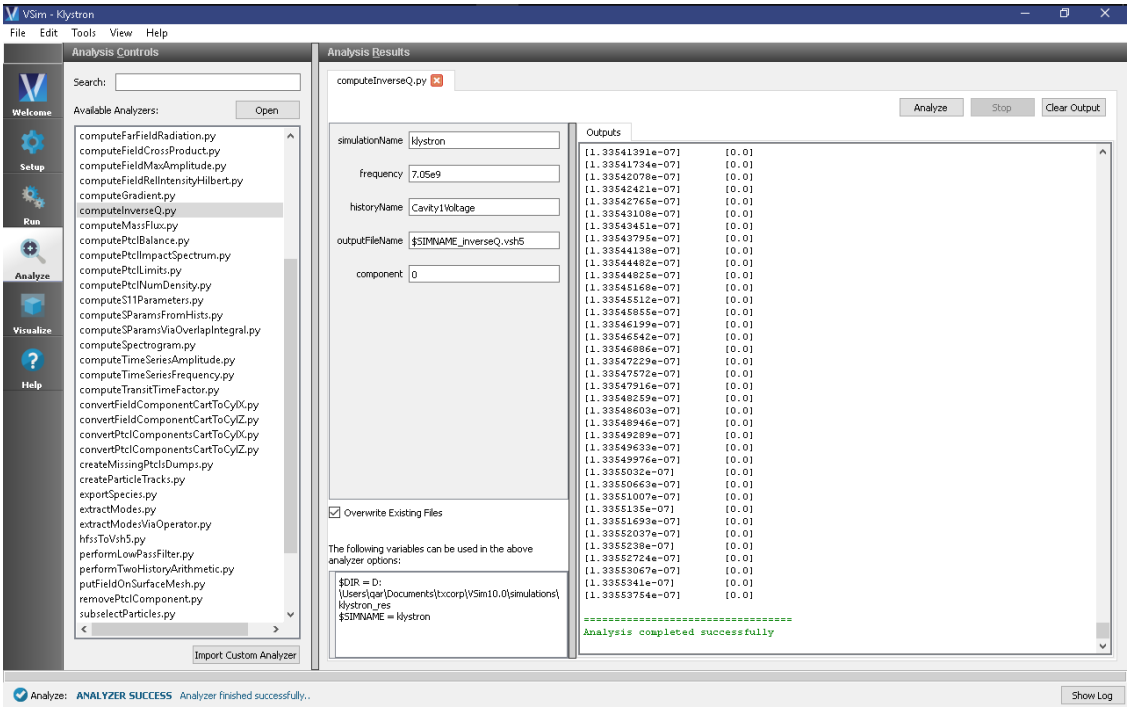


Fig. 4.84: The Analysis window at the end of execution of the computeInverseQ.py script.

- In the resulting *Examples* window expand the *VSim for Microwave Devices* option.
- Expand the *Radiation Generation* option.
- Select "2D Magnetron" 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. 4.85.

Simulation Properties

As seen in Fig. 4.86 of the rising sun magnetron, the radius of the cathode is RCATHODE and the radius of the anode is RANODE. Long cavities have radius RCAVITY1 and opening angle ANGLECAVITY1. Short cavities have radius RCAVITY2 and opening angle ANGLECAVITY2. These dimensions control the spectrum and thus the operating frequency, which for the default parameters is approximately 960 MHz.

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.
- To run the file, 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 is shown in Fig. 4.87 below.

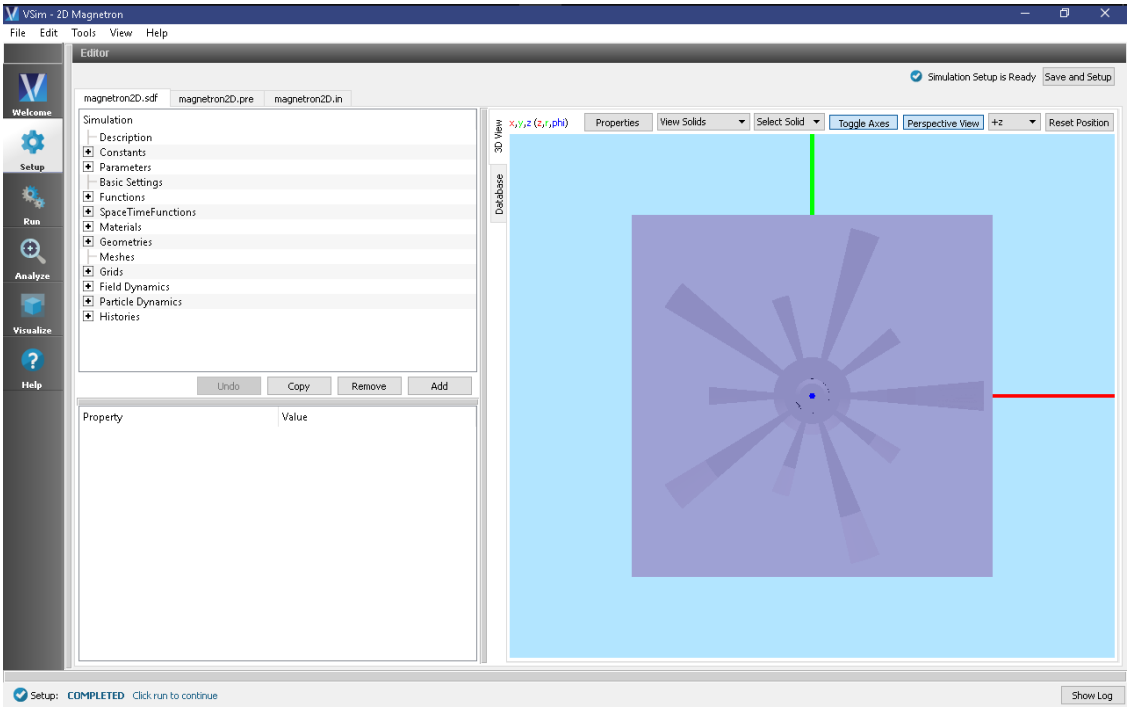


Fig. 4.85: Setup Window for the 2D Magnetron example.

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.
- Expand *Particle Data*
- Expand *electrons*
- Select *electrons*
- Expand *Geometries*
- Select *poly (magnetron2DGeomSolid)*

The electron modes can be viewed in the right pane. Use the dump slider on the bottom of the right pane to step through time. When electrons are emitted from the cathode, the four spoke, 650 MHz is present during startup. At approximately 250 ns, the five spoke begins to dominate and eventually appears as seen in Fig. 4.88.

4.4 Multipacting

4.4.1 Multipacting Growth in Waveguide (multipactingGrowth.sdf)

Keywords:

`multipacting`

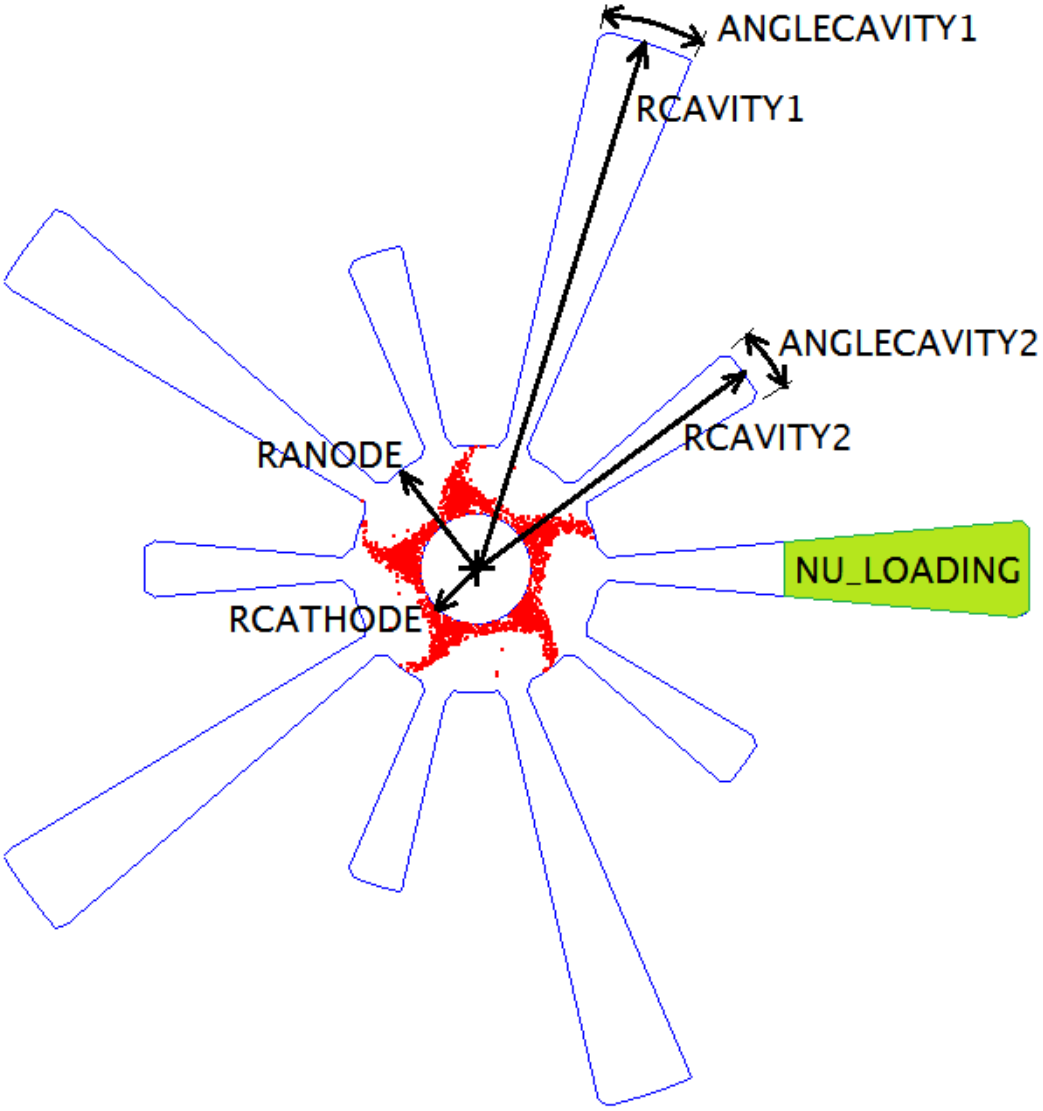


Fig. 4.86: Some exposed variables of the 2D Magnetron example.

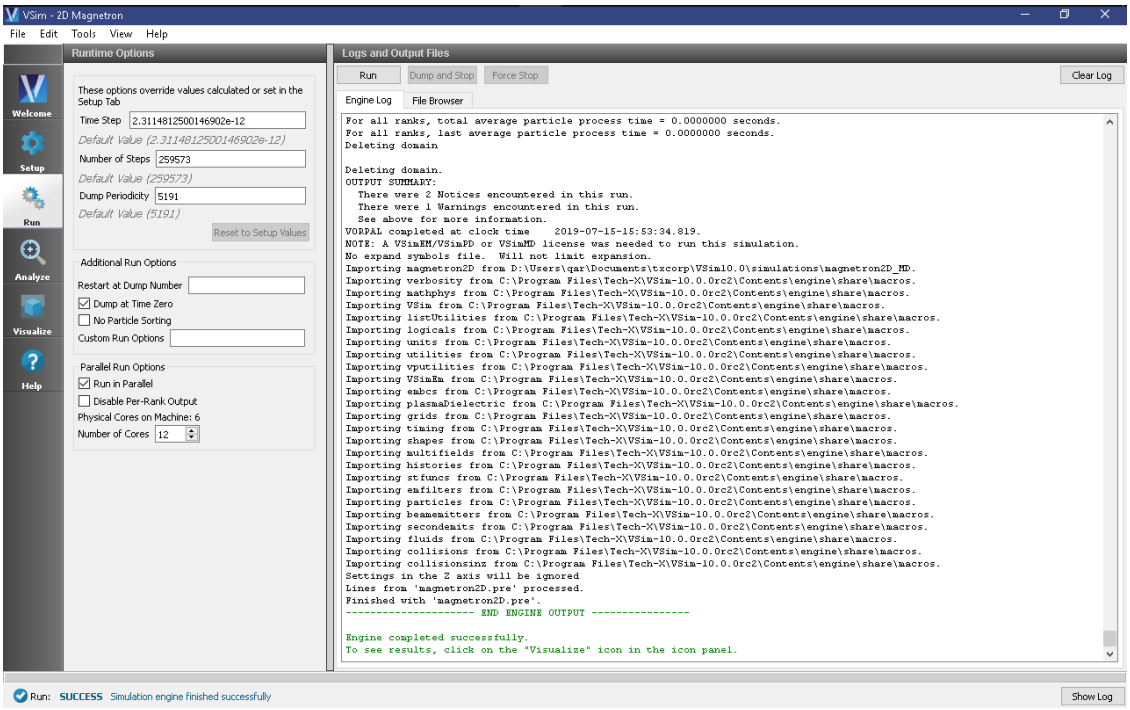


Fig. 4.87: The Run Window at the end of execution.

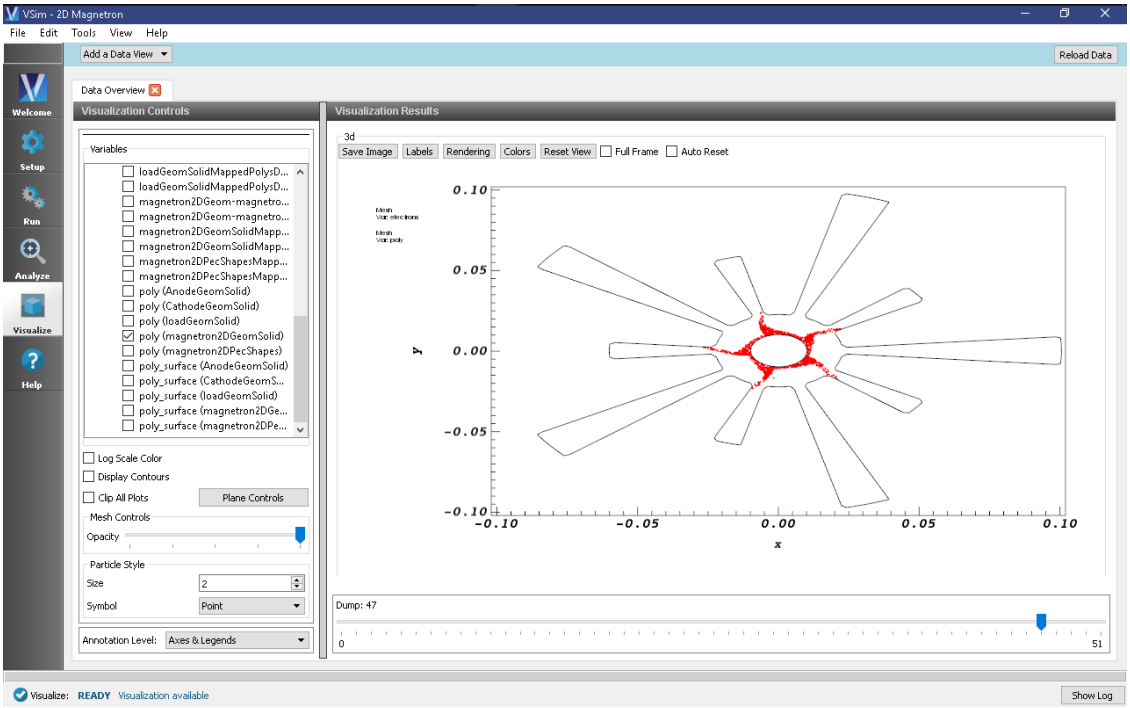


Fig. 4.88: The five spoke pi-mode at 600 ns.

Problem description

Multipacting, which is the resonant buildup of secondary electrons, is often a concern in microwave devices. Anytime there is an oscillating electromagnetic field across a gap between two surfaces there exists the possibility that for the right voltage across the gap a resonance condition will exist allowing the exponential buildup of secondary electrons. A coaxial waveguide is such a type of structure where these conditions can exist.

This simulation can be performed with the VSimMD or VSimPD license.

Opening the Simulation

The Multipacting Growth example is accessed from within VSimComposer by the following actions:

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim for Microwave Devices* option.
- Expand the *Multipacting* option.
- Select "Multipacting Growth in 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. 4.89. 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`.

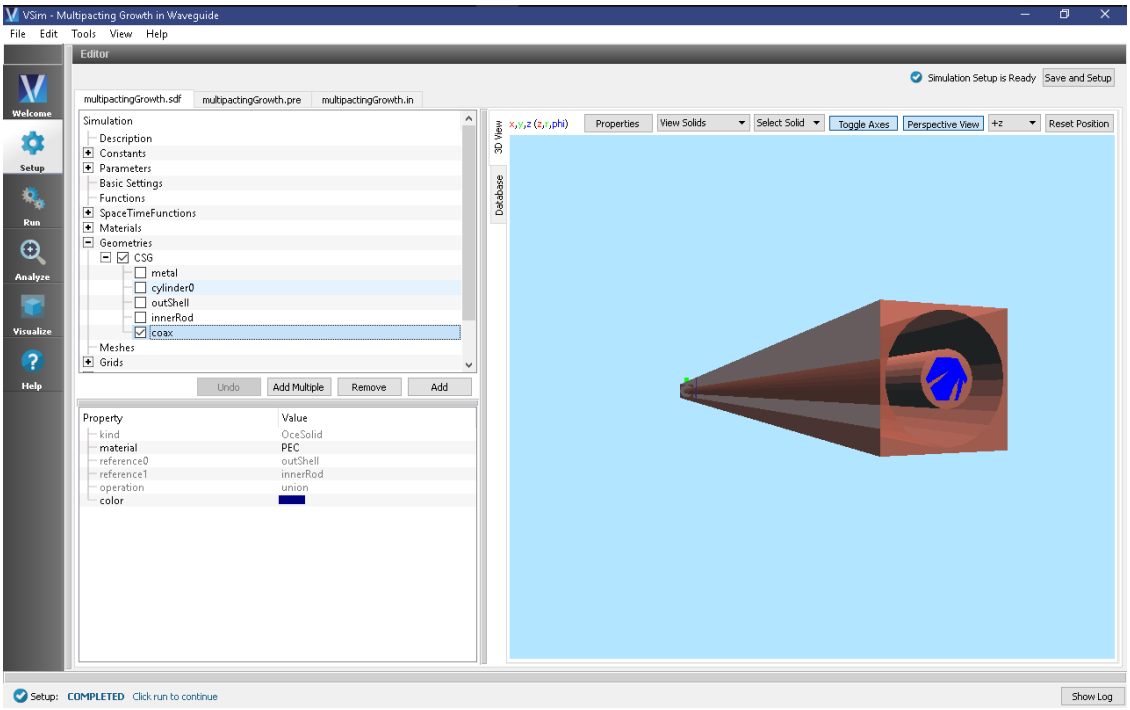


Fig. 4.89: Setup Window for the Multipacting Growth example.

Simulation Properties

This example contains a number of *Parameters* to allow for easy manipulation of the device. Those include:

- R_O: The outer coax radius
- R_I: The inner coax radius
- FREQUENCY: The wave launcher frequency

SpaceTimeFunctions are used to create expressions defining the drive frequency and amplitude of the applied field.

CSG is used to create the coax structure by combining cylinders and cubes.

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.
- 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 when you see the output, "Engine completed successfully." This is shown in Fig. 4.90

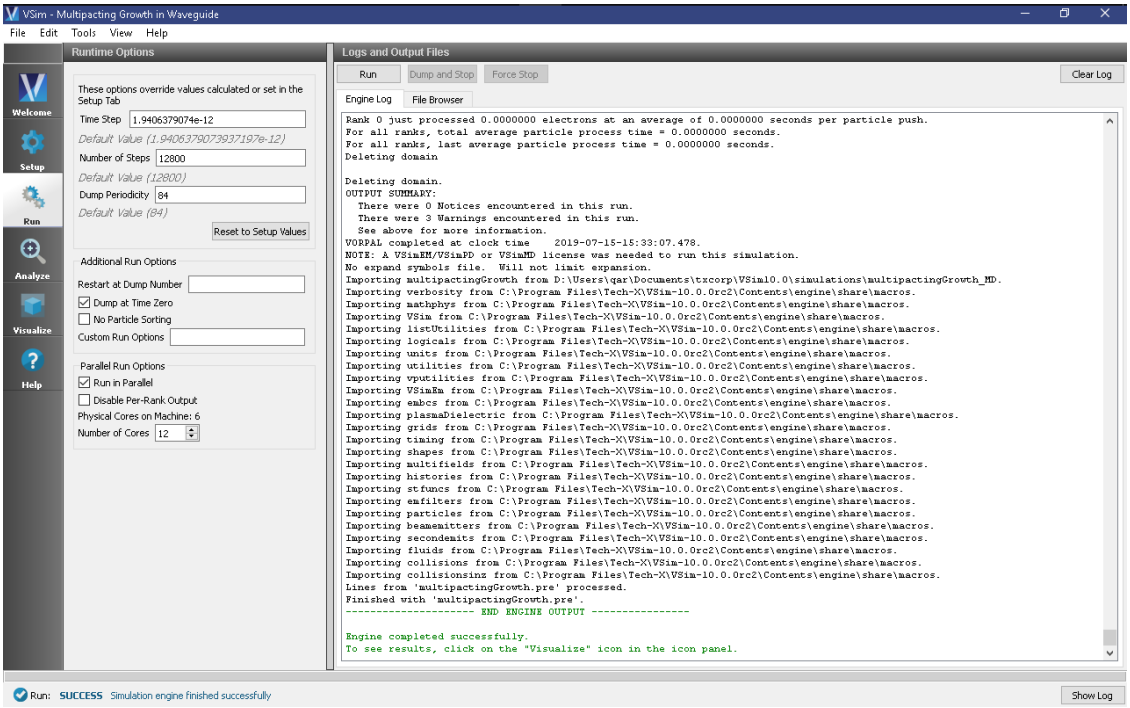


Fig. 4.90: The Run Window at the end of execution.

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 growth in the number of electrons, as shown in Fig. 4.91, do the following:

- Select *History* from the *Data View* pull down menu
- Set Graphs 1&2 to "None"
- Graph 3 should already be set to *numElectrons* (if not, set it)

The overall trend in the number of electrons is an exponential growth with an oscillatory signal that corresponds to the frequency of the traveling wave.

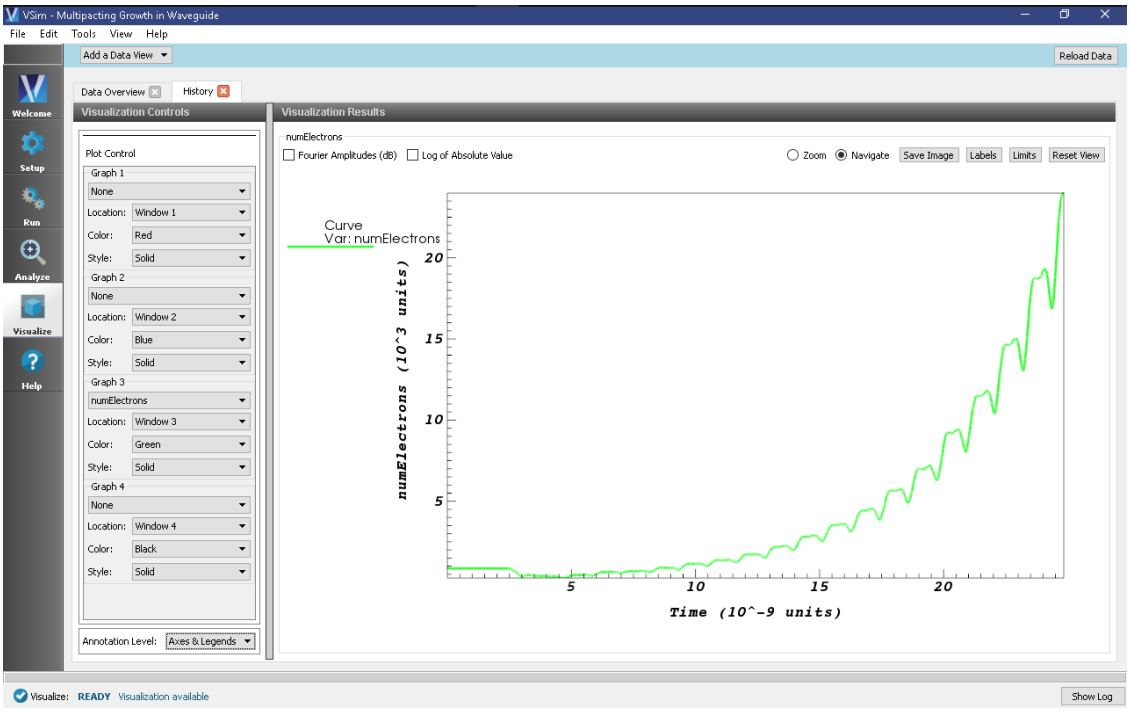


Fig. 4.91: Visualization of the exponential growth of the electrons due to multipacting.

Further Experiments

Try changing the gap voltage or the frequency of the wave to see if one can take the simulation in and out of resonance.

4.4.2 Multipacting Resonances in Waveguide (multipactingResonances.sdf)

Keywords:

`multipacting , multipactingResonances`

Problem description

Multipacting, which is the resonant buildup of secondary electrons, is often a concern in microwave devices. Anytime there is an oscillating electromagnetic field across a gap between two surfaces there exists the possibility that for the right voltage across the gap a resonance condition will exist allowing the exponential buildup of secondary electrons. A coaxial waveguide is such a structure where these conditions can exist.

This simulation can be performed with the VSimMD license.

Opening the Simulation

The Multipacting Resonances example is accessed from within VSimComposer by the following actions:

- Select the *New* → *From Example…* menu item in the *File* menu.

- In the resulting *Examples* window expand the *VSim for Microwave Devices* option.
- Expand the *Multipacting* option.
- Select "Multipacting Resonances in 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 now shown with all the implemented physics and geometries, if applicable. See Fig. 4.92.

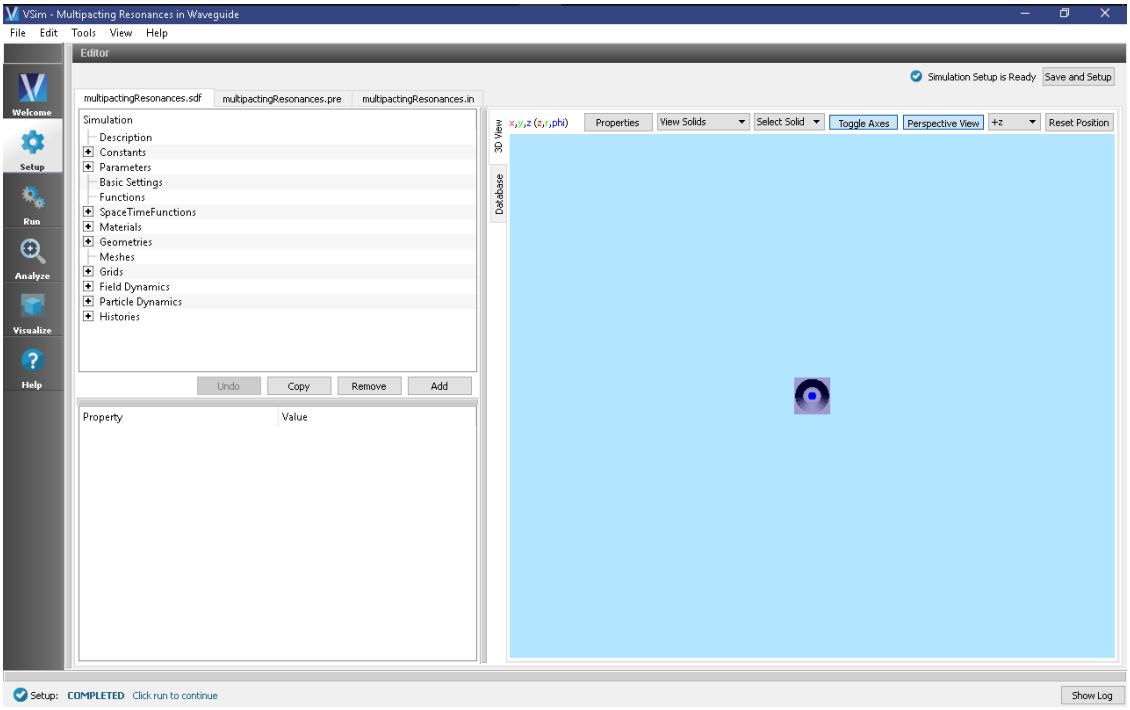


Fig. 4.92: Setup Window for the Multipacting Resonances example.

Simulation Properties

The input file sets the number of cells along the propagation (x) direction to resolve the wavelength. The electrons are seeded in the middle of the waveguide once the wave has passed. A special electron species is used that allows scans over power to be done in a single simulation. The time step is chosen to be at 90% of the CFL (stability) limit.

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.
- To run the file, click on the *Run* button in the upper left corner of the window. 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 is shown in Fig. 4.93.

Visualizing the results

After performing the above actions, continue as follows:

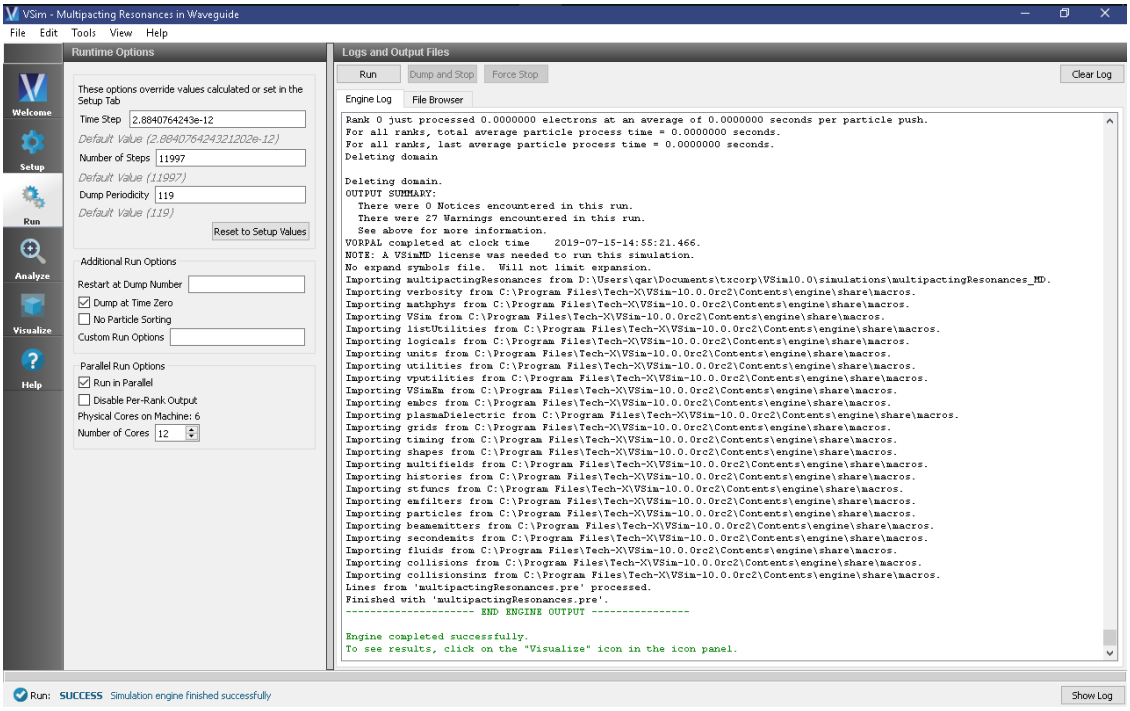


Fig. 4.93: The Run Window at the end of execution.

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

To track the electrons and their field scaled parameters as shown in Fig. 4.94, proceed as follows:

- Select *Phase Space* from the *Add a Data View* pull down menu
- Select *electrons_x* for the *X-axis*
- Select *electrons_y* for the *Y-axis*.
- Select *electrons_fieldScaleParam* for the *Color*.
- Click on *Draw*.
- Move the *Dump* slider to *Dump: 11*.

To view growth in the number of electrons, proceed as follows:

- Select *Binning* from the *Add a Data View* pull down menu
- Select *electrons_fieldScaleParam* for the *Dimension 1*
- Set the *bins* value to 80, the number of scale factors in the simulation
- Select *Count* for the *Operator*.
- Click on *Draw*.
- Move the *Dump* slider to the far right.

Further Experiments

Try seeing how changing the gap voltage or the frequency of the wave changes the multipacting resonances.

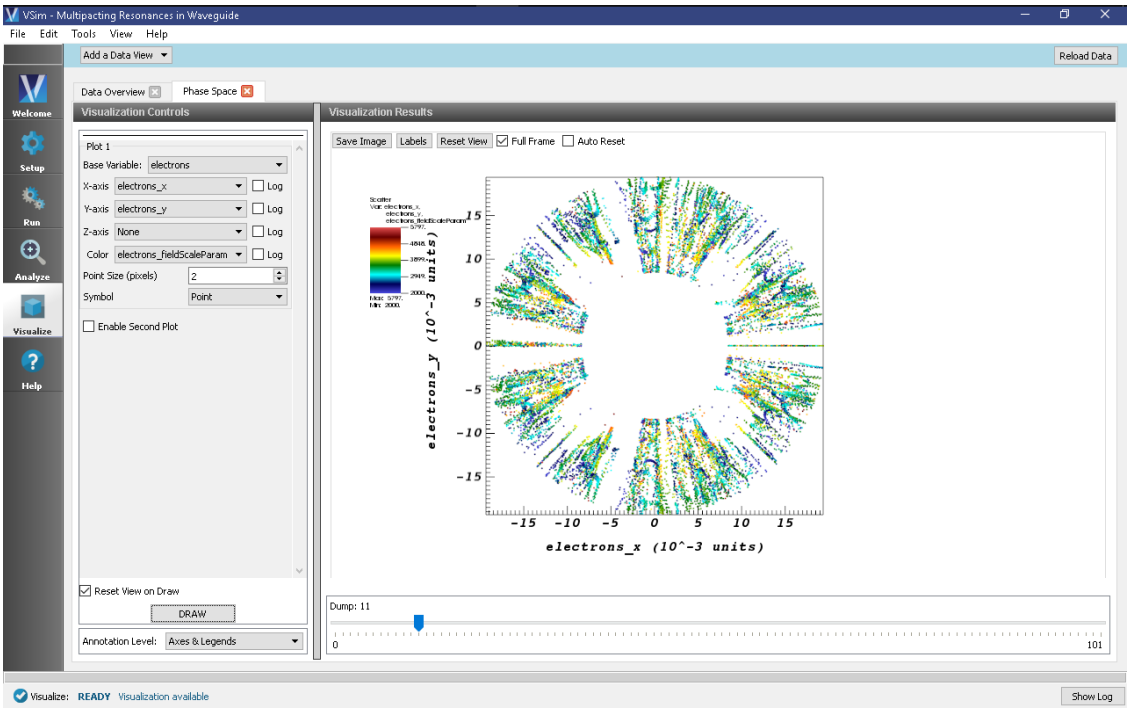


Fig. 4.94: Location of electrons and their field scaled parameters.

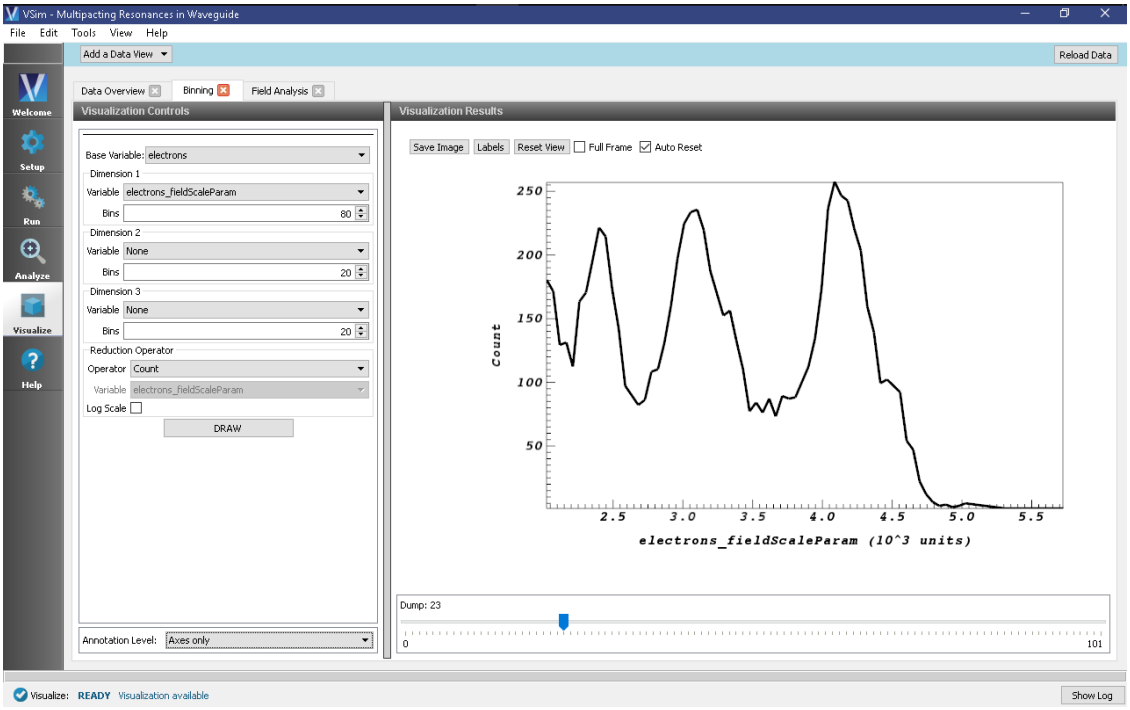


Fig. 4.95: Visualization of the resonance bands.

4.4.3 Multipacting Growth in Spherical PEC Cavity Using Prescribed Fields (multipactingGrowthPrescribedFields.sdf)

Keywords:

multipacting

Problem description

Multipacting, which is the resonant buildup of secondary electrons, is often a concern in microwave devices. Anytime there is an oscillating electromagnetic field across a gap between two surfaces there exists the possibility that for the right voltage across the gap a resonance condition will exist allowing the exponential buildup of secondary electrons. This example simulates multipacting growth in a spherical PEC cavity. The fundamental mode profile for the spherical PEC cavity is imported onto the VSim grid, then advanced in time by a single frequency time signal.

This simulation can be performed with the VSimMD or VSimPD license.

Opening the Simulation

The Multipacting Growth Prescribed Fields example is accessed from within VSimComposer by the following actions:

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim for Microwave Devices* option.
- Expand the *Multipacting* option.
- Select "Multipacting Growth Using Prescribed 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.

All of the properties and values that create the simulation are now available in the Setup Window as shown in Fig. 4.96. 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`.

Simulation Properties

This example contains a number of *Constants* to allow for easy manipulation of the device. Those include:

- **SPHERE_RADIUS**: radius of spherical cavity
- **LENGTH_METAL**: Length of metal box in each dimension (must be larger than 2*SPHERE_RADIUS)
- **RESOLUTION**: The number of cells per wavelength in each dimension

SpaceTimeFunctions are used to create expressions defining the drive frequency and amplitude of the applied field.

The amplitude and frequency of this driving function is defined in *Parameters*:

- **MODE_FREQ**: frequency at which the mode profile oscillates.
- **MODE_AMP**: amplitude that is applied to the mode profile each time step.

CSG is used to create the spherical PEC cavity by subtracting sphere from a cube.

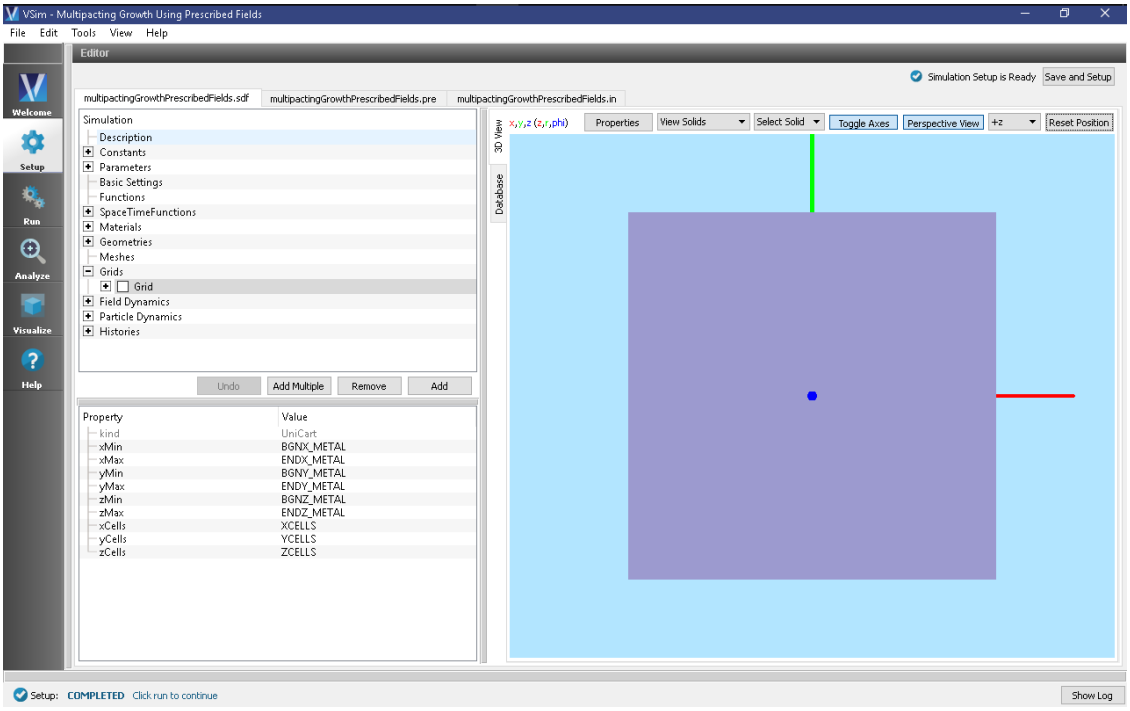


Fig. 4.96: Setup Window for the Multipacting Growth example.

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.
- To run the file, click on the *Run* button in the upper left corner. of the window. 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 is shown in Fig. 4.97

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.
- Select *Binning* from the *Data View* pull down menu.
- Select *Auto Reset* at the top of the *Visualization Window*.

To view the field-scaling values at which mutipacting occurs, as shown in Fig. 4.98, use the following settings in *Visualization Controls*:

- Set *Dimension 1* → *Variable* to *fieldScalingElectrons_fieldScaledParam*.
- Set *Dimension 1* → *Bins* to 100 (Note: always set the number of bins to be greater or equal than the number of scaling factors used in the particle setup).
- Under *Reduction Operator* set *Operator* to *Sum* and set *Variable* to *fieldScalingElectrons_weight*.
- Click *DRAW*.
- Move the dump cursor all the way to the right.

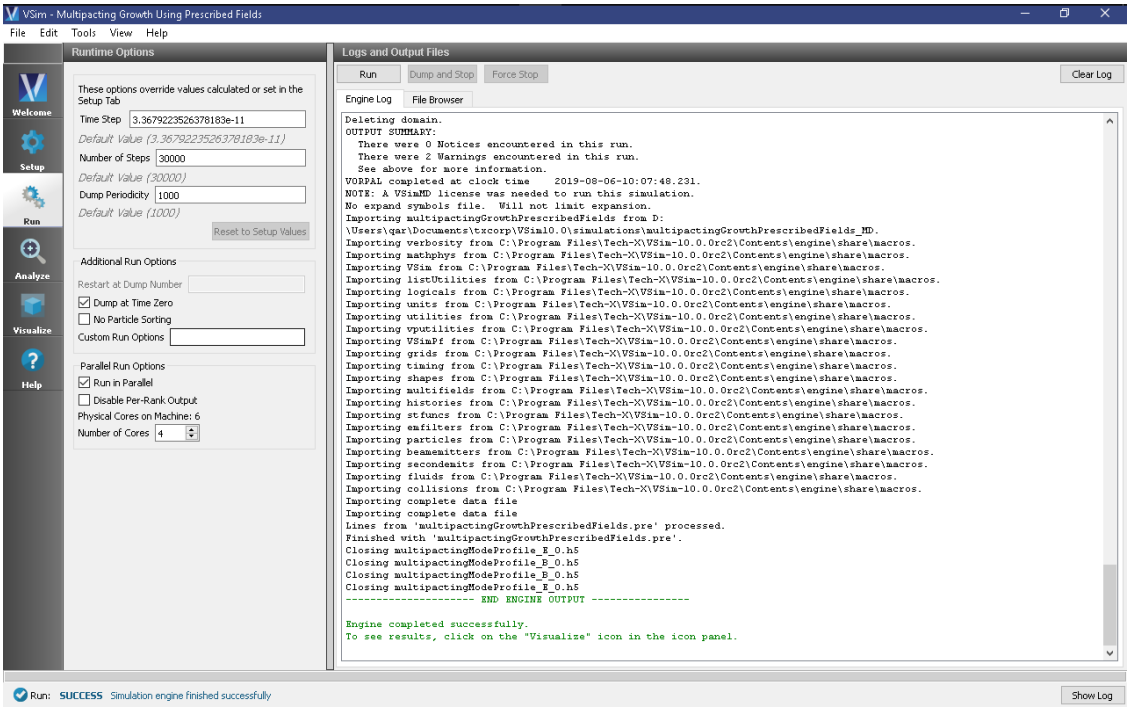


Fig. 4.97: The Run Window at the end of execution.

The overall trend in the number of electrons is that of an exponential growth with time at the correct field strengths that are controlled by the user's power settings during the design of the device.

Further Experiments

Try changing the parameters **`MODE_AMP`** and **`MODE_FREQ`** to see if one can take the simulation in and out of resonance.

4.5 EmissionT (text-based setup)

4.5.1 Vaughan Secondary Emission (VaughanSecondaryElecT.pre)

Keywords:

`VaughanSecondaryElecT`

> **Warning:** Due to a known issue parallel runs, we suggest limiting the run to 8 cores.

Problem description

Sometimes secondary emission processes are not adequately explained by the Furman-Pivi model, and sometimes we deliberately wish to compare how simulation results would be affected by different result files. In this example we show how to set up a user defined secondary emission model

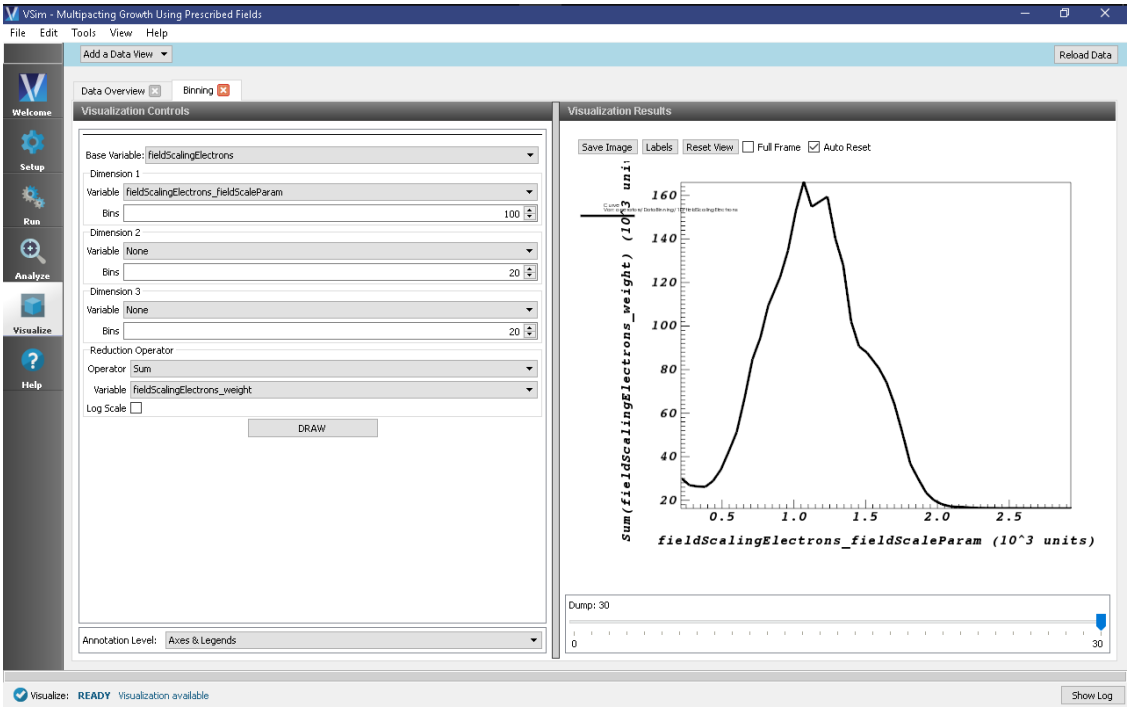


Fig. 4.98: Visualization of the exponential growth of the electrons due to multipacting.

Opening the Simulation

The Vaughan Secondary Emission example is accessed from within VSimComposer by the following actions:

- Select the *New From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim for Microwave Devices* option.
- Expand the *Emission (text-based setup)* option.
- Select "Vaughan Secondary Emission (text-based setup)" 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 basic variables of this problem should now be alterable via the text boxes in the left pane of the *Setup Window*, as shown in Fig. 4.99.

Input File Features

The Vaughan Secondary Emission example has the following input parameters:

- NDIM allows adjusting of the number of dimensions
- NX, NY and NZ allows the setting of the number of cells in x, y and z respectively.
- LX, LY and LZ allows the setting of the size of the physical domain to be modeled respectively.
- E_BEAM_ENERGY specifies the energy of the incident electron beam in eV
- E_BEAM_CURRENT specifies the incident beam current
- E_BEAM_ANGLE chooses the angle at which the electrons are incident on a plane surface. We modify the angle of the geometry in this simulation, rather than moving the beam around.

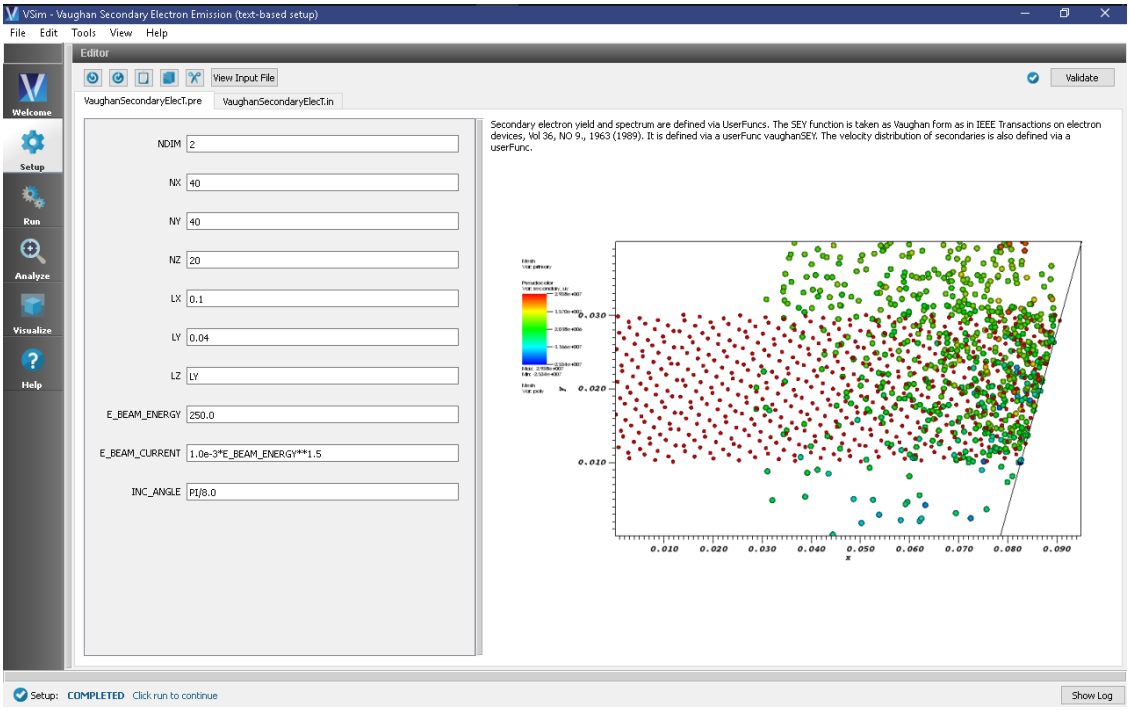


Fig. 4.99: Setup Window for the Vaughan Secondary Emission example.

Many of the interesting features in this file require us to click on *View Input File*.

The `userFunc` vaughanSEY around line 100 pairs up with

```
<UserFunc vaughanSEY>
  kind = expression
  inputOrder = [engInc alpha]
```

in the secondaryEmitter definition. By looking after lines 100 we see how the vaughanSEY arguments are read into this userFunc. We determine the type of data these arguments should contain then build up functions that depend on these arguments. Various terms are built up and then combined in the expression statement

```
expression= sigmaMax * funcF((engInc - Eth) / (EMax - Eth))
```

at the end. The second userFunc defines a cosine squared direction distribution, and the third the outgoing energy spectrum.

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.
- To run the file, click on the *Run* button in the upper left corner of the window. 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 is shown in Fig. 4.100 below.

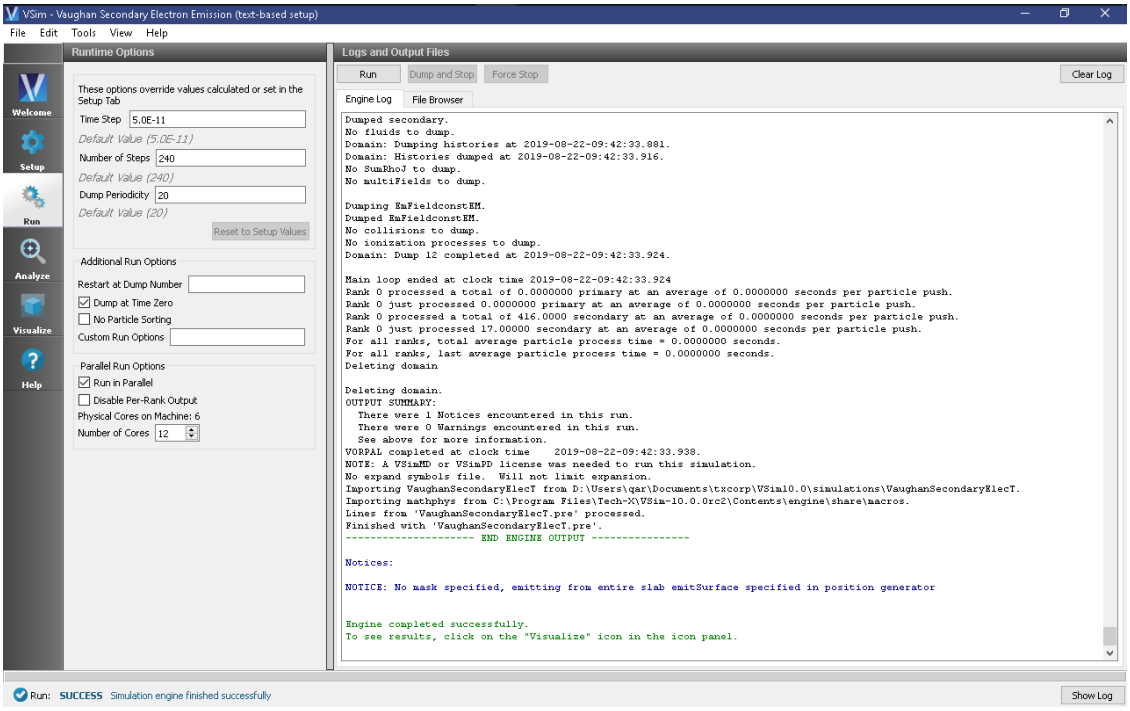


Fig. 4.100: The Run Window at the end of execution.

Analyzer settings

If you proceed directly to visualisation, it is likely the secondary electrons will not show synchronously. In the bottom left corner, setting the annotation level to “all annotations” is important to check that you are visualising primary and secondary electrons at the same time.

One may use an analyzer to add additional dump files to avoid this lack of synchronization. On the analyzer tab, click *Show All Analyzers* and select the createMissingPtclsDumps.py analyzer.

Use the settings shown in Fig. 4.101 below, then click on *Analyze*.

Since our datasets do not allow for completely empty particle datasets, this puts a single particle for each dump for which there are no macroparticles, in the bottom left (low x, low y, low z) corner of the domain, where it will minimally affect viewing of the results, but it will help the visualisation stay synchronous.

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 electrons as in Fig. 4.102:

- Select *Particle Data->primaries* and *Particle Data->secondaries* in the data tree
- If you have a high res monitor you may wish to increase the particle display size in the *Size* option under *Particle Style*.
- If you have run the analyzer since visualising, you will want to click *Reload Data* before continuing.
- Select *Geometries->poly* to view the wall with which the particles are colliding.

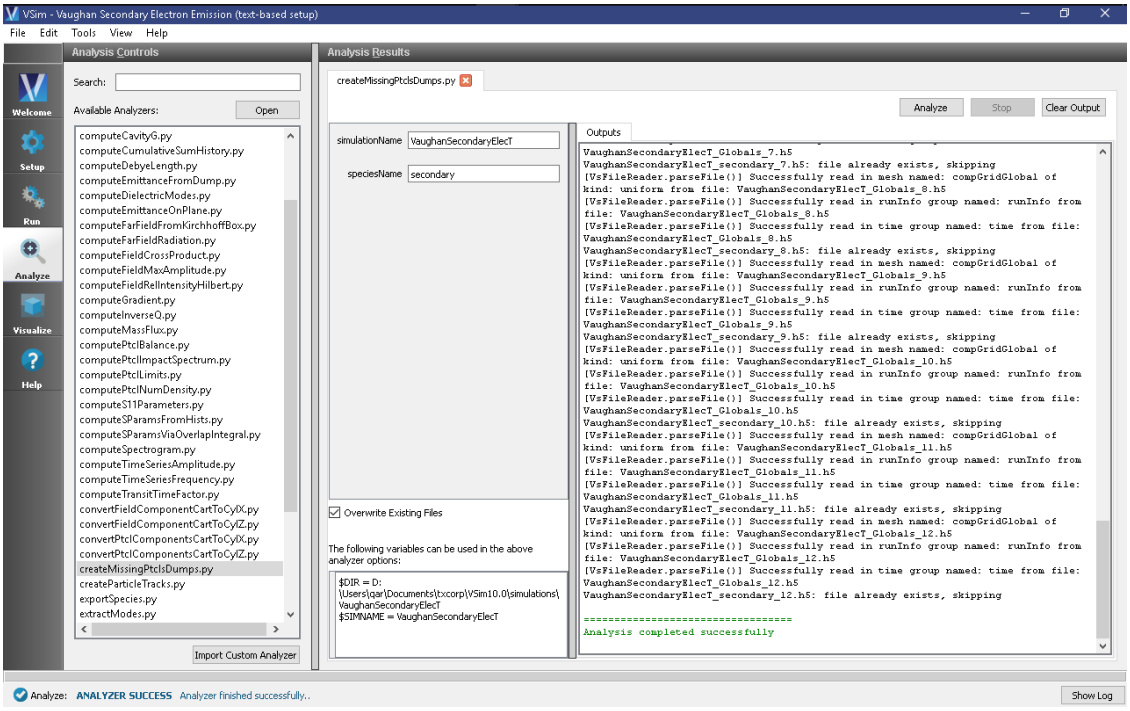


Fig. 4.101: Settings for the Vaughan Secondary Electron Emission Post-processing analysis.

- Move the dump slider to higher dump numbers to see the particles bouncing off the wall and more secondaries be created.

It is also possible to gain information about the effective secondary emission yield, the number of particles out given the particles going in, by comparing the number of primaries and secondaries in the simulation, and the rate at which they are created and lost at steady state.

In this simulation we just start by including histories for the total number of particles of each species, which may be viewed by selecting *Histories* under *Data View* in the top left.

Further Experiments

Extra histories have been included to measure the primary electron current absorbed on the wall, and a particle emission history to measure the rate at which the secondaries are coming away. For example with:

```
<History primCur>
  kind=speciesCurrAbs
  species= [primary]
  ptclAbsorbers = [plateAbsorber]
</History>
<History secCur>
  kind = speciesCurrEmit
  species = [ secondary ]
  ptclSource = secondary.secondaryEmitter
  sourceType = 0
</History>
```

One can then plot the histories as shown in Fig. 4.103. To add a history plot, click on *Add a Data View*, then click on *History*. You will notice a new visualization tab has been added next to "Data Overview". To change the font on any of the plots, click on "Tools", then click on "Settings", and finally click on "Visualization Options".

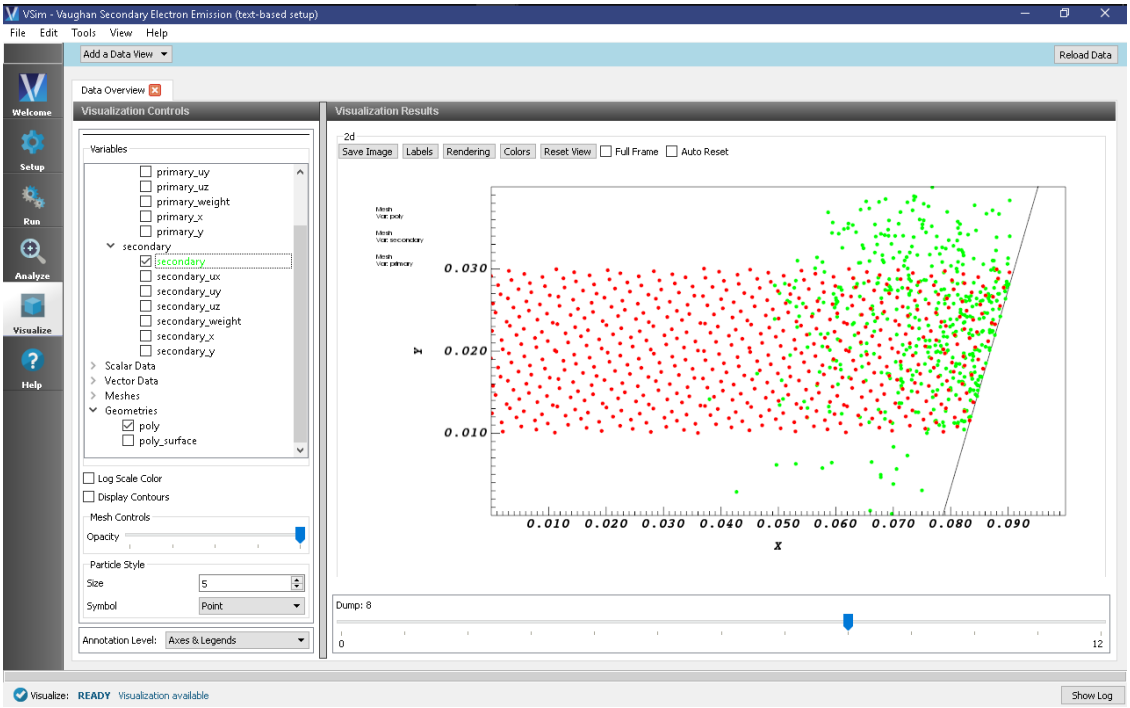


Fig. 4.102: Visualisation of particles hitting the wall in Vaughan Secondary Electron Emission example.

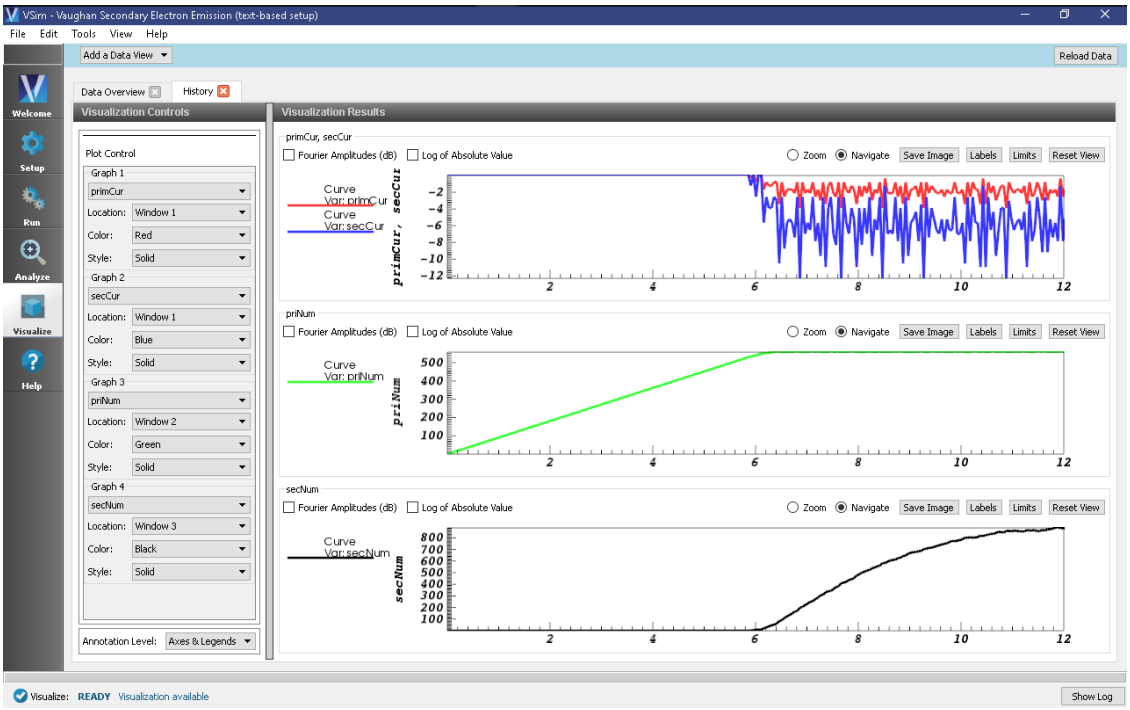


Fig. 4.103: Visualisation of ratio of primary and secondary particles in Vaughan Secondary Electron Emission example.

4.5.2 2D Laminar Brillouin Flow (laminarBrillouinFlowT.pre)

Keywords:

laminarBrillouinFlowT

Problem description

Many VSimMD applications require a smooth beam to achieve good performance. Simulating such devices can be computationally complex, and commonly we choose to work in 2D instead to save time. In this simulation, we demonstrate how to set up a beam that may be used in the interaction region of any vacuum electronic device you might simulate in 2D, with the interaction structure removed for simplicity.

This simulation can be performed with a VSimMD license.

Opening the Simulation

The Brilloin Flow example is accessed from within VSimComposer by the following actions:

- Select the *New From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim for Microwave Devices* option.
- Expand the *Emission (text-based setup)* option.
- Select "Laminar Brillouin Flow (text-based setup)" 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 basic variables of this problem should now be alterable via the text boxes in the left pane of the *Setup Window*, as shown in Fig. 4.104.

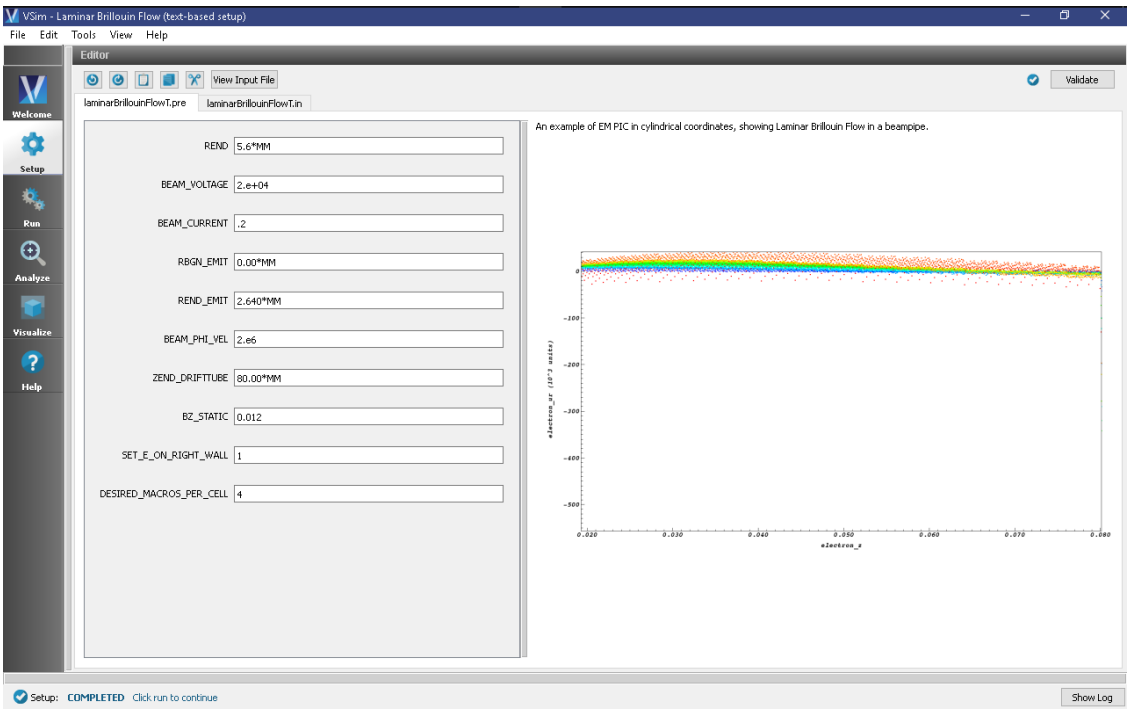


Fig. 4.104: Setup Window for the Brillouin Flow example.

Input File Features

The Laminar Brillouin Flow example has the following input parameters:

- REND is the radial extent of the problem.
- BEAM_VOLTAGE specifies the starting voltage for the electron beam.
- BEAM_CURRENT specifies the beam current.
- RBGN_EMIT and REND_EMIT specify the inner and outer radius respectively of the beam source on the left side of the simulation domain.
- BEAM_PHI_VEL sets the rotational velocity of particles at the outside of the beam. Those inside will be given azimuthal velocity proportional to radius.
- ZEND_DRIFTUBE determines the amount of beam pipe to simulate.
- BZ_STATIC sets the magnetic field (in Tesla)
- SET_E_ON_RIGHT_WALL allows for hard-coding a function for the field distribution on the right wall. When set to 1, and the other parameters have default values, the simulation behaves as if the right side of the simulation domain is open. Set to 0, a wall bounds the right side.
- DESIRED_MACROS_PER_CELL sets the particle discretization.

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.
- To run the file, click on the *Run* button in the upper left corner of the window. 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 is shown in Fig. 4.105 below.

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 exponential increase in electrons as in Fig. 4.106:

- Select *Field Analysis* under the *Data View* drop down
- Set the field to *edgeE_r*
- Pull the slider bar to Dump:40
- Set the layout to stacked 2d/1d

The transverse field increases with radial distance inside the beam, follows a 1/r distribution outside the beam and rapidly drops to zero at the metal interface.

To view the velocity of the particles in the beam, such as that seen in Fig. 4.107:

- Select *Phase Space* under the *Data View* drop down
- Set the X-axis to *electron_z*
- Set the Y-axis to *electrons_uz*
- Set the Color to *electrons_r*

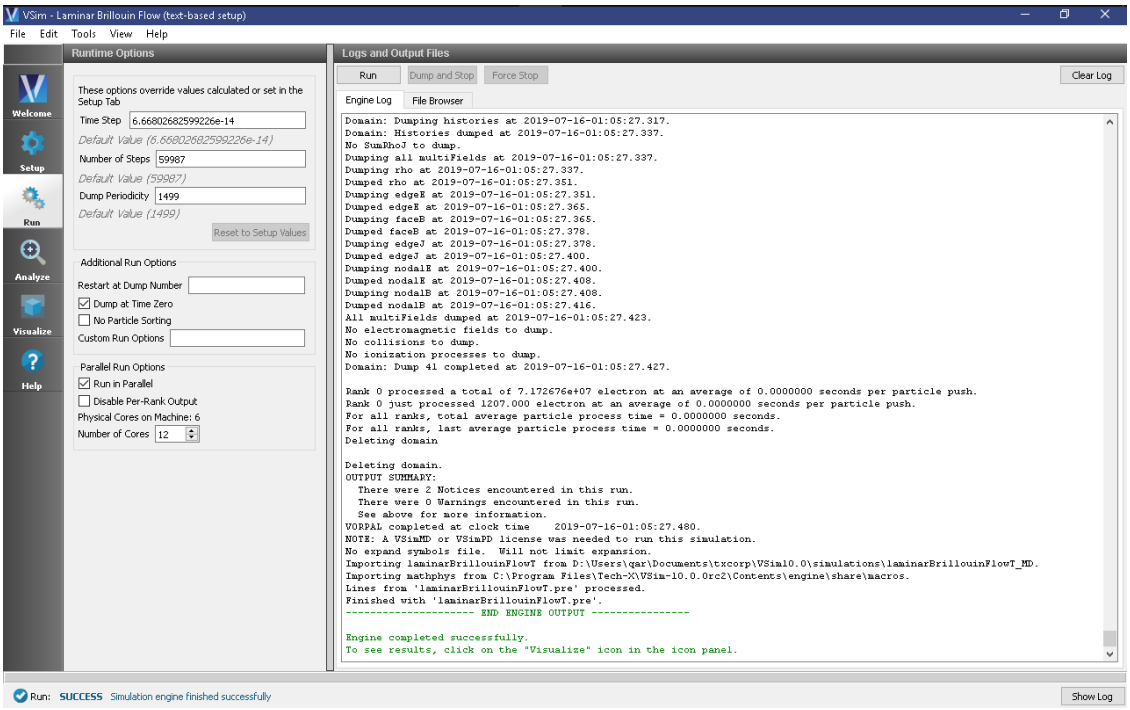


Fig. 4.105: The Run Window at the end of execution.

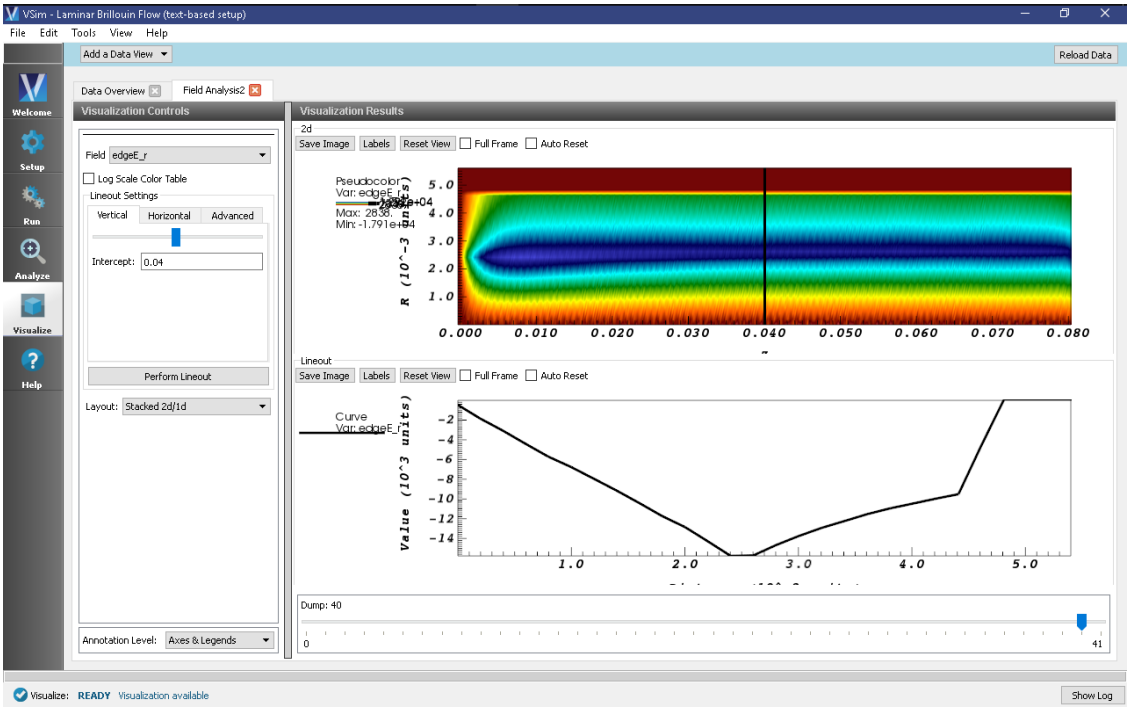


Fig. 4.106: One can see the transverse field profile.

- Press *Draw*.
- Move the slide to the right to see how the beam stabilizes.

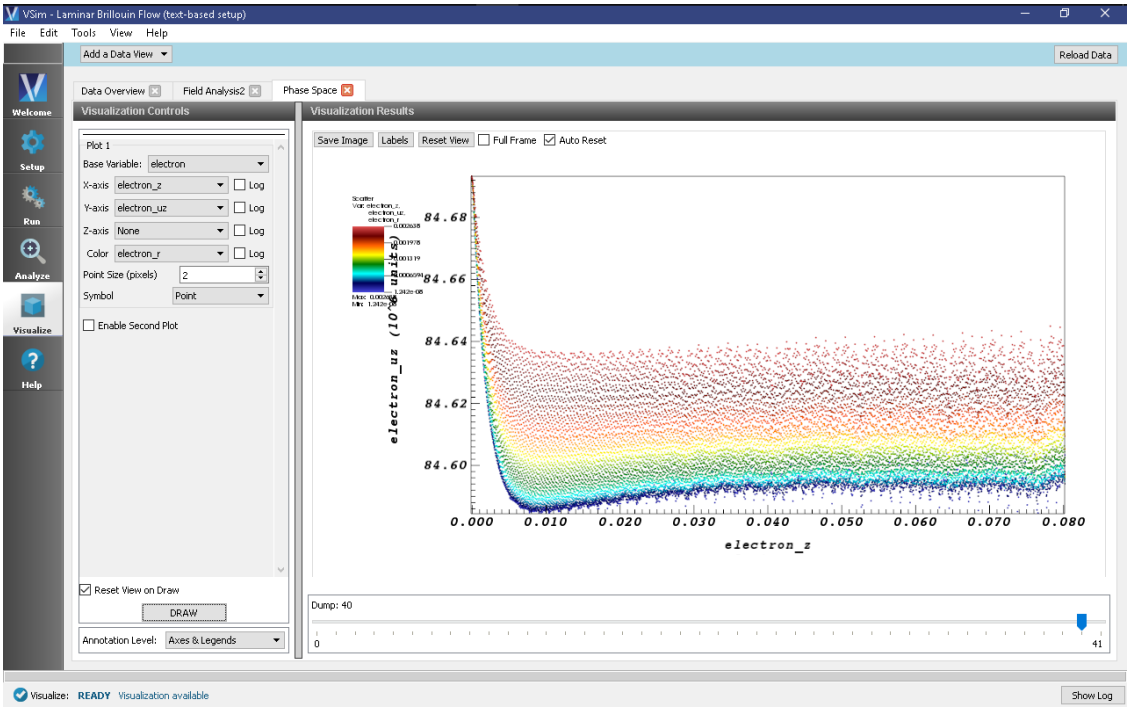


Fig. 4.107: The beam stabilizes fairly quickly and becomes close to monoenergetic with small transverse velocity.

Switch to beam longitudinal velocity (Y-axis to *electrons_uz*), and note the orders of magnitude of difference.

Further Experiments

Try varying the starting and ending voltages, and see if the function used to set the right wall needs to be varied, and by how much.

4.6 Other

4.6.1 Electron Gun (electronGun.sdf)

Keywords:

`electron, gun, beam, collimate`

Problem description

Electron guns are devices that are often found in vacuum electronics as well as in more advanced technologies such as klystrons, electron microscopes, and particle accelerators. They produce narrow, collimated beams of electrons with precisely tuned kinetic energies. They were often found in cathode ray tubes at the heart of television sets prior to the digital revolution. Electron guns are composed of a cathode, an anode, and repulsive rings. A DC or RF signal is applied to the cathode to produce electrons via thermionic emission. The electrodes produce electric fields that focus the electron beam. Often an additional anode is placed between the cathode and the main anode to act as a repulsive

ring that focuses the beam into a small hole in the main anode. The small hole in the main anode acts to collimate the beam.

This example is a specialized electron gun for klystrons and TWTs. It is characterized by high power, a consequence of which is that electrons not successfully collimated can damage the device. To minimize this effect, the gun includes a focusing anode cone, the angle of which is conducive to laminar flow of the electron beam.

Opening the Simulation

The Electron Gun example is accessed from within VSimComposer by the following actions:

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim for Microwave Devices* option.
- Expand the *Other MD* option.
- Select "Electron Gun" 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, if applicable. See Fig. 4.108.

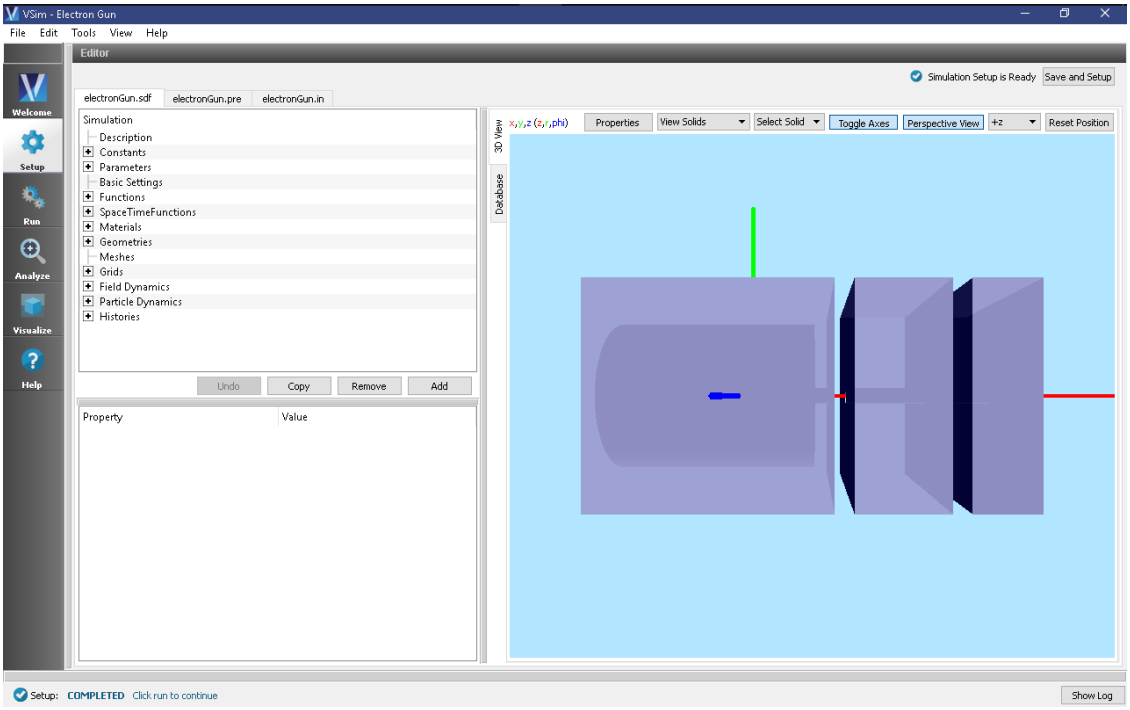


Fig. 4.108: Setup Window for the Electron Gun example.

Simulation Properties

The input parameters give you total flexibility in defining the geometry of the example. Along with these one can define the nominal cell size, the driving voltages, the strength of the magnetic field, and the beam current.

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.
- To run the file, click on the *Run* button in the upper left corner of the right panel. 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 is shown in Fig. 4.109.

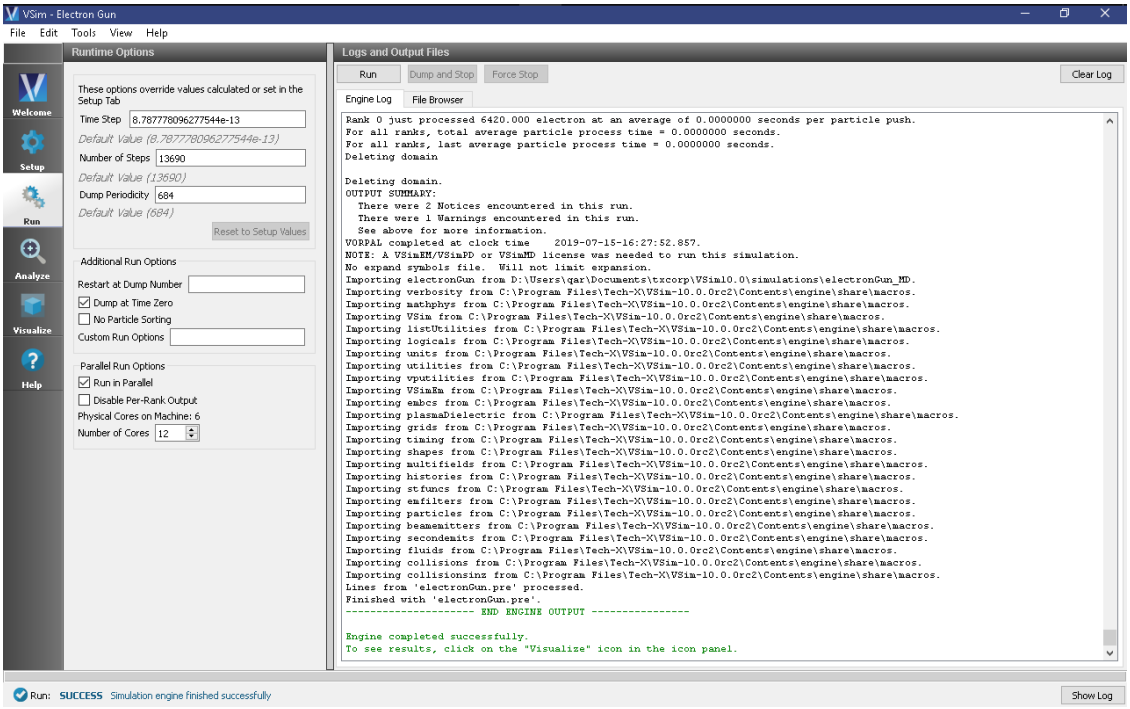


Fig. 4.109: The Run Window at the end of execution.

Visualizing the Results

To reproduce Fig. 4.110 proceed as follows:

- Expand *Particle Data*.
- Expand *electron*.
- Select *electron* in red.
- Expand *Scalar Data*
- Expand *E*.
- Select "E_magnitude".
- Expand *Geometries*.
- Select "poly (electronGunPecShapes)".
- In the lower part of the left pane select "Display Contours".
- Select "Clip All Plots".

- In the lower part of the right pane, move the Dump slider to dump 14.

This will show the electron beam and the electric and magnetic fields.

The phase space diagram can also be viewed by choosing *Phase Space* in the *Add Data View* drop down menu in the top left of the main pane.

The voltages and currents at key locations in the simulation are recorded in Histories and can be viewed by selecting the *History* data view.

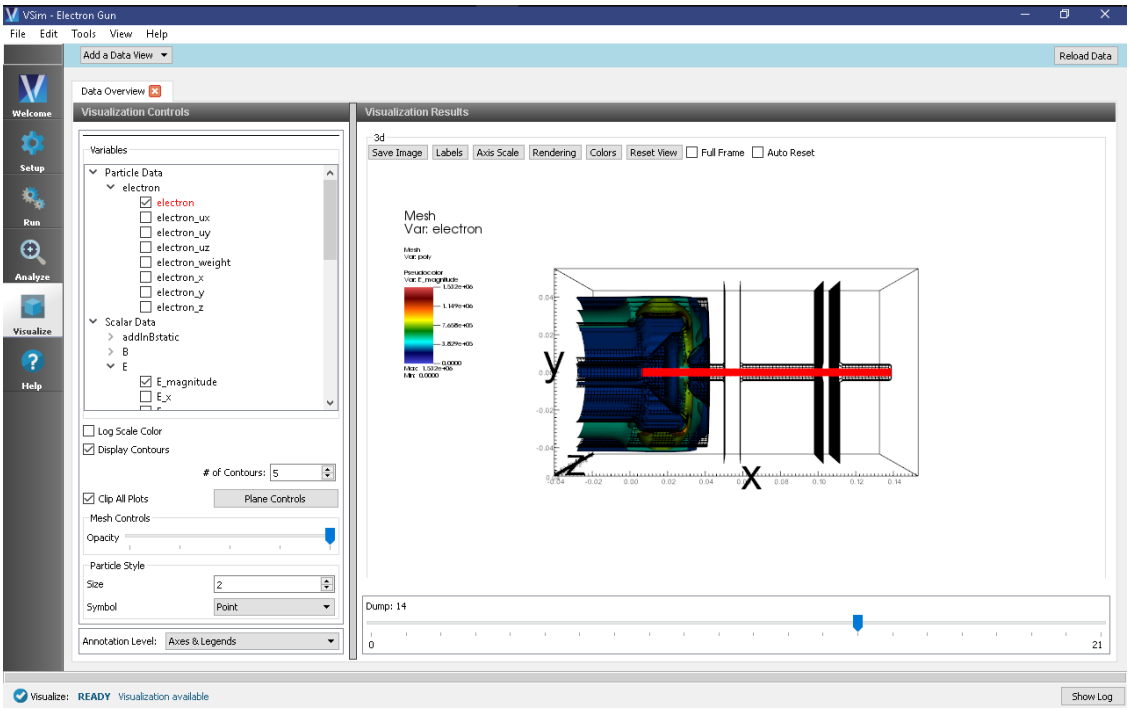


Fig. 4.110: The electron beam and electric field.

Further Experiments

The geometry is extremely important for proper functionality in this example. For example, the angle that the focusing cone makes with the beam axis determines whether the beam will be laminar. If the beam intersects and diverges, the gun can be damaged by its own power. Try altering the dimensions of the geometry and see the effect on the electron beam.

4.6.2 Multistage Collector (multistageCollector.sdf)

Keywords:

`electromagnetics, multistageCollector`

> **Warning:** Due to a known issue in the STL reader during parallel runs, this example may run into trouble if run on too many cores. We suggest limiting this run to 4 cores, though more may still work.

Problem description

Multistage Depressed Collectors (MDCs) are used to recover energy from a spent beam in linear type microwave tubes such as traveling wave tubes (TWTs) and klystrons. VSim provides the capability to simulate these collectors shaped with arbitrarily complex geometries and depressed with different time-dependent voltage profiles to optimize the recovery efficiency of a design. To demonstrate this capability, we show in this example a 4-stage depressed collector. One can adjust the depressed potentials at each electrode individually to see how the performance of the collector is affected.

This simulation can be performed with a VSimMD or VSimPD license.

Opening the Simulation

The Multistage Collector example is accessed from within VSimComposer by the following actions:

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim for Microwave Devices* option.
- Expand the *Other MD* option.
- Select "Multistage Collector" 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, if applicable. See Fig. 4.111.

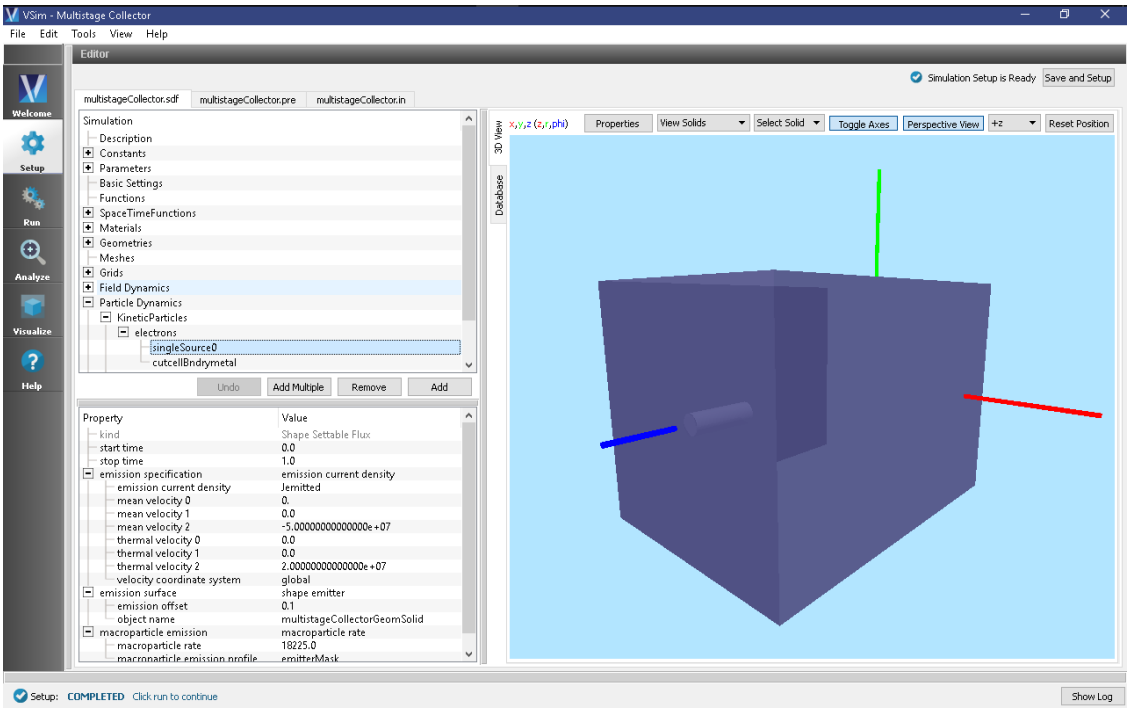


Fig. 4.111: Setup Window for the Multistage Collector example.

Simulation Properties

The simulation geometry consisting of an S-band 4-stage depressed collector is imported into the computational engine from CAD files in stl format. One can easily create new geometry using any CAD program and output or convert the

CAD files into stl files for a new simulation design. The detailed import method is provided in the input file. The spent beam profile is taken from a TWT simulation provided by Prof. H. Song at University of Colorado at Colorado Springs.

An optimized design for a MPM module can be found in reference [1]. Users can set preferred spent beam profiles by employing different emission methods or import data in dat format as in this example. A main feature of this input file is that the depressed voltage profiles are time-dependent and are stabilized with a new external circuit model based on special feedback algorithms only available in VSim. Interested users may refer to the publication for more a detailed description and validation. In addition, the convergence of this example is carefully tested.

In this example, the Z coordinate is the direction aligned with the beam axis of the MDC, and the 4 different voltages can be easily assigned at the input panel. Since it is a time domain simulation, the Dey-Mittra algorithm is employed and the accuracy is second-order for the complex boundaries.

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.
- To run the file, click on the *Run* button in the upper left corner of the right panel. 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 is shown in Fig. 4.112.

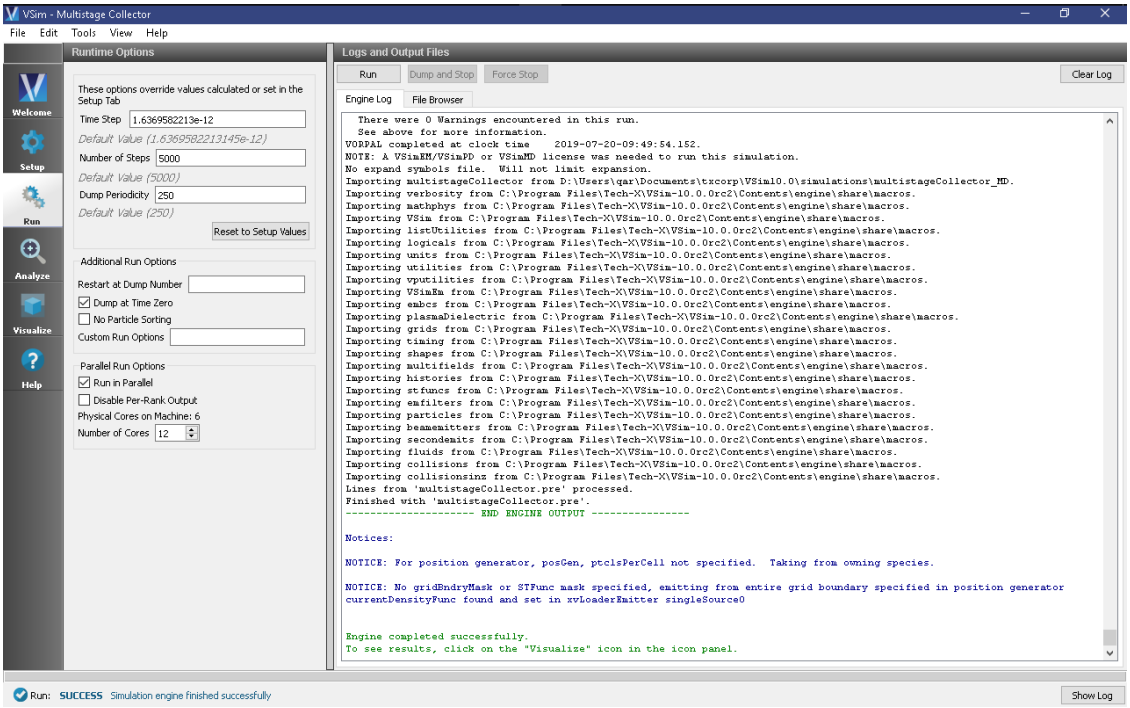


Fig. 4.112: The Run Window at the end of execution.

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.

The results are then read from the Data Overview in the Visualize Window:

- Expand *Particle Data*.
- Expand *electrons*.
- Select *electrons* in red.
- Expand *Scalar Data*.
- Expand *E*.
- Select *E_z*.
- Expand *Geometries*
- Select the second option from the top: *multistageCollectorGeomSolidMappedPolysData_surface*.
- Select *Display Contours* in the bottom left corner.
- Select *Clip All Plots* in the bottom left corner.
- Click on *Plane Controls* and in the *Clip Plane Control* window, under *Clip Plane Normal* select *X (plane normal to x-axis)*, then click *Ok*.
- Move the Dump slider all the way to the end.
- Use the cursor to grab the image and rotate it from right to left to see the image in Fig. 4.113.

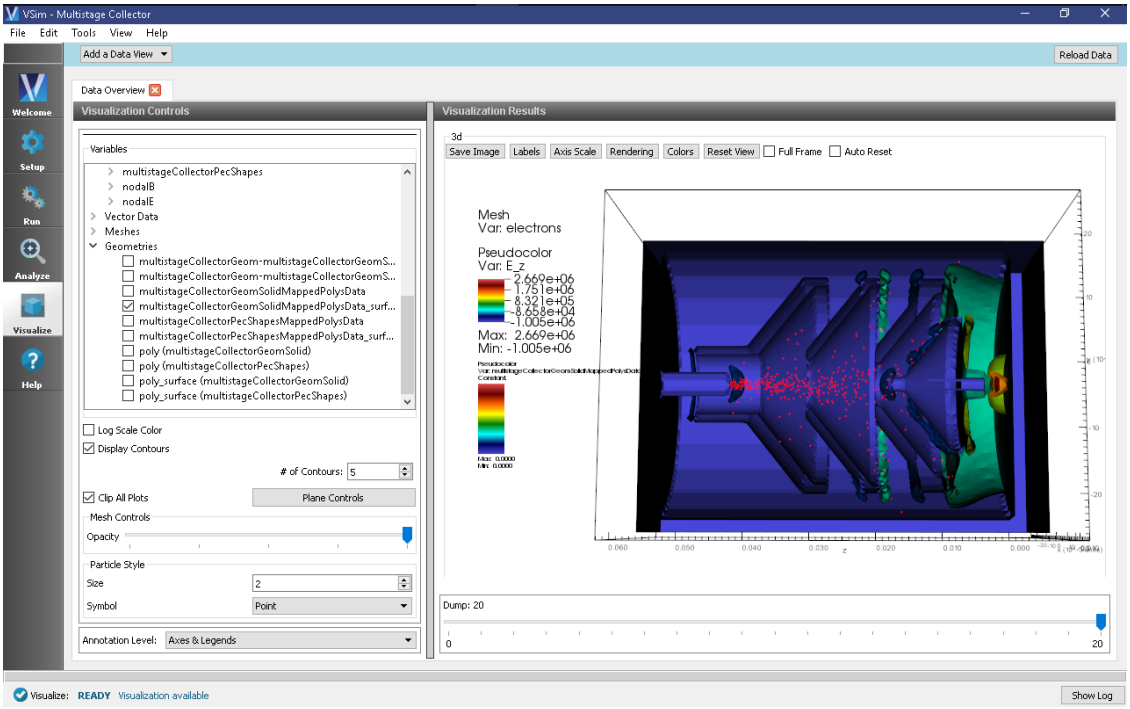


Fig. 4.113: Visualization of the MDC model with a color contour plot of electric fields and electrons in red.

The potential of each of collector surfaces is recorded using a history. To visualize these values as shown in Fig. 4.114, do the following:

- Switch *Data View* to “History”.
- In the left pane, set Graphs 1-4 to each of the different potential histories: “potential10”, “potential20”, “potential30”, and “potential40”, respectively.

- Set the *Location* of each graph to the "Window 1".

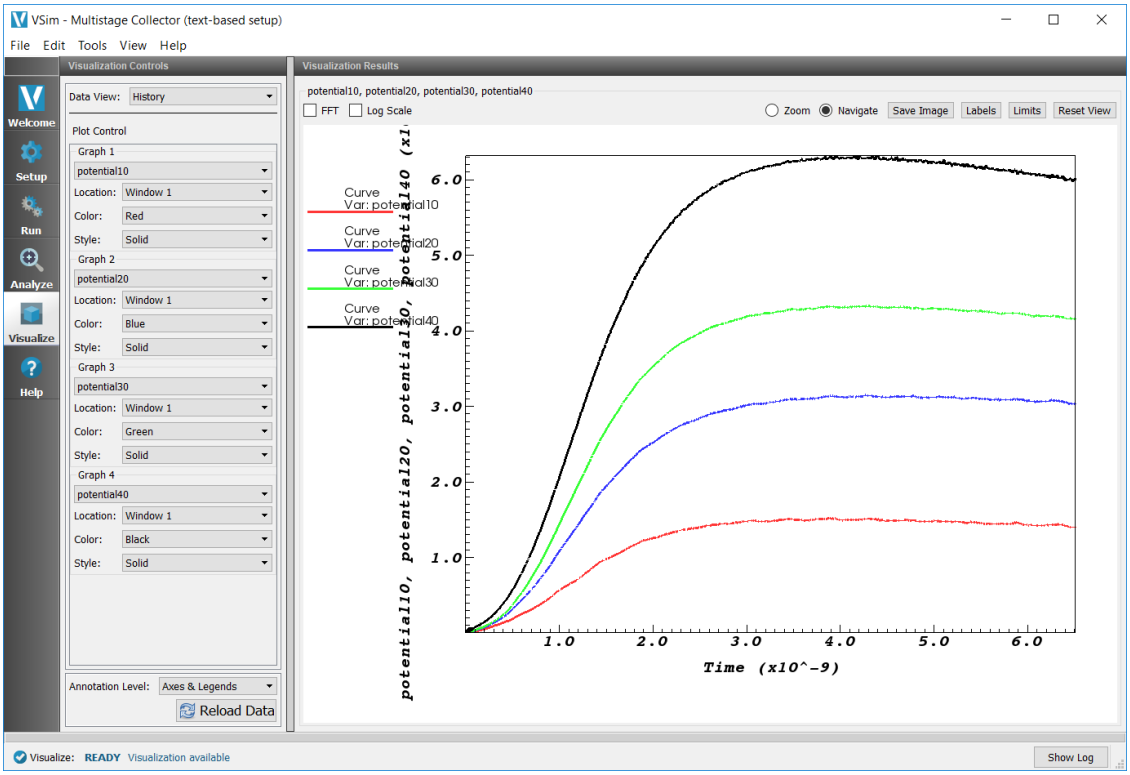


Fig. 4.114: The value of the potential on each of the collectors.

Further Experiments

The depressed voltages or beam current/radius can be varied in the input panel for testing runs. One can also change the grid cell numbers to see the convergence of the simulations.

References

[1] M. C. Lin, P. H. Stoltz, D. N. Smithe, H. Song, H. J. Kim, J. J. Choi, S. J. Kim, and S. H. Jang, "Design and Modeling of Multistage Depressed Collectors Using 3D Conformal Finite-Difference Time-Domain Particle-In-Cell Simulations ", J. Korean Phys. Soc. 60, 731-738 (2012).

CHAPTER

FIVE

VSIM FOR PLASMA ACCELERATION EXAMPLES

These examples illustrate how to solve complex problems in plasma acceleration.

These examples can be run with a VSimPA license.

5.1 Beam Driven

5.1.1 Electron Beam Driven Plasma Wakefield (electronBeamDrivenPlasma.sdf)

Keywords:

electron driven, plasma wakefield, CLARA, PARS, AWAKE

Problem description

This example demonstrates a method to simulate an electron beam driven plasma wakefield accelerator. The electron beam initializes the field using a speed of light frame Poisson equation solve, then the fields and particles are evolved using FDTD EMPIC. We launch the electron beam from x=0 in the positive x direction using the Lorentz boosted Poisson fields to ensure that the simulation is self-consistent from start. The primary bunch generates a region of high field into which one might inject and accelerate a second bunch of charged particles. The example simulation uses parameters that are appropriate to the plasma acceleration research station (PARS) at the CLARA accelerator at Daresbury Laboratory in the UK.

Opening the Simulation

The Electron Beam Driven Plasma Wakefield example is accessed from within VSimComposer by the following actions:

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim for Plasma Acceleration* option.
- Expand the *Beam Drive Acceleration* option.
- Select *Electron Beam Driven Plasma Wakefield* 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. 5.1.

At this stage one can use the element tree on the left-hand side choose constants such as `LONGITUDINAL_RES` and `TRANSVERSE_RES` which represent the longitudinal and transverse number of cells per RMS bunch size. The minimum of 6 or default of 8, generates a simulation that will complete reasonably quickly, but is not adequate to

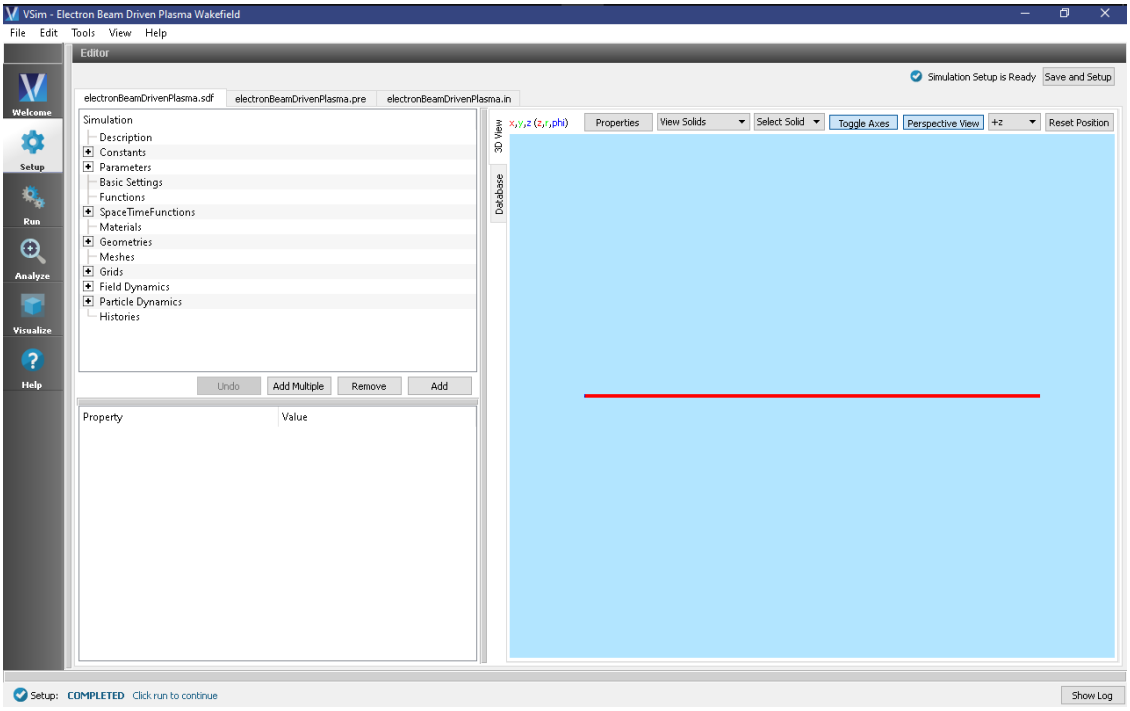


Fig. 5.1: Setup Window for the Electron Driven Plasma Wake example.

generate good results. Consider 12 or 16 cells in both dimensions to avoid a "checkerboard" pattern of numerical noise from developing.

Input File Features

The simulation setup consists of an electromagnetic solver using the Yee algorithm and uses the initBeam macro to set up the initial beam properties. This takes the beam of variable weight particles and calculates self-consistent fields with which to initialize the simulation. As this beam travels near the speed of light, a moving window that co-propagates with the beam is employed. MALs are used on the transverse sides of the window to absorb outgoing waves. The plasma is represented by macro-particles, and both beam and plasma are moved using the Boris push. The particles in the plasma are variably weighted to represent the density ramp. It is assumed the plasma consists of pre-ionized heavy ions, which do not move in the time frame of the simulations.

One can specify the size of the region to be simulated through the constants `LONGITUDINAL_EXTENT` and `TRANSVERSE_EXTENT`, which are measured relative to the longitudinal RMS size `BEAM_LRMS` and transverse RMS size `BEAM_SIGMAR` of the beam. The number of cells is determined by the settings of `LONGITUDINAL_RES` and `TRANSVERSE_RES`, as shown in the figure.

The plasma density is ramped up using a flat top cosine function, by default, over a quarter of the longitudinal size of the simulation window. This can be modified by editing the `STARTRAMP` and `RAMPLEN` parameters.

Running the Simulations

After performing the above actions, continue as follows:

- Proceed to the Run Window by pressing the Run button in the left column of buttons.
- 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 the right pane. This is shown in Fig. 5.2. The run has completed when you see the

output, “Engine output has completed successfully.”

You can expect a 2 core laptop to take a few minutes at the default resolution and run time.

To produce real significant results, a higher resolution is required by changing LONGITUDINAL_RES and TRANSVERSE_RES. Doing so will greatly increase the amount of time to run this simulation. It should also be run for a longer time than the default 3000 steps.

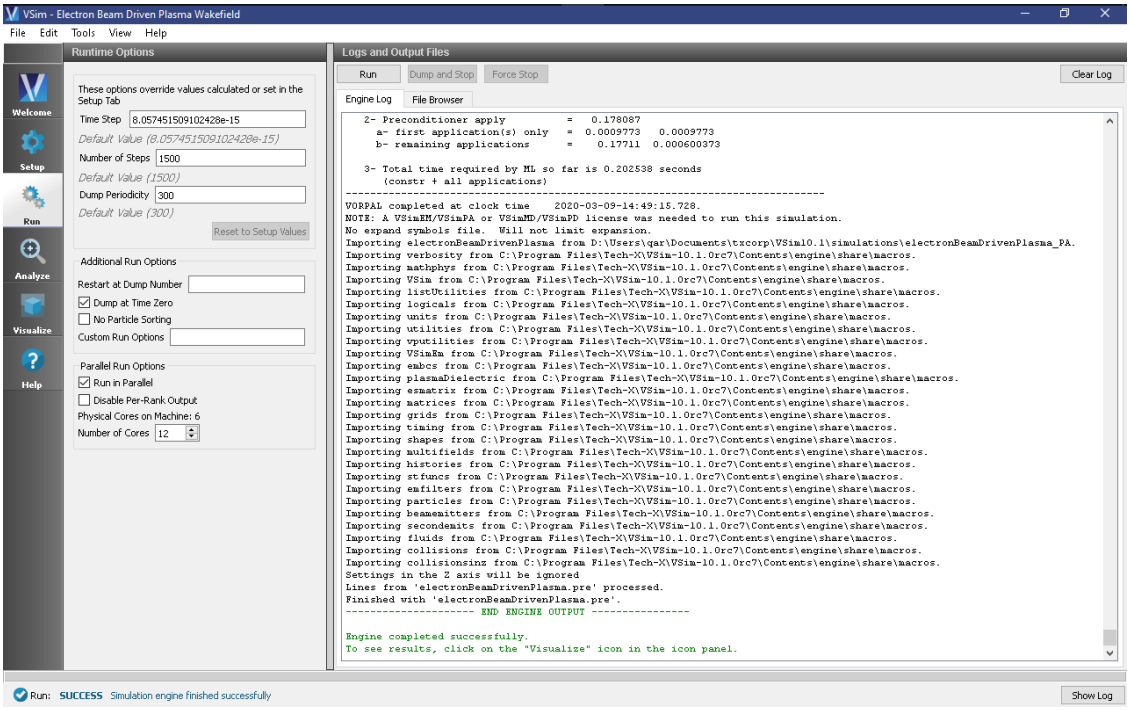


Fig. 5.2: The Run Window.

Visualizing the Output

After performing the above actions, continue as follows:

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

View the electric field generated by the plasma as shown in Fig. 5.3 by doing the following:

- Switch to *Field Analysis* in the *Data View* Controls pane
- Set the *Field* to *E_x*
- Choose the *Horizontal* tab in *Lineout Settings* set the intercept to zero, and click *Perform Lineout*
- Check the *Auto Reset* buttons on both the 2d and the Lineout plots. Sometimes it is necessary to expand the plot size in order for the box to appear. You can do this by pulling the divider between “Visualization Controls” and “Visualization Results” to the left and hiding it. Both the 2d and the Lineout plots should be larger now.
- Move the dump slider forward in time.

The plasma density can be seen as shown in Fig. 5.4 by doing the following:

- Switch to *Data Overview* in the *Data View* drop down
- Expand *Scalar Data*

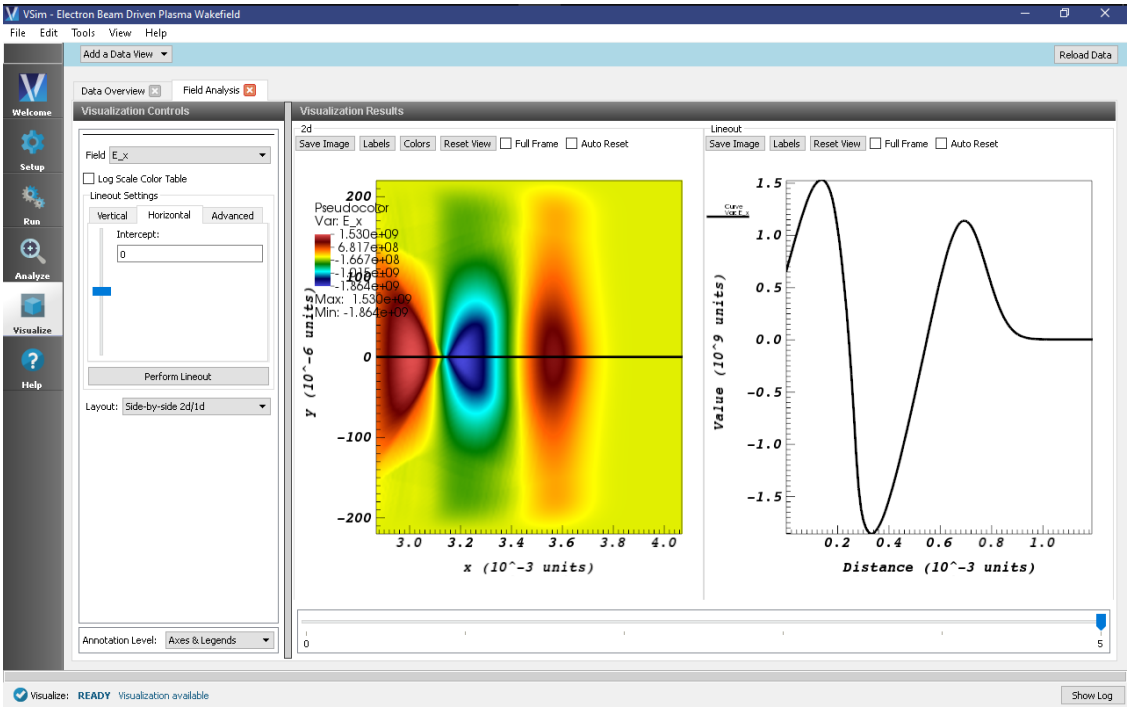


Fig. 5.3: Visualization of the longitudinal electric field as a color contour plot and longitudinal lineout.

- Select *rhoBeam*
- Click *Auto Reset*
- Move slider all the way to the right.

5.2 Beam Driven (text-based setup)

5.2.1 Electron Beam Driven Plasma Wakefield (electronBeamDrivenPlasmaT.pre)

Keywords:

`electron driven, plasma wakefield, CLARA, PARS, AWAKE`

Problem description

This example demonstrates a method to simulate an electron beam driven plasma wakefield accelerator. The electron beam initializes the field using a speed of light frame Poisson equation solve, then the fields and particles are evolved using FDTD EMPIC. We launch the electron beam from x=0 in the positive x direction using the Lorentz boosted Poisson fields to ensure that the simulation is self-consistent from start. The primary bunch generates a region of high field into which one might inject and accelerate a second bunch of charged particles. The example simulation uses parameters that are appropriate to the plasma acceleration research station (PARS) at the CLARA accelerator at Daresbury Laboratory in the UK.

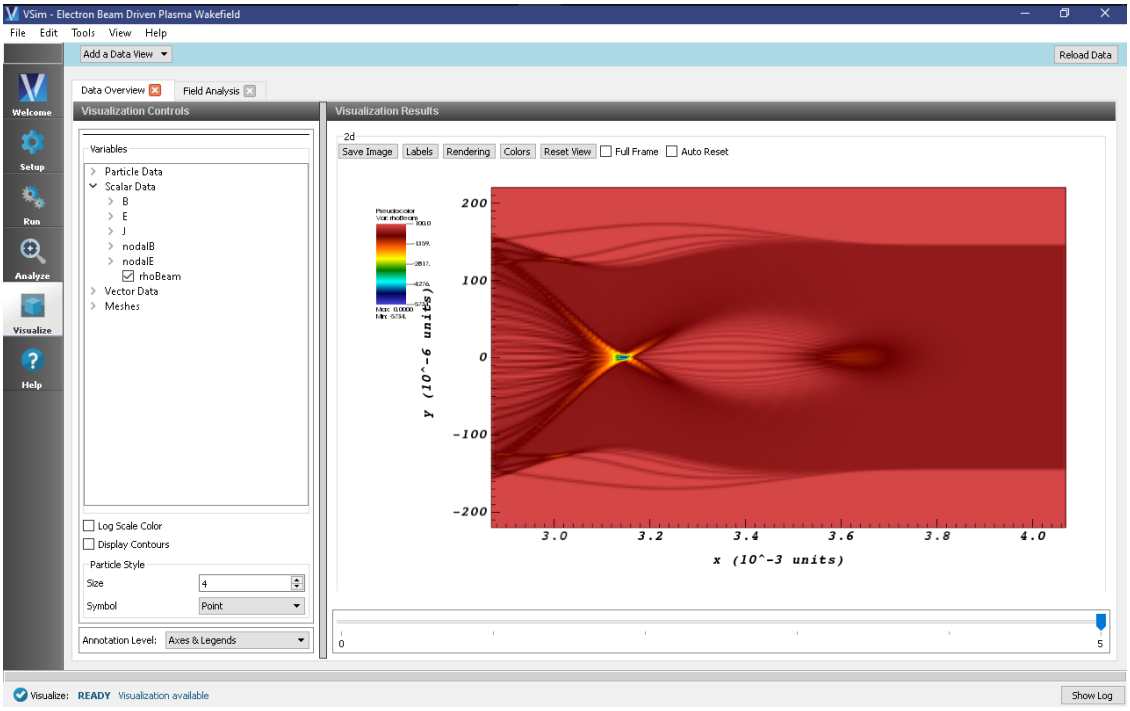


Fig. 5.4: Visualization of the longitudinal plasma density field as a color contour plot.

Opening the Simulation

The Electron Beam Driven Plasma Wakefield example is accessed from within VSimComposer by the following actions:

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim for Plasma Acceleration* option.
- Expand the *Beam Drive Acceleration (text-based setup)* option.
- Select *Electron Beam Driven Plasma Wakefield (text-based setup)* 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 basic variables of this problem should now be alterable via the text boxes in the left pane of the *Setup Window*, as shown in Fig. 5.5.

At this stage one can choose parameters such as `LONGITUDINAL_RES` and `TRANSVERSE_RES` which represent the longitudinal and transverse number of cells per RMS bunch size. The minimum of 6 or default of 8, generates a simulation that will complete reasonably quickly, but is not adequate to generate good results. Consider 12 or 16 cells in both dimensions to avoid a "checkerboard" pattern of numerical noise from developing.

Input File Features

The simulation setup consists of an electromagnetic solver using the Yee algorithm and uses the initBeam macro to set up the initial beam properties. This takes the beam of variable weight particles and calculates self-consistent fields with which to initialize the simulation. As this beam travels near the speed of light, a moving window that co-propagates with the beam is employed. MALs are used on the transverse sides of the window to absorb outgoing waves. The plasma is represented by macro-particles, and both beam and plasma are moved using the Boris push. The particles in

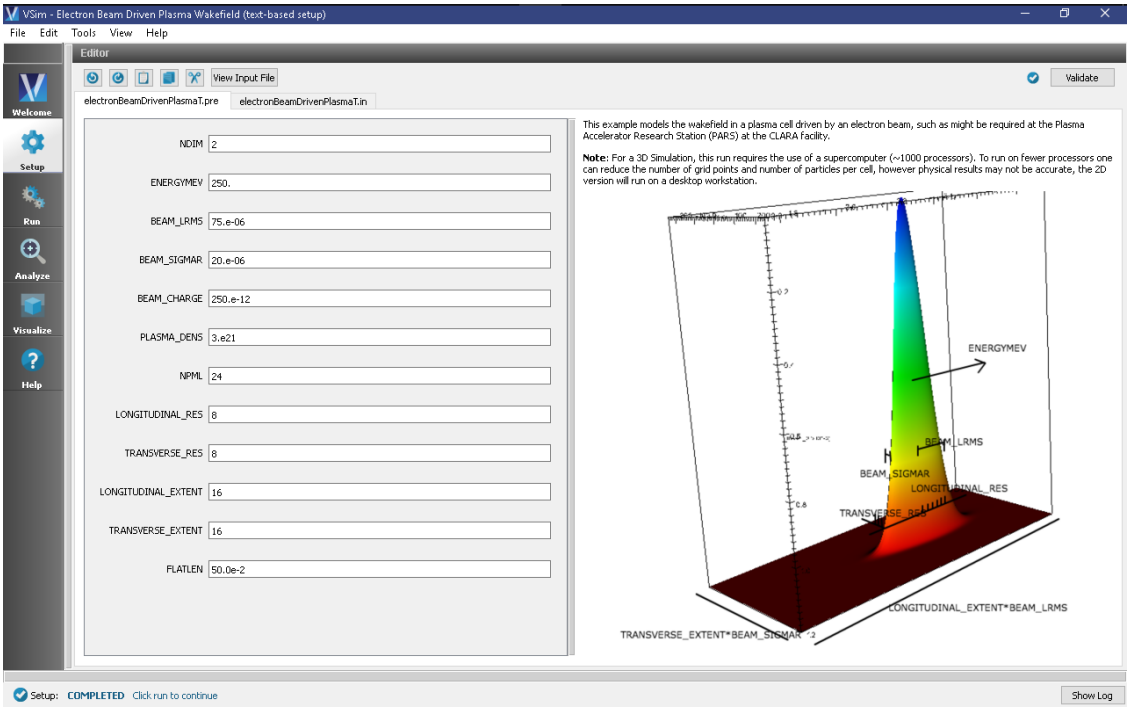


Fig. 5.5: Setup Window for the Electron Driven Plasma Wake example.

the plasma are variably weighted to represent the density ramp. It is assumed the plasma consists of pre-ionized heavy ions, which do not move in the time frame of the simulations.

One can specify the size of the region to be simulated through `LONGITUDINAL_EXTENT` and `TRANSVERSE_EXTENT`, which are measured relative to the longitudinal RMS size `BEAM_LRMS` and transverse RMS size `BEAM_SIGMAR` of the beam. The number of cells is determined by the settings of `LONGITUDINAL_RES` and `TRANSVERSE_RES`, as shown in the figure.

The plasma density is ramped up using a flat top cosine function, by default, over a quarter of the longitudinal size of the simulation window. This can be modified by viewing the input file and editing the `STARTRAMP` and `RAMPLEN` variables.

Running the Simulations

After performing the above actions, continue as follows:

- Proceed to the Run Window by pressing the Run button in the left column of buttons.
- 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 the right pane. This is shown in Fig. 5.6. The run has completed when you see the output, "Engine output has completed successfully."

Running in 2D, you can expect a 2 core laptop to take a few minutes at the default resolution and run time.

To produce real significant results, a higher resolution is required by changing LONGITUDINAL_RES and TRANSVERSE_RES. Doing so will greatly increase the amount of time to run this simulation. It should also be run for a longer time than the default 3000 steps.

The 3D simulation is about 100 times bigger, so 256 cores for a few hours is needed.

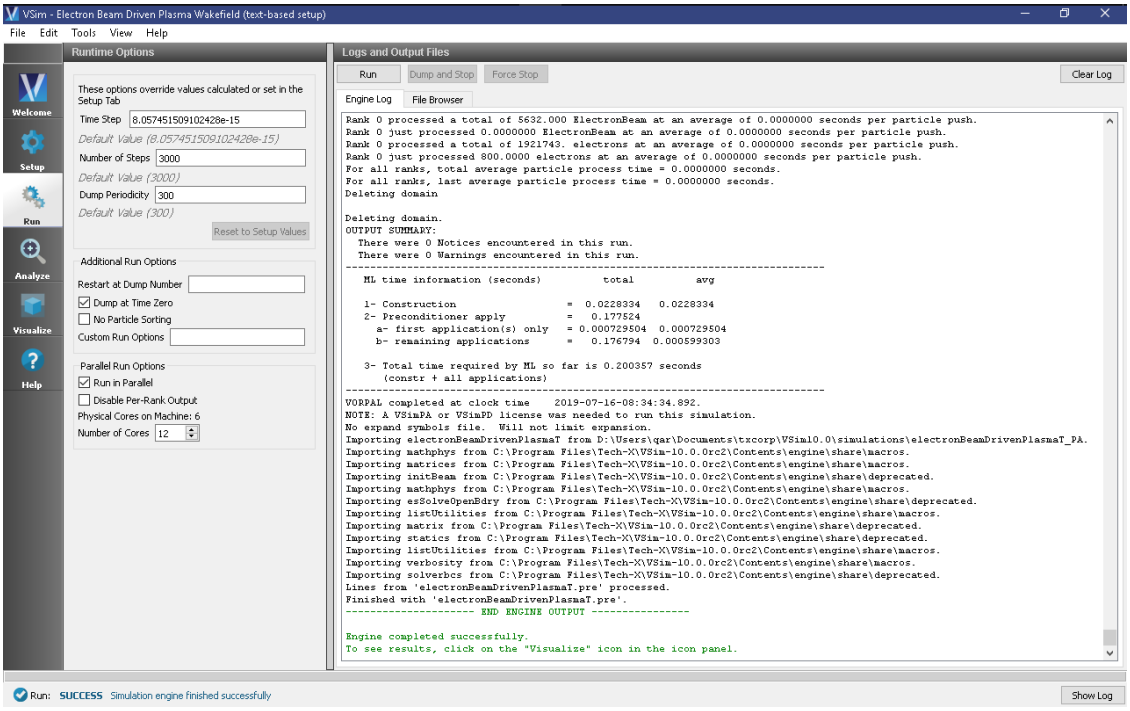


Fig. 5.6: The Run Window.

Visualizing the Output

After performing the above actions, continue as follows:

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

View the electric field generated by the plasma as shown in Fig. 5.7 by doing the following:

- Switch to *Field Analysis* in the *Data View* Controls pane
- Set the *Field* to *ElecFieldPlasma_x*
- Choose the *Horizontal* tab in *Lineout Settings* set the intercept to zero, and click *Perform Lineout*
- Check the *Auto Reset* buttons on both the 2d and the Lineout plots. Sometimes it is necessary to expand the plot size in order for the box to appear. You can do this by pulling the divider between “Visualization Controls” and “Visualization Results” to the left and hiding it. Both the 2d and the Lineout plots should be larger now.
- Move the dump slider forward in time.

The plasma density can be seen as shown in Fig. 5.8 by doing the following:

- Switch to *Data Overview* in the *Data View* drop down
- Expand *Scalar Data*
- Select *rhoPlasma*
- Click *Auto Reset*
- Move slider all the way to the right.

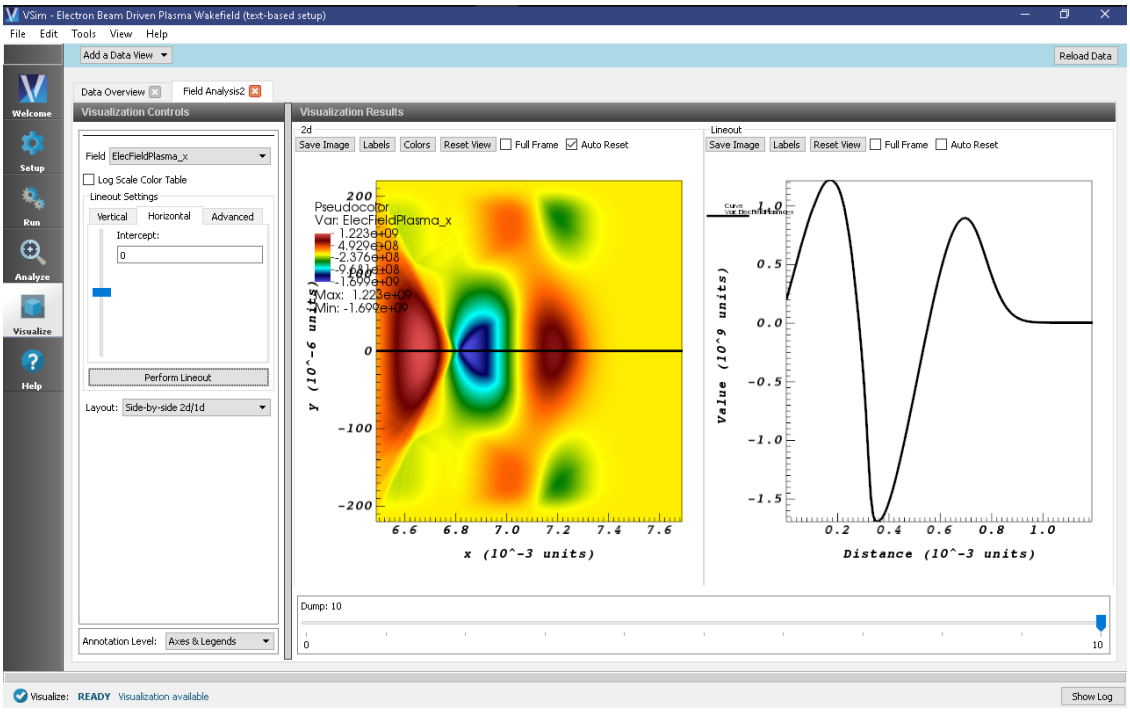


Fig. 5.7: Visualization of the longitudinal electric field as a color contour plot and longitudinal lineout.

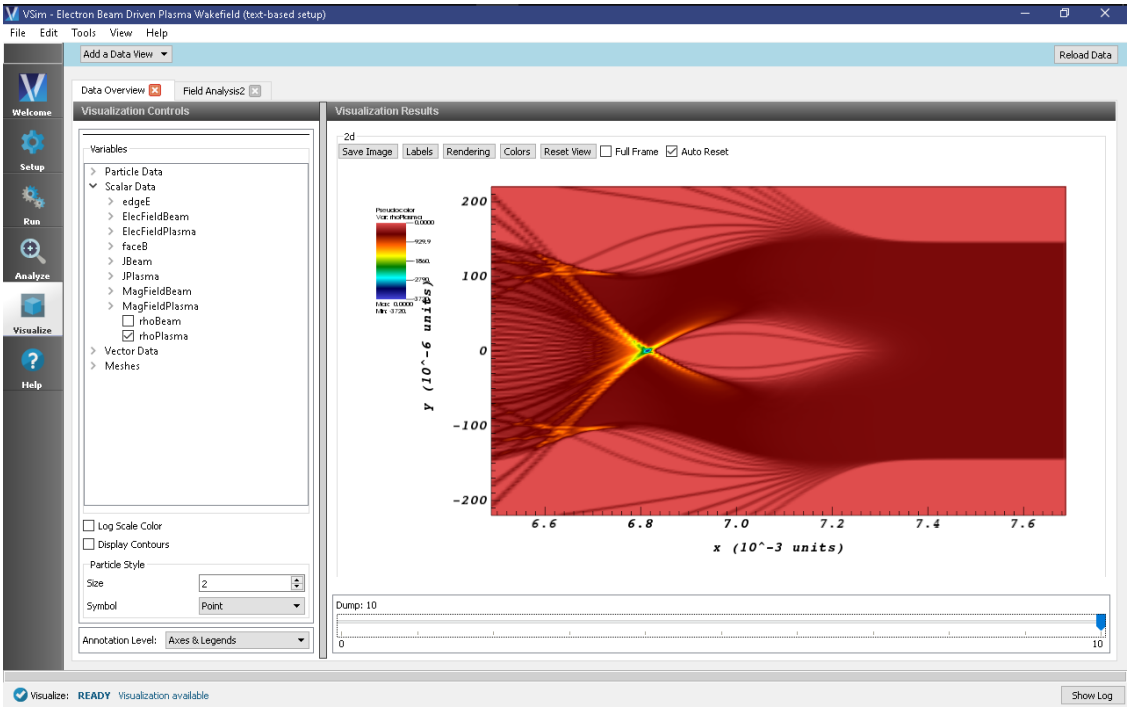


Fig. 5.8: Visualization of the longitudinal plasma density field as a color contour plot.

5.2.2 Dielectric Wall Wakefield Acceleration (dielectricwallaccelerationt.pre)

Keywords:

electron driven, plasma wakefield, CLARA, PARS, AWAKE

Problem description

As an alternative to a full particle in cell approach for accelerator computations is to use a prescribed beam, that is to set the J field directly without using a vector deposition of the current associated with the charge. This is demonstrated in this example which computes the wakefields in a dielectric lined waveguide. The electron beam initializes the field using a custom python technique, which can be adapted for cases where the Poisson solve might not be appropriate (due to memory limitations for example) then the fields (and effective) particles are evolved using FDTD. We launch the electron beam from x=0 in the positive x direction. The primary bunch generates a region of high field into which one might inject and accelerate a second bunch of charged particles. This simulation broadly follows the approach reported on in *[MPSC11]*

Opening the Simulation

The Dielectric Wall Wakefield Acceleration example is accessed from within VSimComposer by the following actions:

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim for Plasma Acceleration* option.
- Expand the *Beam Driven Acceleration (text-based setup)* option.
- Select *Dielectric Wall Wakefield Acceleration (text-based setup)* 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 basic variables of this problem should now be alterable via the text boxes in the left pane of the *Setup Window*, as shown in Fig. 5.9.

Input File Features

The simulation setup consists of an electromagnetic solver using the Yee algorithm and takes advantage of a prescribed current source to avoid computationally intensive particle pushes. The 'top' and 'bottom' y extents of the simulation are metal, and we see the behavior in a dielectric lined waveguide, using a first order accurate dielectric algorithm for the walls.

- `BP` is the beampipe half-width
- `AP` is the dielectric aperture half-width
- `DIELECTRIC_EPSILON` allows the user to experiment with different materials
- `LX` sets the length of the structure and simulation
- `Gamma` sets the relativistic velocity of the beam to be used.

Inside the .pre file, which you can reach by pressing *View Input File*, there are various other settings for modelling this with a laser drive, or with PIC particles.

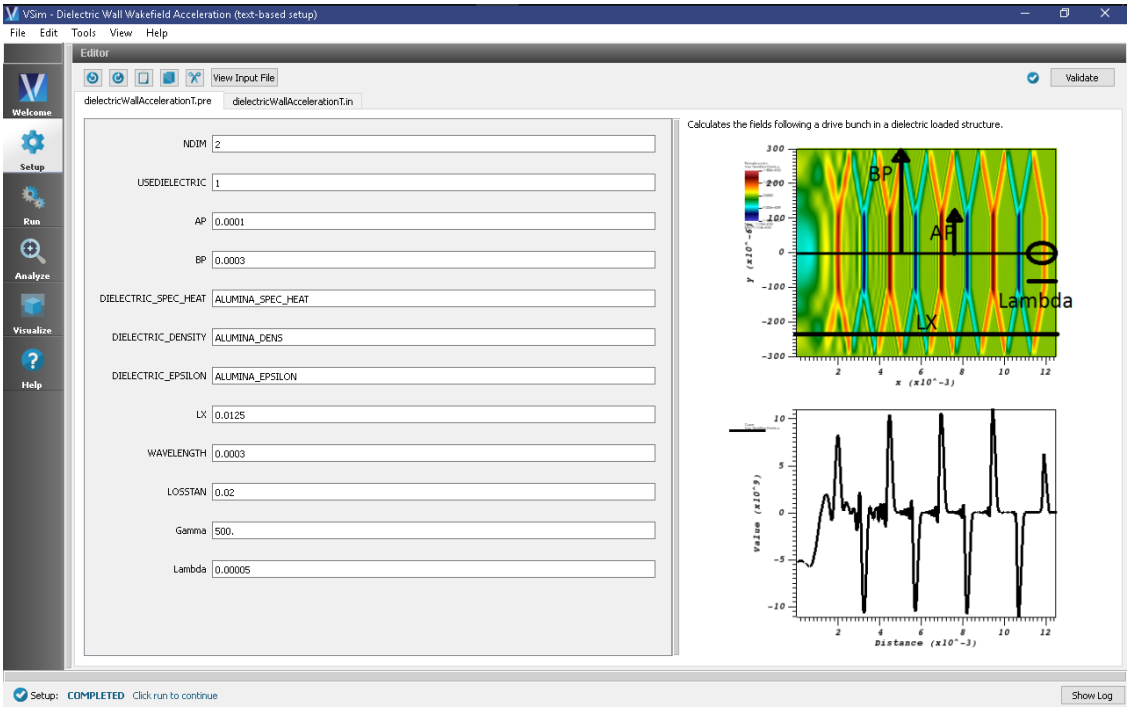


Fig. 5.9: Setup Window for the Dielectric Wall Wakefield Acceleration example.

Running the Simulations

After performing the above actions, continue as follows:

- Proceed to the Run Window by pressing the Run button in the left column of buttons.
- 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 the right pane. This is shown in Fig. 5.10. The run has completed when you see the output, "Engine output has completed successfully."

Running in 2D, you can expect a 2 core laptop to take a few minutes at the default resolution and run time.

Visualizing the Output

After performing the above actions, continue as follows:

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

View the electric field generated by the plasma as shown in Fig. 5.11 by doing the following:

- Switch to *Field Analysis* in the *Data View* Controls pane
- Set the *Field* to *E_x*
- Choose the *Horizontal* tab in *Lineout Settings* set the intercept to zero, and click *Perform Lineout*
- Select stacked 2d/1d view.
- Check the *Auto Reset* buttons on both the 2d and the Lineout plots. Sometimes it is necessary to expand the plot size in order for the box to appear. You can do this by pulling the divider between "Visualization Controls" and "Visualization Results" to the left and hiding it. Both the 2d and the Lineout plots should be larger now.
- Move the dump slider forward in time.

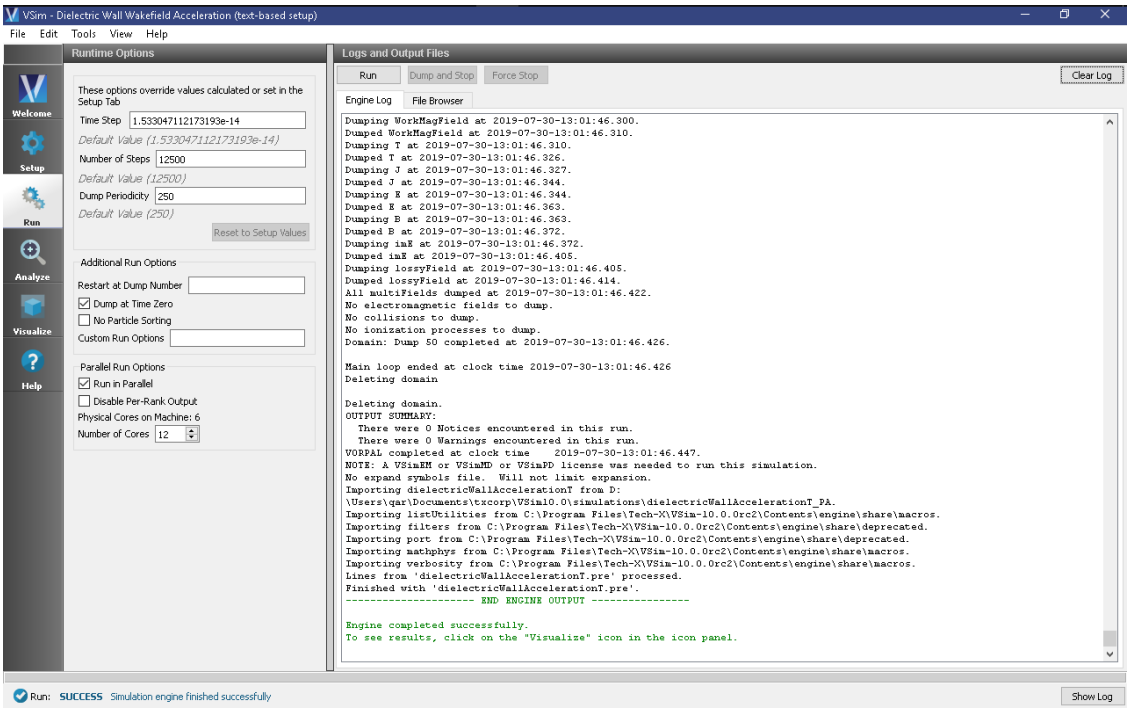


Fig. 5.10: The Run Window.

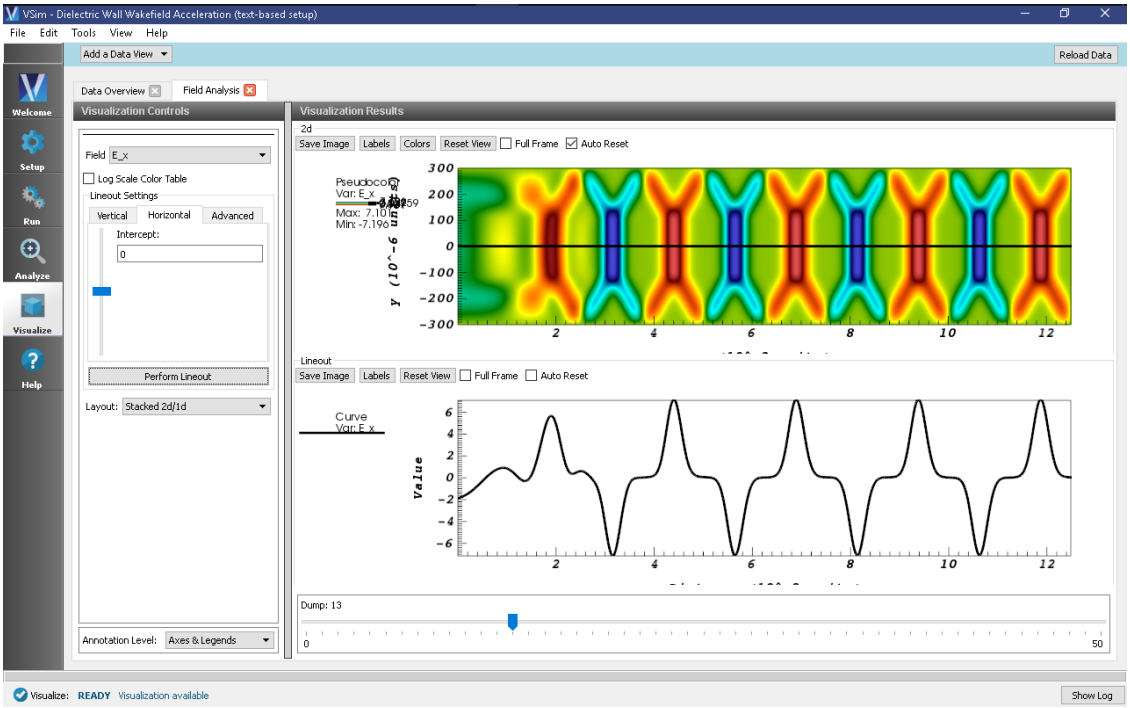


Fig. 5.11: Visualization of the longitudinal electric field as a color contour plot and longitudinal lineout.

Further experiments

There are two bunch shapes provided to start with, you can look at the effect of harmonics in the bunch in the longitudinal field output. There is also a commented block of code that can be switched in that shifts the dielectric into the domain a little, so one can see the process of the electrons beam forming the pattern.

Also, one may add a species propagating from the left hand side to witness the wake, and observe the behavior of particles in or out of phase with the wake.

5.3 Laser Driven

5.3.1 Laser Ionization (laserIonization.sdf)

Keywords:

electromagnetic, particle in cell, field ionization, moving window

Problem description

This example launches an electromagnetic laser pulse into a homogeneous volume of neutral argon gas. The field strength is significant enough to ionize the argon to multiple ionization states, which are included in the simulation. The neutral gas density is depleted as the ionization occurs, with layers of argon atoms at increasing ionization levels towards the center of the Gaussian beam.

This simulation can be performed with a VSimPD license.

Opening the Simulation

The Laser Ionization example is accessed from within VSimComposer by the following actions:

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim for Plasma Discharges* option.
- Expand the *Processes* option.
- Select "Laser Ionization" 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. 5.12. 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.

Simulation Properties

Constants are set up to allow setting the laser amplitude and the neutral argon density (1/m^3).

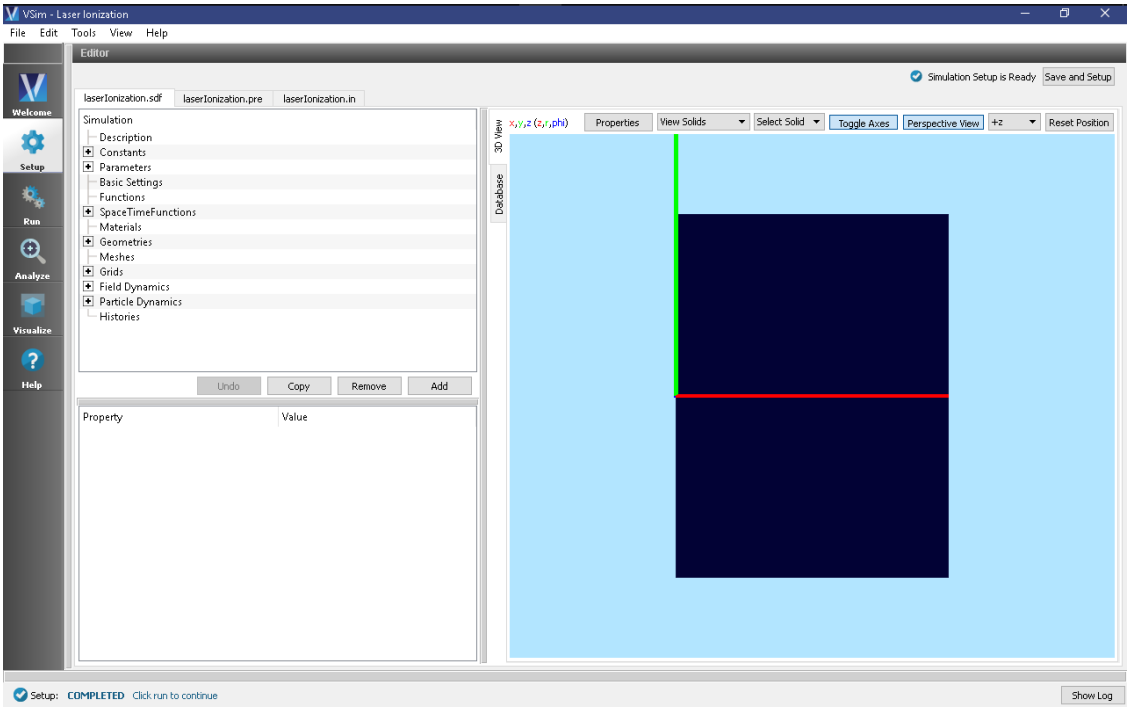


Fig. 5.12: Setup Window for the Laser Ionization example.

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.
- 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 the right pane. The run has completed when you see the output, "Engine completed successfully." A snapshot of the simulation run completion is shown in Fig. 5.13.

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 magnitude, expand *Scalar Data*, expand *E* and select *E_magnitude*. Scrolling through time (by moving the slider at the bottom of the window) will show the laser pulse propagating across the simulation domain. Next, untick the *E_magnitude* and instead tick *neutralArgon*. You can now see the depletion of the neutral background gas as the laser passes through. This will appear the same as in Fig. 5.14.

Further Experiments

Try adding more charge states of Argon (past 6+) and find the limit of ionization that is achievable with this laser pulse.

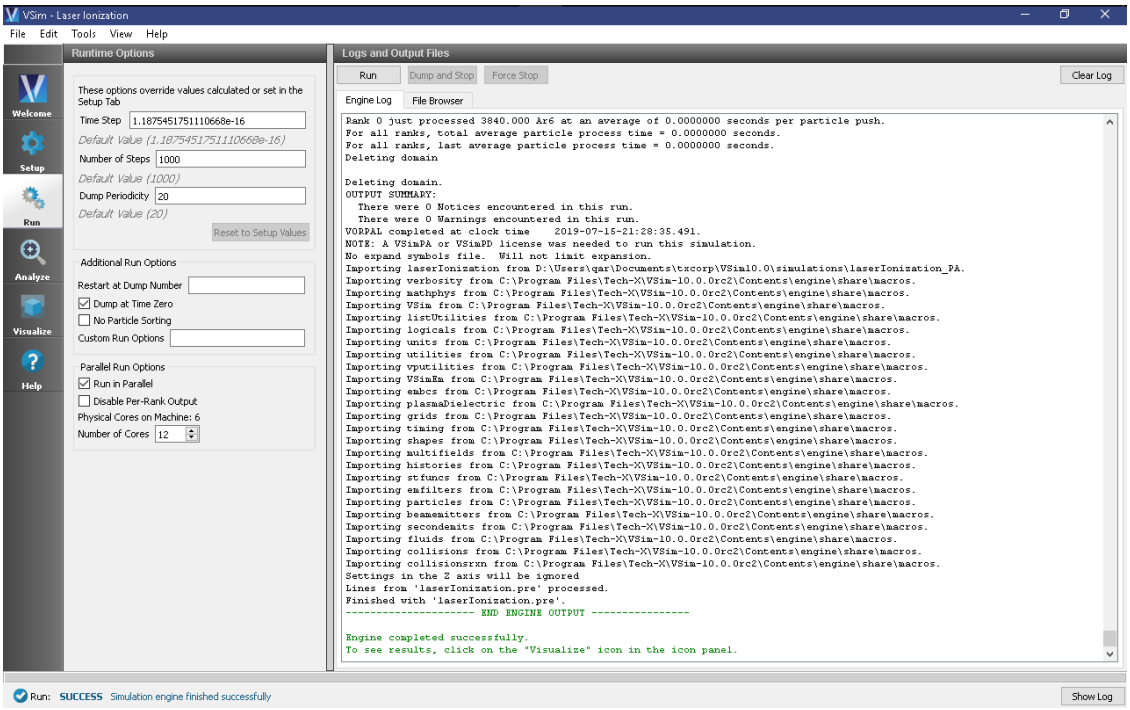


Fig. 5.13: The Run Window at the end of execution.

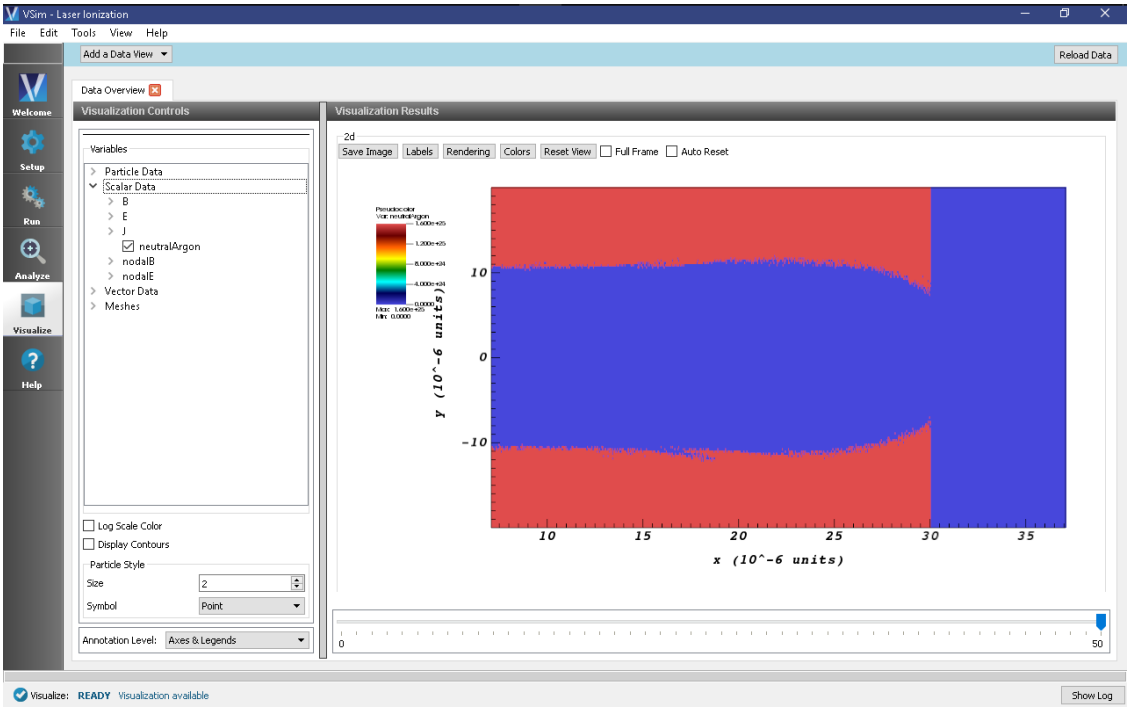


Fig. 5.14: The ionized charge states of argon during laser pulse propagation

5.3.2 Laser Plasma Accelerator (laserPlasmaAccel.sdf)

Keywords:

`Laser Plasma Accelerator`

Problem description

This example demonstrates the use of VSim to simulate a simple laser-plasma accelerator problem using the full PIC algorithm.

An intense, short laser pulse propagating through a plasma can lead to the separation of electrons and ions capable of producing accelerating electric fields of hundreds of GV/m *[GTVT+04]*. VSim is capable of simulating laser plasma accelerators (laserPlasmaAccel) using several different models: envelope, fluid and full particle-in-cell (PIC).

Here we look at the full PIC model with a 1-mm long plasma with uniform density of $1.e25\ \mathrm{m}^{-3}$. A gaussian laser pulse, defined by the transverse electric field

$$E_y = E_0(0.5 - 0.5\cos(\pi t/T))H(T-t)\exp(-(y^2+z^2)/(w_0^2 D_y D_z))\cos(\omega t + \phi_y + \phi_z)$$

where w_0 is the radius at which the wave amplitude drops to $1/e$, T is the temporal duration of the pulse,

$$D_y = 1 + (F/Z)^2$$
$$D_z = 1 + (F/Z)^2$$

are the squares of amplitude reductions from being the launching at a distance, F, from the focus, Z is the Rayleigh length, and

$$\phi_y = -(F/Z)(y/w_0)^2/D_y$$
$$\phi_z = -(F/Z)(z/w_0)^2/D_z$$

are the Gouy phases.

The laser is launched from the left side of the box. The laser amplitude is determined through the normalized vector potential $A_0 = eE_0/\omega m_e c$, where ω is the laser angular frequency.

The simulation setup consists of an electromagnetic solver using the Yee algorithm. The laser pulse is launched from the left side of the window using the pre-defined gaussian pulse launcher at the left boundary. Simple conducting boundary conditions are used at the top and sides. As such, one must ensure that waves reflected off the top or bottom do not get into the simulation, and that no waves hit the right boundary to be reflected back into the simulation.

The plasma is represented by macro-particles which are moved using the Boris push. The particles are variably weighted to represent the density ramp.

The input file allows one to set up plasma and laser parameters. The simulation box size is determined as a function of the laser length and spot size. The resolution was set to have about 24 cells per wavelength longitudinally and 3 transversely. The time step is chosen to be very close to the courant condition limit in order to have good dispersion.

This simulation can be performed with a VSimPA license.

Opening the Simulation

The Laser Plasma Accelerator example is accessed from within VSimComposer by the following actions:

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim for Plasma Acceleration* option.
- Expand the *Introductory Examples* option.

- Select “Laser Plasma Acceleration” 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. 5.15. 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`.

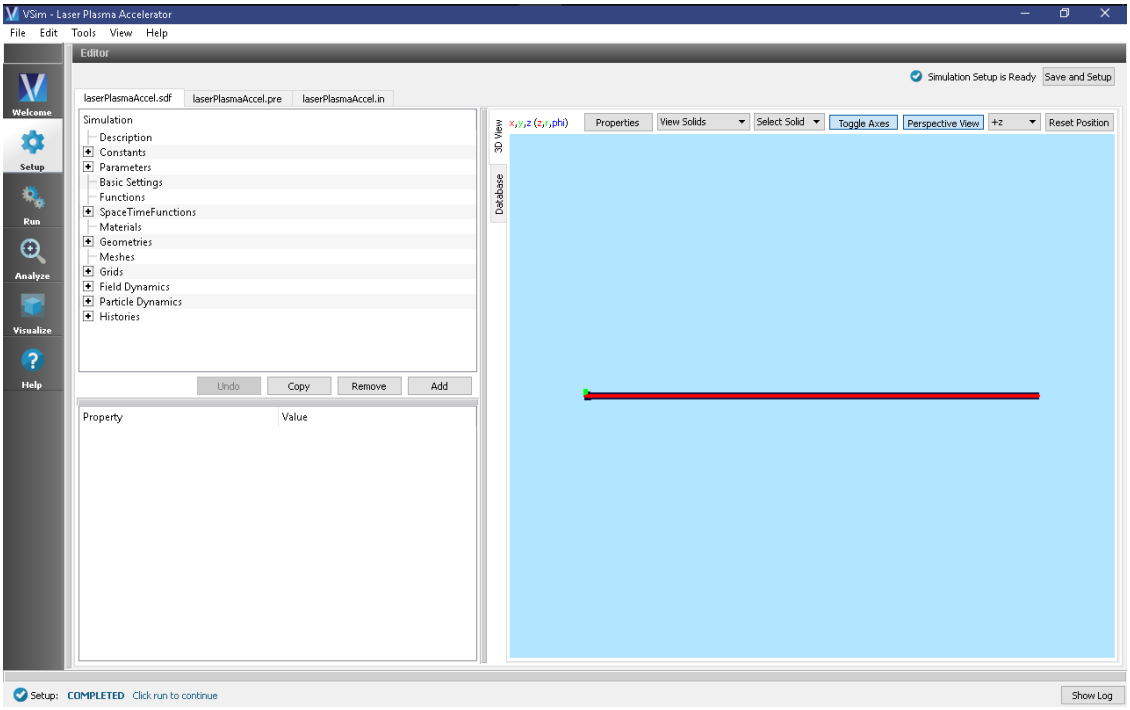


Fig. 5.15: Setup Window for the Laser Plasma Acceleration example.

The Setup Window shows a very long simulation. However, the full length is not simulated at any one time. Instead a moving window is used to simulate only the region where dynamics is occurring. The moving window can be seen in Fig. 5.16 as a small box on at the left end of the electron loader.

To see this view, first toggle off the axes by clicking the *Toggle Axes* button, then expand the *Particle Dynamics* → *KineticParticles* → *electrons0* item in the setup tree and lick *particleLoader*. Finally, zoom in and tranlated to the left to see the left edge of the particle loader and the darker grid. If one wishes to simulate this for longer distances, one can set the Parameter, `NUM_XLENGTHSS`, to a larger number; 400 is more than enough.

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.
- To run the file, click on the *Run* button in the upper left corner of the window. You will see the output of the run in the right pane. The run has completed when you see the output, “Engine completed successfully.” See Fig. 5.17.

Running in 2D, this simulation uses around 225,000 cells and nearly 200,000 particles for 20,000 time steps. The run takes about an hour on a 4-core 2.5 GHz I7.

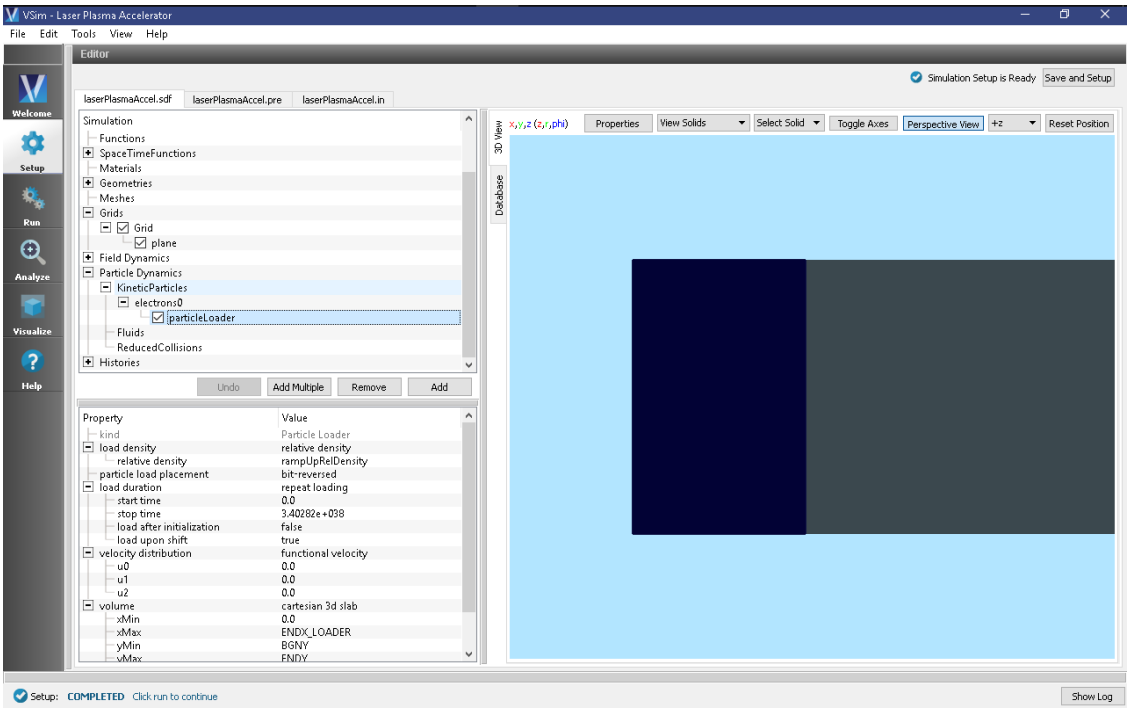


Fig. 5.16: Zoom of Setup Window for the Laser Plasma Acceleration example showing size of grid.

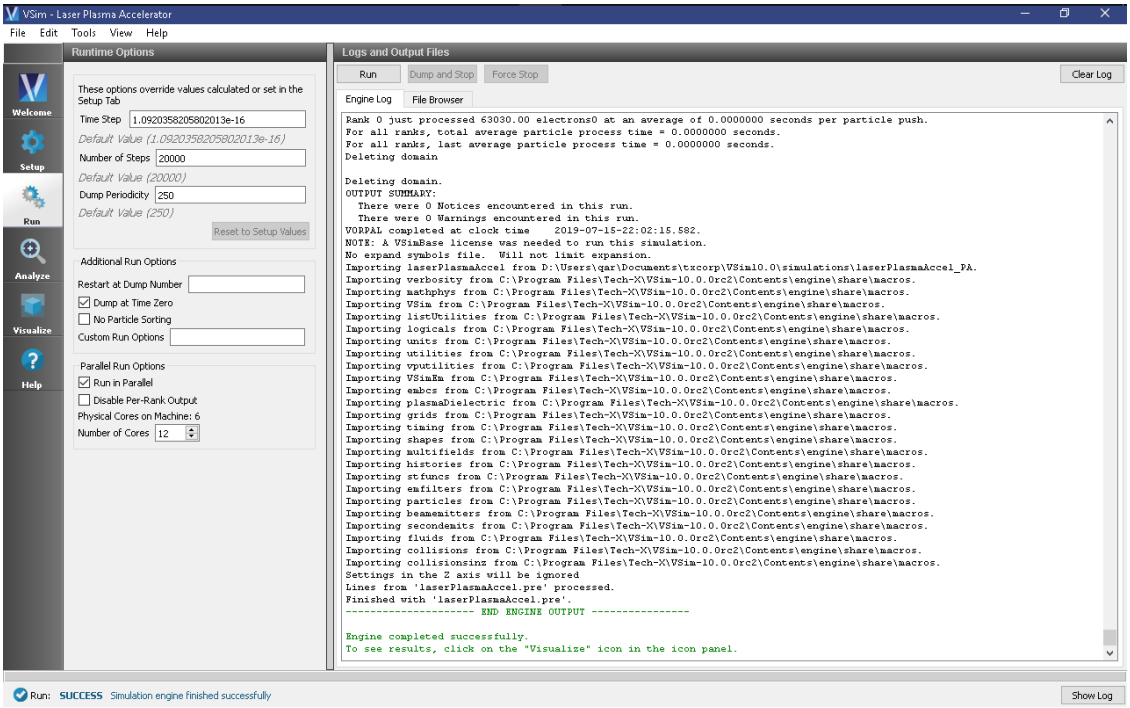


Fig. 5.17: The Run Window at the end of execution.

Visualizing the results

After performing the above actions, click on *Visualize* in the column of buttons at the left. For all plots, it is useful to keep Auto Reset on so that the window moves with the data.

To view the electric field, switch to Field Analysis in the Data View drop-down menu. From the Field drop-down menu, choose the desired component of the electric field, `E`. The `depField` field is the current density. See Fig. 5.18.

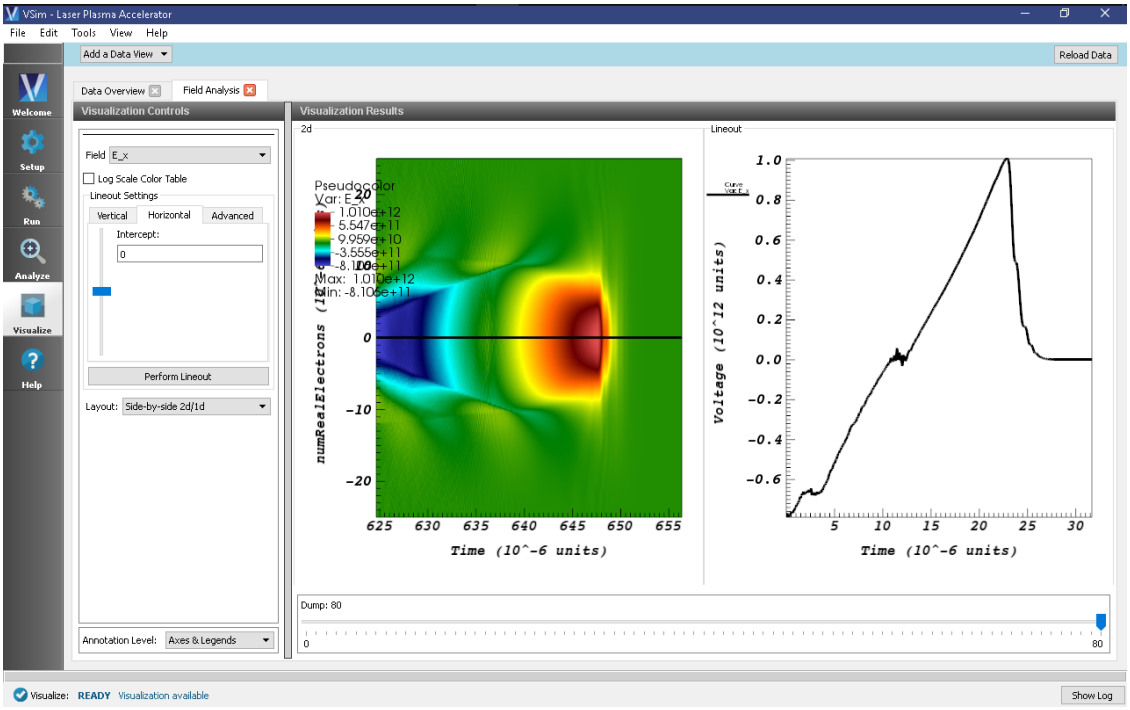


Fig. 5.18: The output of the run shows the accelerating field E_x after about 2 picoseconds.

The acceleration of the particles can be seen by viewing the (x) component of the velocity. To do this, switch to the Phase Space Data View in VSimComposer and set the X-axis variable to electrons0_x, and the Y-axis variable to electrons0_ux. Then click Draw. You will see a color-coded picture of particle velocity like in Fig. 5.19.

Further Experiments

Try increasing or decreasing the intensity of the laser pulse through the parameter `A_0` and see the effect on the shape of the plasma wakefield.

5.4 Laser Driven (text-based setup)

5.4.1 Colliding Pulse Injection (collidingPulseInjT.pre)

Keywords:

`laser plasma accelerator, controlled injection, colliding laser pulses`

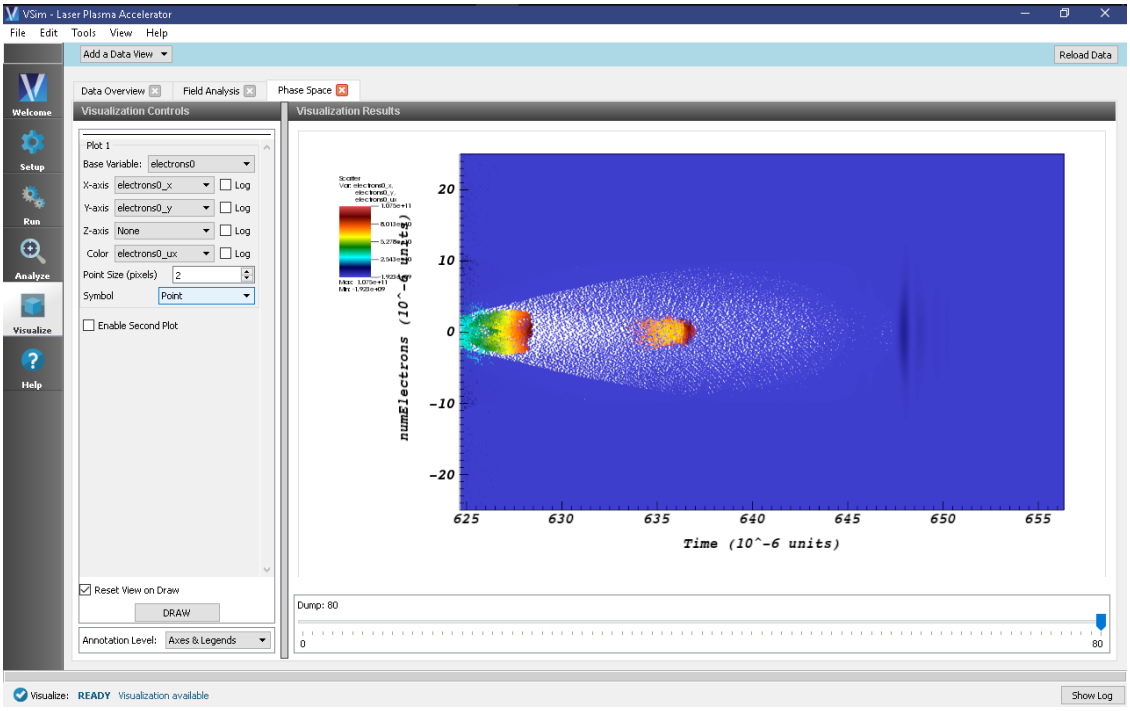


Fig. 5.19: Phase-space plot of plasma electrons at t = 2.1 picoseconds.

Problem description

This example demonstrates the use of VSim to simulate controlled injection in a laser-plasma accelerator using colliding laser pulses *[CMRB+10]*. Two laser pulses are launched from opposite sides (one from the left side and the other one from the right side of the box) and propagate in opposite directions. The laser pulse coming from the left side is the main pulse that drives the plasma wake. The laser pulse coming from the right is the collider pulse, with much lower intensity than the main pulse. It can also propagate with a small angle with respect to the main pulse propagation axis. When the two lasers collide they create a slow beat wave, which allows electrons of the background plasma to be trapped and be accelerated by the wakefield driven by the main pulse.

In this example, the laser pulses are polarized in the y direction and both have a Gaussian profile defined by

$$E_y = E_{\mathrm{pump\_(L,R)}} \exp(-x^2/\mathrm{LPUMP\_(L,R)}^2) \exp(-(y^2+z^2)/\mathrm{W0\_(L,R)}^2)$$

where L and R refer to the left and right pulse respectively. The laser intensity is defined through the normalized vector potential

$$\mathrm{APUMP\_(L,R)} = eE_{\mathrm{pump\_(L,R)}}/\omega_{\mathrm{(L,R)}} m_e c$$

where $\omega = 2\pi c/\mathrm{WAVELENGTH\_(L,R)}$ is the laser frequency.

The pulses enter a plasma channel with density on axis `DENSITY0` through a density ramp of length 20 μ m.

This simulation can be performed with the VSimPA or VSimPD license.

Opening the Simulation

The Colliding Pulse Injection example is accessed from within VSimComposer by the following actions:

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim for Plasma Acceleration* option.
- Expand the *Laser Driven Acceleration (text-based setup)* option.
- Select *Colliding Pulse Injection (text-based setup)* 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 basic variables of this problem can now be changed via the text boxes in the left pane of the *Setup Window*, as shown in Fig. 5.20.

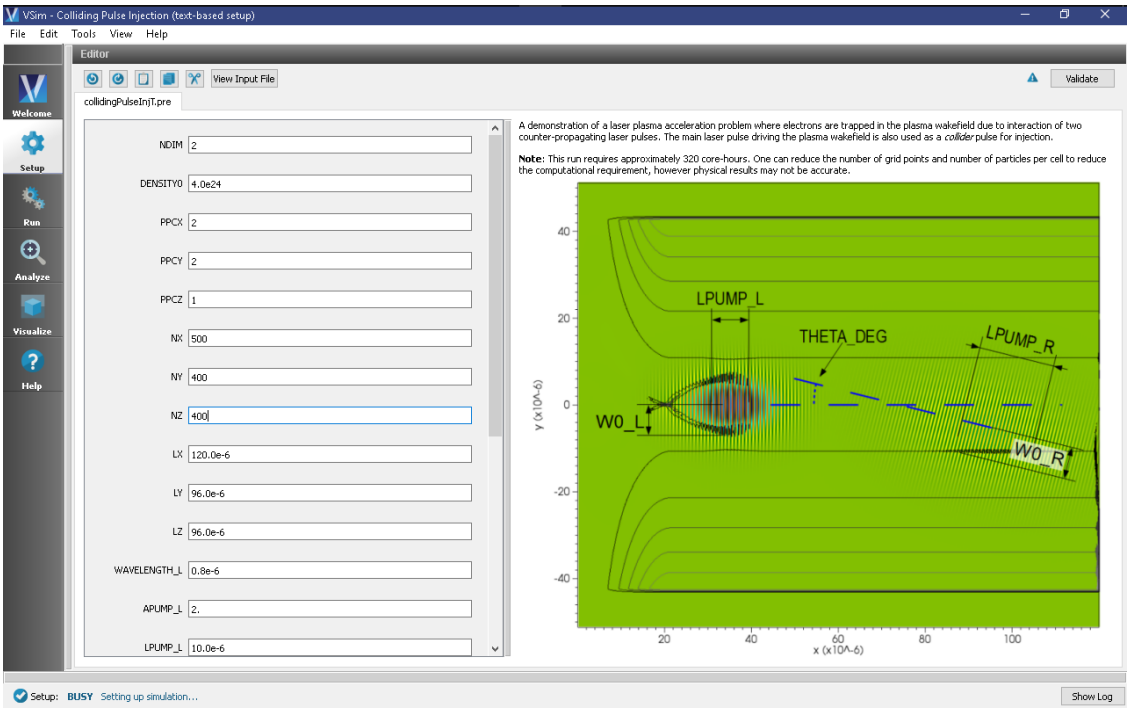


Fig. 5.20: Setup Window for the Colliding Pulse Injection example.

Input File Features

The simulation setup consists of an electromagnetic solver using the Yee algorithm. Two laser launchers are used, one from the left edge and the other from the right edge of the window. PMLs are used on the transverse sides of the window to absorb outgoing waves. The plasma is represented by macroparticles which are moved using the Boris push. The particles are variably weighted to represent the density ramp, and they have a unique tag. The current deposited by the particles is smoothed using four passes of the 1-2-1 filter and subsequently applying a compensator.

The input file allows one to set up both lasers, plasma and grid parameters.

Running the Simulations

Running in 2D, this simulation uses around 2,600,000 cells and nearly 10^7 particles. This run requires bout 320 core-hours for the full 156,000 steps on a 2.5 GHz I7. On less powerful hardware, one can reduce the number of steps to

15000 see just the collision or one can reduce the number of grid points and number of particles per cell to see more of the evolution, but physical results may not be accurate.

To run on local hardware do

- Proceed to the Run Window by pressing the Run button in the left column of buttons.
- Set the number of steps to 5000 and the dump periodicity to 500 in order to see the initial evolution. The collision occurs at step 4500 (dump 9).
- Run in parallel with as many physical cores as are on your machine, because this is a computationally intense problem. Even with the reduced number of steps, this run can take up to 7 hours on four cores for 5000 steps, depending on the processor.
- To run the file, click on the *Run* button in the upper left This is shown in Fig. 5.21. The run has completed when you see the output, "Engine completed successfully." in this same pane.

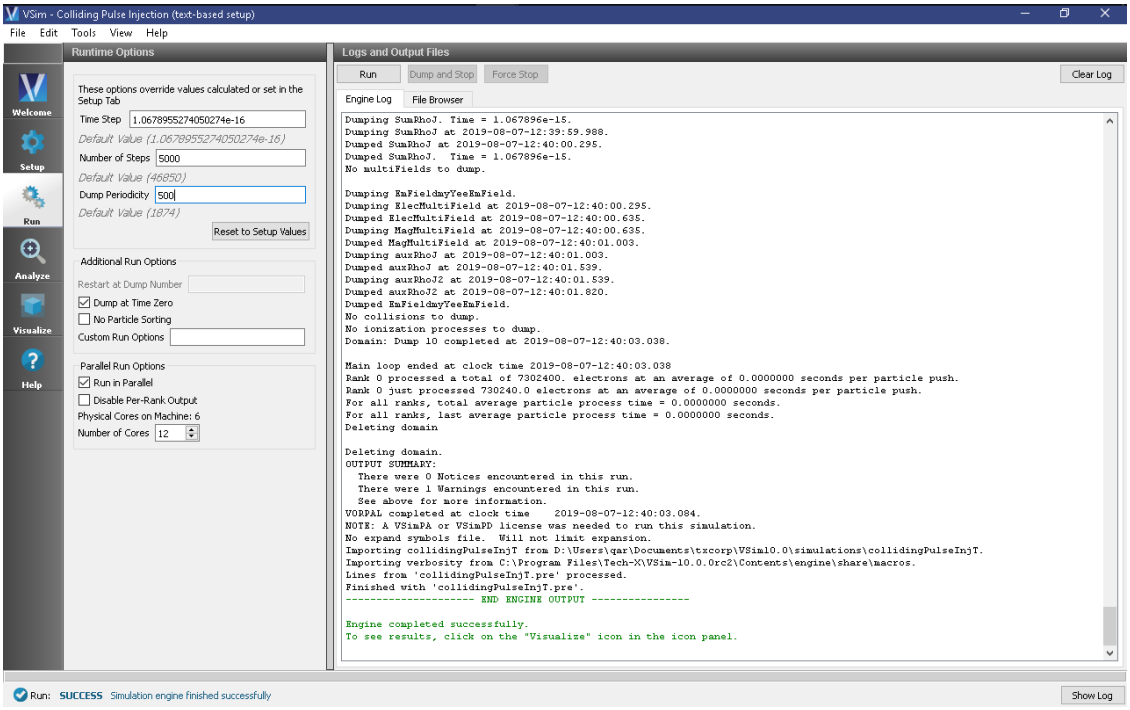


Fig. 5.21: The Run Window.

Alternatively, copy collidingPulseInjT.pre to your more powerful hardware and run it through the command line or submit it to your job queue.

Visualizing the Output

If you have run the job on a remote computer, you would now need to copy back the files that you want to visualize locally into the local directory in which one has the input file open. E.g.,

```
for dmpnum in 0 8 9 10; do
  scp mybigcomputer.mydomain:myspace/collidingPulseInjT_*_${dmpnum}.h5 .
done
```

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

- To view the transverse electric field, switch to the *Data Overview* in the Visualization Controls pane.

- From the Field drop down menu, choose the y component of the `ElecMultiField`.
- Set the scale by clicking on the *Colors* button, then setting the minimum to -2e12 and the maximum to 2e12. Click the Auto Reset check box.
- Move the dump slider to position 8, then 9, then 10 to see the pulses collide.

The collision of the pulses is then seen as shown in Fig. 5.22 below.

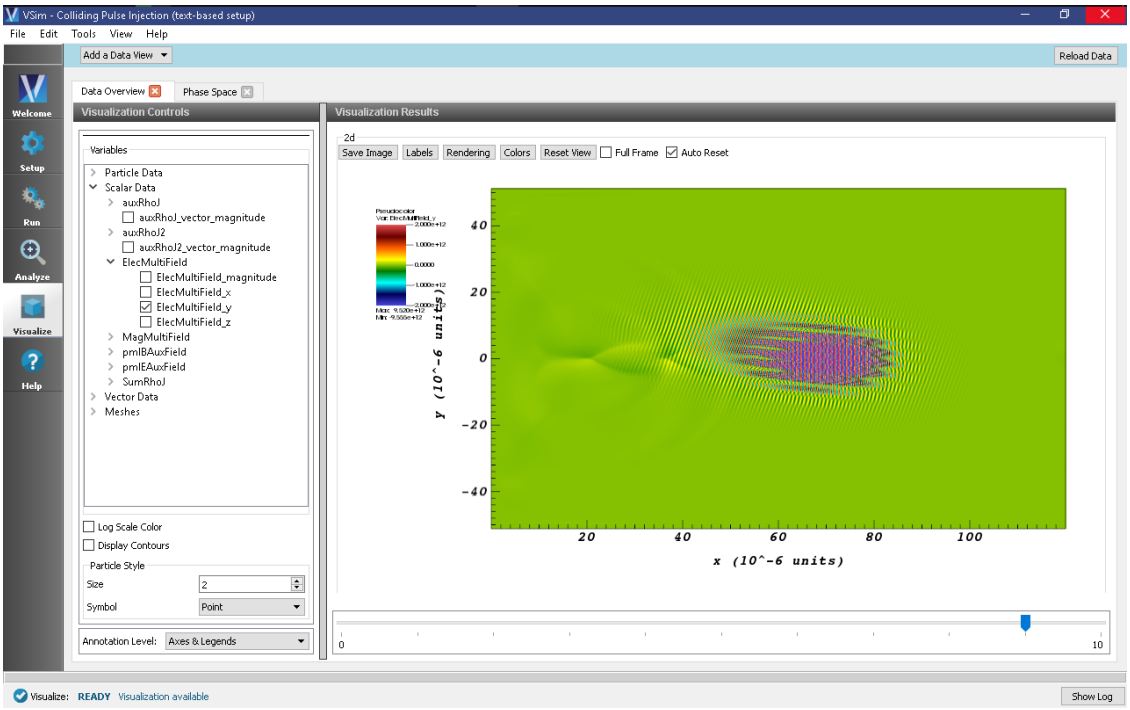


Fig. 5.22: Visualization of the transverse electric field as a color contour plot and longitudinal lineout.

The x-component shows the wake field of the left incoming pulse and some of the electromagnetic field of the incoming collider pulse. The wake field can be better seen by clicking on the *Colors* button and setting the min and max to be -+ 1.e11. The plasma density can be seen in the zeroth component of the SumRhoJ field.

Particle phase-space can be seen by switching to the *Phase Space Data View* in the Controls pane. Fig. 5.23 shows the particle longitudinal momentum as a function of the longitudinal coordinate just after the collision.

Continuing the simulation

The simulation to this point has allowed one to study the initial injection of particles up to high energy, so that they can be trapped by the wake. One can now continue this simulation to study the acceleration in the wake. Since this simulation stopped at dump 10, one can now set the Addition Run Options to unclick *Dump at Time Zero* and then set *Restart at Dump Number* to 10. Since at this point, the evolution changes more slowly, one can set the *Number of Steps* to 10000 and the *Dump Periodicity* to 5000. Again hit *Run*.

At any time one sees that another data dump has occurred, one can switch over to the Visualize pane and hit *Reload Data* to view the new available data, any of the fields or particles as before.

5.4.2 Ionization Injection (fieldIonizeT.pre)

Keywords:

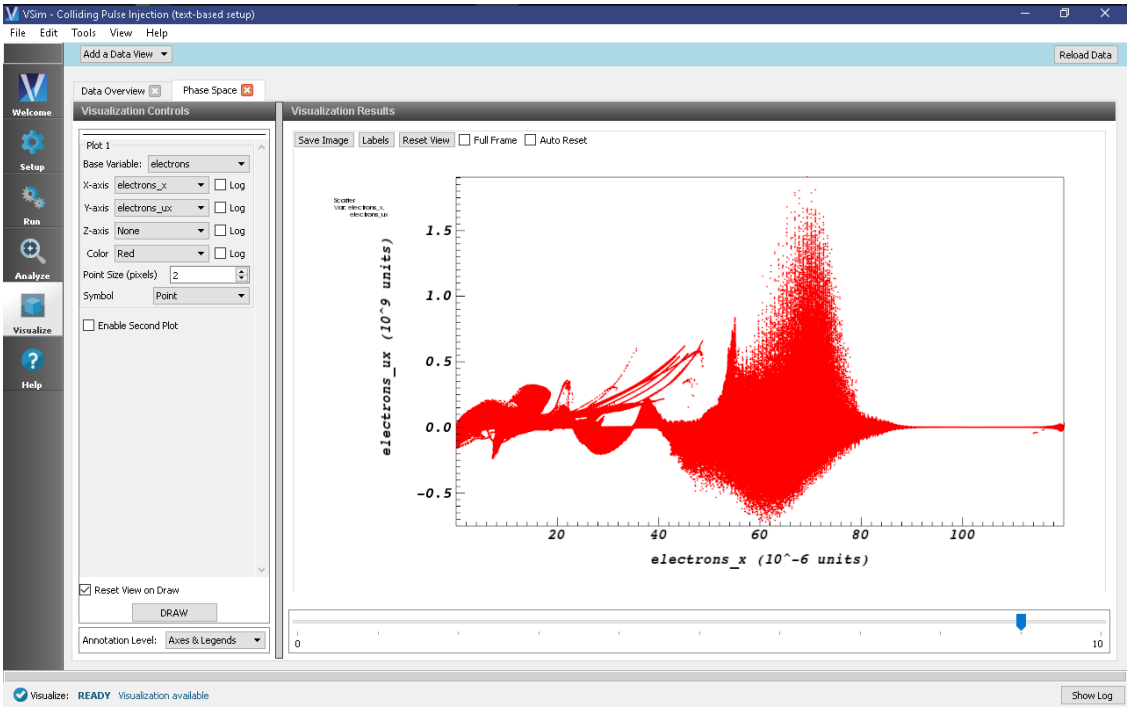


Fig. 5.23: Visualization of the particle longitudinal phase-space during collision. Particles kicked up into the trapped region by the colliding pulses. One can see the acceleration of particle to high energy in later dumps.

laser plasma accelerator, controlled injection, ionization of high-Z gas

Problem description

This example demonstrates the use of VSim to simulate ionization-induced injection in a laser plasma accelerator *[CES+12]*. An intense laser pulse propagates up a plasma density ramp into a uniform plasma, which creates a wakefield. Neutral nitrogen atoms are added to the pre-ionized gas at the beginning of the plasma, where the laser pulse field ionizes them. If the electrons released from the nitrogen ionization are at the correct position relative to the wakefield phase, they can be trapped and accelerated to high energy *[CCMG+13]*.

The laser envelope has a Gaussian profile defined at the waist position by (`X_0_LASER`):

$$E_z = E_0 \exp\left(-x^2/LPUMP^2\right) \exp\left(-(y^2+z^2)/W\_0^2\right) \sin\left(\omega_0 t\right)$$

where $\omega_0 = 2\pi c$ / `WAVELENGTH` is the laser frequency. The laser amplitude is defined through the normalized vector potential `A_0` = $eE_0/\omega_0 m_e c$.

This simulation can be performed with a VSimPA license.

Opening the Simulation

The Ionization Injection example is accessed from within VSimComposer by the following actions:

- Select the *New* → *From Example...* menu item in the *File* menu.

- In the resulting *Examples* window expand the *VSim for Plasma Acceleration* option.
- Expand the *Laser Driven Acceleration (text-based setup)* option.
- Select "Ionization injection (text-based setup)" 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 basic variables of this problem should now be alterable via the text boxes in the left pane of the *Setup Window*, as shown in Fig. 5.24.

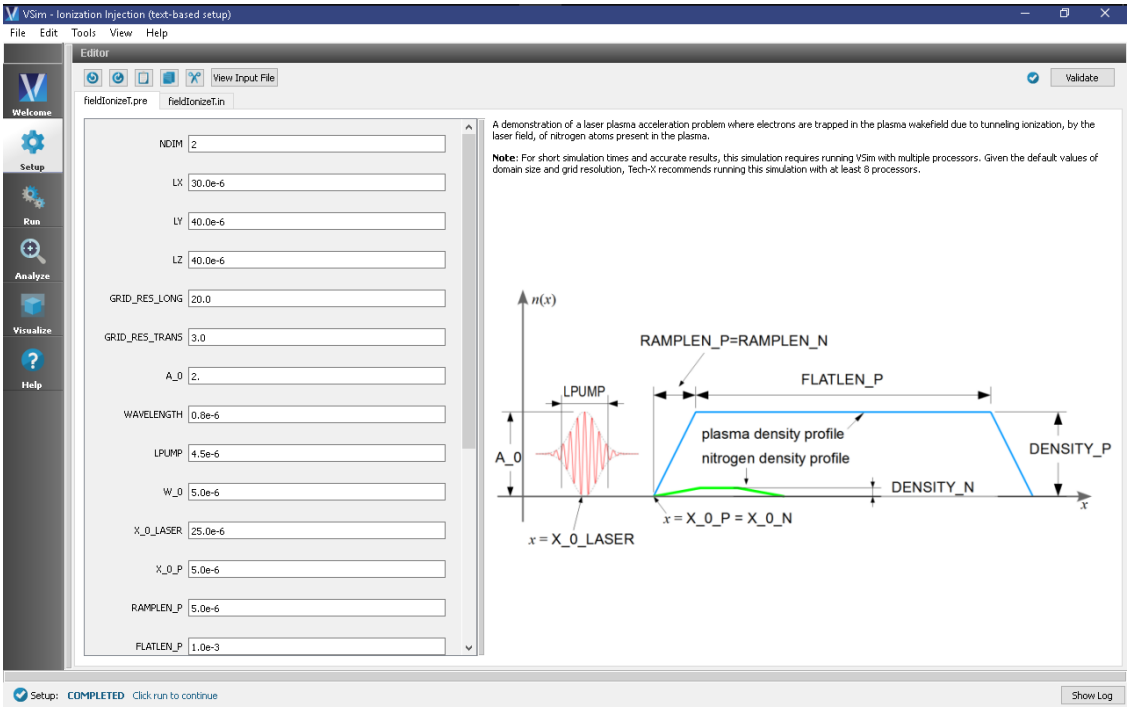


Fig. 5.24: Setup Window for the Ionization injection in Laser Plasma Accelerator example.

Input File Features

The simulation setup consists of an electromagnetic solver using the Yee algorithm. The laser pulse is launched from the left side of the window using an expression launcher at the boundary. MALs are used on the transverse sides of the window to absorb outgoing waves. The plasma is represented by macro-particles which are moved using the Boris push. The particles are variably weighted to represent the density ramp. The nitrogen atoms are represented using a fluid neutral gas. The different excited levels of the nitrogen and electrons product of the ionization are represented through variably weighted macro-particles. The ionization process takes place in MonteCarlo interactions, using the modified time-resolved ADK formula *[CES+12]*.

Running the Simulations

After performing the above actions, continue as follows:

- Proceed to the Run Window by pressing the Run button in the left column of buttons.
- This run is computationally intensive, so you click *Run in Parallel* and select a number of cores equal to the number of physical cores on your machine.

- Unclick *Dump at Time Zero* checkbox (some of the particles do not show up until dump 1, so one would have to run the createMissingPtcls.py analyzer to fill in dump 0, so it is better to not have the dump 0 data for this example).
- To see the initial evolution, set the *Number of Steps* to 1000 and the *Dump Periodicity* to 500.
- To run the file, click on the *Run* button in the upper left corner of the Logs and Output Files pane on the right. of the window. You will see the output of the run in that same pane. The run has completed when you see the output, "Engine completed successfully." This is shown in Fig. 5.25.

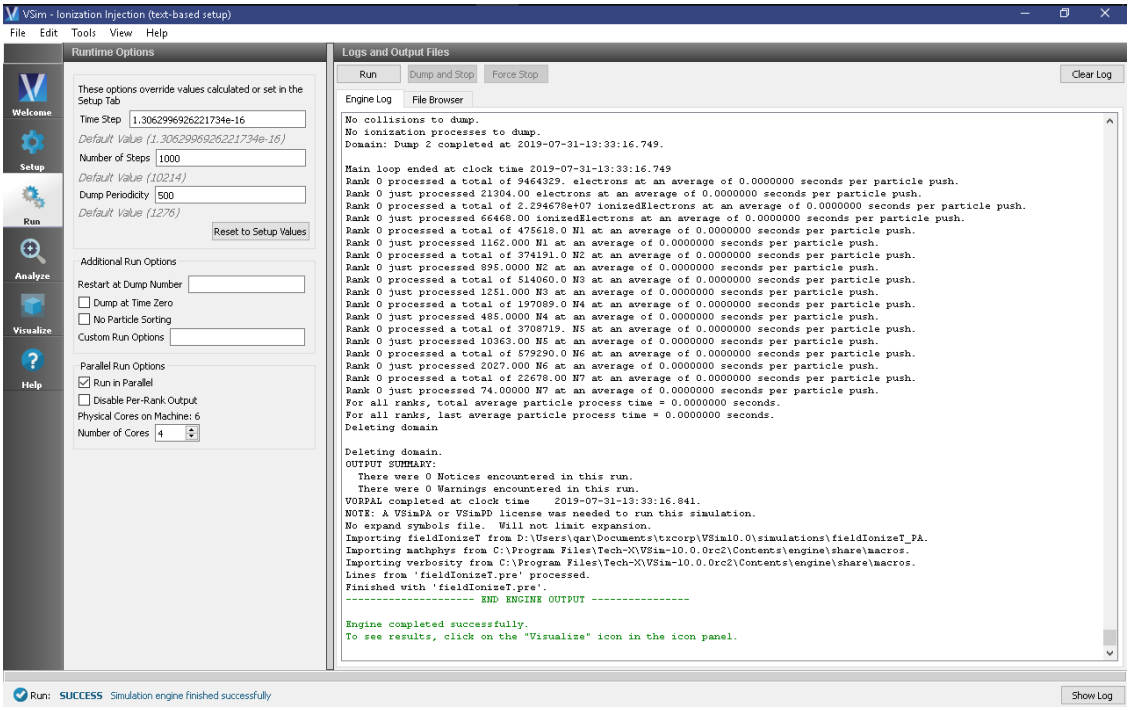


Fig. 5.25: The Run Window at the end of the first execution.

At this point, one can skip ahead to the visualization section to see whether the fields look reasonable. If they do, you can restart:

- Set the *Number of Steps* to 9000 and *Restart at Dump Number* to 2.
- Click on the *Run* button. The run has completed when you see the output, "Engine completed successfully."

This run takes about 70 minutes on a 4 core, 2.5 GHz Intel I7. To run on less powerful hardware one can reduce the number of grid points and number of particles per cell, however physical results may not be as accurate.

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 laser pulse is the z component of the field, while the accelerating field is the x component. The plasma density can be seen in `rho`.

Fig. 5.26 shows the longitudinal laser field along the beam axis. To reproduce:

- Set *Data View* to *Field Analysis*

- Click *Auto Reset* on both the pseudocolor and lineout plots so that the window updates the plot region as one moves the slider. You may need to expand your visualization window for the *Auto Reset* checkbox to appear.
- Select *edgeE_x* from the Field drop down menu
- Select the *Horizontal* tab in the lineout settings
- Set the intercept to 0
- Click "Perform Lineout
- Move the dump slider forward in time

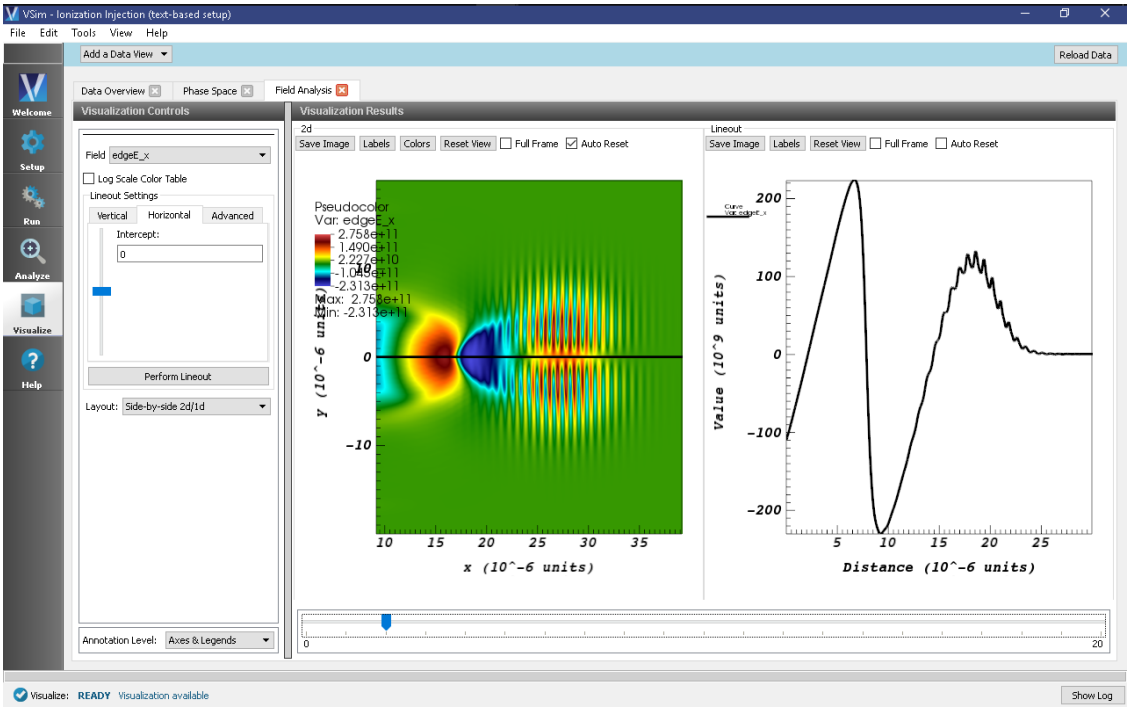


Fig. 5.26: Left: Longitudinal electric field $E_x(x, y)$ at t=1.3 picoseconds. Right: Line-out of field plot at $y = 0$.

The acceleration of the particles can be seen by viewing the (x) component of the velocity as shown in Fig. 5.27

- Set *Data View* to *Phase Space*
- Set *Base Variable* to *electrons*
- Set the X-axis variable to *electrons_x*, the Y-axis variable to *electrons_ux*
- Check *Enable Second Plot*
- Set *Base Variable* to *ionzedElectrons*
- Set the X-axis variable to *ionizedElectrons_x*, the Y-axis variable to *ionizedElectrons_ux*
- Click *Draw*
- Click *Reset View*

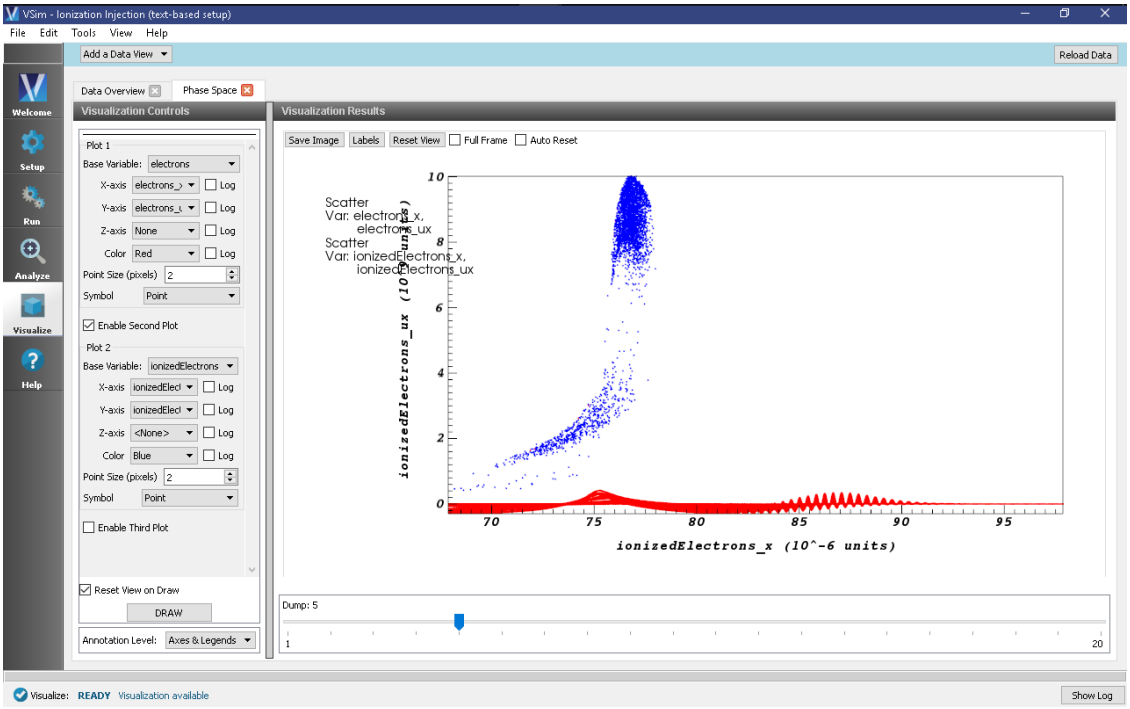


Fig. 5.27: Phase-space plot (x, $\gamma\ v_\mathrm{x}$) of plasma electrons at t=0.33 picoseconds.

CHAPTER

SIX

VSIM FOR PLASMA DISCHARGES EXAMPLES

These examples illustrate how to solve complex problems in plasma discharge modelling.

These examples can be run with a VSimPD license.

6.1 Capacitively Coupled

6.1.1 1D Capacitive Plasma Chamber (capacitivelyCoupledPlasma1D.sdf)

Keywords:

CCP discharge, secondary emission, elastic collision, excitation, ionization.

Problem description

The capacitively coupled plasma (CCP) is one of the most common types of industrial plasma sources. The discharges usually take place between metal electrodes in a reaction chamber and are driven by a radio-frequency (RF) or DC power supply. The plasma is sustained by ohmic heating in the main body and stochastic heating through a capacitive sheath.

This example demonstrates the generation of a capacitively coupled plasma inside two parallel conducting plates separated by 0.05 m. A background Ar neutral gas at approximately 6 mTorr a number density of approximately $2.0 \times 10^{20}\ m^{-3}$) is filled between the electrodes. The right electrode is grounded, while the left one is connected to a voltage source of 200 V at 60 MHz.

This simulation can be performed with a VSimPD license.

Opening the Simulation

The 1D Capacitively Coupled Plasma Discharge example is accessed from within VSimComposer by the following actions:

- Select the *New* $\rightarrow$ *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim for Plasma Discharges* option.
- Expand the *Capacitively Coupled Plasmas* option.
- Select "1D Capacitive Plasma Chamber" 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.

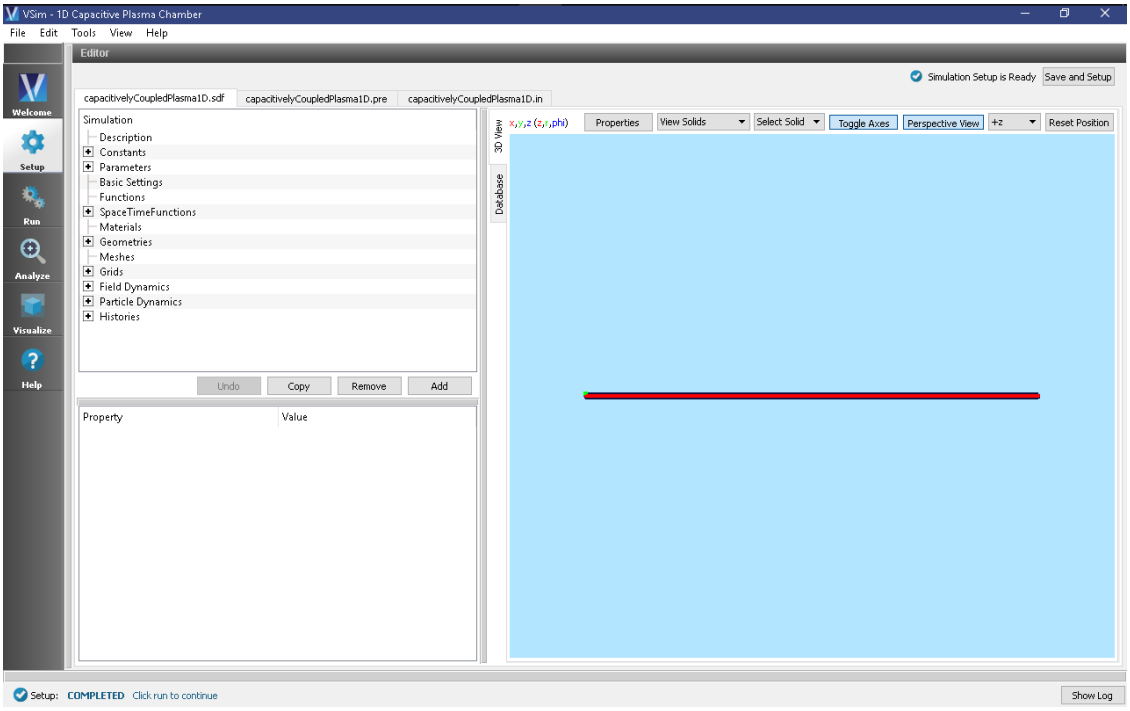


Fig. 6.1: Setup Window for the 1D Capacitively Coupled Plasma Discharge example.

The *Setup Window* and elements tree with all the implemented physics and geometries, is shown in Fig. 6.1.

The time step DT should sufficiently resolve the plasma frequency and collision frequency. The default time step used in this example is TIMESTEP_FACTOR * (0.1 / Plasma frequency) to ensure stability. The initial primary electrons are gradually loaded into the simulation domain over a period of LOADSTEPS timesteps, which has a default value of 5000.

Simulation Properties

This simulation includes some constants and parameters for easy adjustment of the simulation properties. These include:

Constants

- NEUTRAL_ARGON_DENSITY: number density of the background neutral argon gas (number/m^3).
- FREQUENCY: sets the frequency of the driving voltage set on the lower X boundary.
- VOLTAGE: sets the amplitude of the driving voltage set on the lower X boundary.
- NOMINAL_DENSITY: this adjusts the number of physical particles loaded into the simulation.
- LOADSTEPS: Timestep when particle loading will end.
- NSTEPS: How many timesteps to simulate.
- STEPS_PER_DUMP: number of steps to take between data dumps.
- BMAG: sets the strength of the magnetic field (default = 0T).

Time-dependent Dirichlet boundary conditions are used to set up the boundaries of electric fields around the reaction chamber walls, and are set in *Field Dynamics -> FieldBoundaryConditions*. The self-consistent electric field is solved from Poisson's equation by the Generalized Minimum Residual (gmres) electrostatic solver in Cartesian coordinates. This solver is chosen under *Field Dynamics -> PoissonSolver*.

The plasma is represented by macroparticles which are moved using the Boris pusher in Cartesian coordinates and interact with the background neutral argon gas through collisions set up with the Reactions framework. The particles, background gas, and collisions are set up in the *Particle Dynamics* Element.

The simulation includes two electron species: Primary electrons which are electrons loaded into the simulation, and Secondary electrons which are created through physical processes. Both species are managed weight particle species, which will combine or split macro particles based on user choices.

Elastic collisions between electrons and the background gas, excitation collisions in which an electron will lose energy to the background gas, and ionization collision in which electrons create argon ions from the background gas are all included. The cross-sections for this collisions are imported from 2-column data files.

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.
- 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." This is shown in Fig. 6.2.

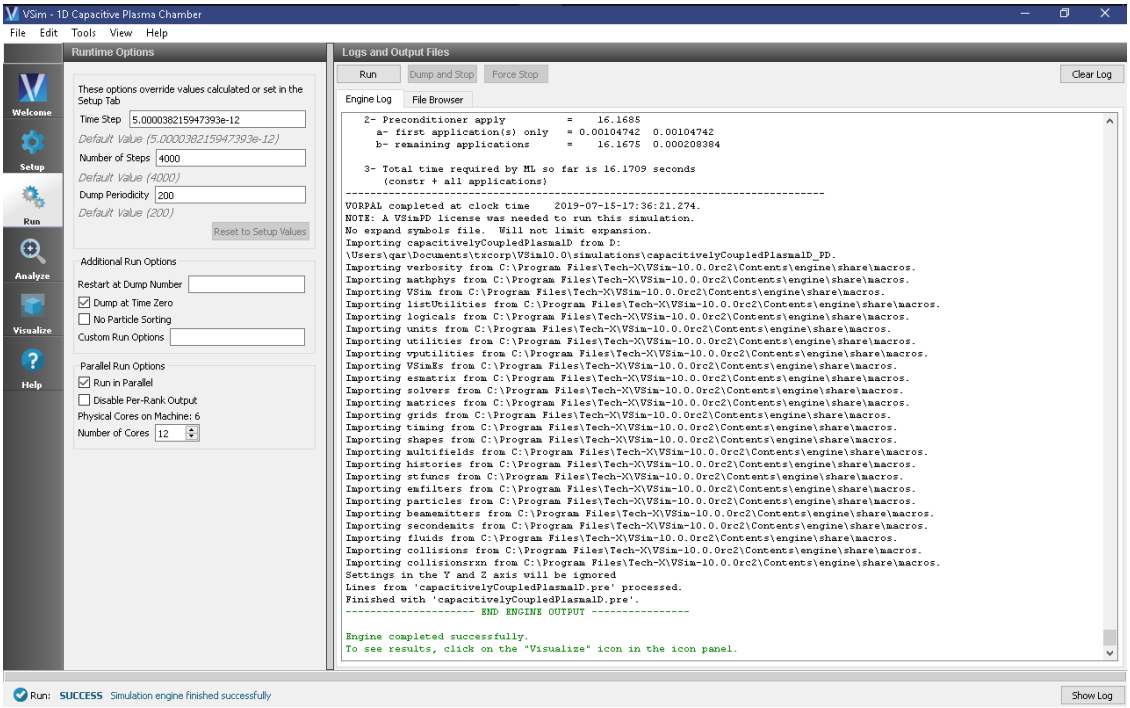


Fig. 6.2: The Run Window at the end of execution.

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.

From the "Data View" option, select "History". There are 6 histories that can be plotted in this window: the number of physical particles and the number of macro particles for each of the three particle species (argon ions, primary electrons, and secondary electrons). To produce the plot in Fig. 6.3 follow these steps:

- Plot the 'numArgon' history in Graph 1.
- Plot the 'numPrimaryElec' history in Graph 2.
- Plot the 'numArgon' history in Graph 3 and change the *Location* to Window 2
- Plot the 'numPhysSecondaryElec" history in Graph 4.

The simulation converges as the number of secondary electrons approaches a constant, indicating a steady state plasma. With the default number of time steps (4000, or 20 nanoseconds), the simulation does not reach steady state (see the black, numPhysSecondaryElec history curve). To reach steady state, the simulation must run for approximately 100 microseconds.

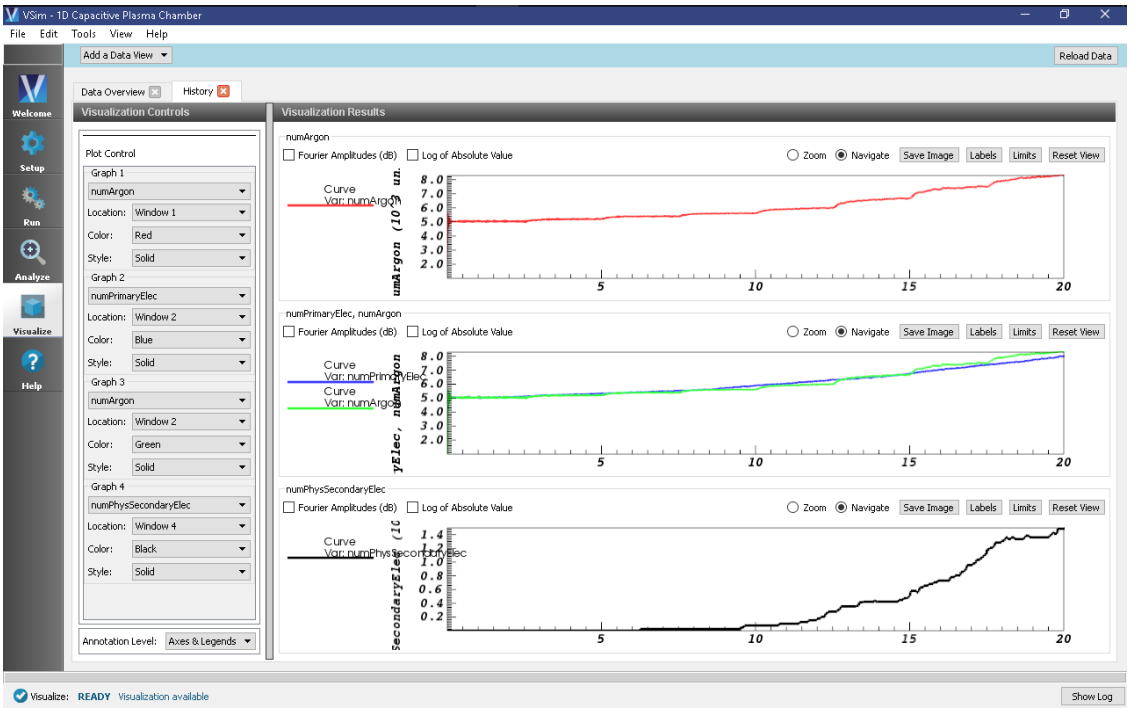


Fig. 6.3: Visualization of number histories of ion, primary electron, and secondary electrons for at 4000 steps or 20 nanoseconds.

Further Experiments

Set up a History that records the electron current flowing into the left and right sides of the simulation. Right click on the "Histories" element and under "Add ParticleHistory" select "Absorbed Particle Current." You can change the name of the history by double clicking on the new "absorbedPtclCurrent0" element in the tree. Then be sure to pick the particle absorber from which you would like to collect data.

The Reactions framework allows one to set up collision interactions flexibly. The collisions involved in this example are electron-neutral collisions that lead to ionization and ohmic heating. As a further experiment, ion-neutral collisions, such as elastic scattering and charge exchange, can also be added to the simulation.

The VSim interface can import any cross sections that are in a 2-column format. There should be NO headings in the data file. The LXcat scattering database (https://fr.lxcat.net/data/set_type.php) and EEDL cross section database contain cross section data for around one hundred different materials. As another experiment, change the cross-section used in the simulation or change the species of the background gas and import new cross-sections.

6.1.2 Turner Case 2 (Turner.sdf)

Keywords:

capacitively coupled plasma, CCP, discharge, steady state, Turner

Problem Description

In this example we demonstrate VSim's ability to simulate capacitively coupled plasmas, using the benchmark cases of Turner et al. *[TDD+13]*. Turner's work documents the successful benchmarking of five independently developed particle-in-cell codes (not including VSim) for four different capacitive coupling scenarios at various background pressures.

Here, we consider the second of the Turner scenarios, though the input file can be readily modified to simulate the others. In addition to being able to accurately reproduce the Turner results, VSim can also employ physics-based initialization methods to enable more rapid convergence of the simulations to their steady-state. The use of such methods will also be explained below.

This simulation can be performed with a VSimPD license.

Opening the Simulation

The Turner example is accessed from within VSimComposer by the following actions:

- Select the *New* → *From Example...* menu item from the *File* menu.
- In the resulting *Examples* window expand the *VSim for Plasma Discharges* option.
- Expand the *Capacitvely Coupled Plasmas* option.
- Select *Turner Case 2* 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. 6.4. In this image, we have unclicked the electrons' particleLoaderE and the HeNeutralFluid so that they will not hide the basic grid. 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`. This is a one-dimensional problem, which is shown by having the grid have only a single cell above and below the x-axis.

Clicking the electrons' particleLoaderE shows that the electron loader is defined to exist over a cartesian 3d slab, even though this is a one-dimensional simulation. The dimensions that do not apply are ignored, with the coordinate set to zero, but this allows easy conversion from a 1D simulation to a 2D simulation.

Simulation Properties

The basic physics of this simulation is a balance between collisional processes and wall losses; a one-dimensional box of length 6.7 cm contains neutral helium gas at room temperature (300 K) and density 3.21e21/m^3 (1 Torr of pressure at that temperature). The gas is weakly ionized, resulting in a population of free electrons and singly ionized helium atoms at density 5.12e14/m^3. The helium ions are also at room temperature, while the electrons are considerably

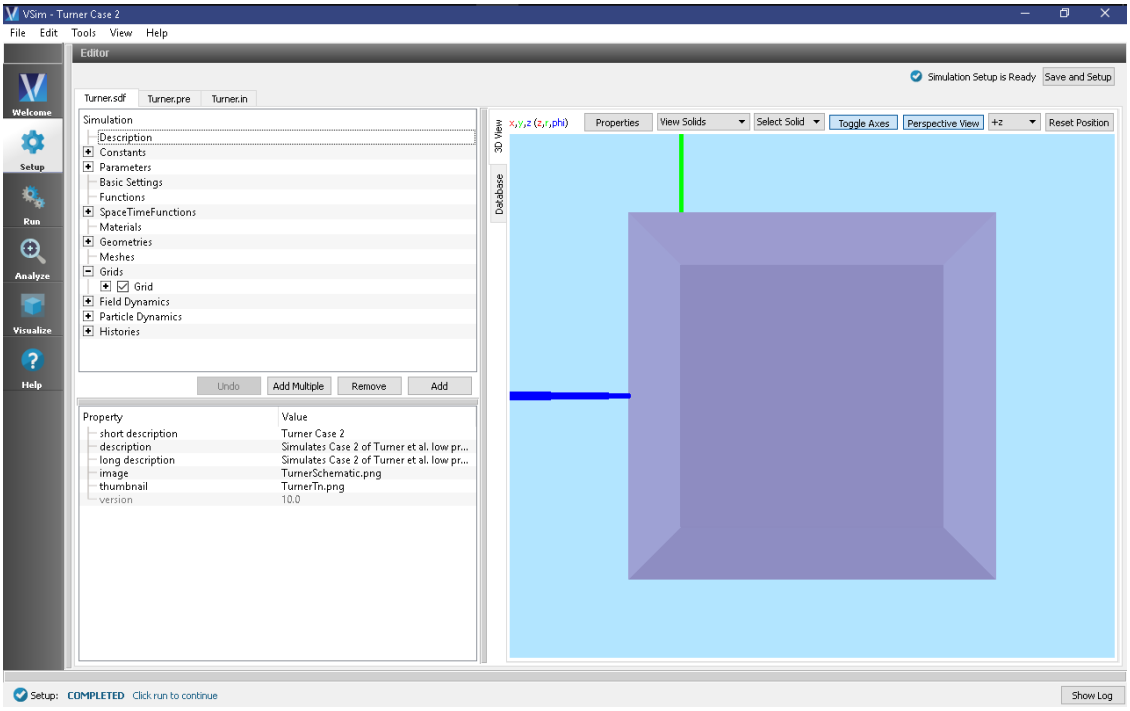


Fig. 6.4: Setup Window for the Turner example.

hotter (30,000 K). The left wall of the box is grounded, while the right wall oscillates with a bias voltage of 200 V at frequency 13.56 MHz.

Charged particles are lost upon collision with the wall and are replenished by ionization of the background neutral gas by the hot electrons; the latter process repopulates both the electrons and helium ions in the plasma (the background neutral gas is treated as an infinite source). Plasma sheaths form near the walls, containing electric fields which are strong relative to those elsewhere in the plasma; the particle density profiles adjust in response to the fields in the sheath. The sheath transit time, for ions, is much longer than the period of the oscillating potential; thus, multiple RF cycles occur while an ion crosses the sheath. A steady state is attained when the loss rate of particles to the wall comes into balance with the ionization rate for a particular profile shape.

In our initial run we are not going to model the full evolution of the discharge to its steady-state parameters; rather, we will explore the basic physics of the discharge and modify the simulation accordingly (with the aim of ultimately hastening convergence to this steady state, while exploring VSim capabilities).

Running the Simulation

The original runs by Turner were for about 4,000,000 steps. However, the asymptotic state is reached after about 300,000 steps. To illustrate how to run this problem, we will run for only 10,000 steps, which takes about 5 minutes on a 4-core i7 Windows workstation.

To run the simulation, continue as follows:

- Proceed to the Run Window by pressing the Run button in the left column of buttons.
- Set ‘Number of Cores’ corresponding to your VSim license in the *MPI* tab at the left side of the run panel.
- To run the file, click on the *Run* button in the upper left corner of the window. You will see the output of the run in the right pane. The run has completed when you see the output, “Engine completed successfully.” A snapshot of the simulation run during execution is shown in Fig. 6.5.

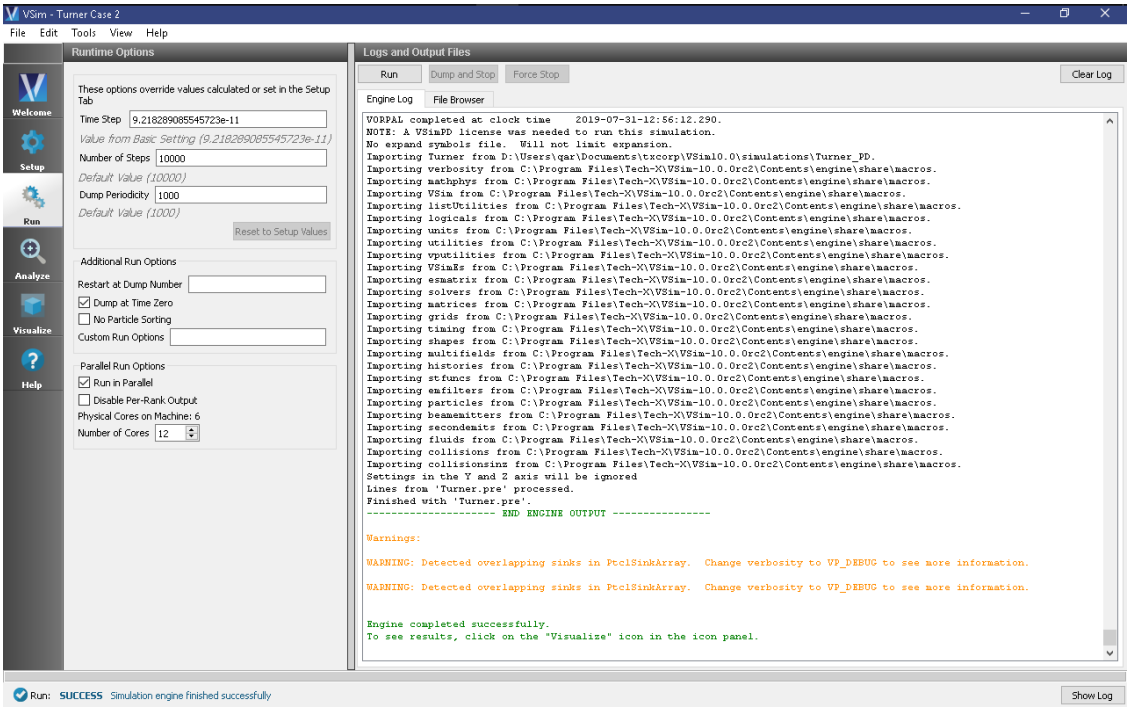


Fig. 6.5: The Run Window during execution.

Analyzing the Results

We are going to run a postprocessing script, computePtclNumDensity.py, which builds density profiles from the particle data generated by VSim, so that we can look at these profiles and their evolution. To do so, we do the following:

- Click the *Analyze* beneath the *run* button in the leftmost pane.
- From the Available Analyzers, choose *computePtclNumDensity.py*. Then click *Open*.
- Fill in the text boxes
  - The simulationName should be already filled in, but if it is not, type in the name of the .sdf file without the .sdf extension.
  - For the speciesName, type in 'electrons' without the quotes.
- Click *Analyze* (in the Analyze Window); this will generate the electron density profiles.
- Now replace 'electrons' in the speciesName box with 'He1', for the helium ions.
- Click *Analyze* (again in the Analyze Window) to generate the ion density profiles.
- The name of the resulting data is *electronDensity* and *He1Density*, which will be visualized in the next section.

Visualizing the Results

Now that we have all of our data, let's look at it.

- Click the *Visualize* button beneath the *Analyze* button in the leftmost pane.

After a brief moment the visualization options for this data should appear.

We will first look at the time evolution of some fundamental one-dimensional quantities. From the *Add a Data View* pulldown menu on the top left, select *History*. The default view here should contain four plots, namely, the electron and ion currents to the left wall and the number of electron and ion macroparticles in the simulation. A number of notable physics effects can be seen here:

- **After a sharp initial decrease in the electron population, both ion and electron populations decline at approximately the same rate.** This is not as apparent from the separate numElec and numIons plots, but clicking on the "Location" drop-down window in Graph 2 and selecting "Window 1" as the new rendering destination, places both ion and electron populations in the same plot. (Select "None" in the plot variable (the topmost menu) for both Graph 3 and Graph 4 to resize the electron/ion plot.) The initial decrease in electron population arises when rapid electron wall losses create a charge imbalance in the plasma and establish plasma sheaths near the walls. Thereafter, this charge imbalance is preserved and the transport of both electrons and ions to the wall becomes ambipolar. A history of the particle popuations can be seen in Fig. 6.6

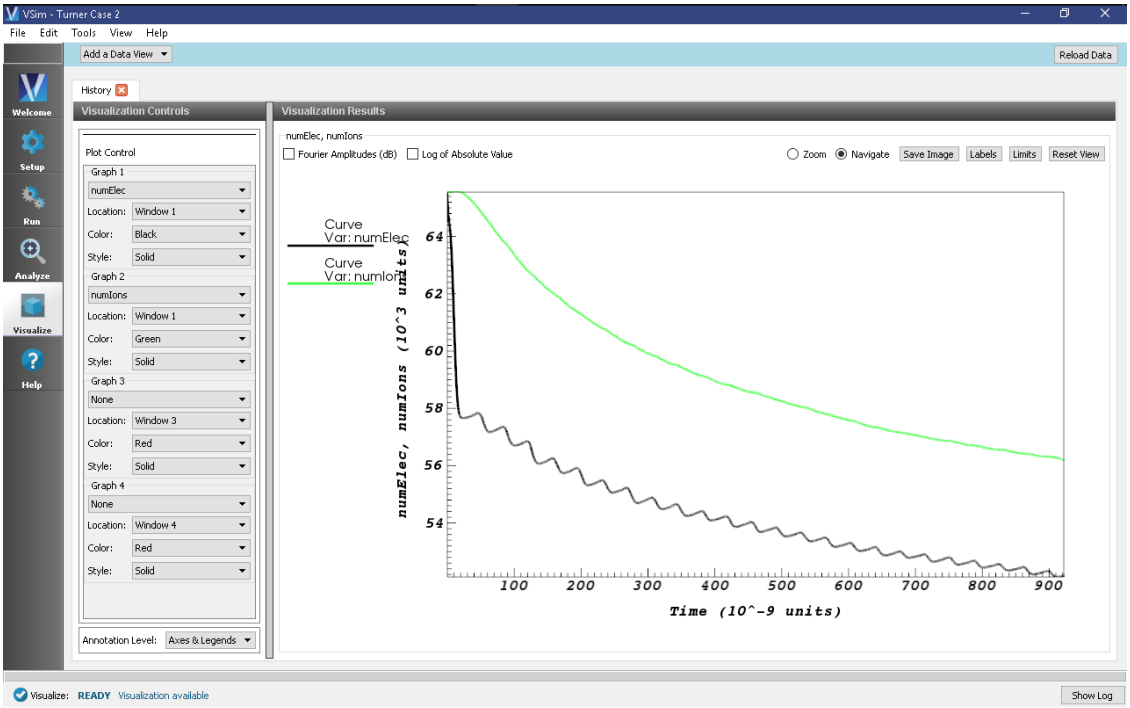


Fig. 6.6: The electron and ion populations versus time.

- **The electron wall currents are quasi-periodic.** The oscillating potential drives the highly mobile electrons alternately into the left and right walls. In the plot variable menu, change "numElec" to "leftElecCurr" in Graph 1 and "numIons" to "rightElecCurr" in Graph 2. The impacts of the electron cloud on the left and right walls, and their phasing in time, can be seen in response to the potential oscillations. A history of the electron currents can be seen in Fig. 6.7
- **The ion currents are non-periodic.** Ions, being much heavier than the electrons, exhibit relatively little response to the oscillating potentials. In the plot variable menu, change the Graph 3 quantity "None" to "leftIonCurrent" and the location to "Window 1", then change the Graph 4 quantity "None" to "rightIonCurrent" and the location again to "Window 1". The ion currents do not have the quasi-periodic structure of the electron currents; rather, ions diffuse outward to the walls in response to the DC sheath potentials, which are established by the initial departure of electrons and may also be rectified by the RF. A history of all the particle currents can be seen in Fig. 6.8
- **Ion losses are negligible before the initial establishment of the sheath.** Change the plot quantity in Graph 3 from "leftElecCurr" to "None". Change the plot quantity in Graph 4 from "rightElecCurr" to "numElec" and for the first two graphs change the "Location" to "Window 3". It is clear that the dominant loss of ions to the

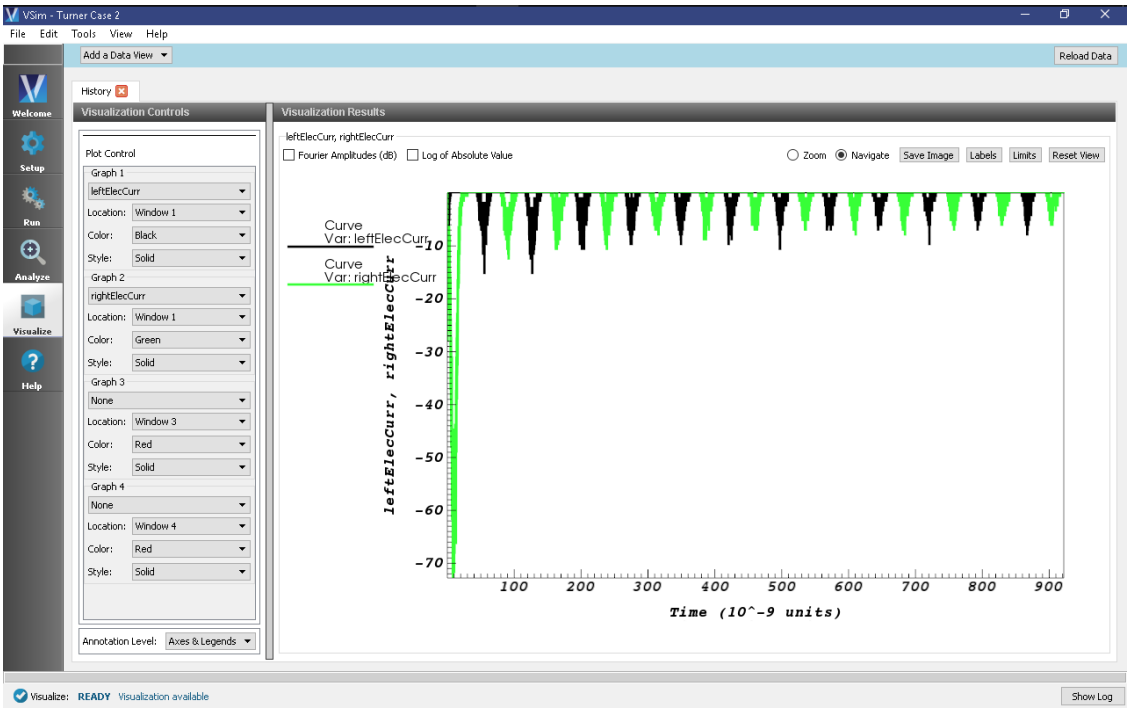


Fig. 6.7: Electron currents on the left (black) and right (green) walls versus time

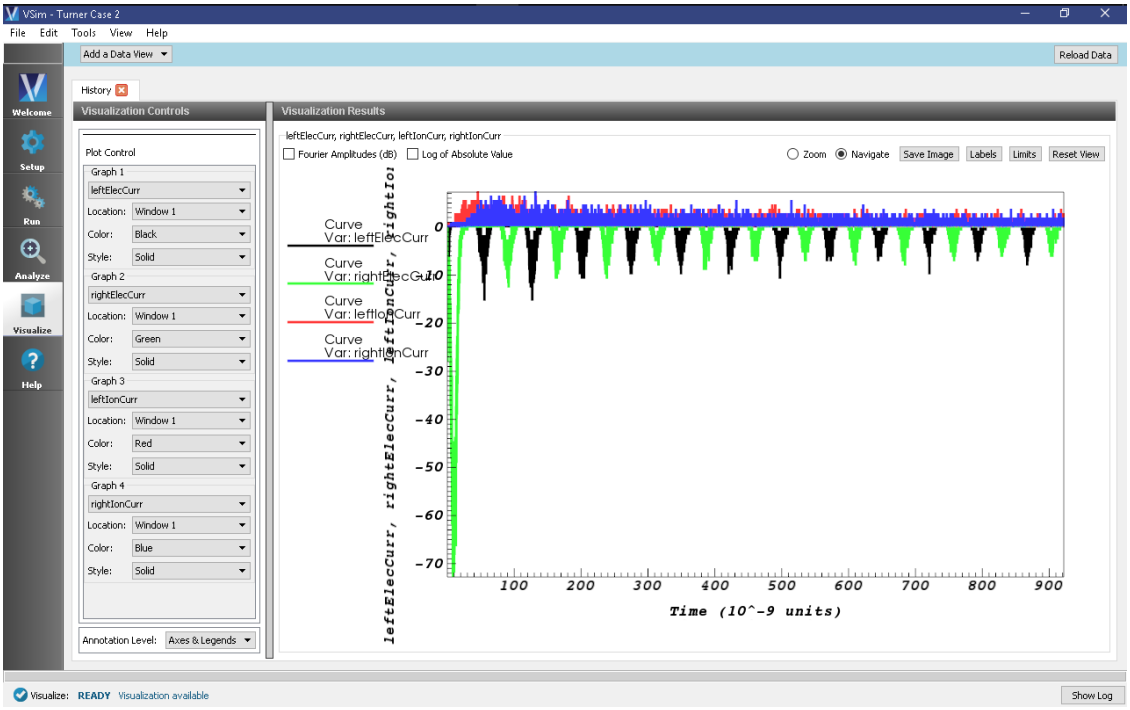


Fig. 6.8: Electron and ion currents on the left and right walls versus time

wall only begins after the initial decrease in electron population (which corresponds to the establishment of the sheath). A history of the electron population against the ion currents can be seen in Fig. 6.9

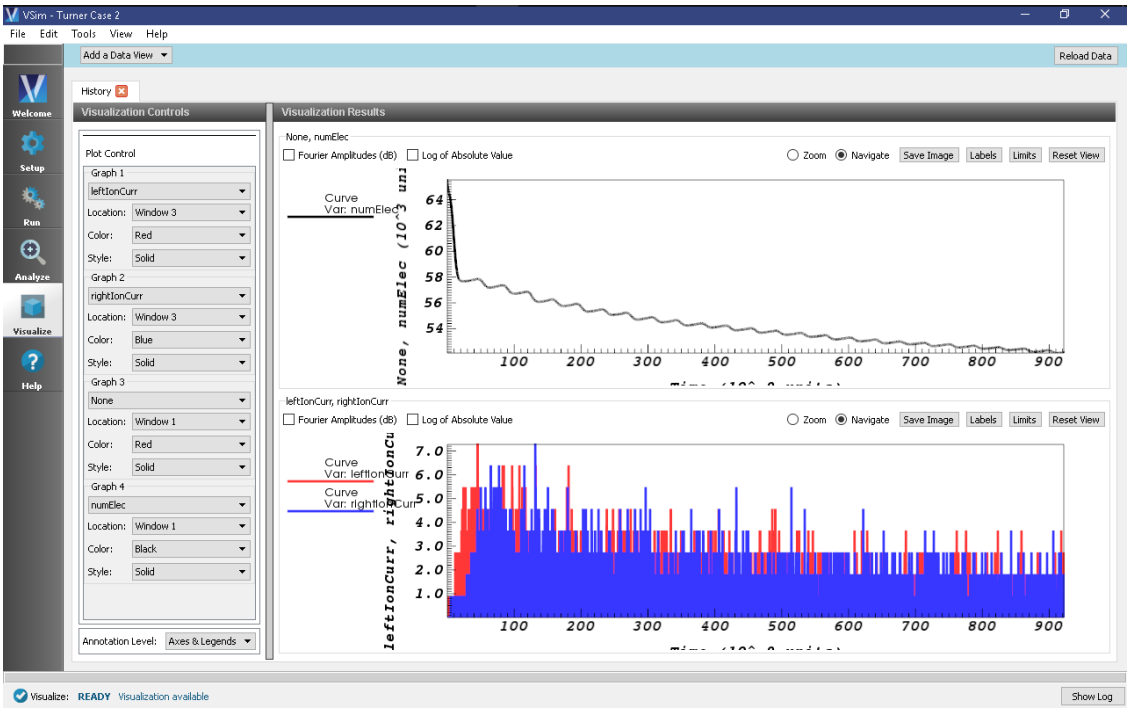


Fig. 6.9: History plots showing the majority of the ion current to the walls only begins after initial decrease in electron population

We can also look at the plasma sheath and the ensuing changes in density profiles directly. In the “Add a Data View” menu at the top left of the Composer window, select “1-D fields”. The plot controls here are similar to those of the history window. Select “E_x” for the plot variable in Graph 1. Select “Phi” for the plot variable in Graph 2. Select “electronsDensity” for the plot variable in Graph 3. Select “He1Density” for the plot variable in Graph 4, and select “Window 3” for the location of this plot. The evolution of the discharge in time can be viewed by moving the time slider below the plots. Slide the bar to dump 10 to view the data at time step 10000 (the data was saved every 1000 time steps).

A number of additional physics features can be seen:

- **Sheath effects are present.** Regions of sharp potential variation, corresponding to strong electric fields, arise near the walls, but such fields are screened out in the bulk plasma. Moving the time slider, it is clear that this sheath behavior persists regardless of the phase of the oscillating wall potential.
- **Electron profiles are altered much faster than ion profiles.** Both ions and electron profiles are initially constant (5.12e14 1/m^3), but by the time the first nontrivial dump file is produced (at time dumpPeriodicity * dt, approximately 1/3 of the way through the period of the first wall oscillation), electron-poor regions corresponding to the sheaths have already been established in the electron profile, while the ions have barely begun to respond to the presence of the sheath. Moving the slider forward in time, one observes that the electron profile predominantly oscillates in response to the wall potential, while the ion profile evolves considerably more slowly, particularly outside the sheath regions. The 1-D fields at time step 10000 (dump 10) can be seen in Fig. 6.10

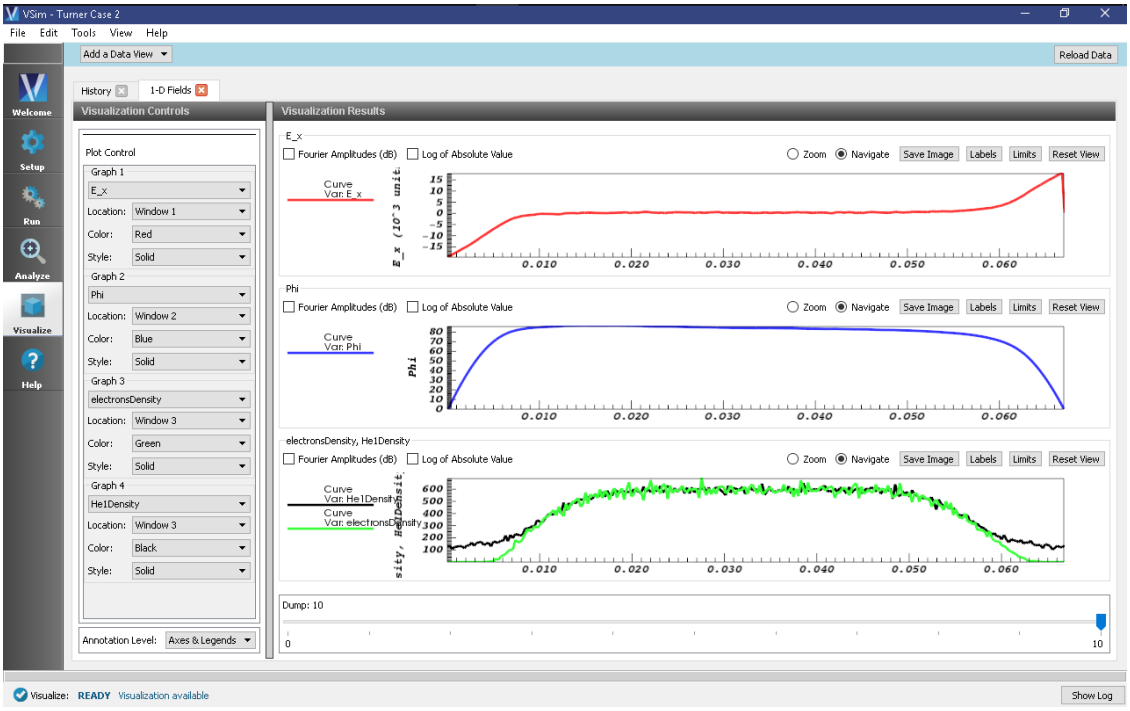


Fig. 6.10: Plots of various 1-D field quantities showing the final state of the run at time step 10000.

Further Experiments

Now that we understand some of the basic physics of the discharge, we are in position to apply physics-based particle loading methods to hasten its eventual convergence to steady-state. The underlying principle here is to identify the 'slow' processes involved in the evolution of the discharge toward steady state, and then alter the loading to more closely mimic the state to which the plasma is being driven. While we cannot entirely predict the parameters of the steady-state, it is not difficult to at least get some idea of how the simulation is evolving and adjust the particle loads accordingly. We have already observed a number of physical processes of possible relevance:

- initial electron loss and the establishment of ambipolarity
- the slow decay of the total ion and electron population following the initial electron loss
- the rapid response of electrons to applied electric fields, particularly in the sheath region
- the slow evolution of ion density profiles.

Of these, we will primarily consider the ion profiles; the high mobility of the electrons suggests that electron profiles will adjust correspondingly on much shorter timescales. Additionally, since the strong electric fields in the plasma sheath region are screened out via Debye shielding as we move away from the walls, it seems clear that profile adjustments in the bulk plasma (where the driving electric fields are weakest) will ensue more slowly than in the plasma edge. We therefore concentrate our attention first on obtaining an approximately correct value for the ion density at the center of the domain.

In the '1-D Fields' tab, set the plot variable to 'None' in plots 2, 3, and 4. In Graph 1, set the plot variable to 'He1Density' and again move the timeslider on the bottom right of the window. The central ion density steadily rises; from its initial value of 5.12e14/m^3, it rises to 6e14/m^3 by the end of our comparatively short run. In addition, a rapid decrease in density near the walls (associated with the plasma sheath) has lowered the edge densities to about 1e14/m^3. The ion density can be seen in Fig. 6.11

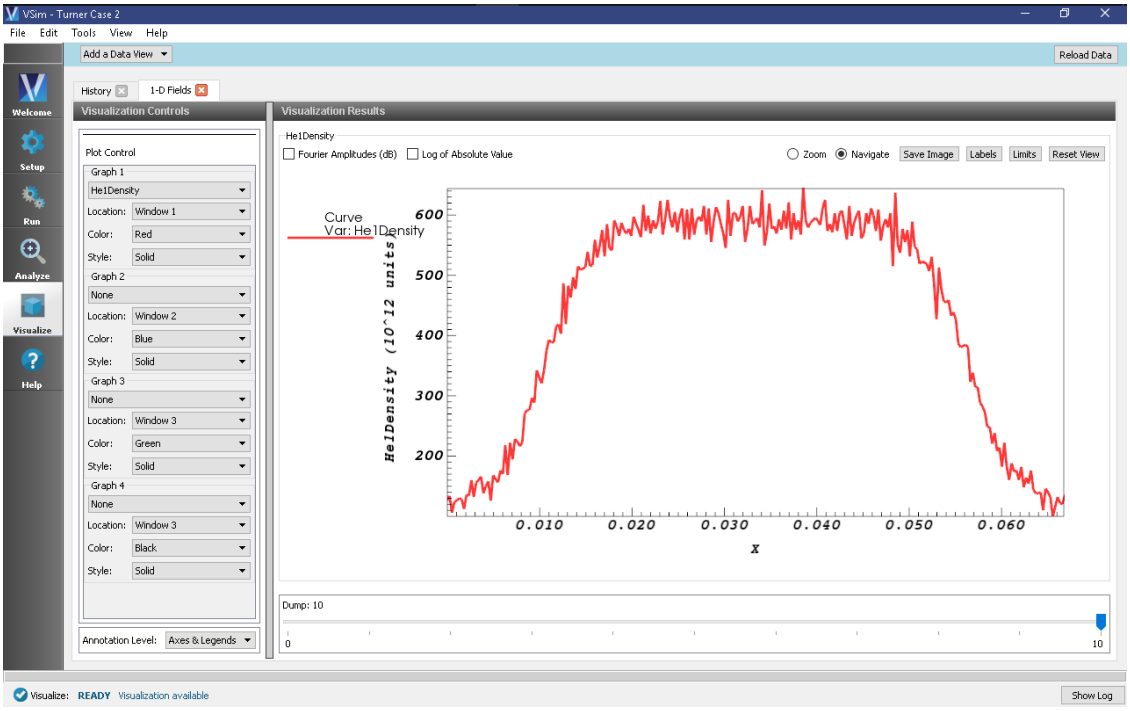


Fig. 6.11: The He ion density at the end of the run (time step 10000).

Additional Studies

It is possible to try other techniques to converge to steady state faster. These techniques are outlined in detail in the text-based version of this example. In summary:

Another thing you can try are by loading the particle with a non-uniform profile that better resembles the outcome.

Yet another is to leave a gap near the walls when loading electrons and ions. What happens in the discharge? The electrons, being highly mobile, rush to fill the gap, but rather than immediately being lost to the wall, they instead produce strong electric fields at the plasma edge which begin to modify the ion profile and bring about ambipolarity. If the gap is sufficiently large, the collisional production of ions and electrons will begin before appreciable wall losses ensue, and we can thus assess the relative rates of production and loss fairly early in the simulation. Since the electrons are highly mobile, let's treat the average electron population as a measure of how well we've achieved this balance; net electron production as we move into the ambipolar phase means that our gap is too low (we have removed too much density), while net losses mean that our gap is insufficiently wide. As the profile shapes near the walls tend to adjust themselves fairly quickly (due to the larger electric fields in this region), we can in this manner obtain approximately correct values for the total ion and electron populations at the simulation outset.

6.2 Capacitively Coupled (text-based setup)

6.2.1 2D Capacitive Plasma Chamber (capacitivelyCoupledPlasma2DT.pre)

Keywords:

`capacitively coupled plasma discharge under RF and DC voltage in 2D cylindrical system.`

Problem description

The capacitively coupled plasma (CCP) is one of the most common types of industrial plasma sources. These plasma discharges typically take place between metal electrodes in a reaction chamber and are driven by a radio frequency (RF) or direct current (DC) power supply. The plasma is sustained by ohmic heating in the main body and stochastic heating through the capacitive sheath.

This example demonstrates the generation of a capacitively coupled plasma inside an axially symmetric reaction chamber with a 50 mm radius and 50 mm length. The top and side walls are grounded at zero potential. A target located at the bottom of the chamber is connected to a 60 MHz AC voltage source at 200 V. There is a small gap of 5 mm between the target and the grounded wall. The chamber is filled with a background gas of Argon at about 0.005 Torr $\left(2.0 \times 10^{20}\ \mathrm{m}^{-3}\right)$. An initial spatially averaged electron density of $5 \times 10^{11}\ \mathrm{m}^{-3}$ is seeded to start the discharge. Note that the particle position (and hence density) is chosen in such a way that the density increases as *R* increases. Therefore, the initial density is less than $5 \times 10^{11}\ \mathrm{m}^{-3}$ for small values of *R* and greater than $5 \times 10^{11}\ \mathrm{m}^{-3}$ for larger values of *R*.

This simulation can be performed with a VSimPD license.

Opening the Simulation

The Capacitively Coupled Plasma 2D example is accessed from within VSimComposer by the following actions:

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim for Plasma Discharges* option.
- Expand the *Capacitively Coupled Plasmas (text-based setup)* option.
- Select "2D Capacitive Plasma Chamber (text-based setup)" 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 basic variables of this problem should now be alterable via the text boxes in the left pane of the *Setup Window*, as shown in figure Fig. 6.12.

Input File Features

The self-consistent electric field is solved from Poisson's equation by an electrostatic solver in cylindrical coordinates. Time-dependent Dirichlet boundary conditions are used to set up the boundaries of electric fields around the reaction chamber walls.

The plasma is simulated with macroparticles which are moved using the Boris pusher in cylindrical coordinates. Various types of elastic and inelastic collisions of the particles are calculated.

The Setup Window has various parameters available for easy manipulation including the density of the argon background gas (DENSITY_Ar), the voltage (VOLTAGE), and the frequency (FREQ).

Running the simulation

Once finished with the problem setup, continue as follows:

- Proceed to the Run Window by clicking the **Run** button in the left column of buttons.
- Choose your desired parallel computing options under *Parallel Run Options*.
- 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 when you see the output, "Engine completed successfully." This is shown in Fig. 6.13.

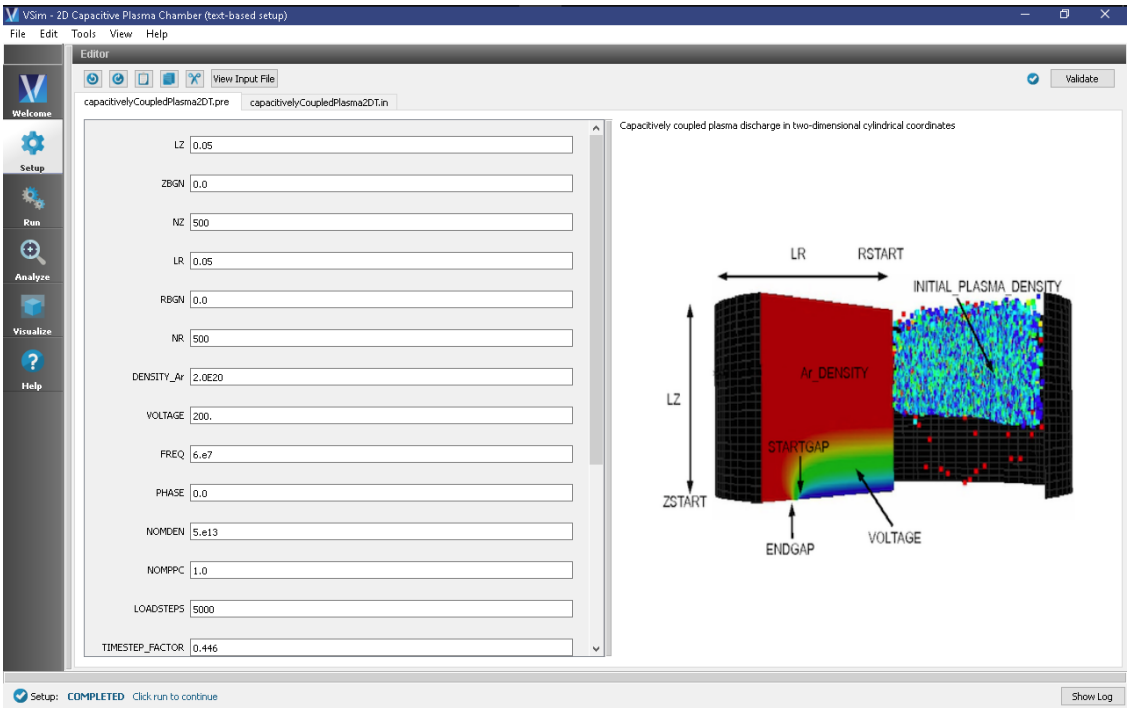


Fig. 6.12: Setup Window for the Capacitively Coupled Plasma 2D example.

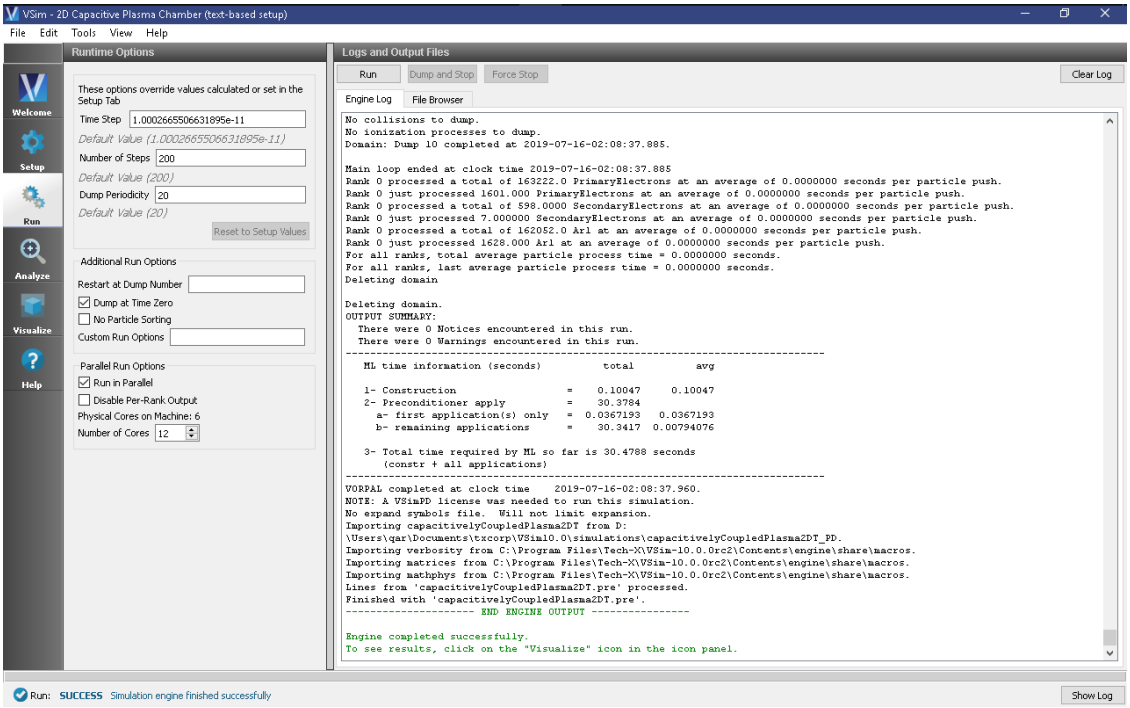


Fig. 6.13: The Run Window at the end of execution.

Visualizing the results

After performing the above actions, continue as follows:

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

To plot the potential:

- In the *Visualization Controls* pane, click on the *Data Overview* tab and expand *Scalar Data*
- Select *phi*
- Move the dump slider at the bottom of the *Visualization Results* pane to the right to move forward in time.

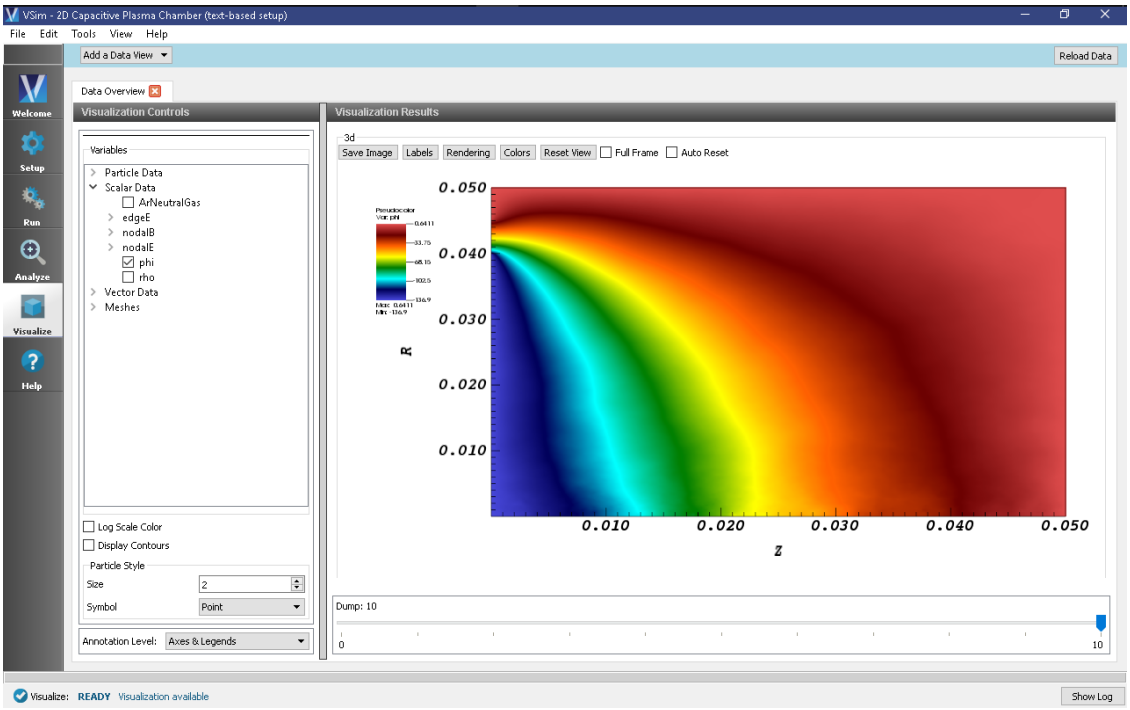


Fig. 6.14: Visualization of the electric potential in r-z coordinates.

Further Experiments

With a time-step of 10^{-11} seconds, running this simulation for the default 200 time-steps will only capture part of the first oscillation. With the frequency set to 6×10^7 Hz, the oscillation period is 1.667×10^{-8} seconds, which corresponds to 1,667 time-steps. To see the approximate steady-state behavior of this example, set the number of time-steps to 10,000 or more and restart or re-run the simulation. When running in parallel on 4 processors, this should take approximately two hours to complete.

After 5×10^{-8} seconds, or 500 dumps, the plasma sheath starts to exhibit oscillating steady-state behavior. To view the behavior of the oscillating plasma sheath, take the following steps:

- In the *Data Overview* tab, expand *Scalar Data*
- Select *rho*
- At the top of the *Visualization Results* pane, click *Colors* to open the *Color Options* window
- Select "Fix Minimum" and set the minimum to `-0.0005`, then select "Fix Maximum" and set the Maximum to `0.0005` (or experiment with minimum and maximum values for best results) and click *OK*

- Move the dump slider at the bottom of the *Visualization Results* pane to dump 250.

After 250 time-steps, the plasma density should appear as shown in Fig. 6.15. The green areas at approximately zero charge density denote the quasi-neutral plasma bulk, while the red areas (positive charge density) denote the non-neutral plasma sheath.

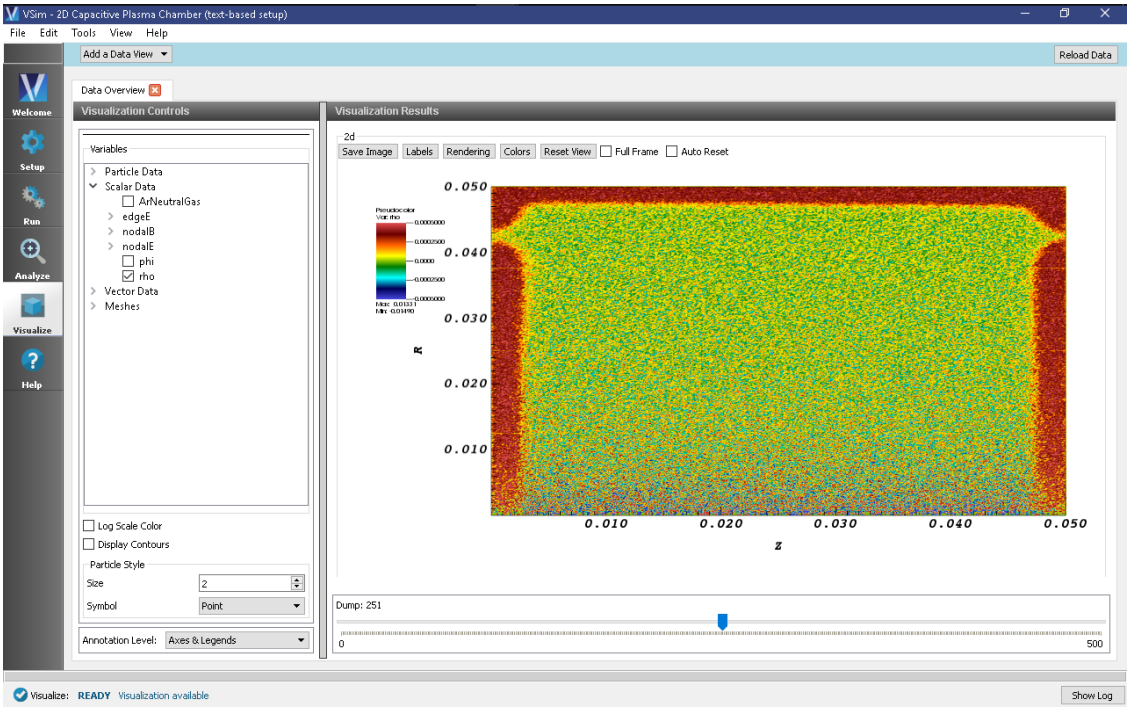


Fig. 6.15: Visualization of the plasma sheath via the charge density.

To view the plasma sheath potential profile, take the following steps:

- In the *Visualize* window, select *Field Analysis* from the *Add a Data View* drop-down menu
- In the *Field Analysis* tab, click on the *Fields* drop-down menu and select *phi*
- Under *Lineout Settings* click on the *Horizontal* tab, and change the intercept value if desired
- Click *Perform Lineout*

The electric potential as well as the axial potential profile should now be visible as shown in Fig. 6.16. Move the slider to the right to see how the plasma sheath potential oscillates in time.

6.3 DC Plasmas

6.3.1 Drifting Electrons (driftingElectrons.sdf)

Keywords:

`electron transport, electron mobility, monte carlo, electrostatic`

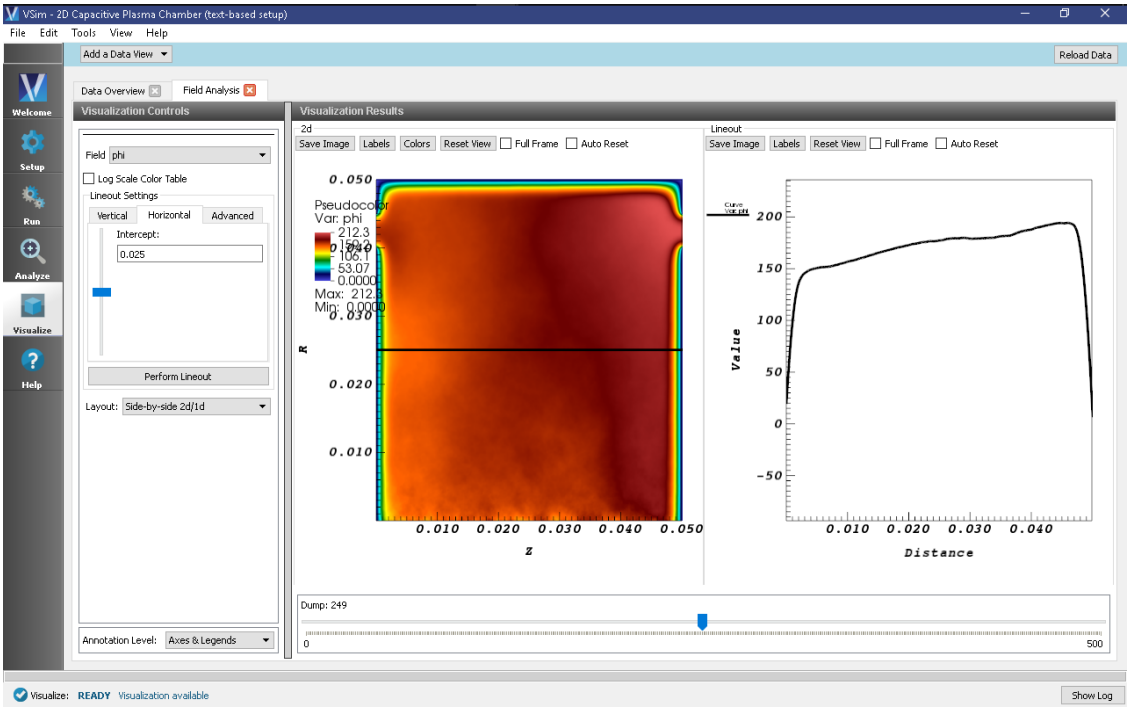


Fig. 6.16: Visualization of the plasma sheath via the charge density.

Problem description

VSim may be used to model charged particles drifting in a background neutral gas. When charged particles, such as electrons, are injected into a background neutral gas, collisions between gas atoms and electrons eventually lead to thermal equilibrium, and electrons will reach the same temperature as the background gas. However, when an external electric field is applied across the neutral gas, the electron collisions and distribution will change due to this applied field. Electrons will gain energy from the applied electric field. The energy loss due to electron-atom collision is small, and most of the energy ends up heating the electrons. Assuming only elastic collisions take place between electrons and atoms, the electron mobility is defined as

$$\mu_e = \left(\frac{\pi\lambda}{2mE}\right)^{\frac{1}{2}}$$

which describes the relation between electron drifting velocity and applied electric field.

This simulation can be performed with a VSimPD license.

Opening the Simulation

The Electron Drifting example is accessed from within VSimComposer by the following actions:

- Select the *New* → *From Example…* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim for Plasma Discharges* option.
- Expand the *DC Plasmas* option.
- Select "Drifting Electrons" 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, if applicable. See Fig. 6.17.

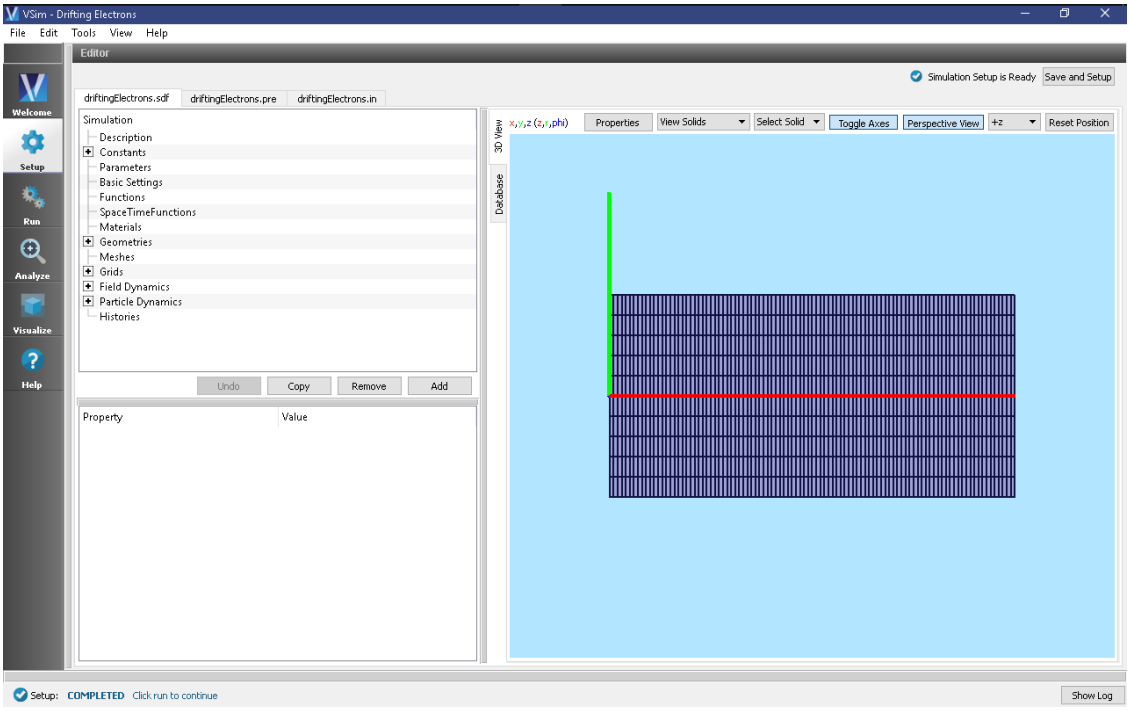


Fig. 6.17: Setup Window for the Electron Drifting example.

Simulation Properties

This input file contains electron as kinetic species as well as a background fluid description of a gas. Elastic collisions between kinetic particles and the background gas are described by Monte Carlo interaction blocks of kind impactIonization.

The fields are solved for electrostatically at each time step, including the fields due to all charged particles, subject to the boundary conditions specified in the input file. There are a number of histories that record the number of particles for electrons.

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.
- Ensure that the *Dump at Time Zero* checkbox is unselected (the electrons show up until dump 1; if you dump at time zero and wish to visualize the electrons over all the dumps, you can run the included createMissingPtclDumps.py analyzer over species "electrons" to add an empty dump 0)
- 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." This is shown in the window below.

Visualizing the Results

After run completion, continue as follows:

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

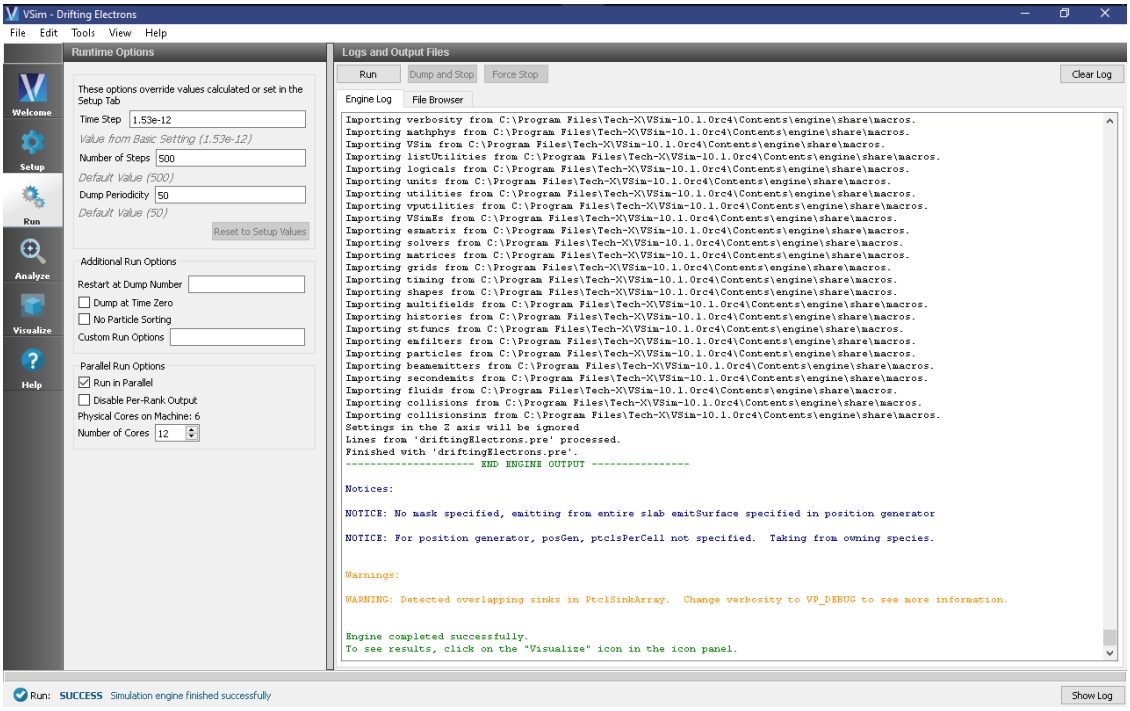


Fig. 6.18: The Run Window at the end of execution.

To view the phase space distribution for the drifting electrons, select *Phase Space* from the drop down *Add a Data View* tab. This will open a new tab called *Phase Space*. In *Base Variable*, select *electrons*. Select *electrons_x* for *X-axis* and *electrons_ux* for *Y-axis*. Click *DRAW* and move the *Dump* slider to view electron accelerating and scattering when they drift over the space. The electron phase space at dump number 10 is shown in Fig. 6.19.

Further Experiments

At lower applied electric fields, electrons are more collisional due to increased cross section. Try reducing the CATHODE_POTENTIAL, and observe more scattered electron distribution when drift over space.

At higher applied electric fields, not only elastic collisions, but also inelastic collisions will take place between electrons and atoms, which further reduce electron drifting velocity and mobility. For further experiments, try adding other collision types, such as excitation and ionization, and observe the effects to electron drifting velocity.

6.3.2 Langmuir Probe (langmuirProbe.sdf)

Keywords:

`electrostatics, particle in cell, sheath, box bounding, internal boundary`

Problem description

This example computes the fields and particles in a box, with an interior probe, modeled as a particle absorber and a constant-voltage (Dirichlet) boundary condition. There is an immobile, background neutralizing charge density. The electrons move to the walls and the probe, creating sheaths at all interfaces.

This simulation can be performed with a VSimBase license.

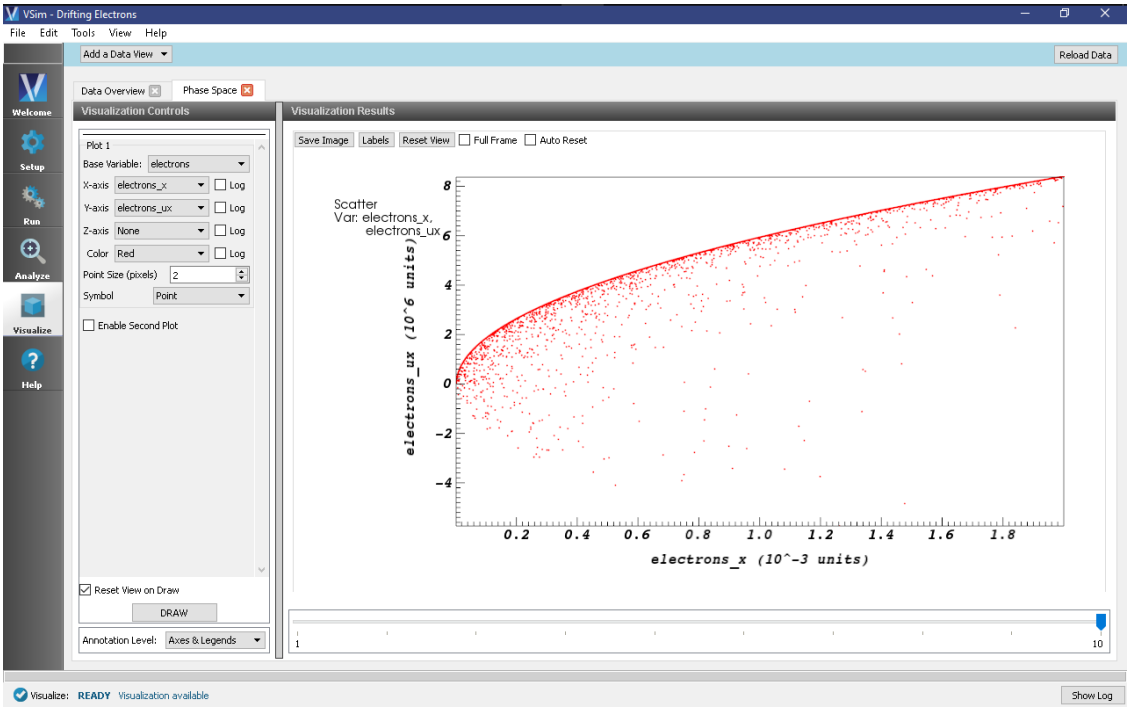


Fig. 6.19: Electron phase space at Dump 10.

Opening the Simulation

The Langmuir Probe example is accessed from within VSimComposer by the following actions:

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim for Plasma Discharges* option.
- Expand the *DC Plasmas* option.
- Select "Langmuir Probe" 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. 6.20. 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.

Simulation Properties

Constants are set up to allow setting the electron temperature in eV (ELEC_TEMP_EV), the electron density (NOM_DENS_E), the number of cells (NCELLS_X, NCELLS_Y) in the x and y directions, the number of particles per cell (PPC), and the size of the simulation (LEN_X, LEN_Y) in the x and y directions.

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.

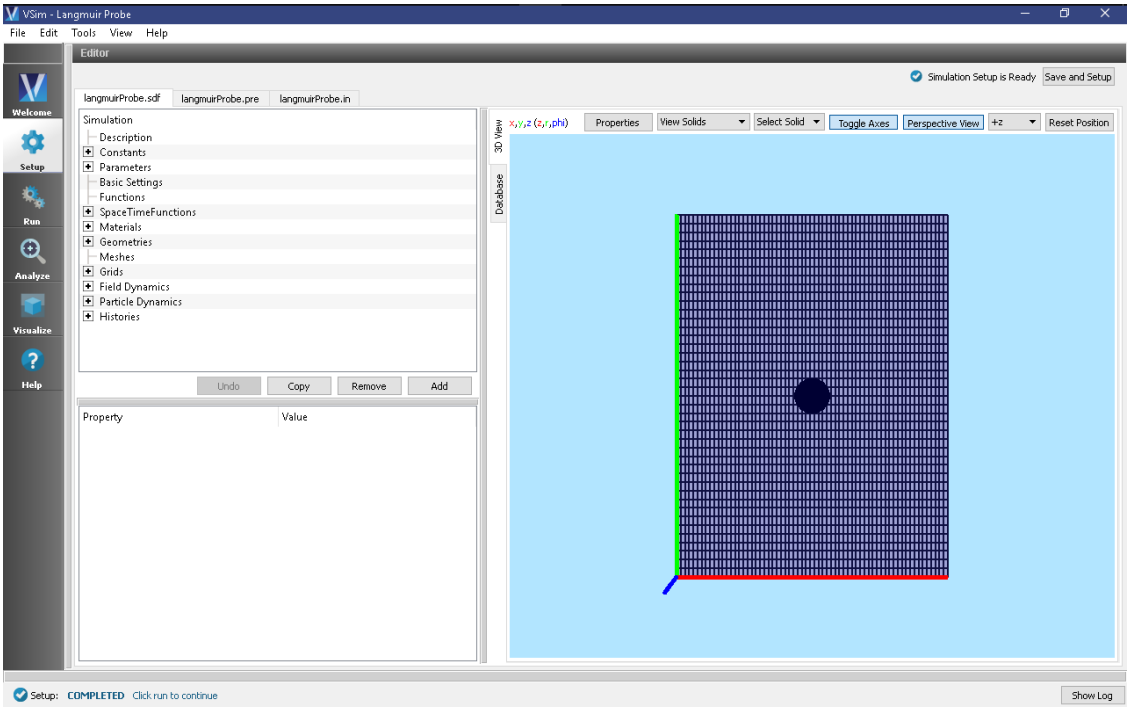


Fig. 6.20: Setup Window for the Langmuir Probe example.

- 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. 6.21.

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 potential, expand *Scalar Data* in the *Data Overview* tab and select *Phi*. The potential in the visualization window resembles that shown in Fig. 6.22.

To view the electrons and sheaths, expand the *Particle Data*, expand *electrons* and select *electrons*. Move the dump slider forward in time to see the formation of the sheaths as seen in Fig. 6.23.

Further Experiments

Try adding in another geometry for inclusion of the support rod or try changing the geometry to represent a different probe.

6.4 Ion Sources

6.4.1 Simple Ion Source (simpleIonSource.sdf)

Keywords:

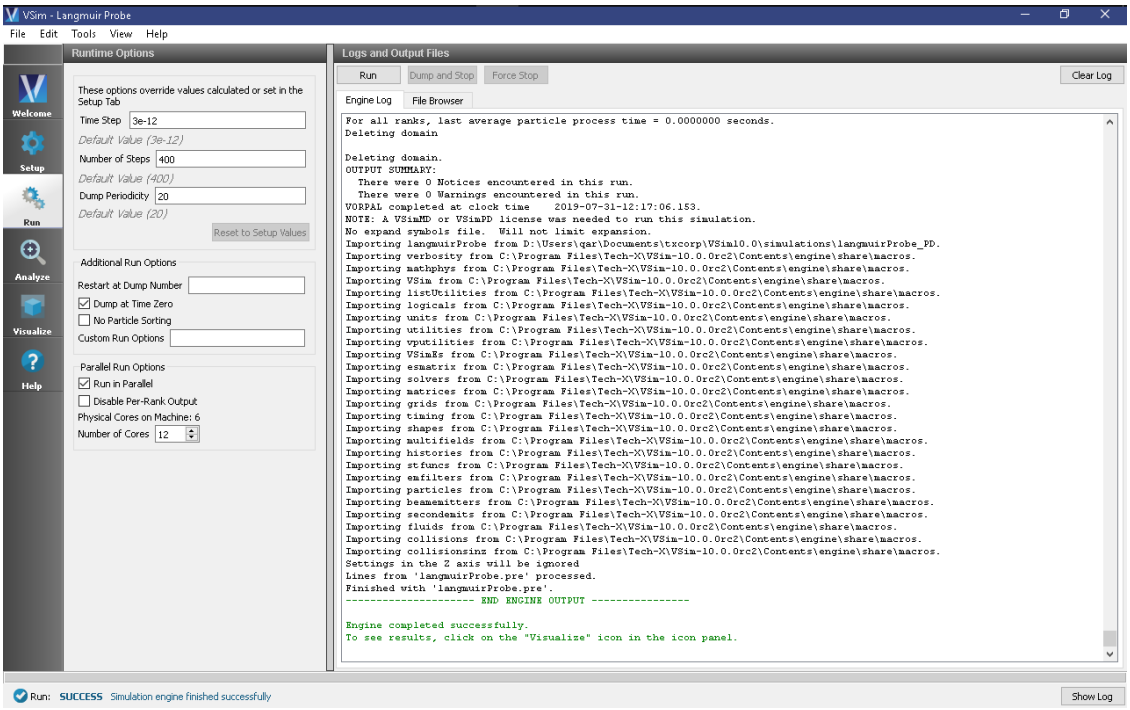


Fig. 6.21: The Run Window at the end of execution.

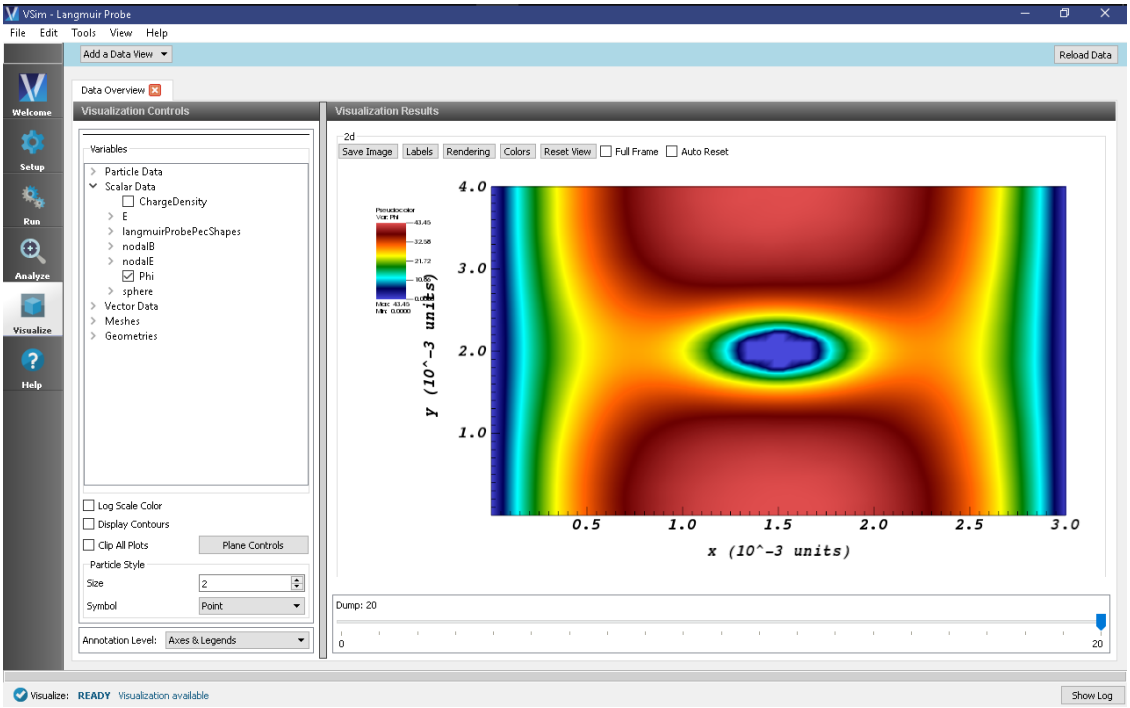


Fig. 6.22: The electrostatic potential

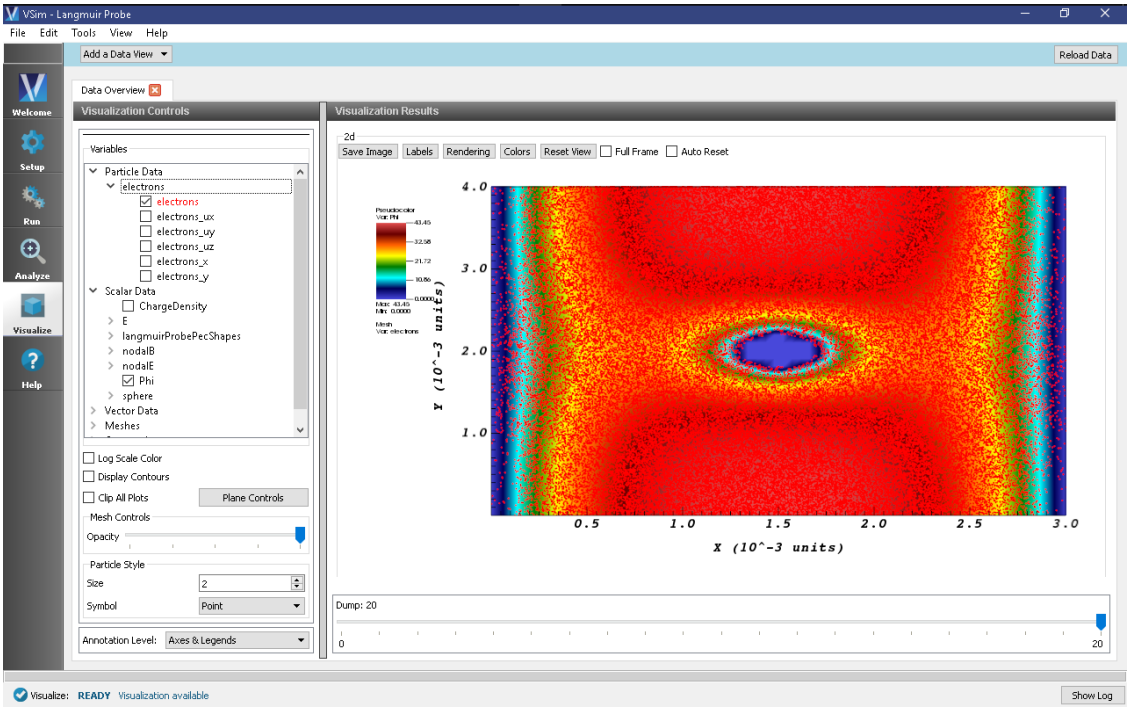


Fig. 6.23: The sheath formation

coherent ion beam, electron emitters, electron-neutral collisions, plasma source

Problem Description

This simple ion source example illustrates how to generate an electron population by emitting electrons off of a cathode. The charge neutral plasma forms due to electron-neutral collisions which ionizes the neutral fluid to generates a second electron (in each collision) and a singly charged positive ion. The ions are then extracted from the plasma source with two extraction plates biased to a large negative potential. The two extraction plates accelerate the ions out of the plasma generating an ion beam which is focused to a small width according the space between the two extraction plates. To further focus the ion beam, absorbing plates are placed above the extraction plates which further focuses the ion beam to a smaller width. A magnetic field can also be imposed to further restrict the motion of the ions. For the example shown here a SpaceTimeFunction is used to define the magnetic field but we have not included the magnetic field in the simulation. The user is free to add this magnetic field to perform further tests on the setup.

This simulation can be run with a VSimPD license.

Opening the Simulation

The simple ion source example is accessed from within VSimComposer by the following actions:

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim for Plasma Discharges* option.
- Expand the *Ion Sources* option.
- Select *Ion Source* 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.

All of the properties and values that create the simulation are now available in the Setup Window as shown in Fig. 6.24. You can expand the tree elements and navigate through the various properties, making any changes you desire. Please note that many options are available by double clicking on an option and also right clicking on option. 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`.

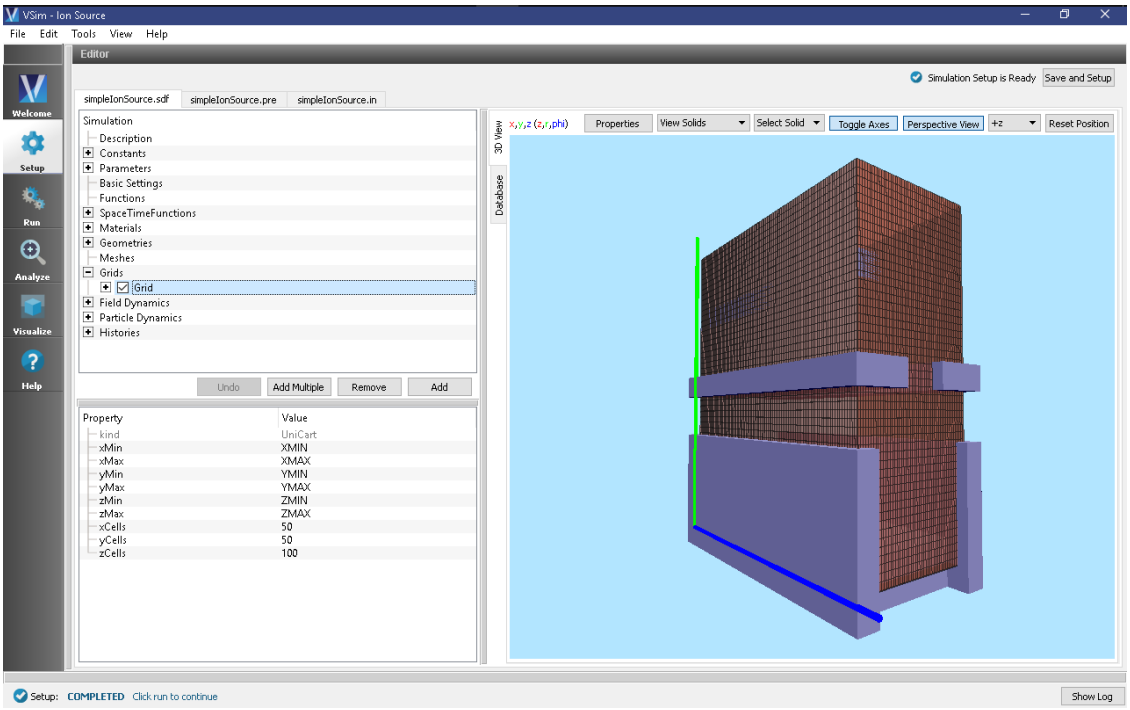


Fig. 6.24: Setup window for the Simple Ion Source example.

Simulation Properties

This example contains many user defined *Constants* which help simplify the setup and make it easier to modify. The following constants can be modified by left clicking on *Setup* on the left-most pane in VSim. Then left click on + sign next to *Constants* and all the constants used in the simulation will be displayed. To add your own constant, right click on *Constants* and left click on *Add User Defined*. Below is an explanation of a few of the constants used. There are several more constants included in the simulation.

XMIN/XMAX,YMIN/YMAX,ZMIN/ZMAX: Physical dimensions of simulation in meters.

CATHPOT: Potential of electron source from which electrons are emitted

B0: Magnetic of external magnetic field (if you choose to include this)

T0: Time that the electrons are emitted from the electron source.

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.” This is shown in Fig. 6.25.

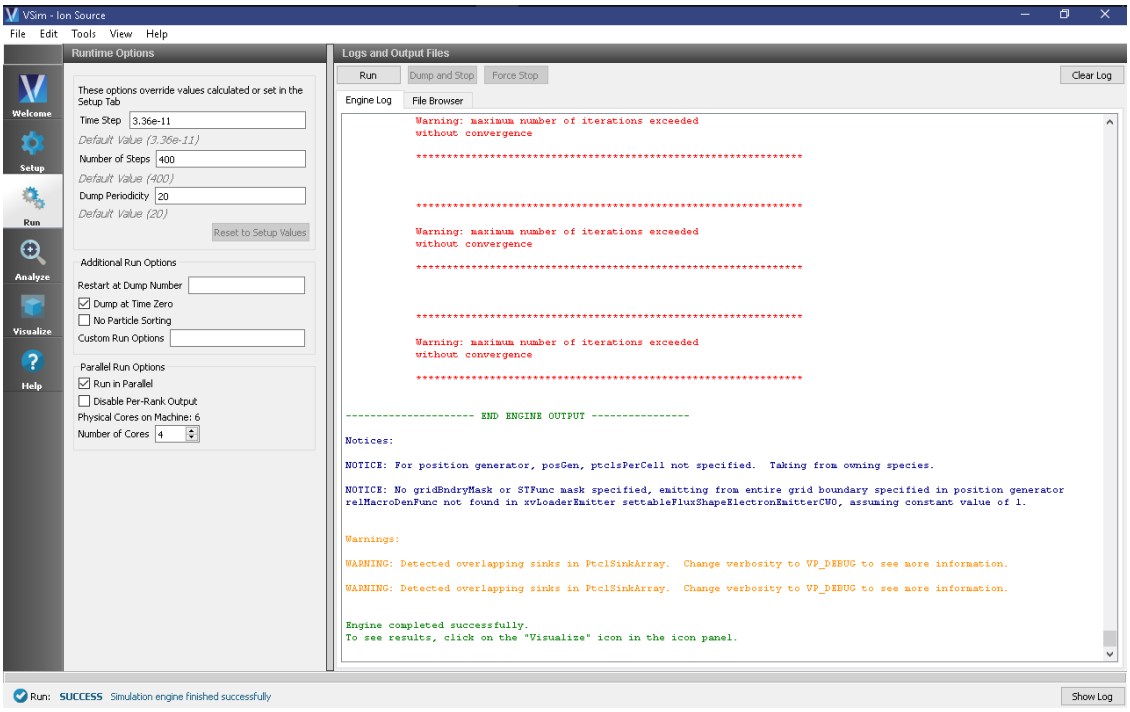


Fig. 6.25: The Run Window at the end of execution.

Analyzing the Results

After performing the above actions, continue as follows:

- Proceed to the Analysis Window by pressing the *Analyze* button in the navigation column.
- In the list of *Available Analyzers* , select *computePtclNumDensity.py* and press *Open** The analyzer fields should be filled as below:
  - *simulationName*: simpleIonSource
- Fill in *SpeciesName* with “electrons” or “Ions” which are the names of the two species in

the “Particle Dynamics” tab in the visual setup.

- Click *Analyze* in the top right corner.
- The analysis is completed when you see the output shown in Fig. 6.26.

The resulting data is called *electronDensity* and *IonDensity* and shows the number density in the 3D simulation domain in units of #/m^3.

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.
- Clicking on the *Data View* dropdown menu shows there are many different types of data to visualize.

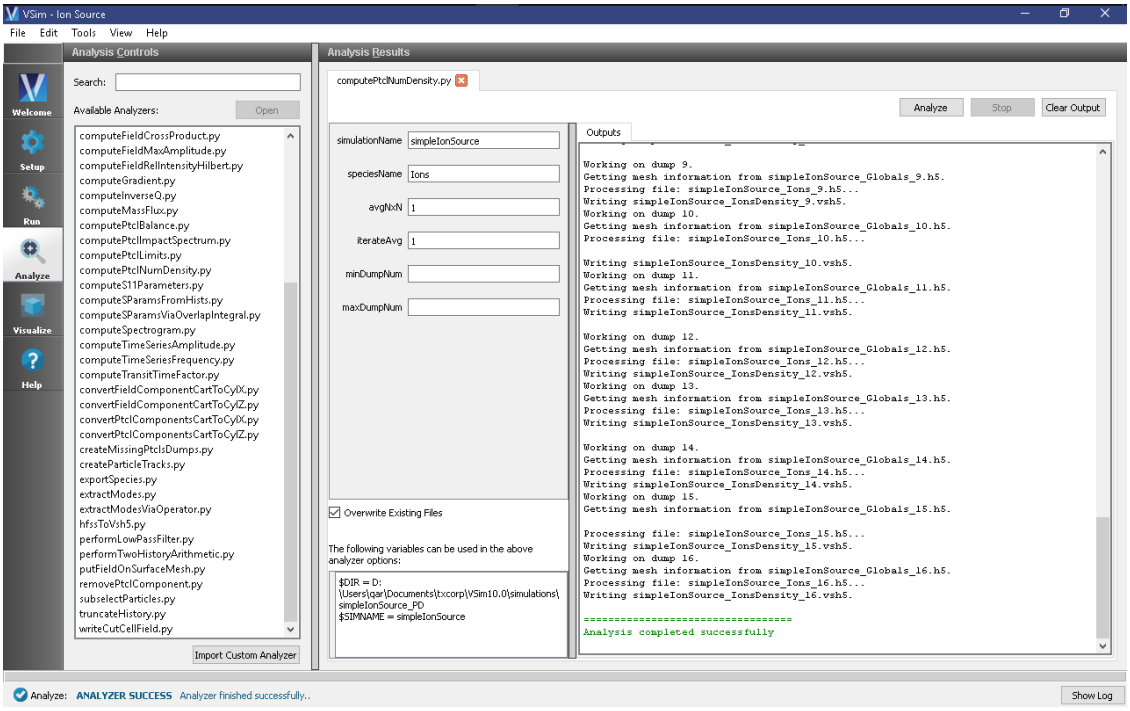


Fig. 6.26: The Analyze Window at the end of a successful run.

- To get started, lets visualize the formation of the plasma at the bottom near the cathode and the resulting ion beam.
- Click on *Data View* then click on *Data Overview*
- Under *Variables*, click on “Particle Data”, “electrons” and the top most box (“electrons”)
- Click on “Ions” then click on the top most box (“Ions”)
- Finally, to visualize the plasma species in the context of the geometries in the simulation, click on “Geometries”, “poly(extractionPlate1)”, “poly(extractionPlate2)”, and “poly(electronSource)”.
- You can rotate the figure by holding down the right mouse button and moving the mouse.

The resulting visualization is shown in Fig. 6.27.

This plot shows that the ions have been extracted from the plasma reservoir to form a coherent ion beam.

Next you can visualize the potential. Click on “Data View” at the top left of the Visualization window. Then click on “Field Analysis”. A new tab will open called “Field Analysis”. In the “Field Analysis” tab, click on the arrow next to the “Field” option and you will see all the available fields for visualizing. Click on “Phi”, which is the potential. To get a 2D view with a 1D slice of the data click on “Side-by-side 2d/1d” next to the *Layout* option. Under “Slice Settings”, change the value from 0 to 0.005, then hit Enter. Now the data is being shown in the 2D plane at z=0.005 m. Finally, you can slide the bar under the “Vertical,Horizontal,Advanced” tabs to change the 1D slice that is shown. You need to click on “Perform Lineout” to see the new slice. To change the axis labels, click on “labels” in the respective plots.

The visualization window showing the potential is shown in Fig. 6.28.

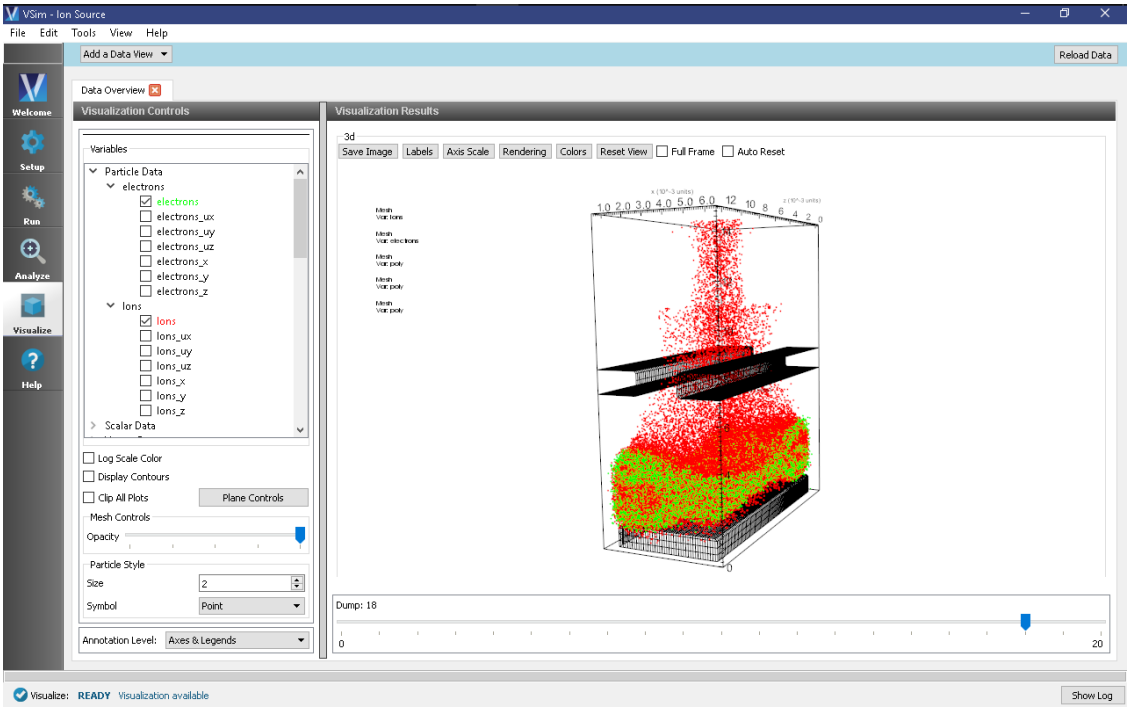


Fig. 6.27: The evolved ion beam near the end of the simulation.

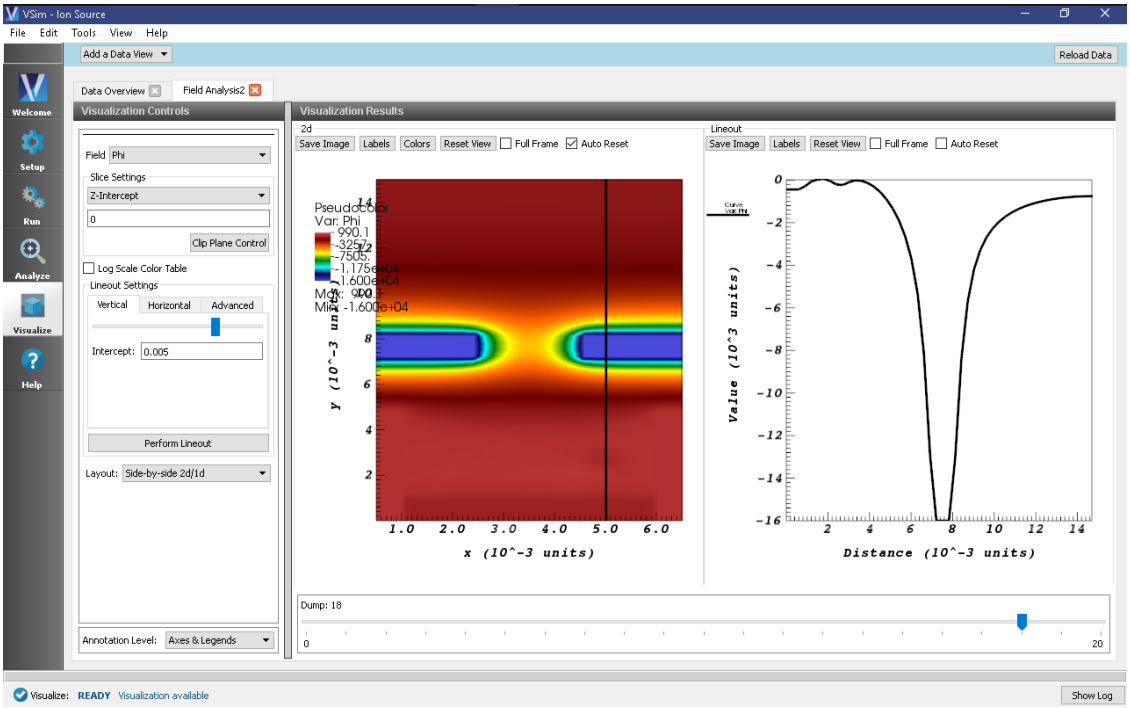


Fig. 6.28: Plot showing the 2D potential in the z=0.005 m plane along with a 1D plot along the y-axis (at a fixed value of x).

Further Experiments

In this example simulation, we have reduced the ion mass to illustrate some key aspects of the capabilities in VSim. For example, we illustrate the formation of the plasma in the plasma source region. As the plasma builds up, the ions are extracted from the plasma source region by the "extraction plates". The ions will also be attracted to electrons in the plasma source region due to the ambipolar electric field which is created by the spatial separation between the ions and electrons. This example can be used to design an experimental setup with realistic ion masses. With larger masses, the ambipolar electric field makes it more difficult for the ions to escape. Therefore, the following modifications can be performed to test more realistic scenarios. (1) Change ion species to hydrogen. (2) Modify "T0" which is the time that the electrons are emitted from the electron source. Increasing T0 means that the ions source becomes more dense. For heavy ions, you will need to modify this so that the ambipolar electric field does not become too large thus pulling the ions back in to the plasma source region. (3) Modify "EPYPosition" which is the distance along the y-axis the extraction plates are plates.

6.4.2 Penning Source (PenningSource.sdf)

Keywords:

coherent ion beam, electron emitters, electron-neutral collisions, plasma source

Problem Description

This Penning Source example illustrates how to generate an electron population by emitting electrons off of a cathode using VSim's Settable Flux Shape Emitter model. The charge neutral plasma forms due to impact ionization from electron-neutral collisions which ionizes the neutral Argon gas to generate a second electron (in each collision) and a singly charged Argon ion. This simulation also includes impact excitation and elastic collisions between electrons and Argon gas. The various types of elastic and inelastic collisions of the particles are calculated with the Vorpal engine's Monte Carlo package. The plasma particles, background gas, and collisions are set up in the *Particle Dynamics* Element. The cross-sections for these collisions are imported from 2-column data files. The electrons are attracted in to the plasma chamber by charging the anode (filled with Argon gas) to a positive potential. Secondary electrons are emitted off the anode due to the primary electrons interacting with the anode walls. This is set up using a *Secondary Emitter* model in VSim.

The Argon ions are then extracted from the plasma source with two extraction plates biased to a large negative potential. The two extraction plates accelerate the ions out of the plasma generating an ion beam which is focused to a small width according to the space between the two extraction plates. To further focus the ion beam, absorbing plates are placed below the extraction plates which are part of the anode. A magnetic field is imposed to further restrict the motion of the ions. For the example shown here a Space Time Function is used to define the magnetic field. The electric field is solved via Poisson's equation using a Generalized Minimal Residual (GMRES) linear solver. This solver is chosen under *Field Dynamics -> PoissonSolver*. A combination of Dirichlet and Neumann boundary conditions involving the shapes and simulation boundaries are imposed. Field boundary conditions are imposed under *Field Dynamics -> FieldBoundaryConditions*. The plasma is represented by macro-particles which are moved using the Boris scheme in a 3D Cartesian coordinate system.

This simulation can be run with a VSimPD license.

Opening the Simulation

The Penning Source example is accessed from within VSimComposer by the following actions:

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim for Plasma Discharges* option.
- Expand the *Ion Sources* option.

- Select *Penning High Intensity Ion Source* 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.

All of the properties and values that create the simulation are now available in the Setup Window as shown in Fig. 6.29. You can expand the tree elements and navigate through the various properties, making any changes you desire. Please note that many options are available by double clicking on an option and also right clicking on an option. The right pane shows a 3D view of the geometry as well as the grid. To show or hide the grid, expand the "Grid" element and select or deselect the box next to `Grid`. You can also show or hide the different geometries by expanding the "Geometries" element and selecting or deselecting "anodePlasmaPlate", "cathode" and/or "extractionPlates". Note that these three shapes are unions of the other more basic shapes included in this simulation.

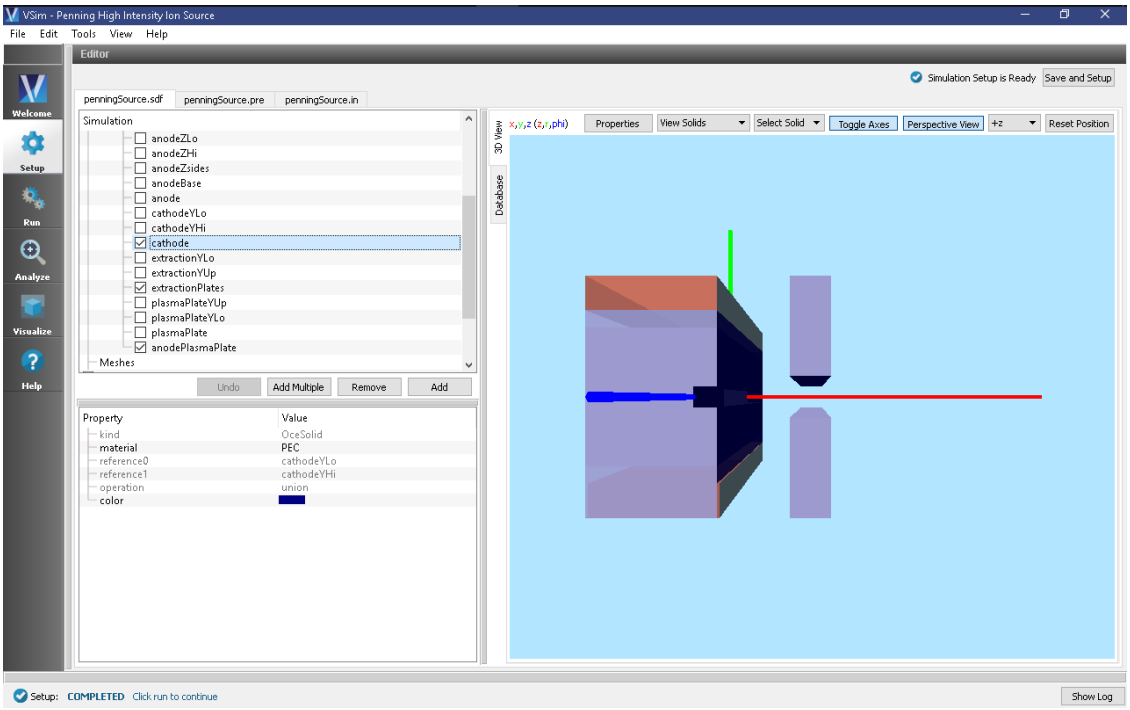


Fig. 6.29: Setup window for the Penning Source example.

Simulation Properties

This example contains many user defined *Constants* which help simplify the setup and make it easier to modify. The following constants can be modified by left clicking on *Setup* on the left-most pane in VSim. Then left click on + sign next to *Constants* and all the constants used in the simulation will be displayed. To add your own constant, right click on *Constants* and left click on *Add User Defined*. Below is an explanation of a few of the constants used. There are several more constants included in the simulation.

XSTART/XEND,YSTART/YEND,ZSTART/ZEND: Physical dimensions of simulation in meters.

CATHODE_VOLTAGE: Potential of electron source from which electrons are emitted

ANODE_VOLTAGE: Potential of anode which attracts the electrons from the cathode

PENNING_MAGNETIC_FIELD: Magnetic of external magnetic field used to confine the ion beam

T1: Time that the electrons are emitted from the electron source.

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 simulation takes about 1500 time steps before an ion beam is observed to begin forming and 4000 time steps before a persistent modestly dense beam has formed. The run has completed successfully when you see the output, "Engine completed successfully." This is shown in Fig. 6.30.

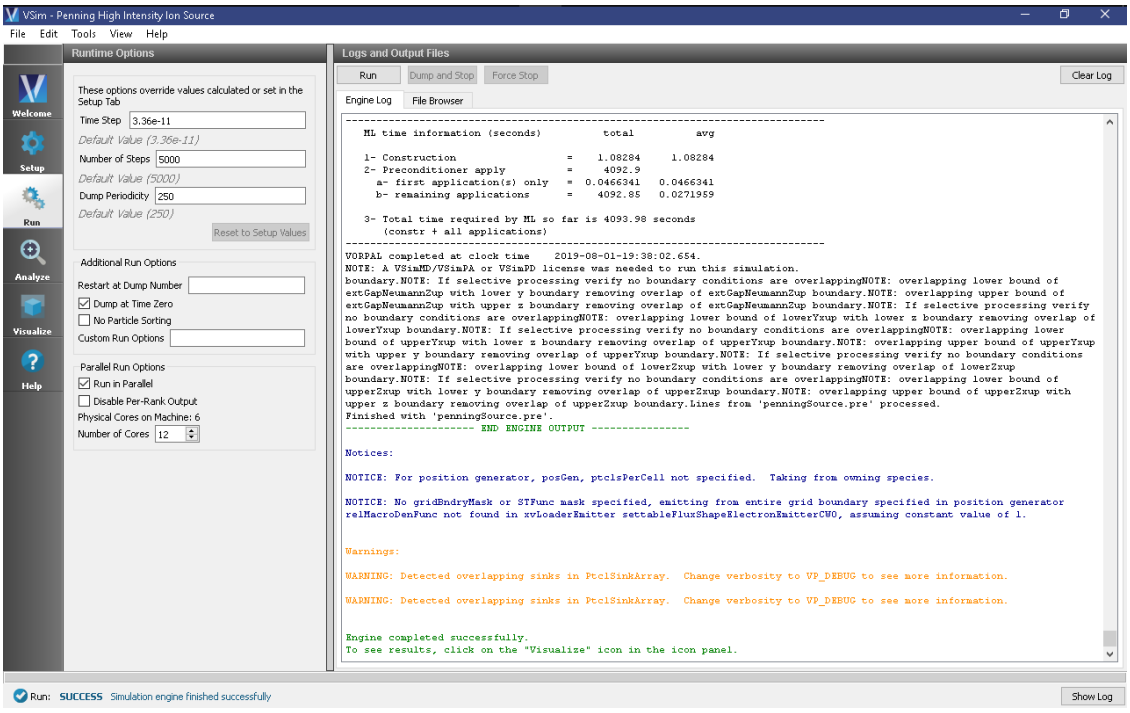


Fig. 6.30: The Run Window at the end of execution.

Analyzing the Results

After performing the above actions, continue as follows:

- Proceed to the Analysis Window by pressing the *Analyze* button in the navigation column.
- In the resulting list of *Available Analyzers*, select *computePtclNumDensity.py* and press *Open*
- The analyzer fields should be filled as below:
  - *simulationName*: PenningSource
- Fill in *SpeciesName* with "electrons" or "Ar1" which are the names of the two species in

the "Particle Dynamics" tab in the visual setup.

- Click *Analyze* in the top right corner.
- The analysis is completed when you see the output shown in Fig. 6.31.

The resulting data is called *electronDensity* and *Ar1Density* and shows the number density in the 3D simulation domain in units of #/m^3.

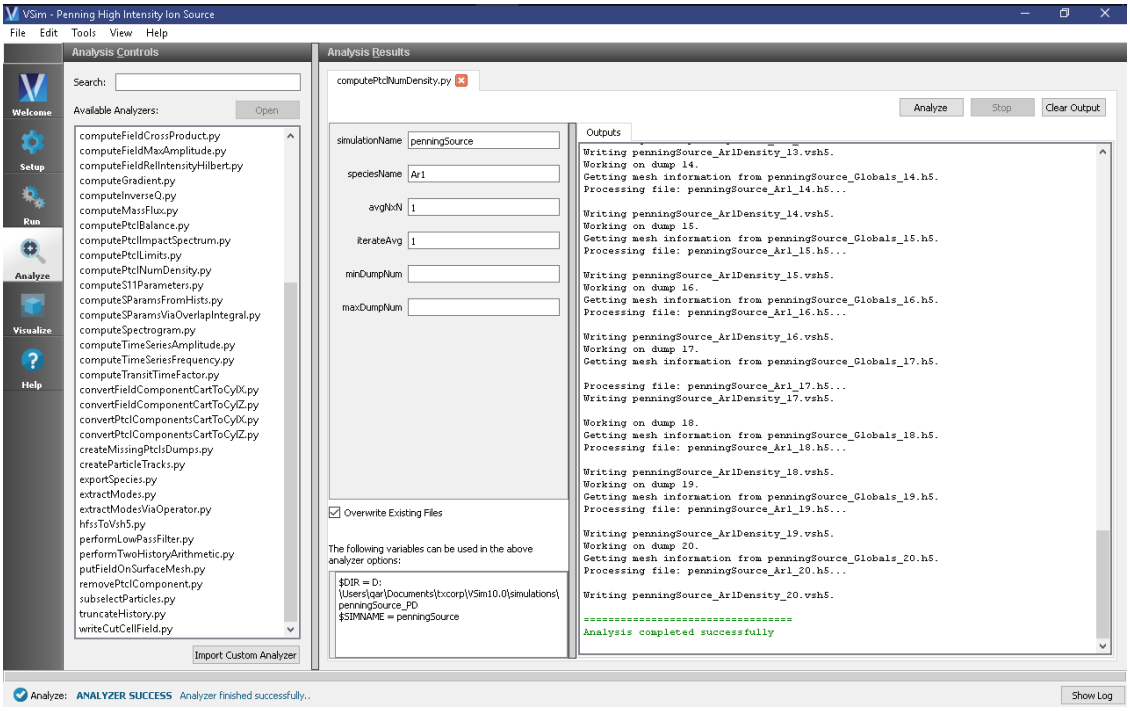


Fig. 6.31: The Analyze Window at the end of a successful run.

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.
- Clicking on the *Add a Data View* dropdown menu shows there are many different types of data to visualize.
- To get started, lets visualize the formation of the plasma in the anode and the resulting ion beam.
- Click on the *Data Overview* tab which is a default tab already loaded in the "Visualize" section of VSim.
- Under *Variables*, click on "Particle Data", "electrons" and the top most box ("electrons")
- Click on "Ar1" then click on the top most box ("Ar1")
- Finally, to visualize the plasma species in the context of the geometries in the simulation, click on "Geometries", "poly(cathode)", "poly(extractionPlates)", and "poly(anodePlasmaPlate)".
- You can rotate the figure by holding down the left mouse button and moving the mouse.
- Slide the bar to the right at the bottom of the display window to watch the ions form and accelerate out of the anode.

The resulting visualization is shown in Fig. 6.32.

This plot shows that the ions have been extracted from the plasma reservoir to form a coherent ion beam.

Next you can visualize phase space which is a plot of velocity vs distance. Click on "Add a Data View" at the top left of the Visualization window. Then click on "Phase Space". A visualization tab will open next to "Data Overview". Next to "Base Variable", click "Ar1". For the "X-axis", choose *Ar1_x* and for the "Y-axis", choose *Ar1_ux*. This produces a plot of velocity vs. position. Finally, you can include position along the z-axis by clicking on the "Color" tab and choosing *Ar1_z* and clicking "DRAW" at the bottom of the screen. The color indicates the location along the z-axis of each particle.

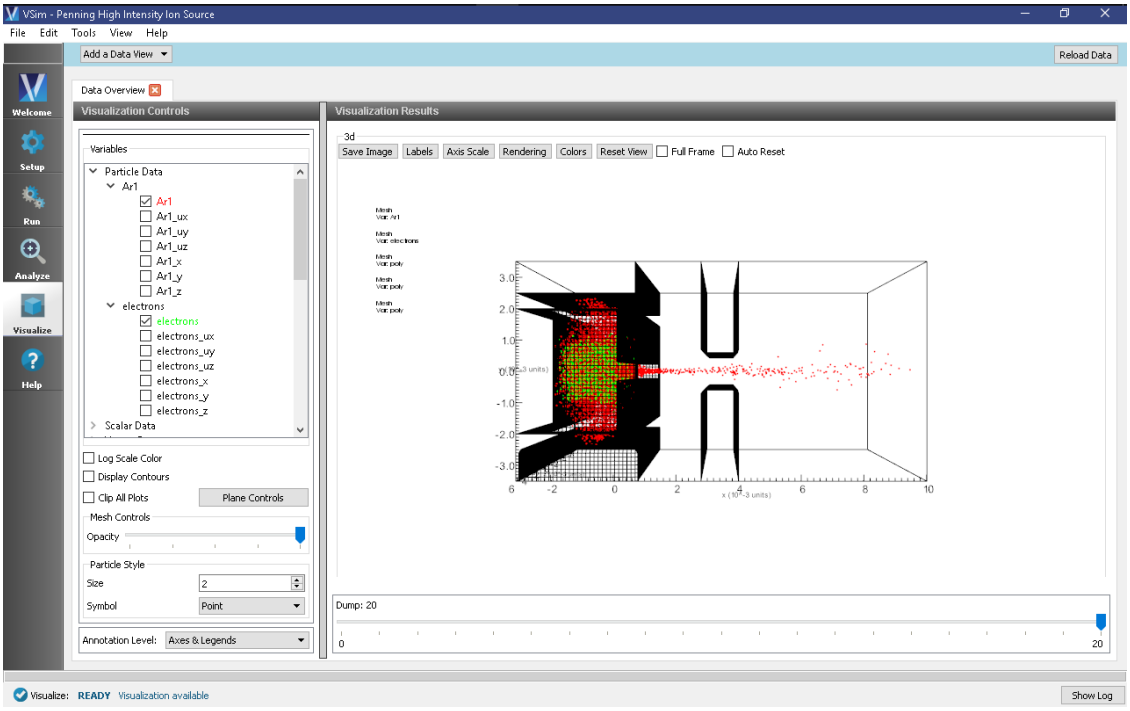


Fig. 6.32: The evolved ion beam near the end of the simulation.

The visualization window showing phase space is shown in Fig. 6.33.

Further Experiments

For this simulation, sample cross sections were used which are not necessarily the correct ones. The VSim interface can import any cross sections that are in a 2-column format. There should be NO headings in the data file. The LXcat scattering database (https://fr.lxcat.net/data/set_type.php) and EEDL cross section database contain cross section data for around one hundred different materials. As another experiment, change the cross-section used in the simulation or change the species of the background gas and import new cross-sections.

Set up a History that records the Ar1 current flowing out of the plasma chamber on to the x=XEND boundary. Right click on the "Histories" element and under "Add ParticleHistory" select "Absorbed Particle Current." You can change the name of the history by double clicking on the new "absorbedPtclCurrent0" element in the tree. Then be sure to pick the particle absorber from which you would like to collect data. Note that to add a history on a boundary, that boundary needs to be of type "Absorb and Save".

The Reactions framework allows one to set up collision interactions flexibly. The collisions involved in this example are electron-neutral collisions that lead to ionization and ohmic heating. As a further experiment, ion-neutral collisions, such as elastic scattering and charge exchange, can also be added to the simulation.

6.5 Processes

6.5.1 Negative Ion Beam (negativeIonBeam.sdf)

Keywords:

ion beam, beam transport, reactions, electrostatic

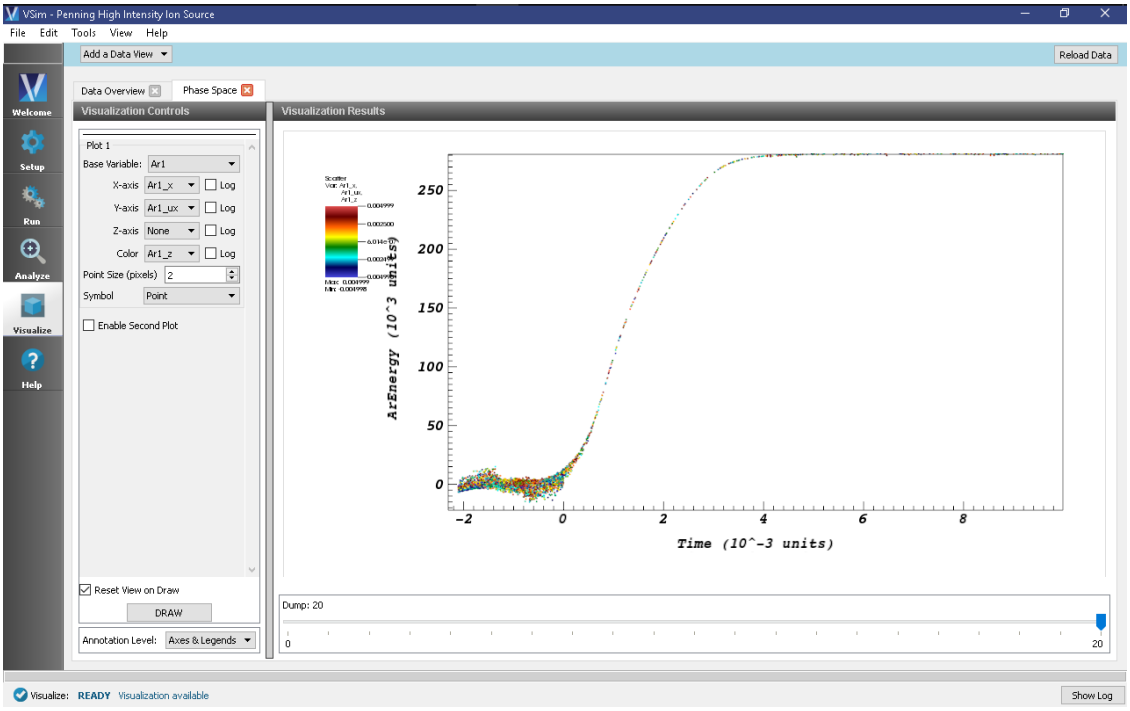


Fig. 6.33: Plot of phase space (vx vs x).

Problem description

VSim may be used to model ion beam transport and particle dynamics where the beam is represented by kinetic simulation particles. Low density background gasses can cause instabilities in the beams due to collisions between the beam particles and the background gas.

In this simulation, a beam of H- ions propagates through a background H2 gas. Collisions between the beam ions and the background gas produce electrons, H2+, and neutral H through the following reactions:

$$H^- + H_2 \rightarrow H^- + H_2^+ + e^- \text{ (ion impact ionization)}$$

$$e^- + H_2 \rightarrow H_2^+ + 2e^- \text{ (electron impact ionization)}$$

$$H^- + H_2 \rightarrow H + H_2 + e^- \text{ (detachment)}$$

$$H^- + e^- \rightarrow H + 2e^- \text{ (stripping)}$$

There are other reactions that are not included in this tutorial simulation. Typically these reactions have low cross sections. Fig. 6.34 shows the cross sections for the above reactions as a function of incident energy.

This simulation can be performed with a VSimPD license.

Opening the Simulation

The Kinetic Collisions example is accessed from within VSimComposer by the following actions:

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim for Plasma Discharges* option.
- Expand the *Processes* option.
- Select "Negative Ion Beam" and press the *Choose* button.

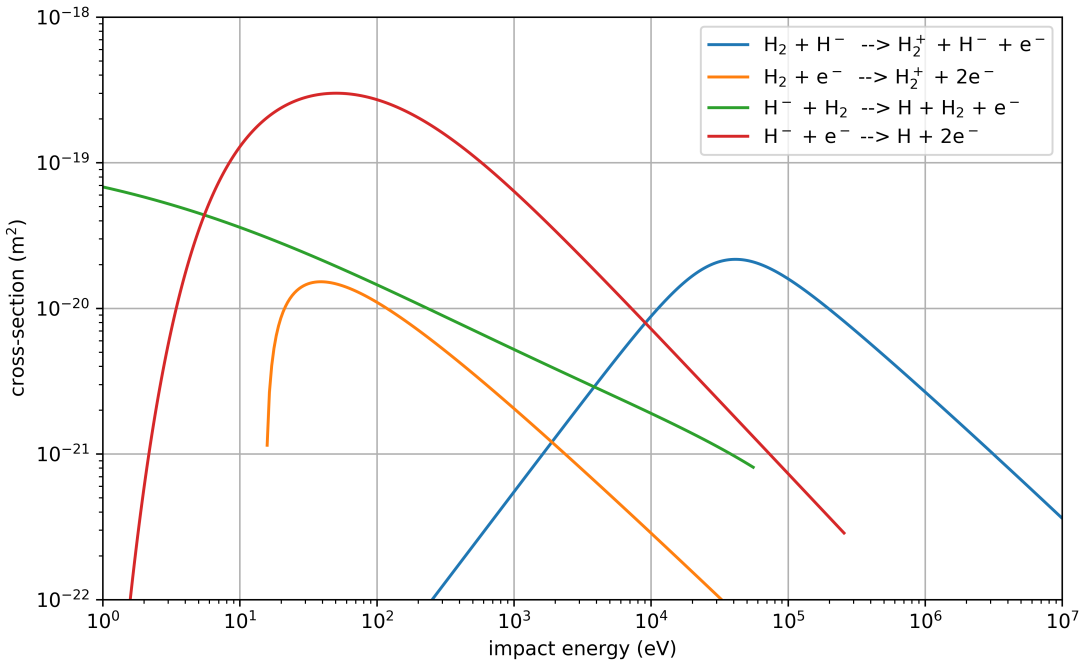


Fig. 6.34: Cross sections for the four collision reactions included in this example.

- 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, if applicable. See Fig. 6.35.

Simulation Properties

This input file contains a number of different kinetic species as well as a background fluid description of a gas. Ionization collisions between kinetic particles and the background gas are described by Monte Carlo interaction blocks of kind impactIonization, and detachment of electrons due to a collision with the background gas are of kind negativeIonDetachment. Collisions between kinetic particles and other kinetic particles are described in the input file by an interaction of kind binaryIonization.

The fields are electrostatically solved for at each time step, including the fields due to all charged particles, subject to the boundary conditions specified in the input file. There are a number of histories that record the number of particles for different species, their energies, as well as currents absorbed at the boundaries.

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.
- 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." This is shown in the window below.

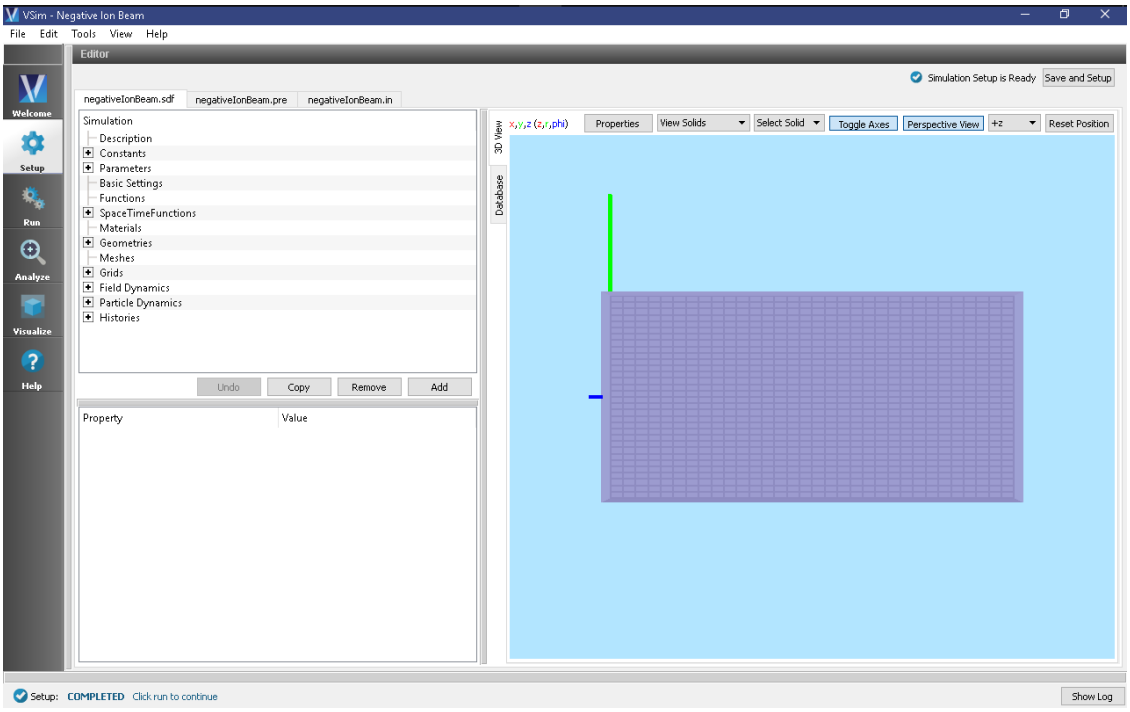


Fig. 6.35: Setup Window for the Negative Ion Beam example.

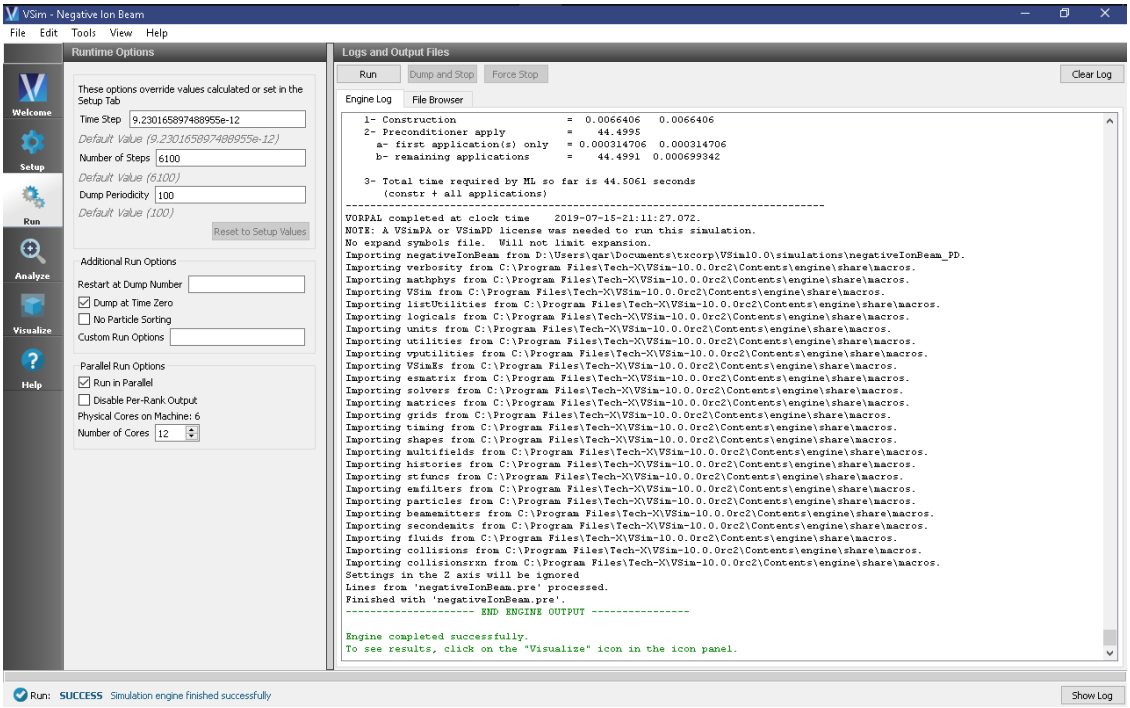


Fig. 6.36: The Run Window at the end of execution.

Analyzing the Results

If it is desired to calculate the density of the electrons the analysis script *computePtclNumDensity.py* must be used.

- First click on the *Analyze* Tab.
- From the list of *Available Analyzers* and choose *computePtclNumDensity.py*. Then click *Open* at the bottom of the *Analysis Controls* pane.
- Ensure that the "simulationName" field is "negativeIonBeam" and enter "Electrons" in the "speciesName" field. Leave the "aveNxN" and "iterateAve" with their default values.
- Press the *Analyze* button on in the upper right corner of the window to run the analysis. Below, the Analyze Tab is shown at the end of a successful run.

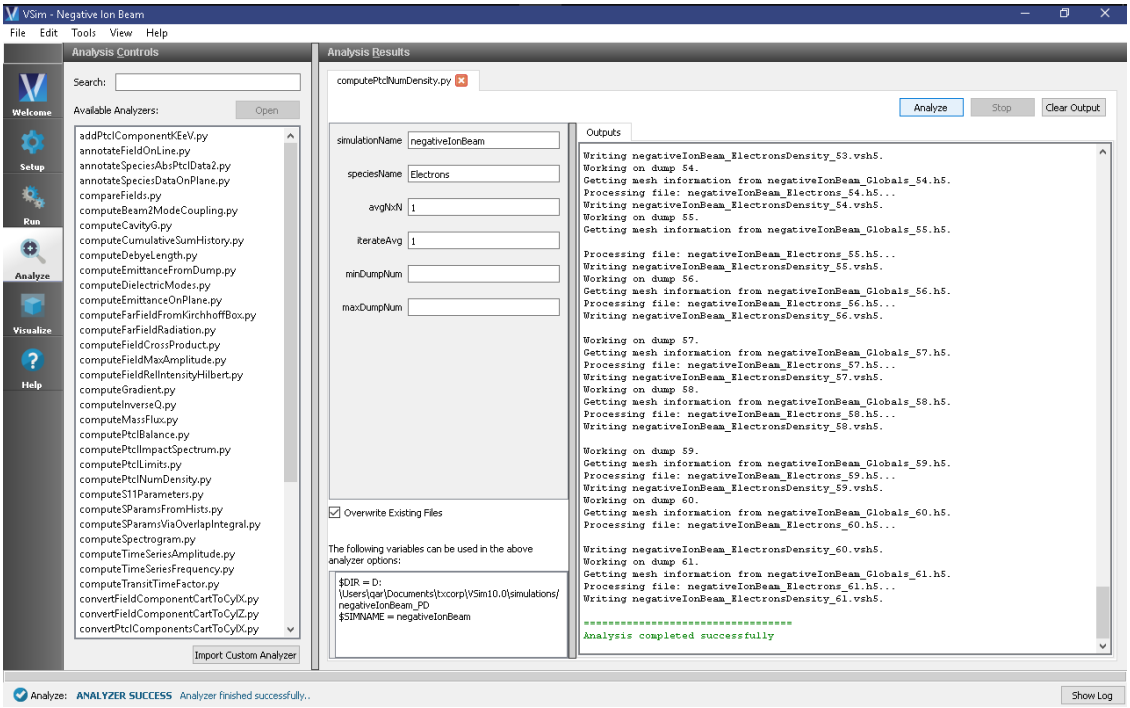


Fig. 6.37: The Analyze Window at the end of execution.

The resulting data can be visualized as "ElectronsDensity" under the Scalar Data menu in the *Visualize* Tab. A plot of this data is shown below in Fig. 6.38. The density of H2plus, Hminus or Hneutral can also be calculated if those species names are used in place of "Electrons" and the analyzer is re-run. If you have previously navigated to the *Visualize* Tab, you will need to press the *Reload Data* button at the bottom of the *Visualize* Tab to view the data.

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.
- Expand "Particle Data" and select "Electrons," "H," "H2Plus," and "Hminus".
- Then expand "Scalar Data" and select "Phi."
- Check the *Display Contours" box, which is below the *Variables* box in the *Visualization Controls* pane.
- Set the "# of Contours" to 20. The scroll through the dumps to produce the image in Fig. 6.39.

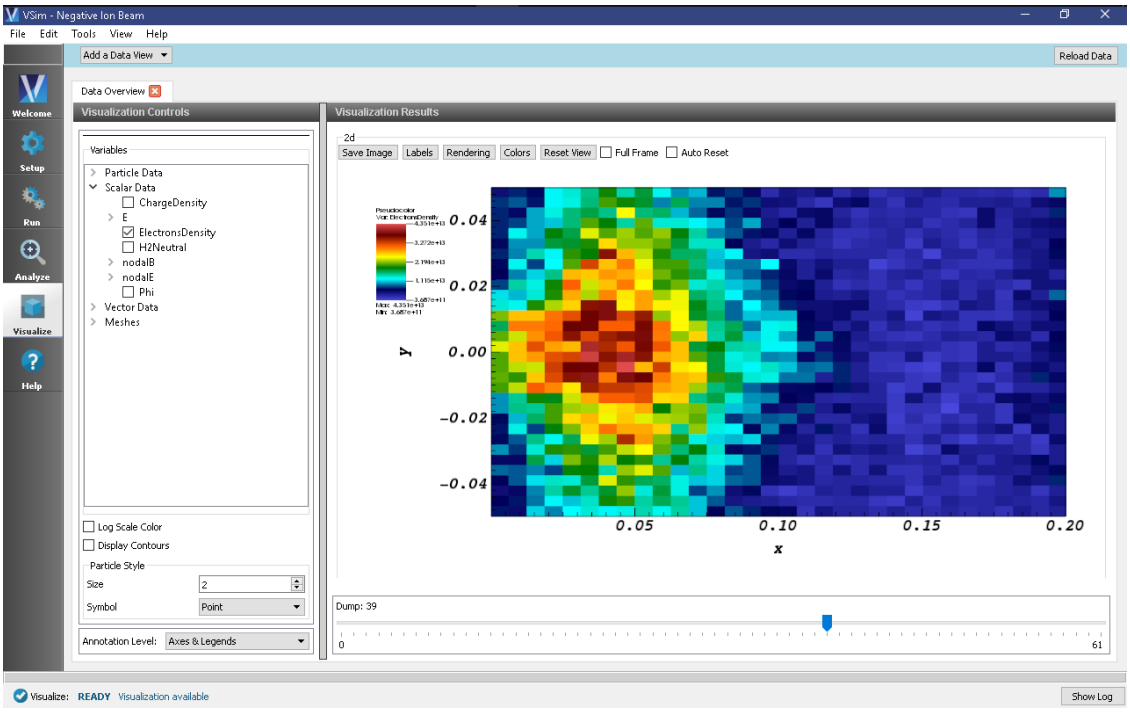


Fig. 6.38: Plot of the electron density at dump 39.

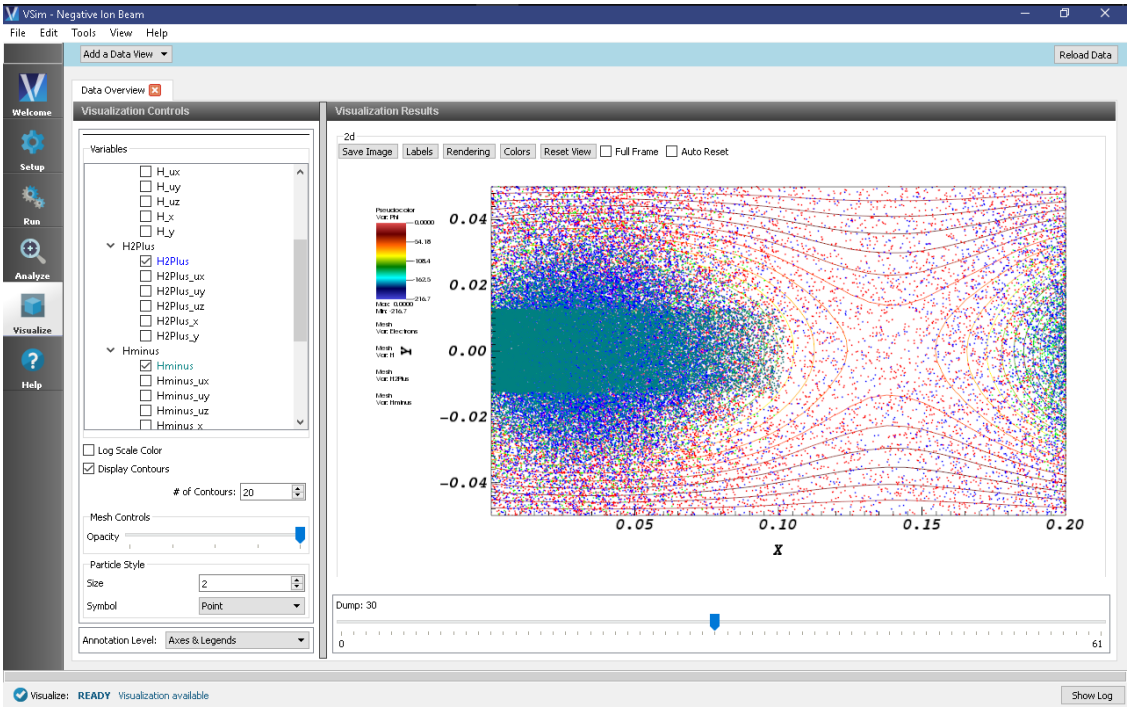


Fig. 6.39: Visualization of particle densities as a color contour plot, overlaid with a scatter plot of the particle positions.

Further Experiments

The background gas pressure is higher than one would typically see in an accelerator in this example so that the example will produce results quickly. Decreasing the pressure will give the same results, but over longer time scales.

Since this beam is negatively charged, it repulses electrons from the region near the beam. Decreasing the beam current will produce more neutralizing H2+ near the beam as the electrons can more effectively ionize the background H2 gas in that area.

6.5.2 Neutral Heat Transport (neutralHeatTransport.sdf)

Keywords:

`heat transport, DSMC, elastic collisions, reactions`

Problem description

VSim may be used to model the heat flux through a neutral gas confined between two plates of different temperatures. This problem is a common benchmark for DSMC simulations, and is described by Bird in "Molecular gas dynamics and the direct simulation of molecular gas flows" (1994) on page 280. In this example, we model the heat transport between cold (250K) and hot (1000K) plates separated by a meter. Between the plates is a volume of neutral Argon gas that transports the heat through either free-molecular motion (in the case of lower pressure) or through collisional transport via elastic collisions (in the case of higher pressure). The simulated heat flux can then be compared to the analytic result, validating the reactions framework in VSim.

This simulation can be performed with a VSimPD license.

Opening the Simulation

The Neutral Heat Transport example is accessed from within VSimComposer by the following actions:

- Select the *New –> From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim for Plasma Discharges* option.
- Expand the *Processes* option.
- Select "Neutral Heat Transport" 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 with all the implemented physics and geometries in Fig. 6.40.

Simulation Properties

This input file contains one kinetic species of neutral Argon, the required thermalizing boundary conditions for the hot and cold plates, and the Ar-Ar elastic collisions. The constants and parameters are set up so that the Argon pressure (ARPRES) in Pa can be changed, and the simulation grid will adjust resolution to ensure that the mean-free path is always resolved. This means that multiple simulations can be run to match the analytic result for a variety of pressures/collisionality.

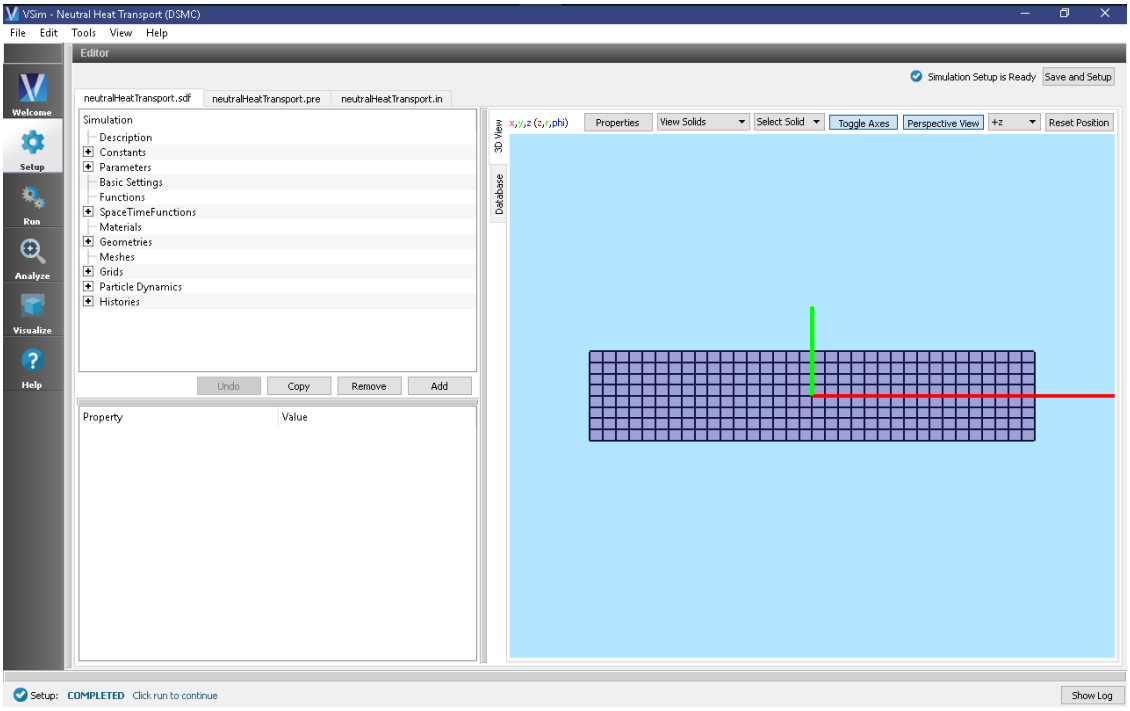


Fig. 6.40: Setup Window for the Neutral Heat Transport example.

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.
- 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." This is shown in the window below.

Visualizing the Results

After run completion, continue as follows:

- Proceed to the *Visualize* Tab by pressing the Visualize button in the left column of buttons.
- Select "History" from the *Data View* drop down menu, which is located in the upper right corner the window.

Two graphs will be shown in the resulting window (see Fig. 6.5.2). The first graph, ArEnergy, shows the total kinetic energy of the argon species. The second, AverageEnergyExchange, shows the average energy transferred between the particles and the plates as a function of time. The AverageEnergyExchange plot divided by the cross-sectional area of a plate gives the average heat flux. A python script, validation.py, is provided to calculate this heat flux from the simulation data, and plot the heat flux versus the analytic heat flux. To run this script, go to the examples directory and run python from the command line (using the command "python validation.py"). The first plot is the same histories seen in the VSim Composer visualization. The second plot is the validation.

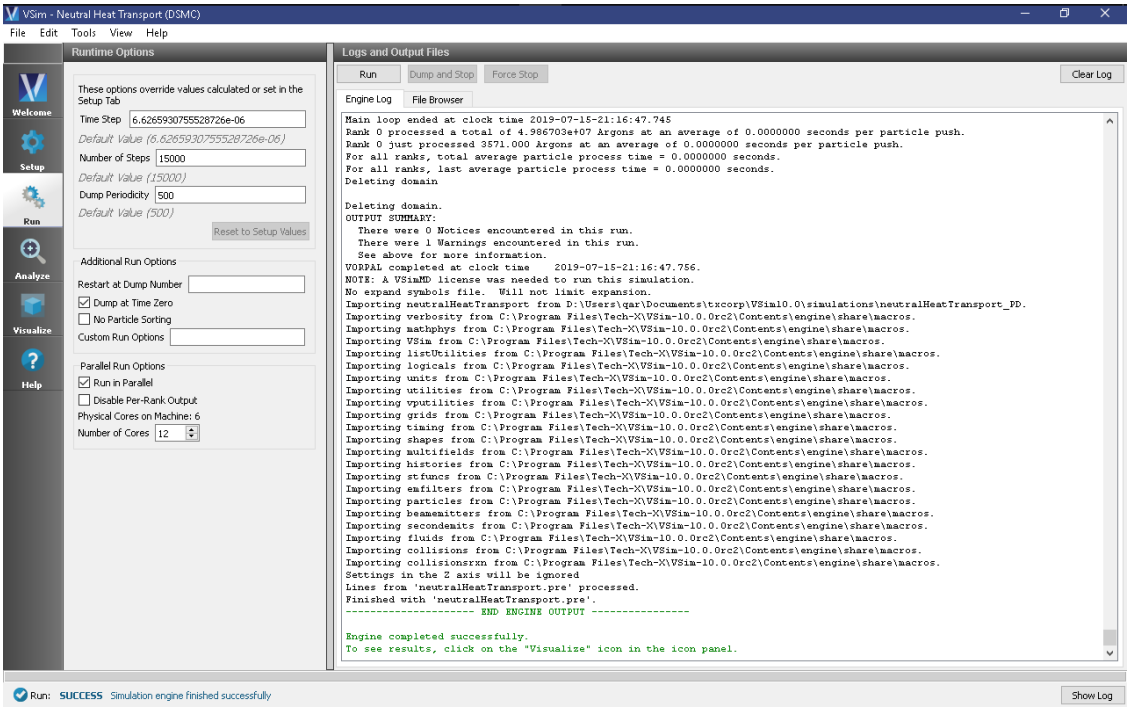
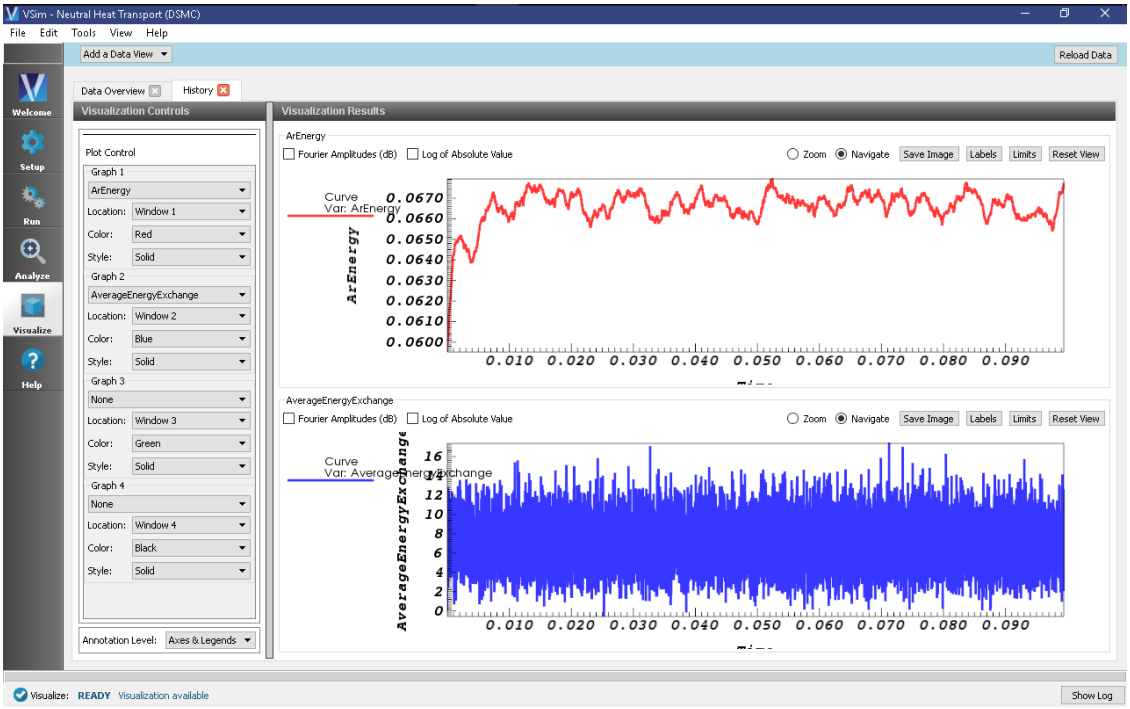
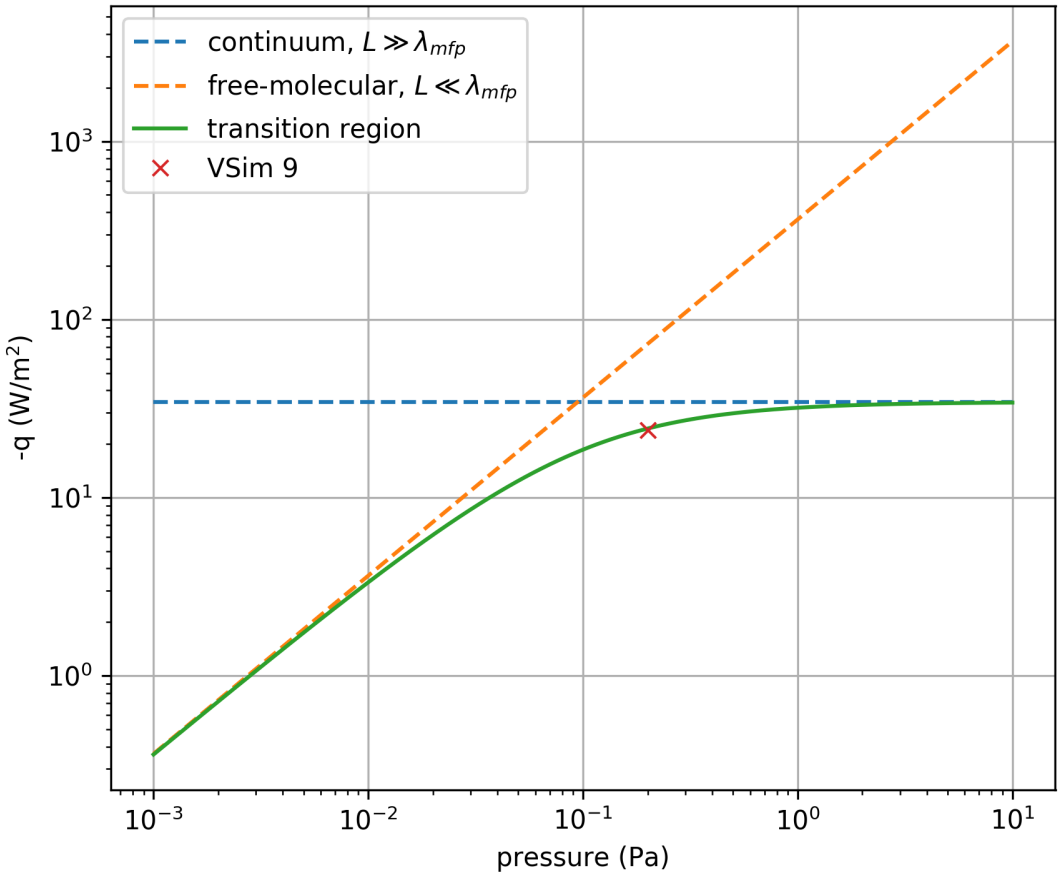


Fig. 6.41: The Run Window at the end of execution.



Further Experiments

As stated in the simulation properties section, simulations can be run with varying pressures (maintaining all else constant) and the resulting heat fluxes plotted against the analytic result, as shown in Fig. 6.5.2. The provided python script will only plot one simulation result at a time, but it can be modified easily to overplot multiple simulations. Each simulation should lie on the analytic green line. It is important to ensure that the statistics of the collisions are good enough, so when moving to lower collisionality (pressure) the number of macro particles per cell should be increased. Additionally, it is useful to switch the kinetic particle type so that it is variable weight with managed weights. This allows an isotropic macroparticle density while accounting for a variable physical particle density. Alternatively, the temperature of the plates, distance between them, species of neutral gas, etc. can all be modified to test the generality of the model and collisions.



6.5.3 Proton Beam (protonBeam.sdf)

Keywords:

electromagnetic, particle in cell, material boundary, reactions, particle emitter

Problem description

This example injects a proton beam into a column of neutral H2 gas. The geometry is setup like an electron column in an accelerator beamline (i.e. external solenoidal B-field and negative electrodes on either end for electron

confinement). Upon entering the neutral gas multiple reactions begin to occur including ionization, charge exchange, dissociation, H3+ formation, and others. The beam leaves the column, leaving behind a combination of ions, electrons, and neutrals that are either confined or ejected by the background electrode potential.

In this simulation, a beam of H+ ions propagates through a background H2 gas. Collisions between the beam ions and the background gas produce electrons, H2+, neutral H, and H3+ through the following reactions:

$H^+ + H_2 \rightarrow H^+ + H_2^+ + e^-$ (ion impact ionization)

$e^- + H_2 \rightarrow H_2^+ + 2e^-$ (electron impact ionization)

$H^+ + H_2 \rightarrow H^+ + H_2$ (elastic)

$H^+ + H_2 \rightarrow H + H_2^+$ (charge exchange)

$H_2^+ + H_2 \rightarrow H_3^+ + H$ (H3+ formation)

$H_2^+ + H_2 \rightarrow H_2^+ + H_2$ (charge exchange)

$H_2 + e^- \rightarrow H^+ + H + 2e^-$ (dissociative ionization)

$H_3^+ + H_2 \rightarrow H_3^+ + H_2$ (elastic)

Fig. 6.42 shows the cross sections for the above reactions as a function of incident energy.

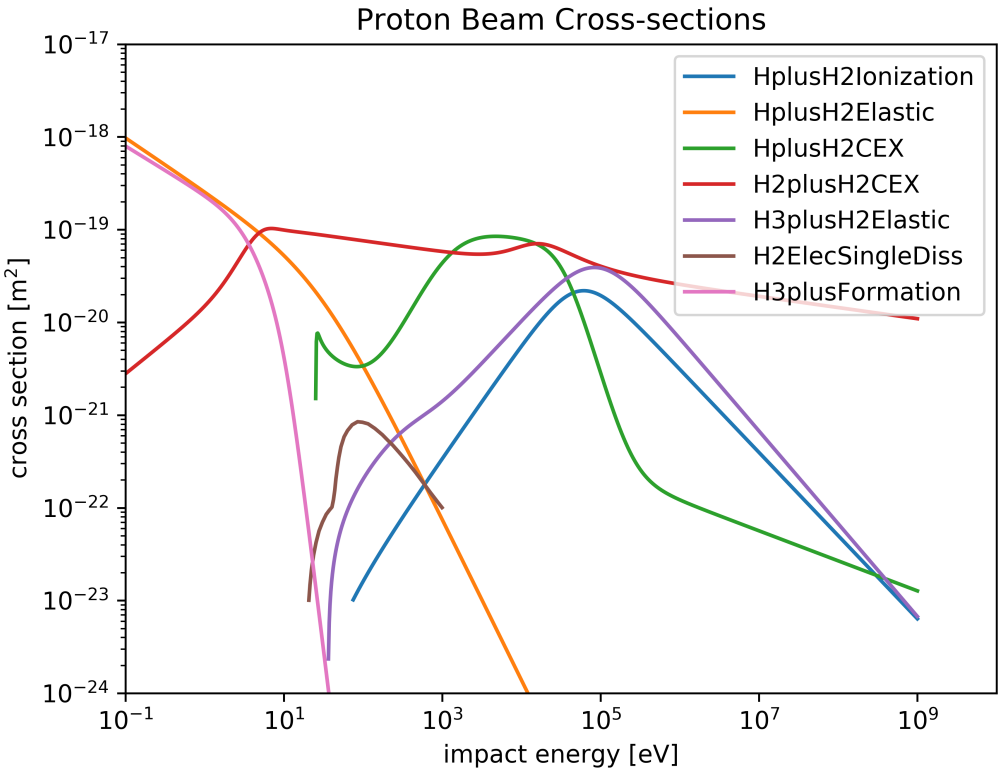


Fig. 6.42: Cross sections for the collisions included in this example.

This simulation can be performed with a VSimPD license.

Opening the Simulation

The Proton Beam example is accessed from within VSimComposer by the following actions:

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim for Plasma Discharges* option.
- Expand the *Processes* option.
- Select "Proton Beam" 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. 6.43. 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.

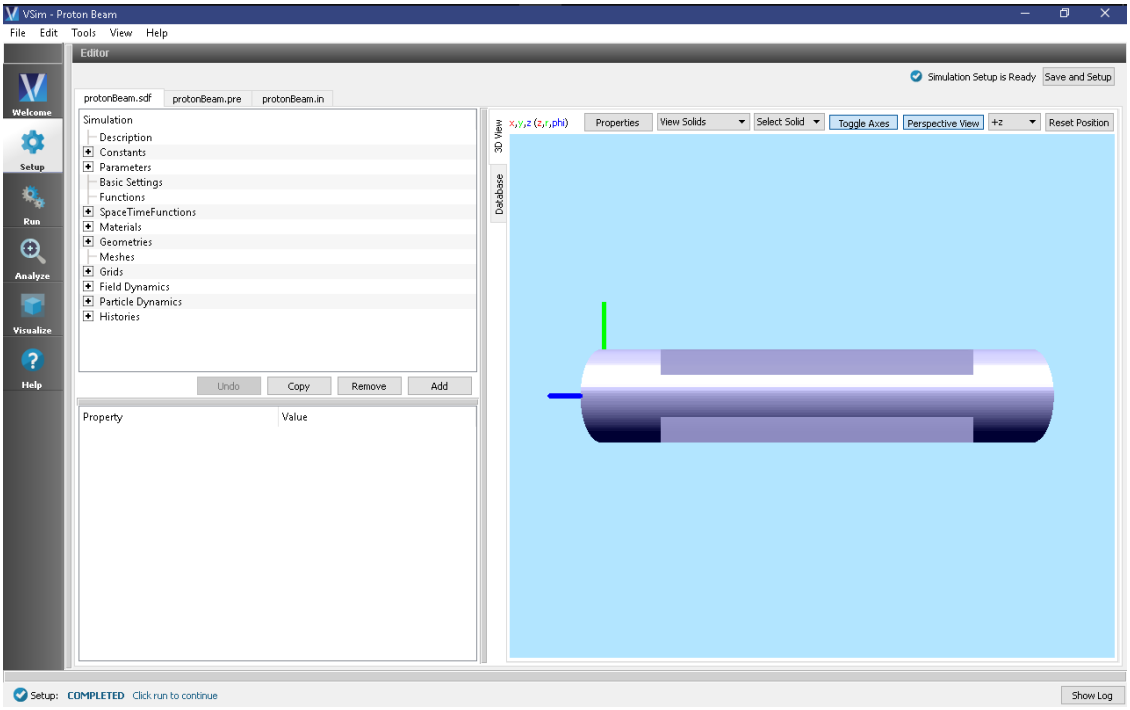


Fig. 6.43: Setup Window for the Proton Beam example.

Simulation Properties

Constants are set up to allow setting the proton beam energy and current, the background H2 pressure and temperature, and the cross-sectional size of the beam emission.

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.

- 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 the right pane. The run has completed when you see the output, "Engine completed successfully." A snapshot of the simulation run completion is shown in Fig. 6.44.

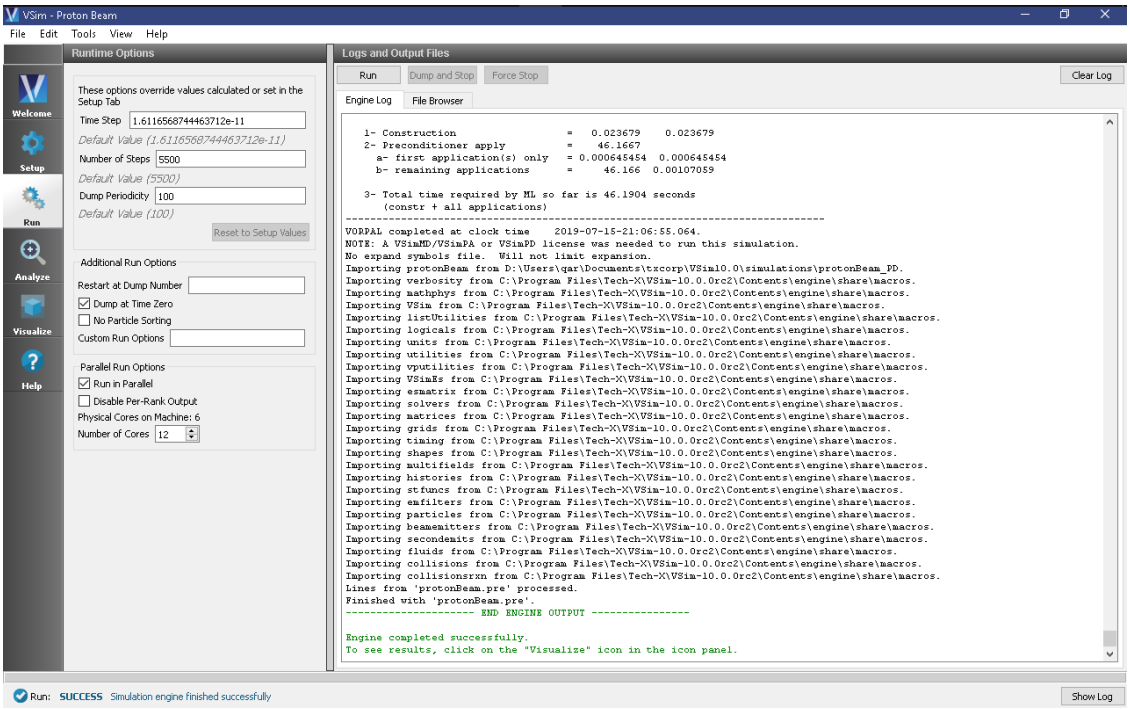


Fig. 6.44: The Run Window at the end of execution.

Analyzing the Results

The reactions do not occur at the beginning of the simulation because the protons must propagate to the neutral gas. This means that some species, such as the electrons, do not exist in the simulation until a later time. For the time slider to be consistent across species in the visualization, we must generate empty time slices for these particles for early in time. This can be quickly and easily accomplished in the Analyzers tab.

- First click on the *Analyze* Tab.
- From the *Available Anaylzers* list choose *createMissingPtclsDumps.py*. Then click *Open* at the bottom of the *Analysis Controls* pane.
- Ensure that the "simulationName" field is "protonBeam" and enter "Electrons" in the "speciesName" field.
- Press the *Analyze* button on in the upper right corner of the window to run the analysis. Below, the Analyze Tab is shown at the end of a successful run.
- Repeat this for the H2Plus and any other species that are generated (be default, only these two)

Visualizing the Results

We can now visualize all of the particles at a particular time slice. To do this:

- Proceed to the Visualize Window by pressing the *Visualize* button in the left column of buttons
- Expand *Particle Data* and select Electrons, H2Plus, and Hplus.

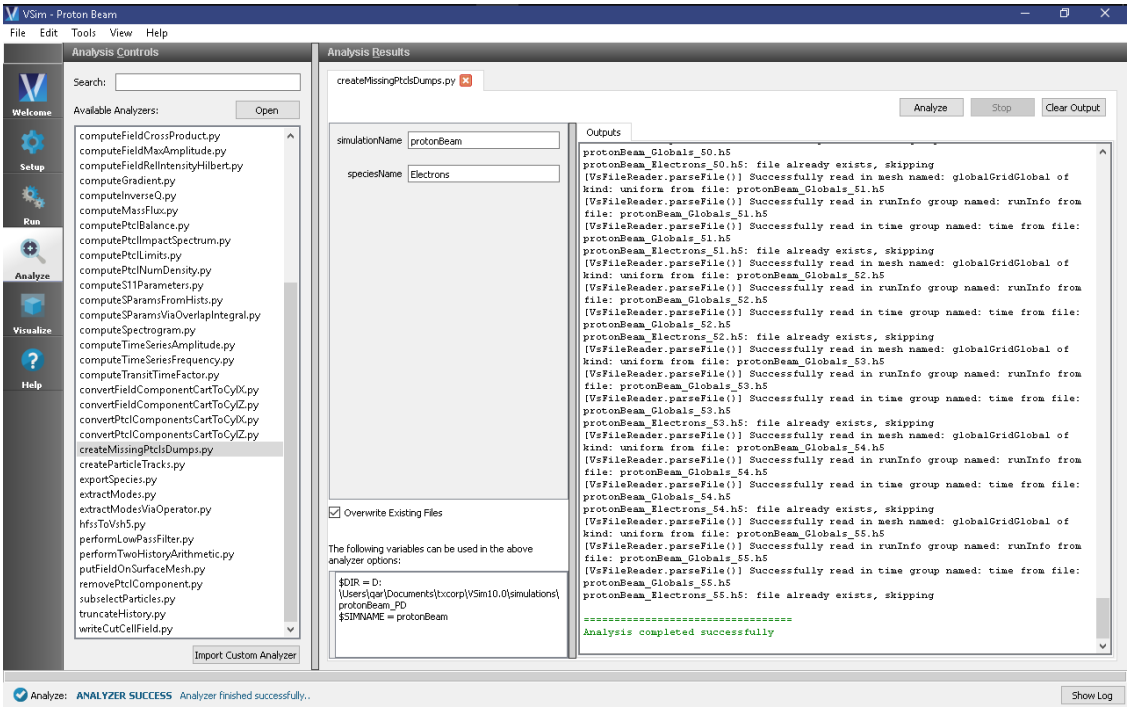


Fig. 6.45: The Analyze Window at the end of execution.

- Slide the time slider to advance the simulation in time (step 50 is shown in Fig. 6.46)

Next we can visualize the potential due to the particles and the electrodes:

- Unselect the particle data (Electrons, H2Plus, and Hplus).
- Expand *Scalar Data* and select *Phi*.
- Check the *Clip All Plots* box and scroll through the dumps.

The potential shown in Fig. 6.47 is the total potential, that is, the potential due to the static electrodes, the proton beam, and other charged species resulting from the reactions.

Further Experiments

Try changing the neutral gas pressure (which in turn will modify its density). At higher densities more reactions will occur and the proton beam will not be able to traverse the column intact. For lower densities, which are more in line with experiment, the proton beam will cause small amounts of ionization in the background gas, generating an electron cloud that is confined by the electrodes that can provide space-charge compensation for the beam. Lowering the beam energy will allow some lower energy reactions, such as H3+ formation, to occur.

6.5.4 Single Particle Circular Motion (singleParticleCircularMotion.sdf)

Keywords:

`single particle, circular motion, finite difference effects`

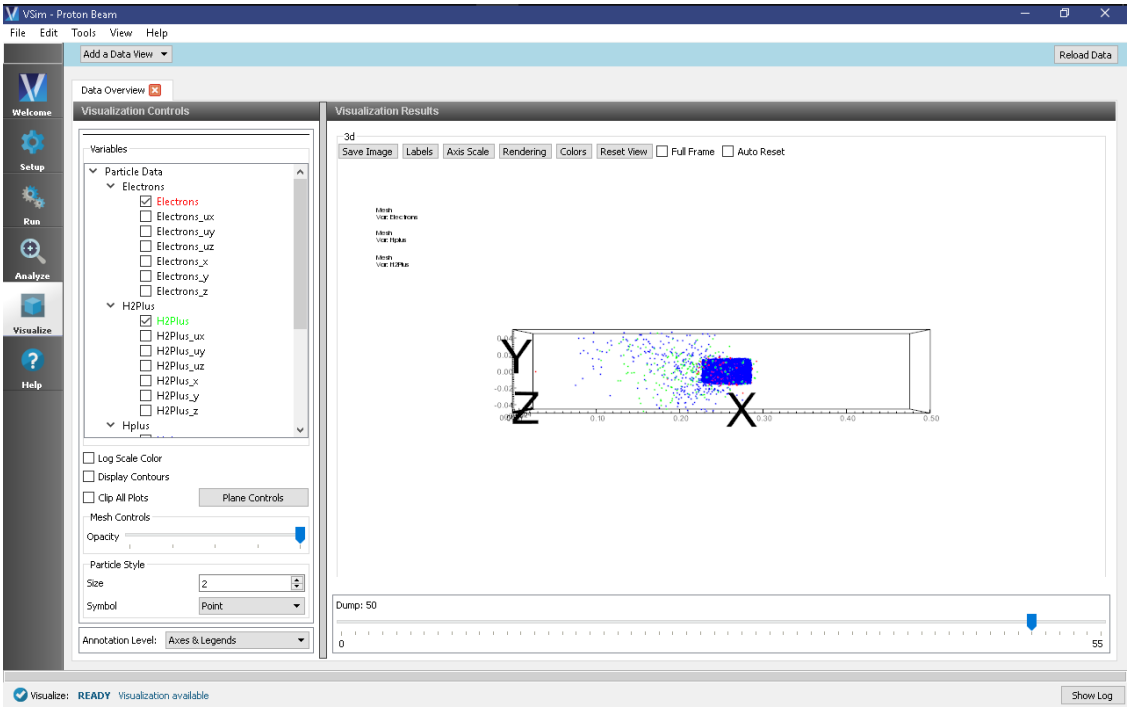


Fig. 6.46: Plot of all the particles at timestep 50. Notice that the electrons are confined by the magnetic field to the inner radius of the device. Some will also be confined by the electrodes to oscillate along the device.

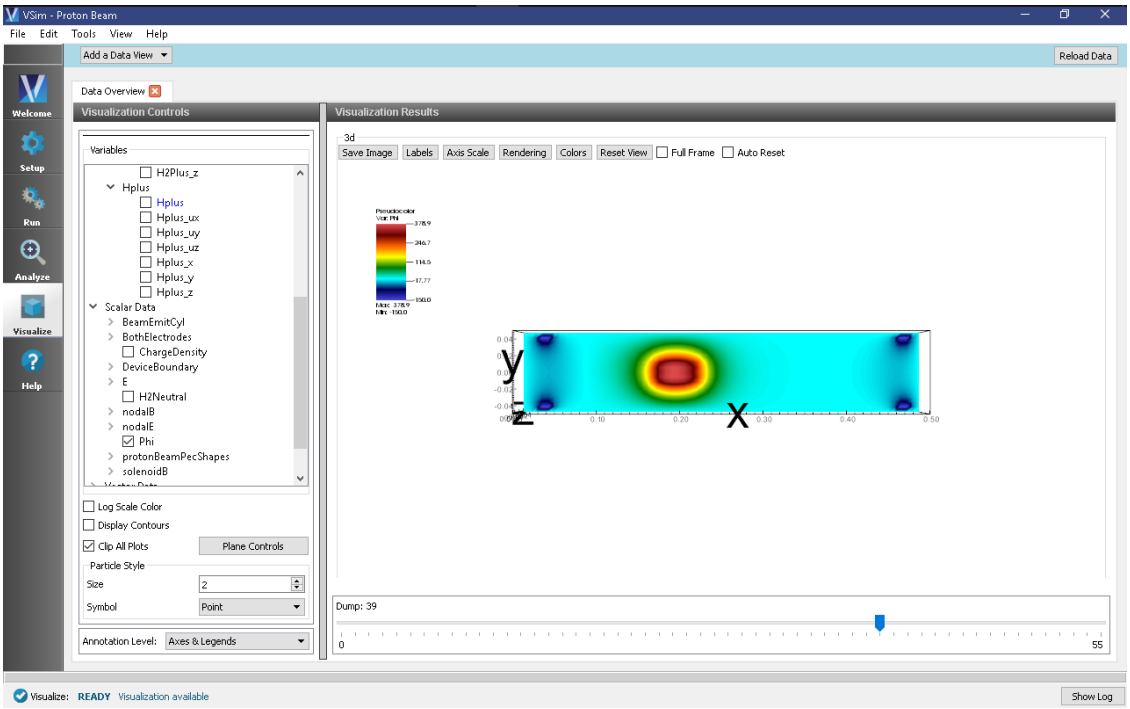


Fig. 6.47: The electrostatic potential

Problem Description

This example shows how to simulate the uniform circular motion of a single electron in a constant, uniform magnetic field in VSim. The electron is loaded inside a cylindrical capacitor with grounded walls to eliminate any stray electric fields. The electron is loaded far from the walls to reduce any effects from image charges. The magnetic field points down the positive z-axis.

Due to the finite difference algorithm utilized by VSim, two corrections must be made in order to get the electron to take a true circular trajectory. The first correction is to the cyclotron frequency. In the finite difference world of VSim, the electron does not move along a circular arc from time step to time step, instead it moves along a straight line. To correct for this we need to set our $\omega_{cyclo_{FD}} = \frac{2}{\Delta T} \arctan\left(\frac{\omega_{cyclo}\Delta T}{2}\right)$ [1] (see chapter 4 section 3).

The next correction is to account for the implementation of the Boris Method [1], the algorithm used in VSim to push particles. In the Boris Method, the position of the particle, $\vec{x}(t)$, is defined at full time steps, while the velocity, $\vec{v}(t)$, is defined at half time steps. This scheme of 'well-centered' derivatives means that VSim is automatically accurate to second order, but it means we have to be careful about our initial conditions for the electron's velocity. The initial velocity is set under Particle Dynamics → Kinetic Particle → electrons0 → particleLoader0 then velocity distribution. VSim will assume that this is the particle's velocity a **ONE HALF** time step before the start of the simulation, so we must load the particle with the velocity it would have a half time step before the start of the simulation.

This simulation can be performed with a VSimBase license.

Opening the Simulation

The Single Particle Circular Motion example is accessed from within VSimComposer by the following actions:

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim for Plasma Discharges* option.
- Expand the *Processes* option.
- Select *Single Particle Circular Motion* 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. 6.48. 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`.

Simulation Properties

The Single Particle Circular Motion example includes some constants for easy adjustment of simulation properties:

- B0: The magnitude of the magnetic field
- VOLTAGE_OUTER and VOLTAGE_INNER: sets the value of the radial electric field experienced by electron (default value for both is zero)

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.
- Here you can set run parameters. The default is to run in serial.

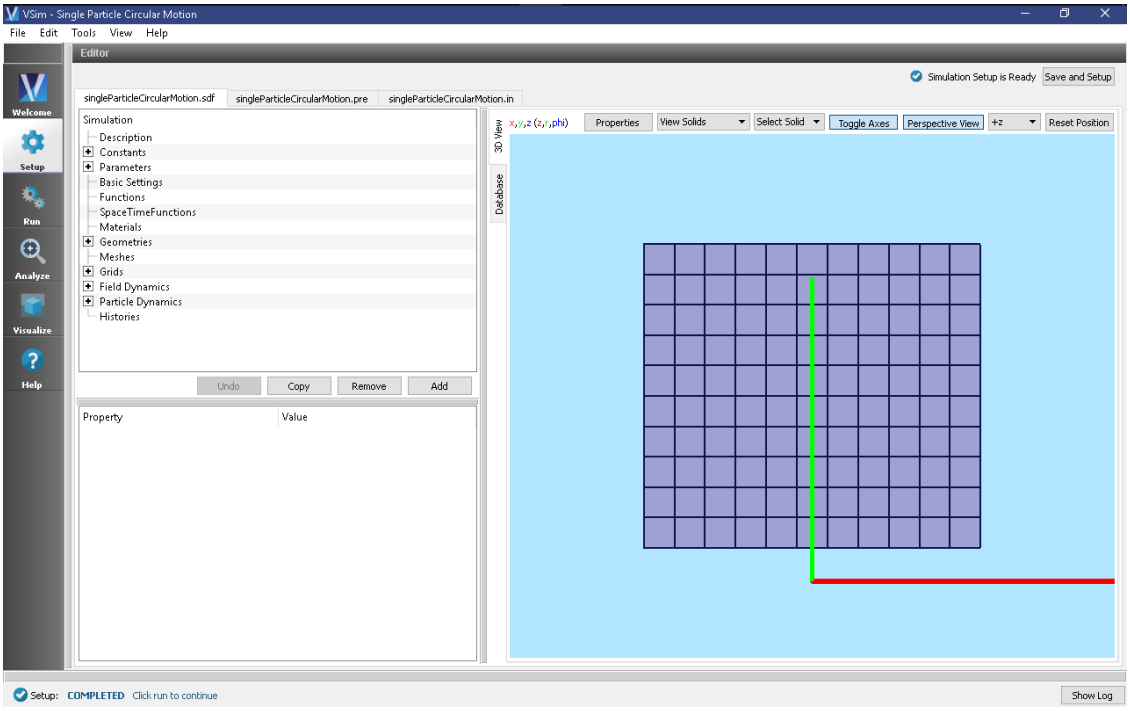


Fig. 6.48: Setup Window for the Single Particle Circular Motion example.

- 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." This is shown in Fig. 6.49

In serial, this simulation only takes seconds to run.

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.

- In the Visualize Window, expand 'Particle Data' then 'electrons0' and check the box next to the red 'electrons0.' This will plot our single electron.
- Expand 'Meshes' then 'globalGridGlobal' and check the box next to 'globalGridGlobal (ChargeDensity)' as shown in Fig. 6.50.
- Scroll through the dump slider (found below the plot), the electron will be stationary because the axial coordinate (phi) has been compressed. This means the electron remains at the same r and z coordinate (this is a 2D simulation).

Further Experiments

Simulations are correct only to some accuracy. The corrections we made to the cyclotron frequency and the initial velocity make this simulation correct to second order. By looking at the phase space plot, we can explore the second order accuracy of this simulation. Navigate to the Visualize Window, select 'Phase Space' from the 'Data View' drop down menu, and plot 'electrons0_r' vs 'electrons0_ur.'

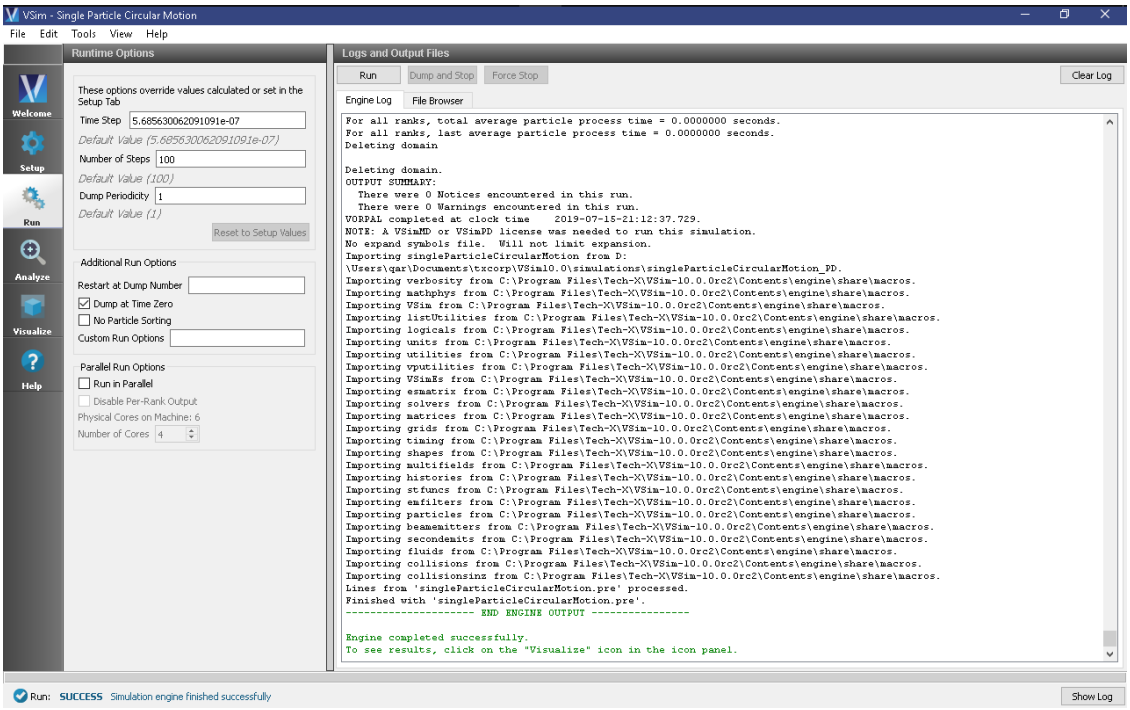


Fig. 6.49: The Run Window at the end of execution in serial.

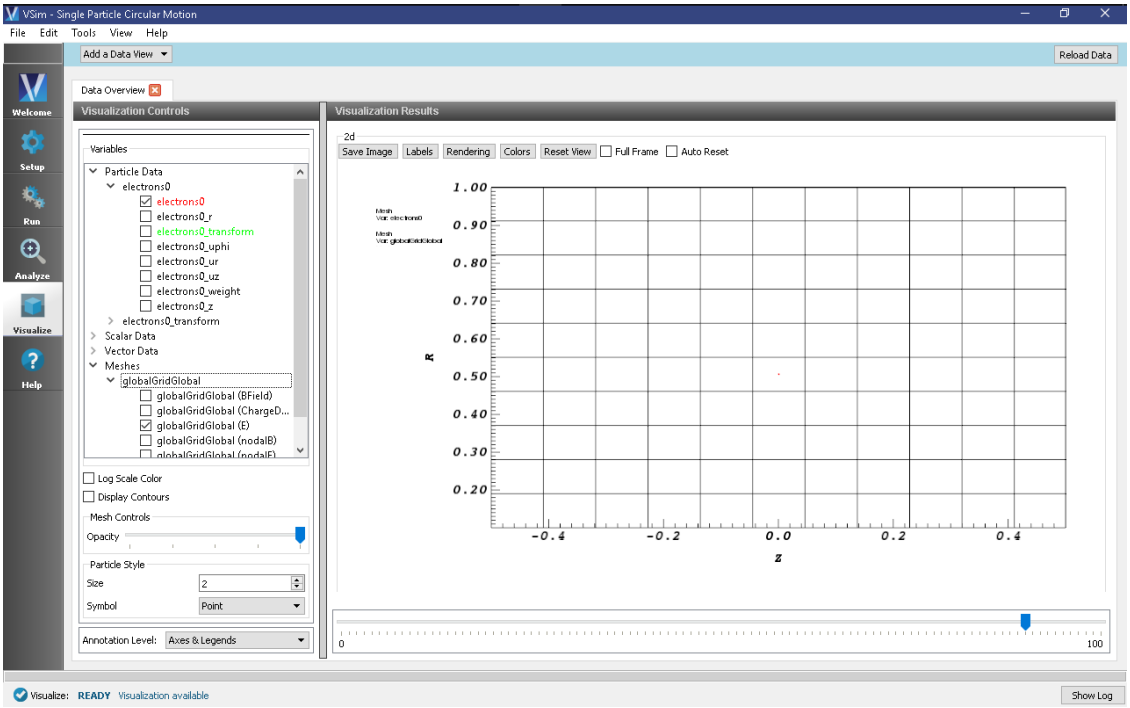


Fig. 6.50: Visualization of Single Particle Circular Motion at dump 100.

As you scroll through the dumps for the first time (with the 'Auto Reset' box UN-checked), the axes will adjust. The particle is taking an elliptical path in phase space. In a perfect simulation, the electron would remain at the same position in phase space with constant radius and zero radial velocity. Instead, the electron oscillates between the positions $r = 0.50500$ m and $r = 0.50542$ m for a $\Delta r = 0.00042$ m. Cut the time step in half and double the number of timesteps taken (so that the simulation runs through the same amount of time). Now look at the phase space plot again. By approximately what factor did Δr drop? Since the simulation is correct to second order, dropping the time step by a factor of 2 should drop the error by a factor of 4.

Other things you can play around with:

- Reset the electron speed, electron loading position, or the cyclotron frequency, OMEGA, back to the uncorrected versions and redo the error analysis described above.
- Change the values for VOLTAGE_INNER and VOLTAGE_OUTER to see the effects of a radial electric field on the single electron.

References

[1] Birdsall, C. K., & Langdon, A. B. (1985). Plasma Physics via Computer Simulation. New York: McGraw-Hill.

6.5.5 Townsend Avalanche (townsend.sdf)

Keywords:

background gas, particle emission, ionization, inelastic anisotropic scattering

Problem Description

In a Townsend avalanche, electrons are accelerated by an electric field and ionize a background gas of neutral atoms or molecules. Each ionization event creates an additional electron that will also be accelerated and eventually produce more ionization events of its own. This process of repeated doubling results in an exponential increase in the number of electrons. The Townsend avalanche occurs when the electron and ion densities are too low to behave collectively and form a plasma.

This type of discharge is named after John Townsend who first proposed the ionization model to explain the phenomena. In his experiments, Townsend measured the current across a gas-filled chamber with a of pair of parallel plates on the two ends. Townsend illuminated the cathode plate with X-rays which produced electrons via the photo-electric effect. Townsend noticed that the current between the plates depended on the strength of the electric field between the plats and the pressure of the gas.

His observations lead to the conclusion that electrons were ionizing the gas and causing an exponential increase in the measured current. The current measured across the discharge chamber is described by the following formula:

$$\frac{I(x)}{I_0} = \frac{e^{\alpha(x-x_0)}}{1-\gamma(e^{\alpha(x-x_0)}-1)}$$

where I_0 is the photo-current produced by the X-rays, $I(x)$ is the current through the chamber as a function of the plate separation, x. The parameter x_0 is a characteristic distance that a collection of electrons has to travel away from the cathode before an equilibrium is reached. The parameters α and γ are the first and second Townsend coefficients, respectively. The first Townsend coefficient, α is a measure of how many ionization events a single electron will produce per unit length, and γ is a parameter accounting for electron generation from secondary processes, like ion impact at the cathode.

In this example simulation, we will loosely follow the experimental setup of L.M. Chanin and G.D. Rork who measured the first Townsend coefficient for Helium *[CR64a]*, Neon, and Hydrogen *[CR64b]* in the 1960s. For low voltage

discharges, the factor γ is negligible and the equation above reduces to

$$I(x) = I_0 e^{\alpha(x-x_0)}.$$

In reality, α is a complicated function of pressure, accelerating field, and the species of background gas molecule/atom. Chapter 14 section 3 of *Principles of Plasma Discharges and Materials Processing*, *[LL05]* presents an analytical model for α as a function of pressure and electric field and provides the fitting constants for different gases. This simulation will estimate a value for α for a Helium gas keeping the electric field and pressure constant at $E = 2.12e5$ V/m and $p = 21.2$ torr. At this pressure and electric field, the accepted value for α is 1.3 *[CR64a]*, *[LL05]*.

This simulation can be run with a VSimPD license.

Opening the Simulation

The Townsend Avalanche example is accessed from within VSimComposer by the following actions:

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim for Plasma Discharges* option.
- Expand the *Processes* option.
- Select *Townsend Avalanche* 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. 6.51.

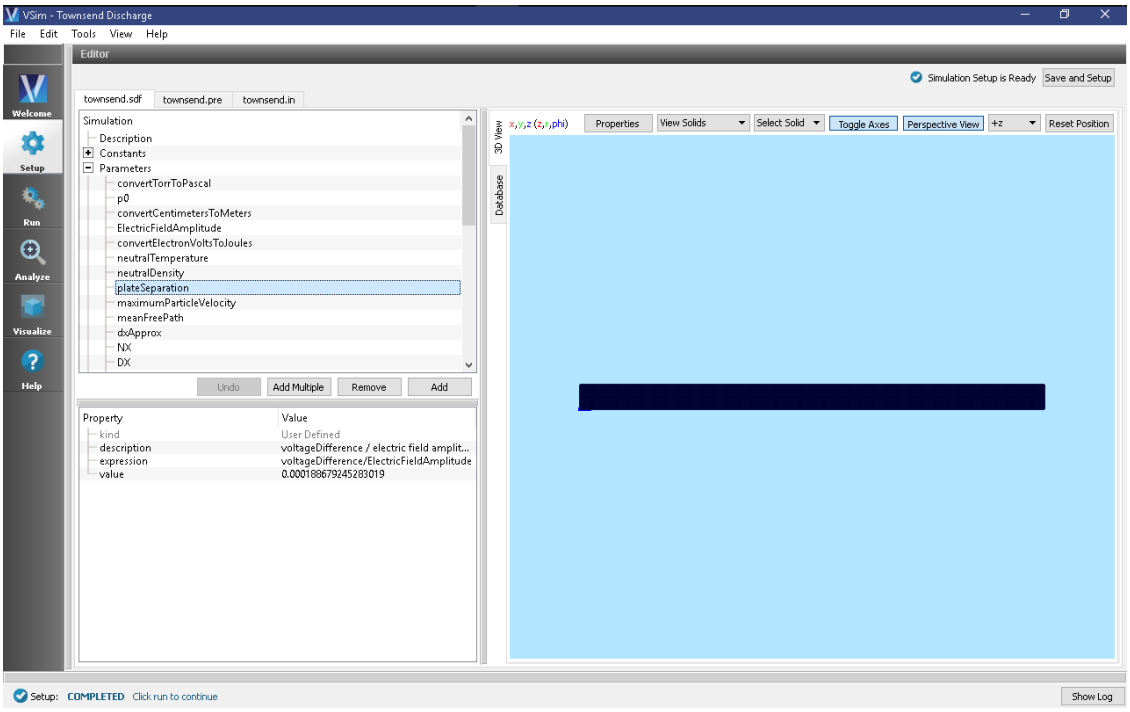


Fig. 6.51: Setup Window for the Townsend Avalanche example.

Simulation Properties

In order to calculate a value for the Townsend coefficient, this simulation will need to be run multiple times at different plate separations while keeping the electric field and pressure constant. The data points from these multiple runs will be used in a least squares fit to an exponential function of the form

$$F(x) = Ae^{\alpha(x-\beta)}$$

where α and β are the constants being fitted, with α being the Townsend coefficient of interest. Details on how to fit the data yourself is covered in the section **Computing the Townsend Coefficient** below.

The simulation input file is set up to allow for rapid iteration on the multiple plate separations. The *voltageDifference* actually controls the separation of the two plates in such a way that keeps the electric field constant, and also is used to calculate an estimate for the maximum electron speed which goes into determining the time step.

In **Basic Settings** the field solver is set to *prescribed fields*. This means we manually set the electric field as a function of space (and have an option on add time dependence). In the **Field Dynamics** element, under `Fields` is where we actually set the value for the constant electric field present between the two plates at either end of the discharge tube. In the `externalElectricField` element we set `component0`, the x-component of the electric field to a value calculated from the parameters in *[CR64a]*.

The **Particle Dynamics** element is where we set up the which particle species the simulation will contain, how particles of each species get added and removed from the simulation, and in what processes/interactions they participate.

This simulation contains an electron species, a positive helium ion species, and a background gas of helium neutral gas. The `settableFluxSlabElectronEmitterCW0` is set up to emit a total of 100,000 physical electrons. Both the electron and helium ion species have been set up such that each macro particle corresponds to a single physical particle.

The electric field points from upper x (right side of screen) to lower x (left side of screen). The electrons are emitted into the simulation from the lower x and are accelerated to the right. There are seven interactions in which an electron can be involved as it crosses the simulation grid. These processes are set in the `Reactions` element, which has been enabled by choosing *include particles* and *reactions* in **Basic Settings.**

The processes included in this simulation are electron/neutral helium elastic scattering, five different electron/neutral helium excitation processes in which electrons lose some energy to putting a neutral helium atom into an excited state, and an electron/neutral helium ionization process which is the crucial bit of physics for this simulation. We found that the it was important to use the reaction type `Inelastic Electron Scattering` with `scatter type` set to *VahediSurendra* for getting an accurate value for the Townsend coefficient.

To add collision processes to VSim, cross-sections data must be supplied by the user. To get the most accurate results from this validation study, we used cross sections from the LxCat database. According to the terms of the database, we are not permitted to distribute data obtained from the database, so the data included as part of this example is less accurate. The following section provides a set of instructions for obtaining the more accurate cross-section data for yourself.

Downloading Cross-Sections from LxCat Database

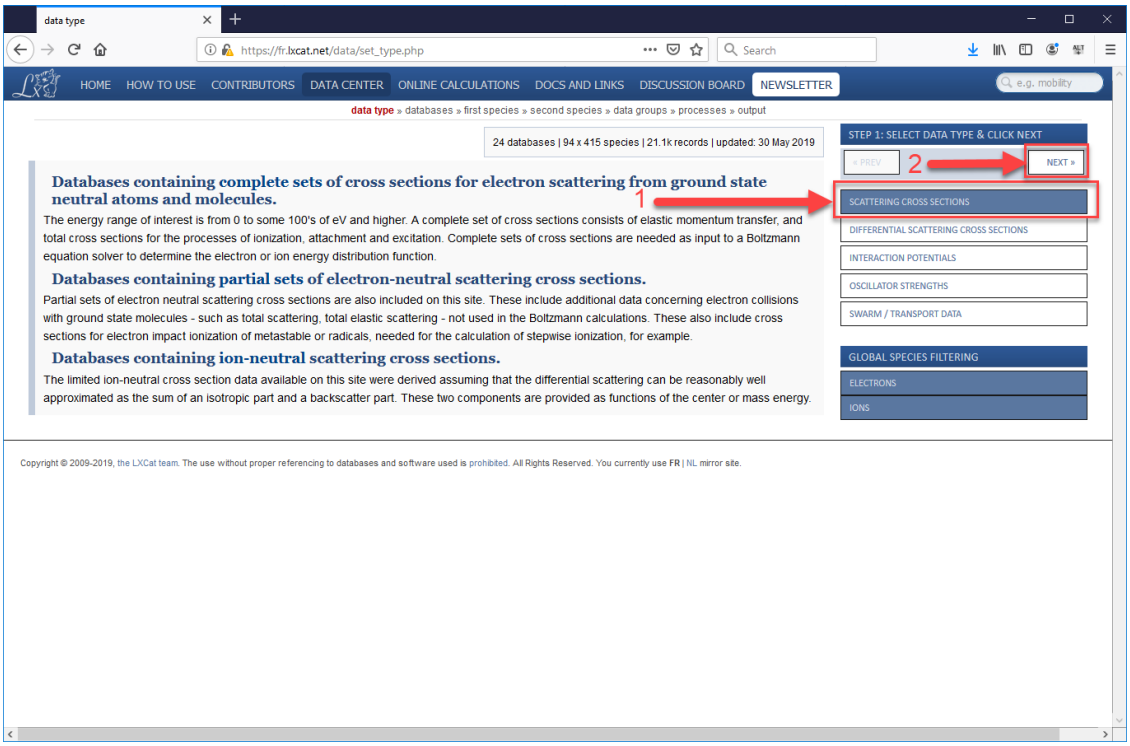
In our validation study, the cross-sections used in the elastic, excitation, and ionization interactions between the electrons and the background neutral Helium gas were taken from the LxCat Database.

The terms of use of the LxCat database do not allow Tech-X to distribute the cross sections we used in our study which calculated a Townsend coefficient within 10% of accepted value of 1.31. Instead, less accurate estimates of the cross-sections are provided so that the simulation can will run.

To obtain a more accurate set of cross-sections from the LxCat Database, follow these instructions:

1. Open a Browser and follow this (https://fr.lxcat.net/data/set_type.php) link to the database.

2. From the menu bar on the left, select “Scattering Cross Sections” and press “Next.”



3. On the next page, first press the “minus” button to de-select all options, then select the “Biagi (transcription of data from SF Biagi’s Fortan code, Magboltz.)” Alternatively, you could choose another sub-database from which to obtain cross-sections, the data from all the sources agree fairly well, but we are taking you to the specific set we used in our validation study. After the sub-database is selected, press “Next”.

4. Since we are using the Biagi data, which are all processes which involve electrons, we skip over the choice of first species (since it will be an electron) and go straight to choosing the second species. Choose “He” (or your species of choice), then “Next”.

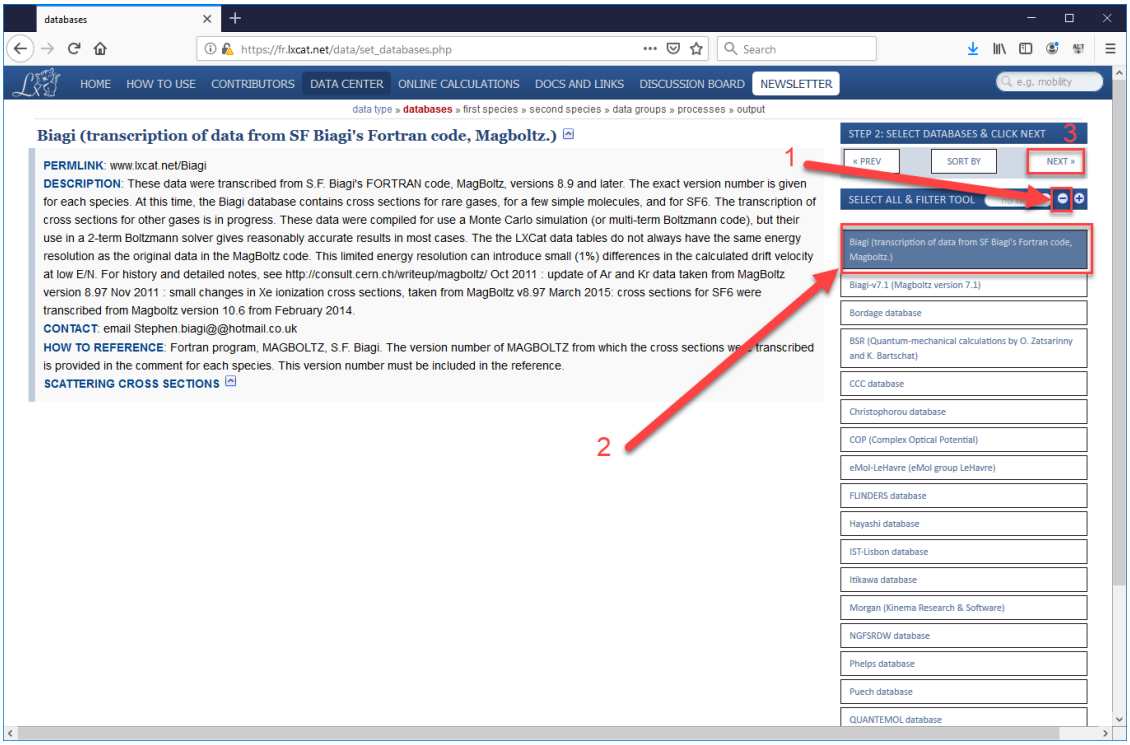
5. We will be using Elastic Scattering, Excitation, and Ionization processes for this simulation, so select all three options (“Elastic”, “Excitation”, and “Ionization”) the press “Next”.

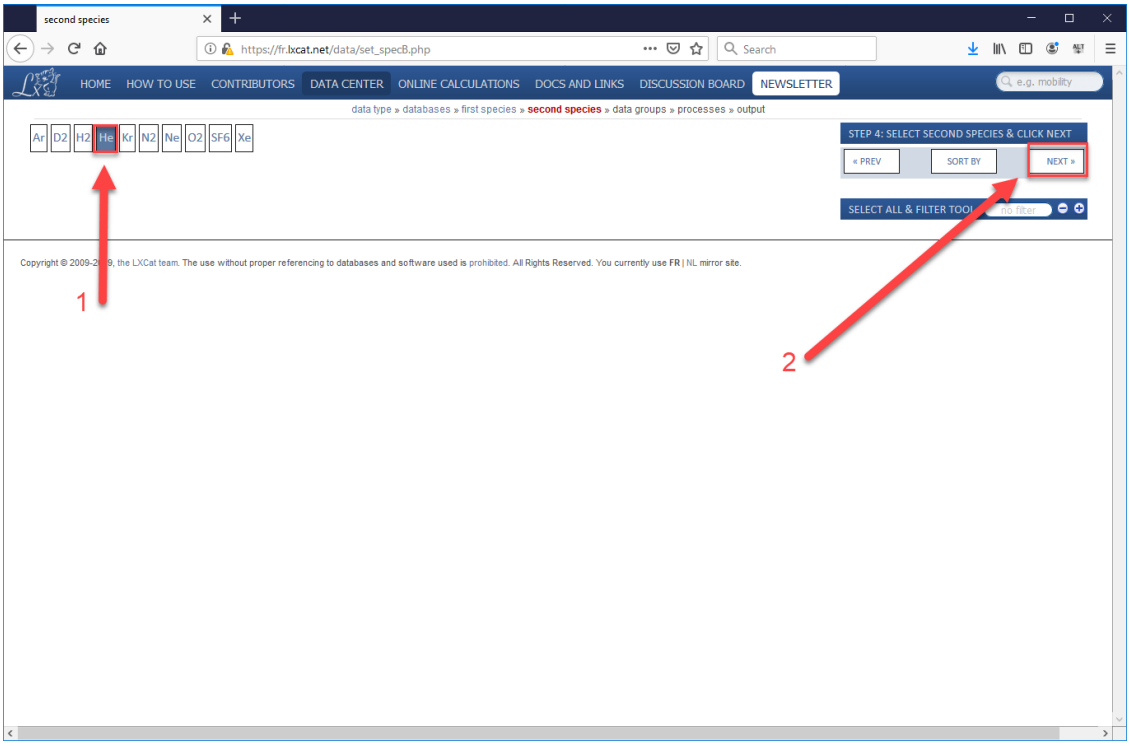
6. One the next page that opens will be a long list of electron-helium processes, the majority of which will be excitation processes which involve an electron passing energy to a neutral helium atom, sending it into an excited state, which will eventually cause the helium atom to emit a photon (which will not be tracked in VSim) when the atom returns to its ground state. In our validation study, we used the cross-sections for the elastic scattering process, five of the most common excitation processes, and the ionization process. Be sure to check the boxes for all 7 processes, scrolling down to the very bottom to find ionization (as shown in images Fig. 6.5.5 and Fig. 6.5.5). Then scroll back up and press “Next”.

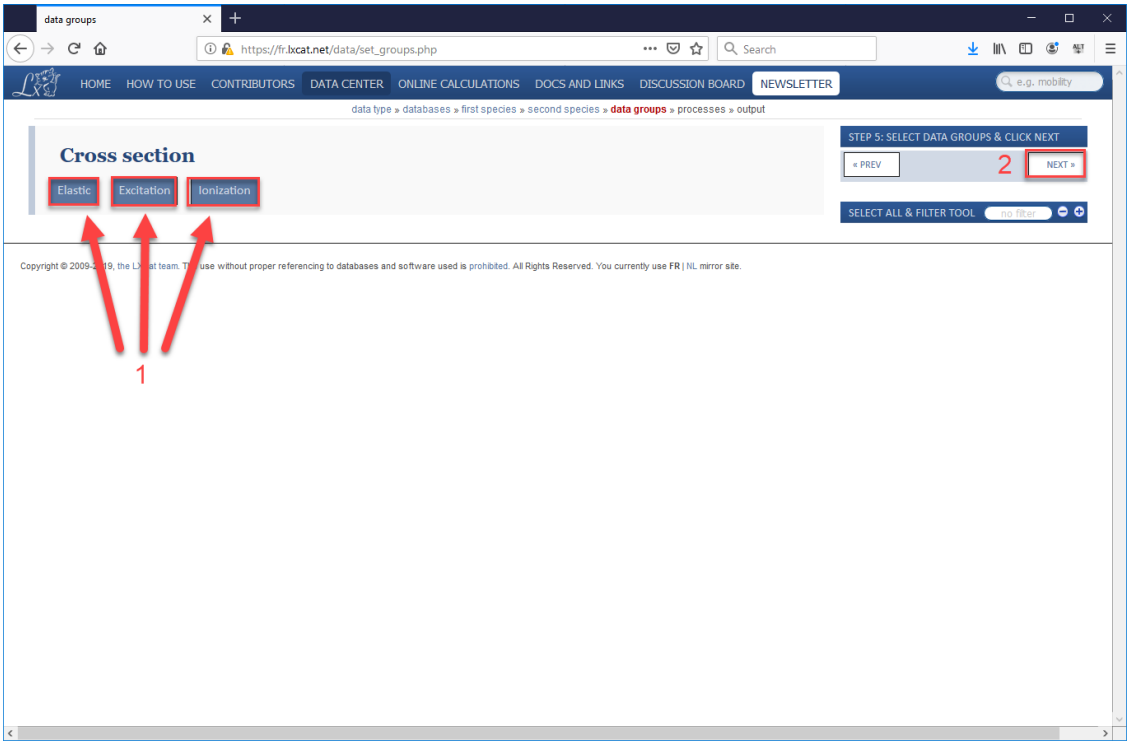
7. This will take you to the Terms of Use page for the LxCat database. All the data in this database represents someone’s original research, and therefore must be appropriately cited when used in further research. Please read this page (it is very short), then press “Yes, I have read and understood >>” to proceed.

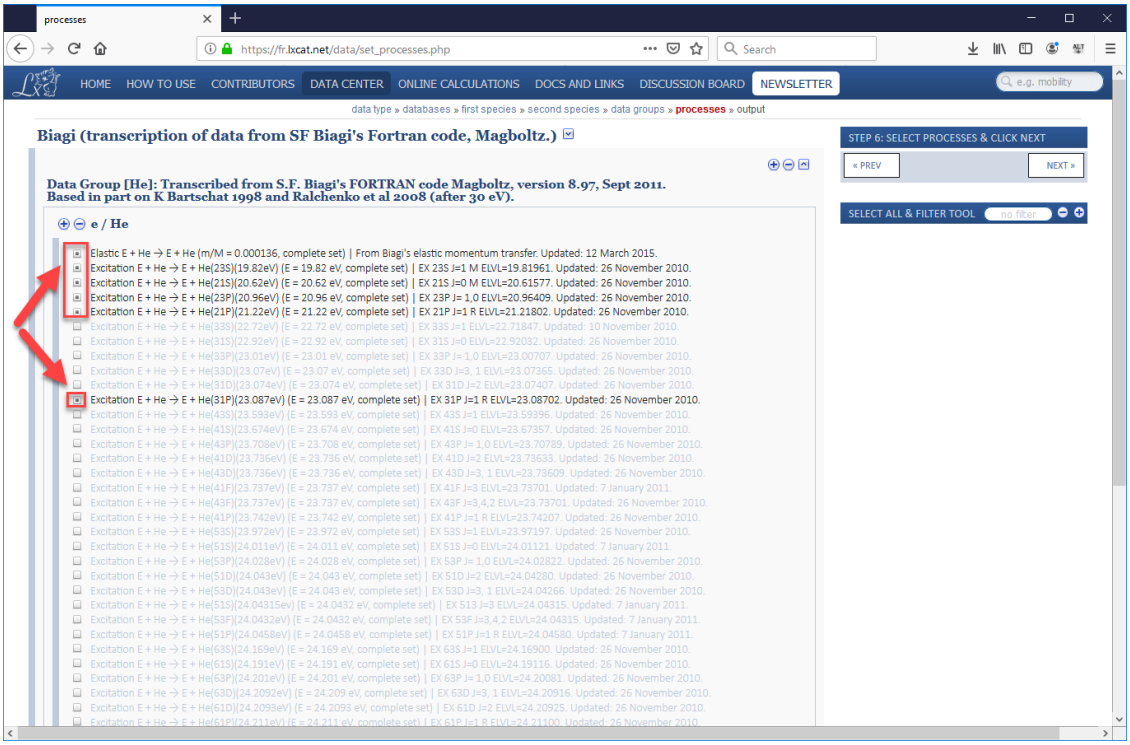
8. After agreeing to the terms of use, you will once again be shown the list of cross-sections from which you selected the 7 processes in **Step 6**. Just press “Next” one final time to retrieve the data.

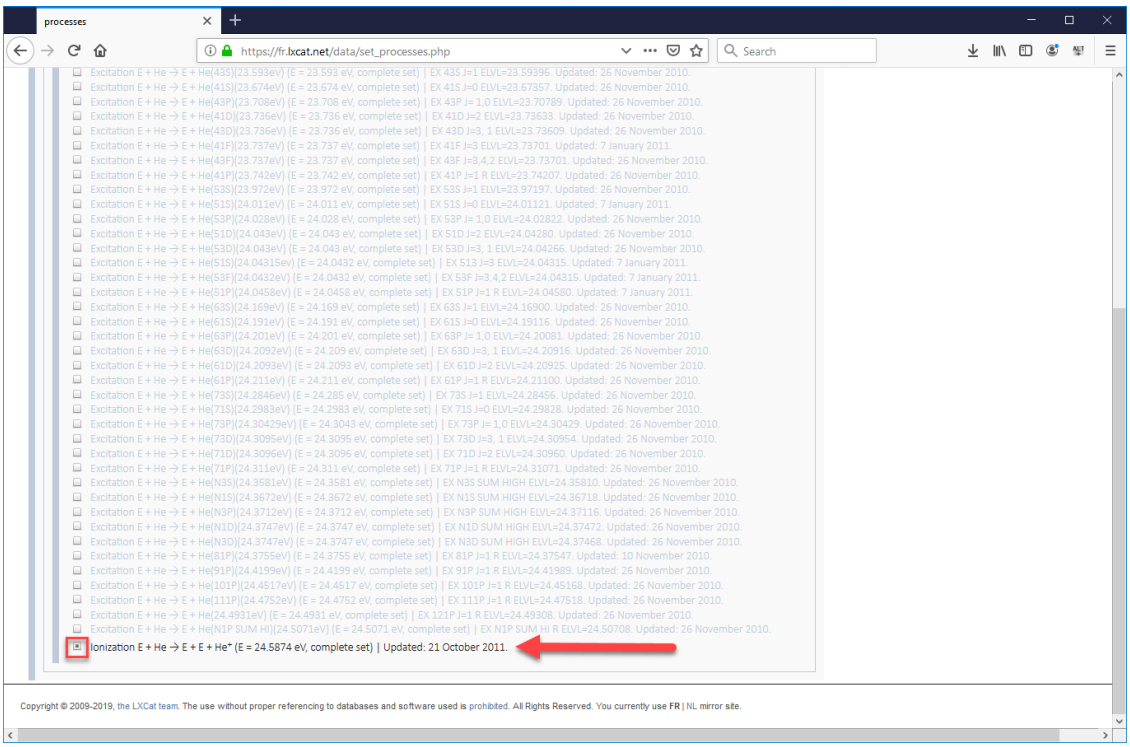
9. On the next page there will be a plot showing the cross-section data you selected in **Step 6**. To use cross-section data in VSim, we need the data in a two-column format with no header. The two column need to have the CM energy (in eV) in the first column, and the cross section (in square meters) in the second column. To retrieve the data in this form, press the “TXT” text in the upper right corner (see Fig. 6.5.5).

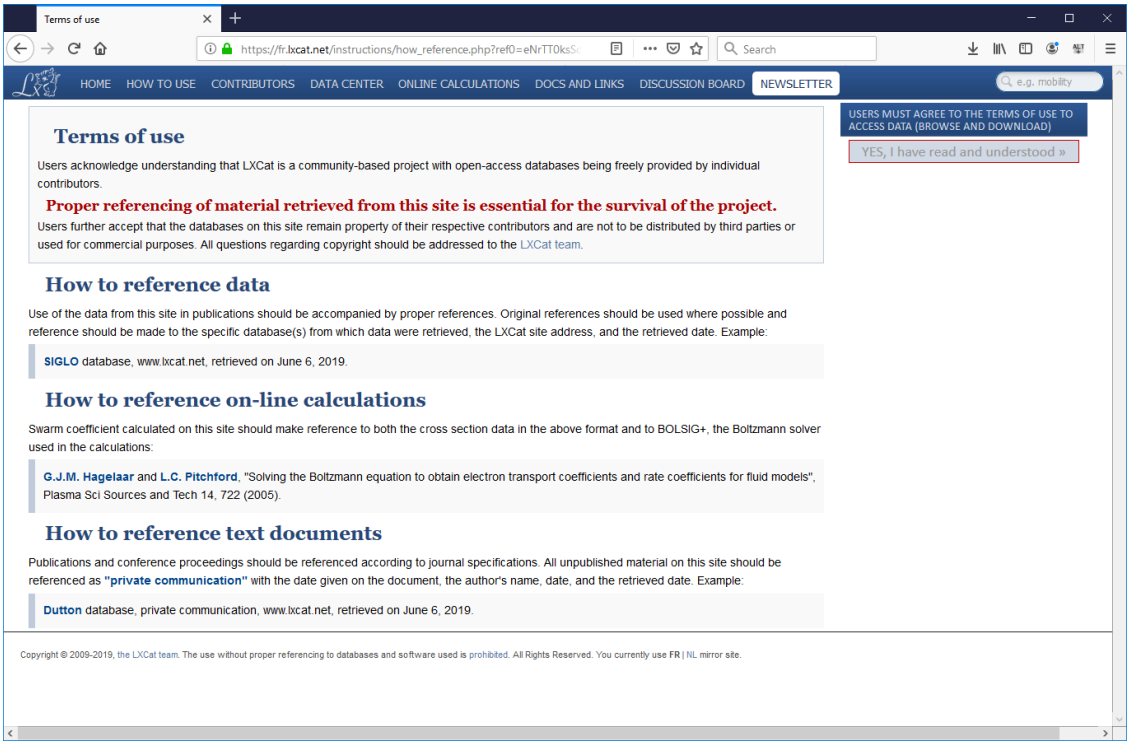

databases
https://fr.lxcat.net/data/set_databases.php
HOME
HOW TO USE
CONTRIBUTORS
DATA CENTER
ONLINE CALCULATIONS
DOCS AND LINKS
DISCUSSION BOARD
NEWSLETTER
data type » databases » first species » second species » data groups » processes » output
Biagi (transcription of data from SF Biagi's Fortran code, Magboltz.)
PERMLINK: www.lxcat.net/Biagi
DESCRIPTION: These data were transcribed from S.F. Biagi's FORTRAN code, MagBoltz, versions 8.9 and later. The exact version number is given for each species. At this time, the Biagi database contains cross sections for rare gases, for a few simple molecules, and for SF6. The transcription of cross sections for other gases is in progress. These data were compiled for use a Monte Carlo simulation (or multi-term Boltzmann code), but their use in a 2-term Boltzmann solver gives reasonably accurate results in most cases. The the LXCat data tables do not always have the same energy resolution as the original data in the MagBoltz code. This limited energy resolution can introduce small (1%) differences in the calculated drift velocity at low E/N. For history and detailed notes, see http://consult.cern.ch/writeup/magboltz/ Oct 2011 : update of Ar and Kr data taken from MagBoltz version 8.97 Nov 2011 : small changes in Xe ionization cross sections, taken from MagBoltz v8.97 March 2015: cross sections for SF6 were transcribed from Magboltz version 10.6 from February 2014.
CONTACT: email Stephen.biagi@@hotmail.co.uk
HOW TO REFERENCE: Fortran program, MAGBOLTZ, S.F. Biagi. The version number of MAGBOLTZ from which the cross sections were transcribed is provided in the comment for each species. This version number must be included in the reference.
SCATTERING CROSS SECTIONS
STEP 2: SELECT DATABASES & CLICK NEXT
« PREV
SORT BY
NEXT »
SELECT ALL & FILTER TOOL
Biagi (transcription of data from SF Biagi's Fortran code, Magboltz.)
Biagi-v7.1 (Magboltz version 7.1)
Bordage database
BSR (Quantum-mechanical calculations by O. Zatsarinny and K. Bartschat)
CCC database
Christophorou database
COP (Complex Optical Potential)
eMol-LeHavre (eMol group LeHavre)
FLINDERS database
Hayashi database
IST-Lisbon database
Itikawa database
Morgan (Kinema Research & Software)
NGFSRDW database
Phelps database
Puech database
QUANTEMOL database
1
2
3


second species
https://fr.lxcat.net/data/set_specB.php
HOME
HOW TO USE
CONTRIBUTORS
DATA CENTER
ONLINE CALCULATIONS
DOCS AND LINKS
DISCUSSION BOARD
NEWSLETTER
data type » databases » first species » second species » data groups » processes » output
Ar
D2
H2
He
Kr
N2
Ne
O2
SF6
Xe
STEP 4: SELECT SECOND SPECIES & CLICK NEXT
« PREV
SORT BY
NEXT »
SELECT ALL & FILTER TOOL
Copyright © 2009-2019, the LXCat team. The use without proper referencing to databases and software used is prohibited. All Rights Reserved. You currently use FR | NL mirror site.
1
2


data groups
https://fr.lxcat.net/data/set_groups.php
Search
HOME
HOW TO USE
CONTRIBUTORS
DATA CENTER
ONLINE CALCULATIONS
DOCS AND LINKS
DISCUSSION BOARD
NEWSLETTER
e.g. mobility
data type » databases » first species » second species » data groups » processes » output
Cross section
Elastic
Excitation
Ionization
STEP 5: SELECT DATA GROUPS & CLICK NEXT
« PREV
NEXT »
2
SELECT ALL & FILTER TOOL
no filter
1


processes
https://fr.lxcat.net/data/set_processes.php
Search
HOME
HOW TO USE
CONTRIBUTORS
DATA CENTER
ONLINE CALCULATIONS
DOCS AND LINKS
DISCUSSION BOARD
NEWSLETTER
e.g. mobility
data type » databases » first species » second species » data groups » processes » output
Biagi (transcription of data from SF Biagi's Fortran code, Magboltz.)
STEP 6: SELECT PROCESSES & CLICK NEXT
« PREV
NEXT »
SELECT ALL & FILTER TOOL
no filter
Data Group [He]: Transcribed from S.F. Biagi's FORTRAN code Magboltz, version 8.97, Sept 2011.
Based in part on K Bartschat 1998 and Ralchenko et al 2008 (after 30 eV).
e / He
Elastic E + He → E + He (m/M = 0.000136, complete set) | From Biagi's elastic momentum transfer. Updated: 12 March 2015.
Excitation E + He → E + He(23S)(19.82eV) (E = 19.82 eV, complete set) | EX 23S J=1 M ELVL=19.81961. Updated: 26 November 2010.
Excitation E + He → E + He(21S)(20.62eV) (E = 20.62 eV, complete set) | EX 21S J=0 M ELVL=20.61577. Updated: 26 November 2010.
Excitation E + He → E + He(23P)(20.96eV) (E = 20.96 eV, complete set) | EX 23P J= 1,0 ELVL=20.96409. Updated: 26 November 2010.
Excitation E + He → E + He(21P)(21.22eV) (E = 21.22 eV, complete set) | EX 21P J=1 R ELVL=21.21802. Updated: 26 November 2010.
Excitation E + He → E + He(31P)(23.087eV) (E = 23.087 eV, complete set) | EX 31P J=1 R ELVL=23.08702. Updated: 26 November 2010.


processes
https://fr.lxcat.net/data/set_processes.php
Search
Ionization E + He → E + E + He+ (E = 24.5874 eV, complete set) | Updated: 21 October 2011.
Copyright © 2009-2019, the LXCat team. The use without proper referencing to databases and software used is prohibited. All Rights Reserved. You currently use FR | NL mirror site.


Terms of use
https://fr.lxcat.net/instructions/how_reference.php?ref0=eNrTT0ksS
Search
HOME
HOW TO USE
CONTRIBUTORS
DATA CENTER
ONLINE CALCULATIONS
DOCS AND LINKS
DISCUSSION BOARD
NEWSLETTER
e.g. mobility
Terms of use
Users acknowledge understanding that LXCat is a community-based project with open-access databases being freely provided by individual contributors.
Proper referencing of material retrieved from this site is essential for the survival of the project.
Users further accept that the databases on this site remain property of their respective contributors and are not to be distributed by third parties or used for commercial purposes. All questions regarding copyright should be addressed to the LXCat team.
USERS MUST AGREE TO THE TERMS OF USE TO ACCESS DATA (BROWSE AND DOWNLOAD)
YES, I have read and understood »
How to reference data
Use of the data from this site in publications should be accompanied by proper references. Original references should be used where possible and reference should be made to the specific database(s) from which data were retrieved, the LXCat site address, and the retrieved date. Example:
SIGLO database, www.lxcat.net, retrieved on June 6, 2019.
How to reference on-line calculations
Swarm coefficient calculated on this site should make reference to both the cross section data in the above format and to BOLSIG+, the Boltzmann solver used in the calculations:
G.J.M. Hagelaar and L.C. Pitchford, "Solving the Boltzmann equation to obtain electron transport coefficients and rate coefficients for fluid models", Plasma Sci Sources and Tech 14, 722 (2005).
How to reference text documents
Publications and conference proceedings should be referenced according to journal specifications. All unpublished material on this site should be referenced as "private communication" with the date given on the document, the author's name, date, and the retrieved date. Example:
Dutton database, private communication, www.lxcat.net, retrieved on June 6, 2019.
Copyright © 2009-2019, the LXCat team. The use without proper referencing to databases and software used is prohibited. All Rights Reserved. You currently use FR | NL mirror site.

10. A new browser tab will open up. Scroll down past the headers to find the data. Notice that in the headers for each of the data sets, the units of each of the columns are given. This data happens to be in the correct units. *Not all the cross-section data on the LxCat site is in eV and square meters.*

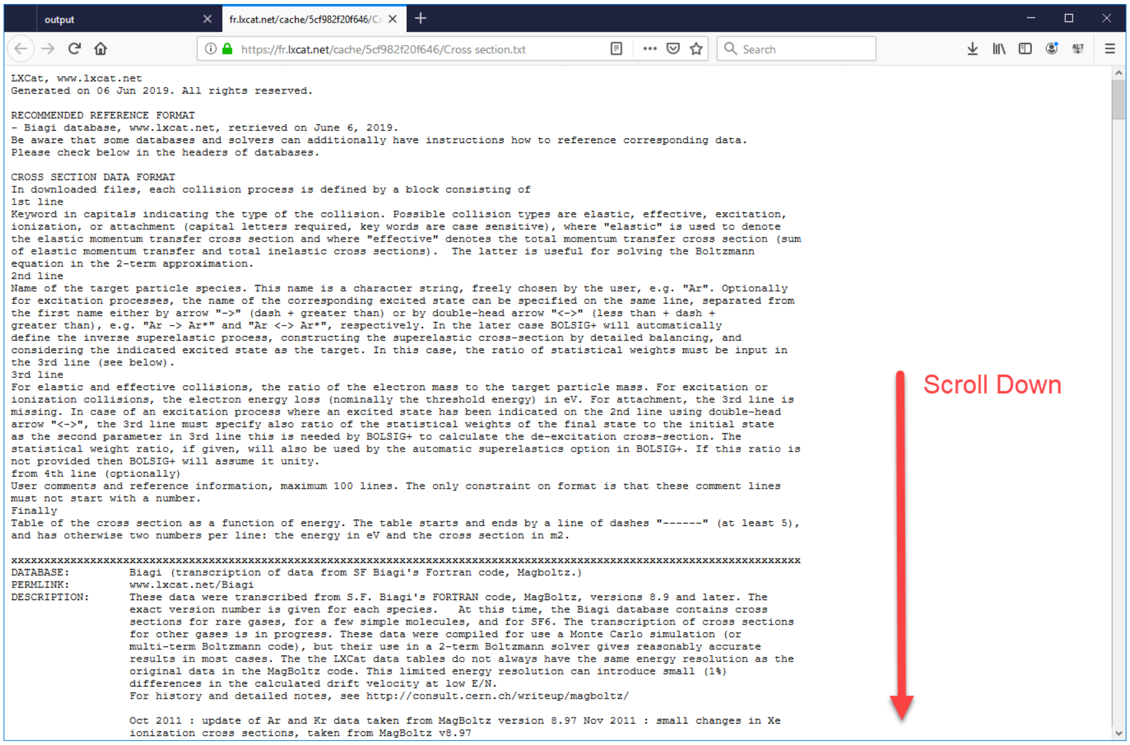
11. Highlight the data for one set, and copy it into a .dat file. Be sure to only select the data and no headers or other characters. The best way to create a .dat file is to copy the data into a .txt file, then change the .txt extension with .dat. This can be done in a Windows file browser.

12. Each of the seven data sets will need to be copied into their own .dat file, with the appropriate name. The name will need to match the string in the `cross-section data file` field of the particle reaction that uses the data set. This is how to import cross-section into collision processes that use the `Reactions` framework. For use with older frameworks like the Monte Carlo or Impact Collider frameworks, consult the appropriate documentation.

Running the Simulation

To run the simulation:

- Proceed to the run window by pressing the Run button in the left column of buttons.
- Here you can set run parameters, including how many cores to run with (under the MPI tab).
- When you are finished setting run parameters, 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." This is shown in Fig. 6.52.



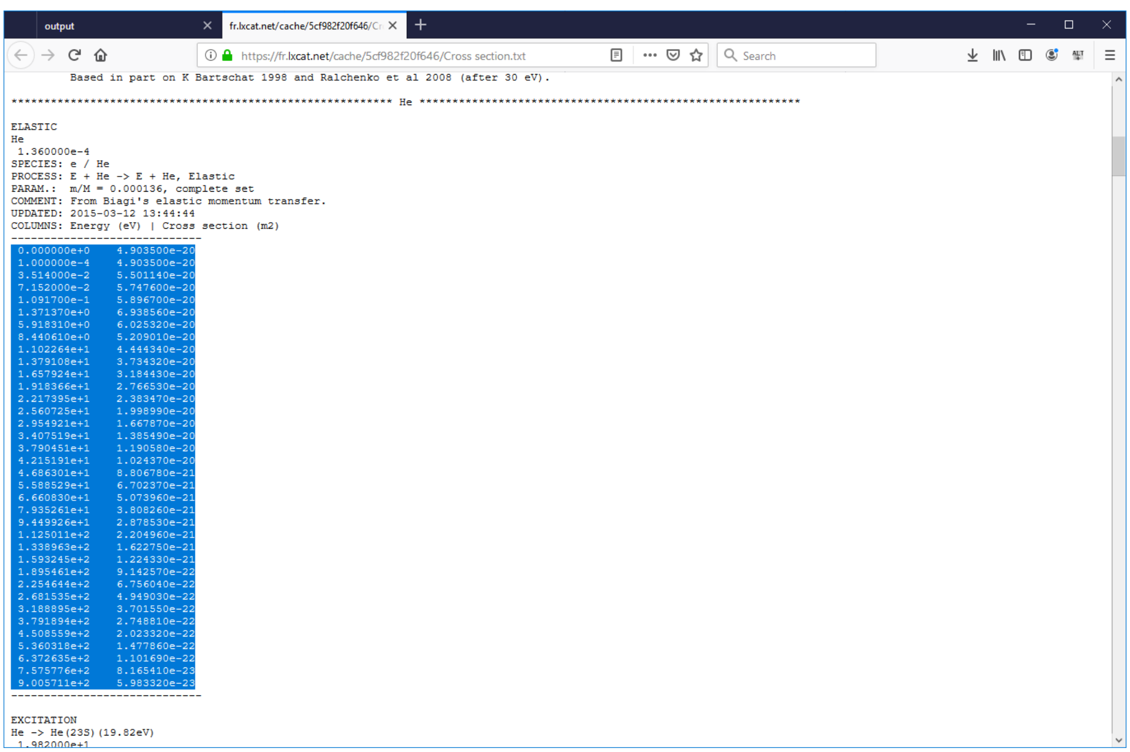

```
LXCat, www.lxcat.net
Generated on 06 Jun 2019. All rights reserved.

RECOMMENDED REFERENCE FORMAT
- Biagi database, www.lxcat.net, retrieved on June 6, 2019.
Be aware that some databases and solvers can additionally have instructions how to reference corresponding data.
Please check below in the headers of databases.

CROSS SECTION DATA FORMAT
In downloaded files, each collision process is defined by a block consisting of
1st line
Keyword in capitals indicating the type of the collision. Possible collision types are elastic, effective, excitation,
ionization, or attachment (capital letters required, key words are case sensitive), where "elastic" is used to denote
the elastic momentum transfer cross section and where "effective" denotes the total momentum transfer cross section (sum
of elastic momentum transfer and total inelastic cross sections).  The latter is useful for solving the Boltzmann
equation in the 2-term approximation.
2nd line
Name of the target particle species. This name is a character string, freely chosen by the user, e.g. "Ar". Optionally
for excitation processes, the name of the corresponding excited state can be specified on the same line, separated from
the first name either by arrow "->" (dash + greater than) or by double-head arrow "<->" (less than + dash +
greater than), e.g. "Ar -> Ar*" and "Ar <-> Ar*", respectively. In the later case BOLSIG+ will automatically
define the inverse superelastic process, constructing the superelastic cross-section by detailed balancing, and
considering the indicated excited state as the target. In this case, the ratio of statistical weights must be input in
the 3rd line (see below).
3rd line
For elastic and effective collisions, the ratio of the electron mass to the target particle mass. For excitation or
ionization collisions, the electron energy loss (nominally the threshold energy) in eV. For attachment, the 3rd line is
missing. In case of an excitation process where an excited state has been indicated on the 2nd line using double-head
arrow "<->", the 3rd line must specify also ratio of the statistical weights of the final state to the initial state
as the second parameter in 3rd line this is needed by BOLSIG+ to calculate the de-excitation cross-section. The
statistical weight ratio, if given, will also be used by the automatic superelastics option in BOLSIG+. If this ratio is
not provided then BOLSIG+ will assume it unity.
from 4th line (optionally)
User comments and reference information, maximum 100 lines. The only constraint on format is that these comment lines
must not start with a number.
Finally
Table of the cross section as a function of energy. The table starts and ends by a line of dashes "------" (at least 5),
and has otherwise two numbers per line: the energy in eV and the cross section in m2.

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
DATABASE:         Biagi (transcription of data from SF Biagi's Fortran code, Magboltz.)
PERMLINK:         www.lxcat.net/Biagi
DESCRIPTION:      These data were transcribed from S.F. Biagi's FORTRAN code, MagBoltz, versions 8.9 and later. The
                  exact version number is given for each species.   At this time, the Biagi database contains cross
                  sections for rare gases, for a few simple molecules, and for SF6. The transcription of cross sections
                  for other gases is in progress. These data were compiled for use a Monte Carlo simulation (or
                  multi-term Boltzmann code), but their use in a 2-term Boltzmann solver gives reasonably accurate
                  results in most cases. The the LXCat data tables do not always have the same energy resolution as the
                  original data in the MagBoltz code. This limited energy resolution can introduce small (1%)
                  differences in the calculated drift velocity at low E/N.
                  For history and detailed notes, see http://consult.cern.ch/writeup/magboltz/

                  Oct 2011 : update of Ar and Kr data taken from MagBoltz version 8.97 Nov 2011 : small changes in Xe
                  ionization cross sections, taken from MagBoltz v8.97
```




```
        Based in part on K Bartschat 1998 and Ralchenko et al 2008 (after 30 eV).

********************************************************** He **********************************************************

ELASTIC
He
 1.360000e-4
SPECIES: e / He
PROCESS: E + He -> E + He, Elastic
PARAM.:  m/M = 0.000136, complete set
COMMENT: From Biagi's elastic momentum transfer.
UPDATED: 2015-03-12 13:44:44
COLUMNS: Energy (eV) | Cross section (m2)
-----------------------------
 0.000000e+0    4.903500e-20
 1.000000e-4    4.903500e-20
 3.514000e-2    5.501140e-20
 7.152000e-2    5.747600e-20
 1.091700e-1    5.896700e-20
 1.371370e+0    6.938560e-20
 5.918310e+0    6.025320e-20
 8.440610e+0    5.209010e-20
 1.102264e+1    4.444340e-20
 1.379108e+1    3.734320e-20
 1.657924e+1    3.184430e-20
 1.918366e+1    2.766530e-20
 2.217395e+1    2.383470e-20
 2.560725e+1    1.998990e-20
 2.954921e+1    1.667870e-20
 3.407519e+1    1.385490e-20
 3.790451e+1    1.190580e-20
 4.215191e+1    1.024370e-20
 4.686301e+1    8.806780e-21
 5.588529e+1    6.702370e-21
 6.660830e+1    5.073960e-21
 7.935261e+1    3.808260e-21
 9.449926e+1    2.878530e-21
 1.125011e+2    2.204960e-21
 1.338963e+2    1.622750e-21
 1.593245e+2    1.224330e-21
 1.895461e+2    9.142570e-22
 2.254644e+2    6.756040e-22
 2.681535e+2    4.949030e-22
 3.188895e+2    3.701550e-22
 3.791894e+2    2.748810e-22
 4.508559e+2    2.023320e-22
 5.360318e+2    1.477860e-22
 6.372635e+2    1.101690e-22
 7.575776e+2    8.165410e-23
 9.005711e+2    5.983320e-23
-----------------------------

EXCITATION
He -> He(23S)(19.82eV)
 1.982000e+1
```

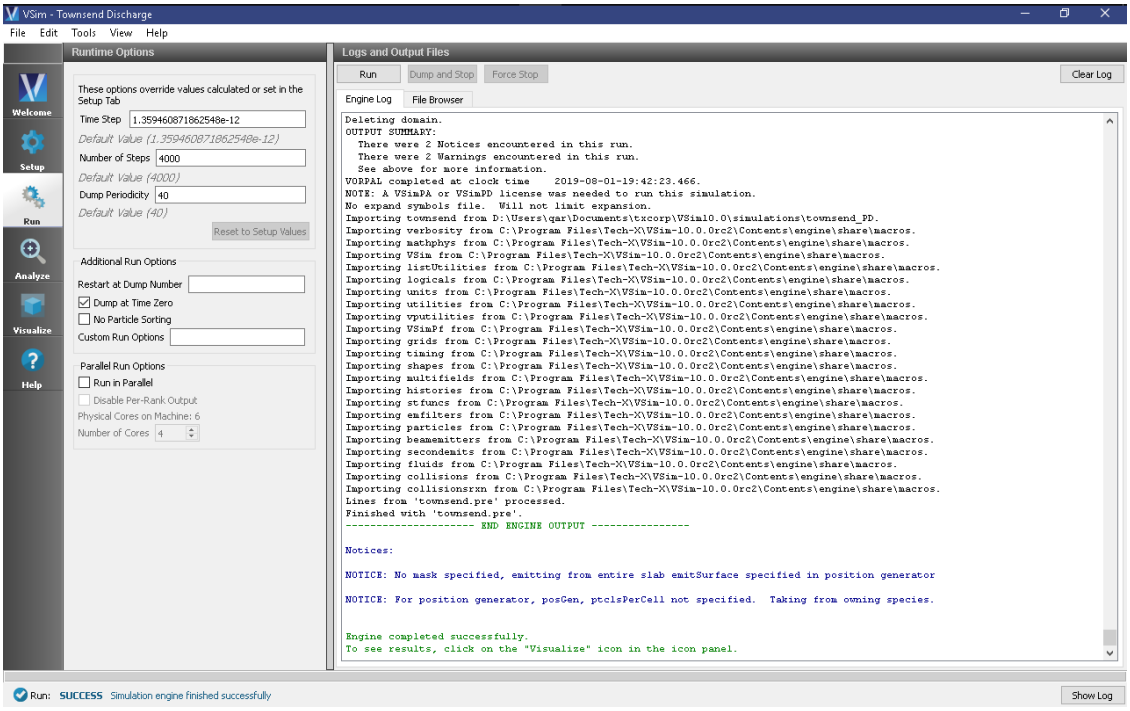


Fig. 6.52: The Run window at the end of execution.

Computing the Townsend Coefficient

To collect enough data and make a fit to

$$\frac{I(x)}{I_0} = e^{\alpha(x-x_0)}$$

this simulation will need to be run multiple times with different plate separations. Since the electric field is constant, different plate separations correspond to different voltage differences between the right and left plates. As the plates get further apart, electrons will have more space to accelerate and ionize the neutral helium gas. Since the total number of helium ions created in collisions with electrons depends on the total number of electrons in the space between the two electrodes, the exponential form of the equation above is expected.

To adjust the plate separation, change the *voltageDifference* constant which is under **Constants** in the setup tree. The simulation will open with this constant set to 40 volts. When we performed this study, we ran at voltage of 40, 50, 60, 70, 80, 90, and 100 volt differences, then ran a least squares fit of the data to the shifted exponential in Mathematica. Our code for performing the fit is below:

```
In[1]:= Clear["Global`*"];

(* number of electrons *)
In[2]:= Ne = 100000
Out[2]= 100000

(*  number of ions from various runs *)
In[3]:= Ni = {5460, 13258, 22548, 33778, 46605, 60772, 77181}
Out[3]= {5460, 13258, 22548, 33778, 46605, 60772, 77181}

(* corresponding voltages *)
In[4]:= Voltage = {40, 50, 60, 70, 80, 90, 100}
```

(continues on next page)

(continued from previous page)

```
Out[4]= {40, 50, 60, 70, 80, 90, 100}

(* distance increment corresponding to 10 volts *)
In[5]:= dL = 1/21200.
Out[5]= 0.0000471698

(* plate separations *)
In[6]:= Lvals = Voltage/10*dL
Out[6]= {0.000188679, 0.000235849, 0.000283019, 0.000330189, \
0.000377358, 0.000424528, 0.000471698}

(* Current ratio (measured/initial - for Nate's simulations, we will \
just define this directly from the data instead of computing from Ne, \
Ni *)
In[7]:= Ivals = N[1 + 2*Ni/Ne]
Out[7]= {1.1092, 1.26516, 1.45096, 1.67556, 1.9321, 2.21544, 2.54362}

(* Deviation from exponential growth *)
In[8]:= S = Total[(Log[Ivals] - m*Lvals - b)^2]
Out[8]= (0.933588 - b - 0.000471698 m)^2 + (0.795451 - b -
   0.000424528 m)^2 + (0.658607 - b - 0.000377358 m)^2 + (0.516147 -
   b - 0.000330189 m)^2 + (0.372225 - b -
   0.000283019 m)^2 + (0.235199 - b - 0.000235849 m)^2 + (0.103639 -
   b - 0.000188679 m)^2

(* Minimize with respect to m and b *)
In[9]:= mval = Expand[m /. Solve[D[S, m] == 0, m][[1]]]
Out[9]= 1668.61 - 2800. b

In[10]:= bval = Expand[b /. Solve[D[S, b] == 0, b][[1]]]
Out[10]= 0.516408 - 0.000330189 m

In[11]:= sols = Solve[{m == mval, b == bval}, {m, b}][[1]]
Out[11]= {m -> 2950.38, b -> -0.457775}

(* Predicted Townsend *)
In[12]:= AlphaTownsend = m /. sols
Out[12]= 2950.38

(* Predicted Alpha/p0 *)
In[13]:= AlphaTownsend/21.2/100
Out[13]= 1.39169
```

Note: The data in this code snippet were taken from simulations that used the cross-sections obtained from the LxCat database.

The `In[3]:=` line contains a list of helium ions counts collected from the 7 runs at different voltages, and correspond to the voltages in the list on the `In[4]:=` line. Plate separations are calculated based on the voltages in the `In[6]:=` line and are values for the independent variable, x_0 in the equation to which we are fitting.

In the `In[7]:=` line we calculate an estimate for the total current that would be measured should the discharge be sustained continuously. We consider each particle to represent one "unit" of current. Therefore, the total current will be $N_0 + N_{e,i} + N_{He,i}$ where N_0 is the initial number of electrons emitted from the cathode, $N_{e,i}$ is the number of electrons created in ionization collisions, and $N_{He,i}$ is the number of helium ions created in collisions. Since math:*N_{e,i} = N_{He,i}*, ie the number of electrons created through collisions are the same as the number of helium

ions created in collisions, we calculate the normalized current (quantity on the left side of the equation above) through the chamber here. The rest of the lines are performing the mathematics of the fit.

To obtain the data used on the `In[3]:=` line, follow these instructions:

1. After a simulation has finished running, proceed to the Visualize Window by pressing the *Visualize* button in the navigation column.
2. From the `Add a Data View` menu at the top left of the Visualize Window choose **History**. A new tab will open in which you can visualize data collected every timestep.
3. Set the history in `Graph 1` to "numPhysHeIons" and set the other three Graphs to "None". When the data is finished loading, your screen should look like the image in Fig. 6.53.

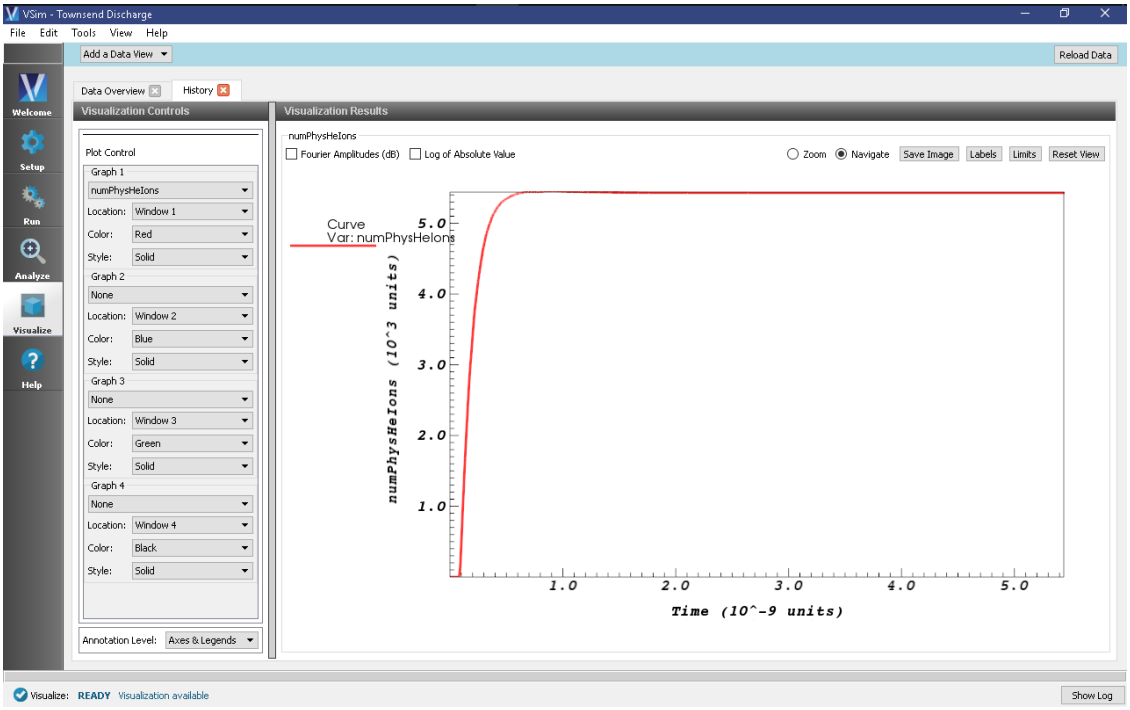


Fig. 6.53: The total number of helium ions in the simulation as a function of time.

4. Zoom into the end section of the plot to easily read off the total number of helium ions at the end of the simulation. An astute user might notice that some helium ions are absorbed while the simulation is running, so the number of helium ions isn't an exact measure of the number of ionization events that occurred. To get the number of ions absorbed during the simulation, one can plot the `HeUpperXAbsorbedCurrent` and `HeLowerXAbsorbedCurrent` histories, where each spike in the histories represent the absorption of one ion. Since the total number of absorbed ions is around a percent of total ions, this isn't a terrible approximation.
5. Copy this value into the Mathematica (or whichever program you choose to run the fitting).

Further Experiments

1. Run the simulation at a different pressure to measure the Townsend coefficient in a different regime.
2. Change the gas type by switching the ion species, gas species, and cross-sections.

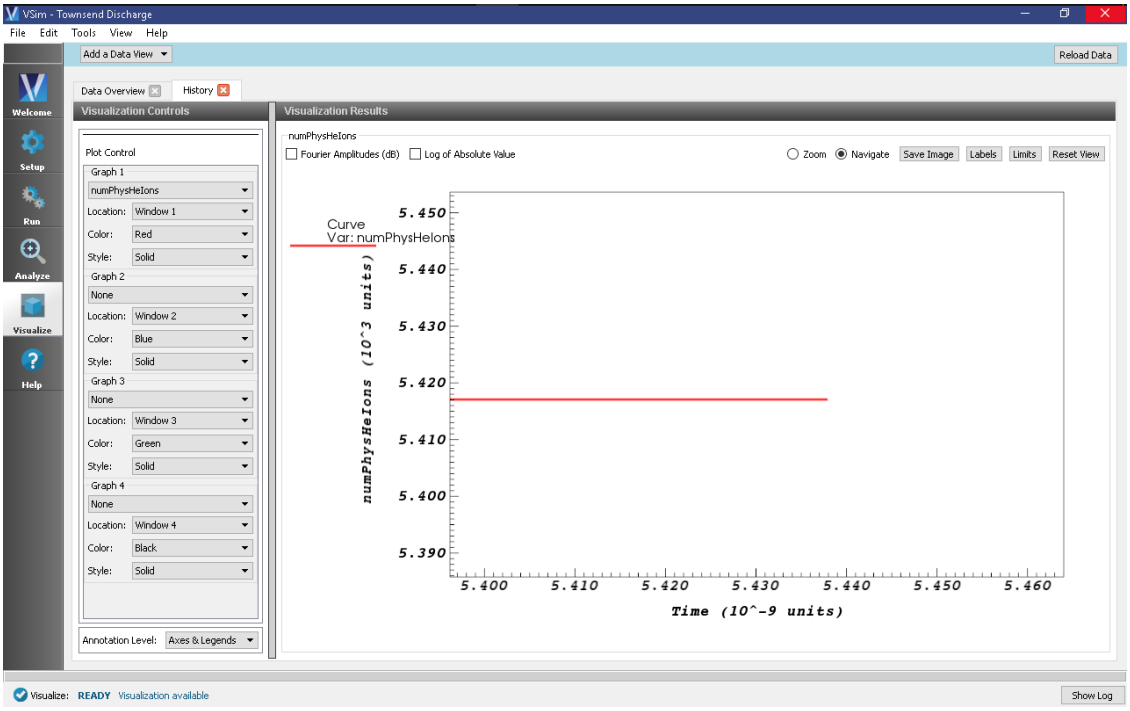


Fig. 6.54: A zoomed in view of the helium ion count at the end of the simulation.

6.6 Spacecraft

6.6.1 Coupon Array Charging (couponArrayCharging.sdf)

Keywords:

`solar wind, electrostatics, surface charging`

> **Warning:** Due to a known issue in parallel runs on Windows, this example should only be run serial on Windows.

Problem description

In orbit, insulating outer surfaces of satellites will develop a surface charge due to the impinging solar wind. If enough surface charge accumulates electric breakdown can occur across or through the satellite and damage the craft.

This simulation models the accumulation of solar wind particles on an array of solar cells (coupons). The array includes 6 coupons, a kapton backing, and 6 metal busbars. Using post-simulation analysis, the component of the electric field normal to the surface of the satellite is calculated.

With additional data specific a particular spacecraft and materials, this simulation can indicate locations where breakdown is likely to occur.

This simulation can be performed with a VSimPD license.

Opening the Simulation

The Electron Drifting example is accessed from within VSimComposer by the following actions:

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim for Plasma Discharges* option.
- Expand the *Spacecraft* option.
- Select "Coupon Array Charging" 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* as seen after opening the example is shown in Fig. 6.55.

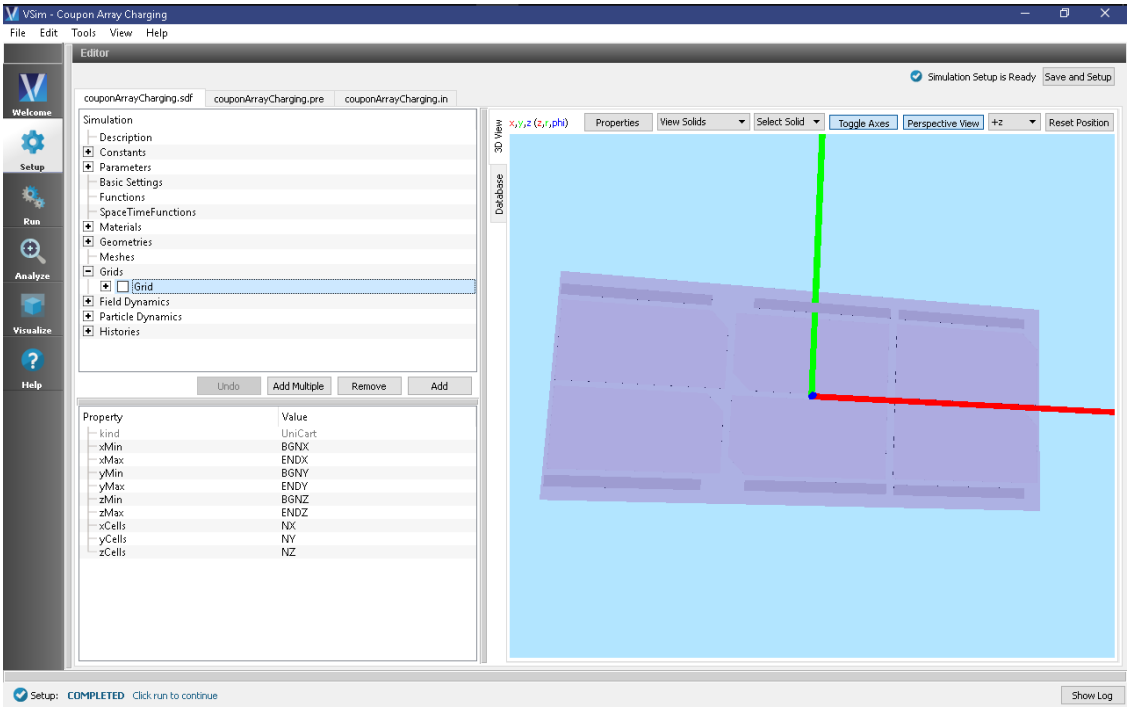


Fig. 6.55: Setup Window for the Coupon Array Charging example.

Simulation Properties

The geometry for the busbars and coupon array are imported from stl files. Material properties are set on the geometries: perfect electrical conductor (PEC) for the busbars, and absorbium, an insulating particle absorbing material, on the array of cells.

A voltage of 5 volts is set on the busbars. The upper z boundary is set as the $V = 0$ point, a Neumann boundary condition is set on the lower z boundary of the simulation grid, which enforces that the gradient of the electric field normal to this surface is zero. Periodic boundary conditions (for particles and fields) are set on all other simulation boundaries.

The solar wind is emitted off the upper z boundary of the simulation domain with a number density of 1.e7 particles per meter cubed. The masses of the ions are artificially set to 100x the mass of the electrons. Particle accumulation boundary conditions are set on the insulating surface of the coupons, and a particle absorbing boundary condition is set on the metal busbars.

Histories save the absorbed particle energy deposited onto the satellite surface.

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.
- 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." This is shown in Fig. 6.56 below.

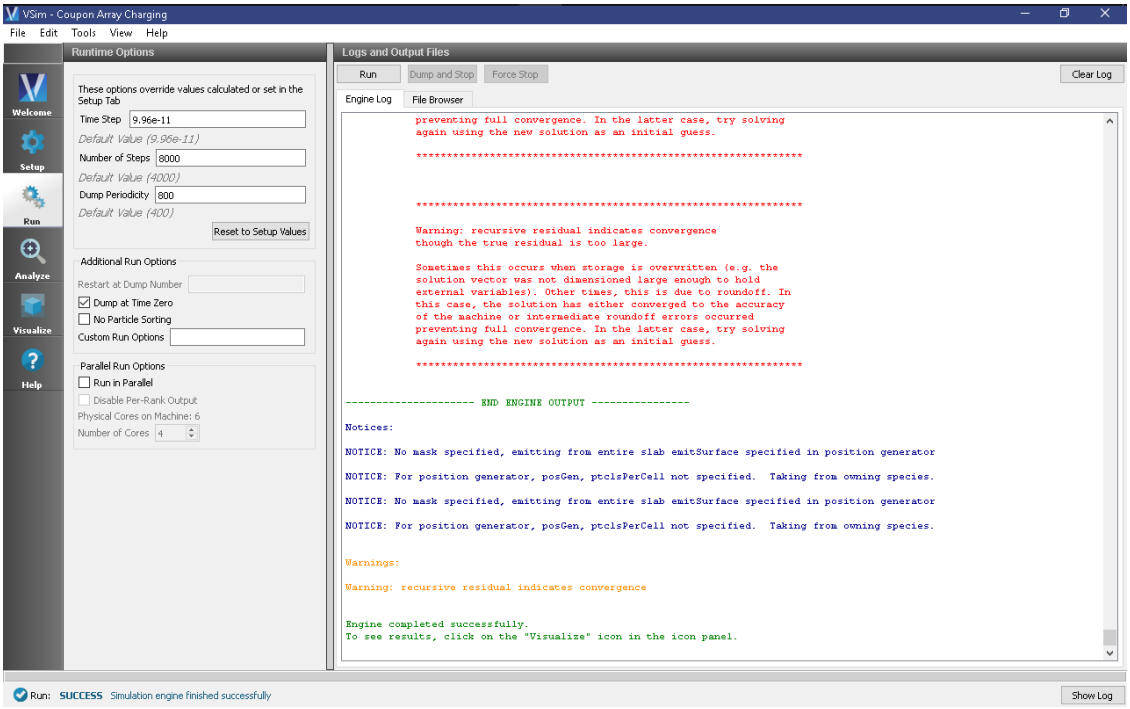


Fig. 6.56: The Run Window at the end of execution.

Analyzing the Results

The physics engine, vorpal, inside VSim only calculates field values on edges, nodes, or faces of grid cells. The *putFieldOnSurfaceMesh.py* analyzer can interpolate the values calculated on the grid to the surface of a geometry in the simulation.

To calculate the normal component of the electric field on the surface of the array, proceed to the *Analyze* Tab. The *putFieldOnSurfaceMesh.py* analyzer is included by default to this simulation. Click on the text "putFieldOnSurfaceMesh.py (Default)" to highlight it, then click the "Open" button at the bottom of the *Analysis Controls* pane. Ensure the following is entered into each field:

- **`simulationName:`** "couponArrayCharging"
- **`geometryName:`** "satelliteSurfaceGeomSolid"
- **`fieldName:`** "E"
- **`beginDump:`** "1"
- **`endDump:`** "9"
- **`outputFileName:`** "elecFieldOnSurface"

- Click *Analyze* in upper right corner of the window. When the analysis is finished, you should see a window similar to Fig. 6.57.

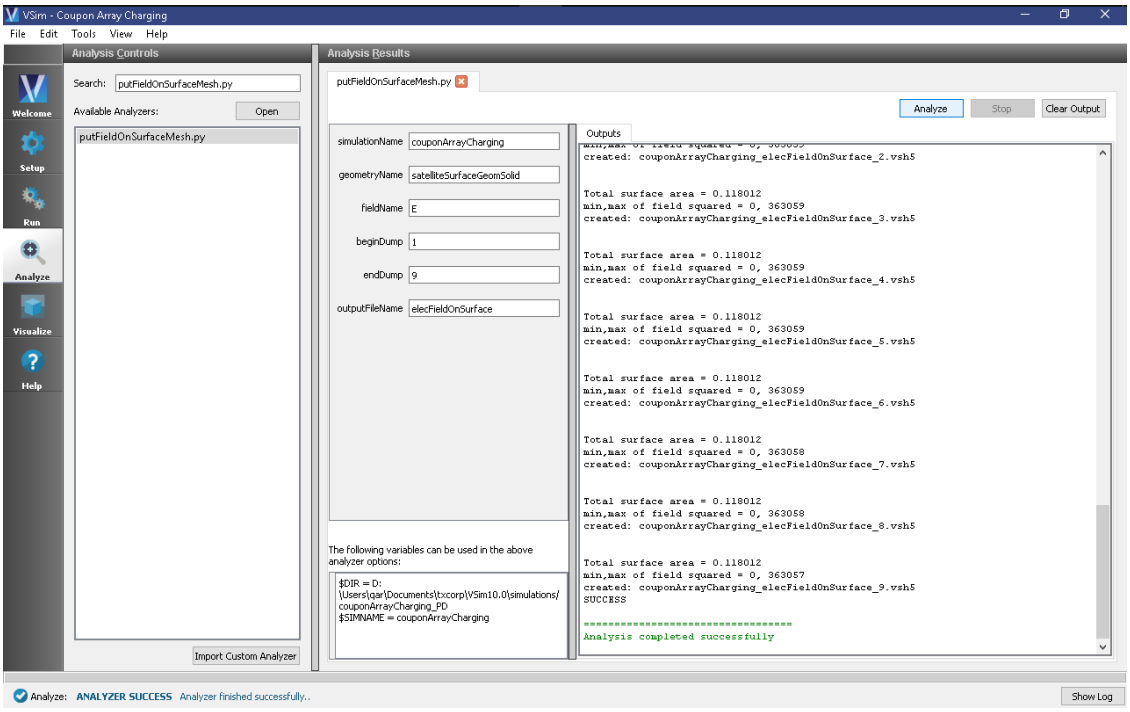


Fig. 6.57: The Analyze Window at the end of execution.

Visualizing the Results

After run completion, continue as follows:

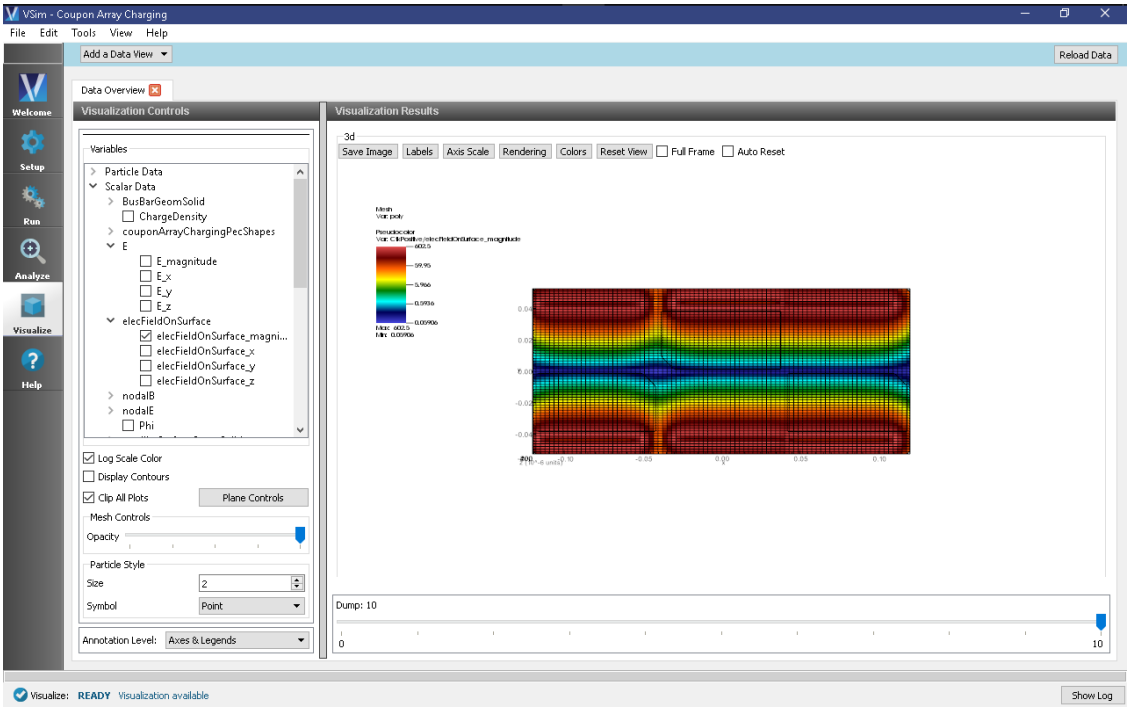
Proceed to the Visualize Tab by pressing the Visualize button in the left column of buttons. To view the normal component of the electric field on the surface of the array follow the following steps.

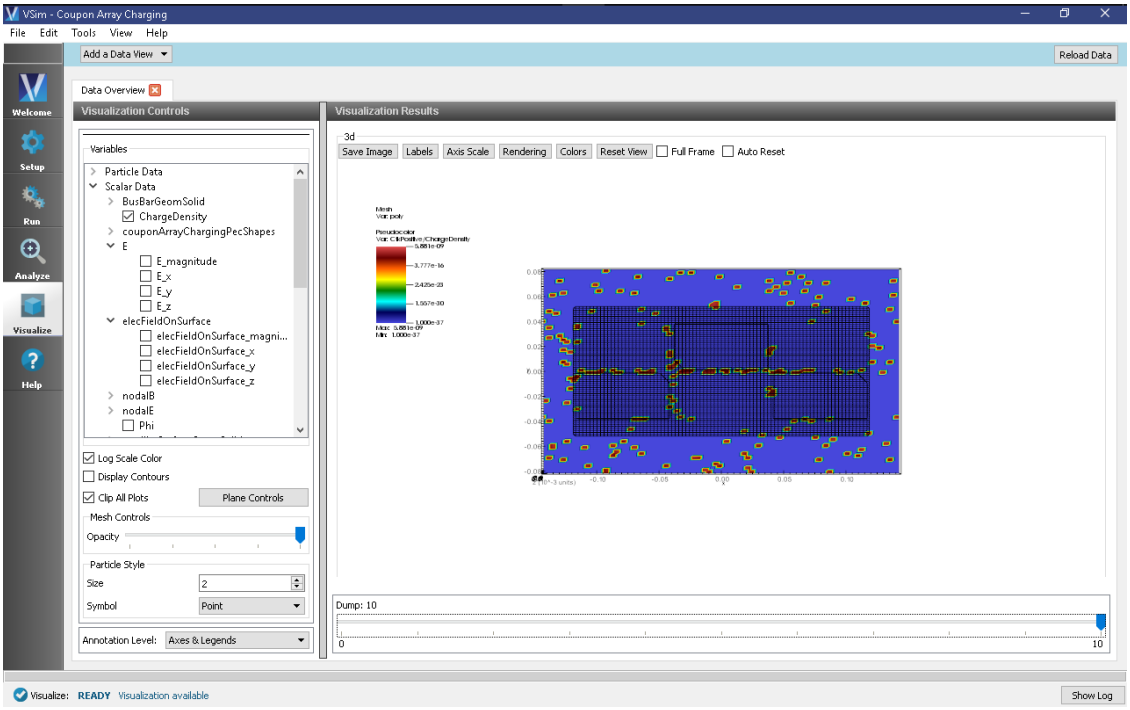
- If you have previously switched to the Visualize Tab, you will have to click the *Reload Data* button at the bottom of the *Visualization Controls* pane.
- Expand "Scalar Data" then expand "elecFieldOnSurface" and check the box for "elecFieldOnSurface_magnitude" to plot the component of the electric field normal to the surface of the coupon array. It is also possible to plot the two tangential components of the field, as well as the magnitude.
- To compare to the satellite geometry, expand "Geometries" then select "poly (satelliteSurfaceGeomSolid)".
- To see the fields more clearly, select "Log Scale Color"
- The resulting visualization is shown in Fig. 6.6.1.

We can see the accumulation of electric charge by unselecting "elecFieldOnSurface_magnitude" and viewing "ChargeDensity" instead. Move the dump slider to view the accumulation of charge over time, as shown in Fig. 6.6.1.

Further Experiments

Perform the same analysis done above for the electric potential, "Phi," and charge density, "ChargeDensity," to create plots of those fields on the surface of the satellite geometry.


VSim - Coupon Array Charging
File Edit Tools View Help
Add a Data View
Reload Data
Data Overview
Visualization Controls
Visualization Results
Welcome
Setup
Run
Analyze
Visualize
Help
Variables
Particle Data
Scalar Data
BusBarGeomSolid
ChargeDensity
couponArrayChargingPecShapes
E
E_magnitude
E_x
E_y
E_z
elecFieldOnSurface
elecFieldOnSurface_magni...
elecFieldOnSurface_x
elecFieldOnSurface_y
elecFieldOnSurface_z
nodalB
nodalE
Phi
Log Scale Color
Display Contours
Clip All Plots
Plane Controls
Mesh Controls
Opacity
Particle Style
Size
2
Symbol
Point
Annotation Level:
Axes & Legends
3d
Save Image
Labels
Axis Scale
Rendering
Colors
Reset View
Full Frame
Auto Reset
Dump: 10
0
10
Visualize: READY Visualization available
Show Log


VSim - Coupon Array Charging
File Edit Tools View Help
Add a Data View
Reload Data
Data Overview
Visualization Controls
Visualization Results
Welcome
Setup
Run
Analyze
Visualize
Help
Variables
Particle Data
Scalar Data
BusBarGeomSolid
ChargeDensity
couponArrayChargingPecShapes
E
E_magnitude
E_x
E_y
E_z
elecFieldOnSurface
elecFieldOnSurface_magni...
elecFieldOnSurface_x
elecFieldOnSurface_y
elecFieldOnSurface_z
nodalB
nodalE
Phi
Log Scale Color
Display Contours
Clip All Plots
Plane Controls
Mesh Controls
Opacity
Particle Style
Size
2
Symbol
Point
Annotation Level:
Axes & Legends
3d
Save Image
Labels
Axis Scale
Rendering
Colors
Reset View
Full Frame
Auto Reset
Dump: 10
0
10
Visualize: READY Visualization available
Show Log

Import your own geometry and reset the materials, grid size, and particle absorbers as necessary.

Increase the grid resolution to get finer data on the electric field that develops on the surface of the spacecraft.

Change the number density and speeds of the incident particles to values for orbits at different altitudes.

6.6.2 Cylindrical Hall Thruster (cylHallThruster.sdf)

Keywords:

`electric propulsion, Hall thruster channel, erosion models.`

Problem description

Electric Hall thrusters are used for in-space propulsion and satellite station-keeping needs. The discharge plasma inside the Hall thruster channel is produced by the ionization of electrons with a neutral propellant gas such as xenon. The electrons are emitted from the neutralizer cathode placed at the exit of the Hall thruster (cathode end). The neutral gas is fed into the channel from the anode end of the Hall thruster channel. The electrons are confined inside the Hall thruster channel by the radial magnetic field applied through the solenoidal magnetic fields. Plasma xenon ions are accelerated out of the channel at high velocity, which produces the thrust necessary for space propulsion. Recently these thrusters are being designed to support long life time, high-power and high-thrust operations. The channel wall erosion occurring inside of the Hall thruster is one of the main limitations to these design needs. It becomes important to understand the plasma discharge processes occurring inside the Hall thruster channel and predict the lifetime of the Hall thruster based on the calculations of sputtered material from the Hall thruster channel.

This example demonstrates elements of the full cylindrical Hall thruster text-based example. Please refer to that documentation for a detailed description of the simulation geometry, and physical properties of the full model. This visual-setup model employs the same physical geometry (2D, cylindrical coordinates) as the text-based setup model. As with the text-based setup example, there is an electron source and a background xenon gas that is ionized by kinetic electrons to produce kinetic xenon ion particles. Particle sinks, and the physical extent of the background gas are the same as in the text-based example.

The primary difference is that there are no dielectric materials in this example, and so the electromagnetic fields are different. This leads to a different pattern for electrons exiting the simulation than in the text-based example. Also, there is no neutral particle sputtering of wall material included in this example. This example does demonstrate ionization of neutral gas in cylindrical geometry.

This simulation can be performed with a VSimPD license.

Opening the Simulation

The Cylindrical Hall Thruster example is accessed from within VSimComposer by the following actions:

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim for Plasma Discharges* option.
- Expand the *Spacecraft* option.
- Select *Cylindrical Hall Thruster* 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. 6.58. 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`.

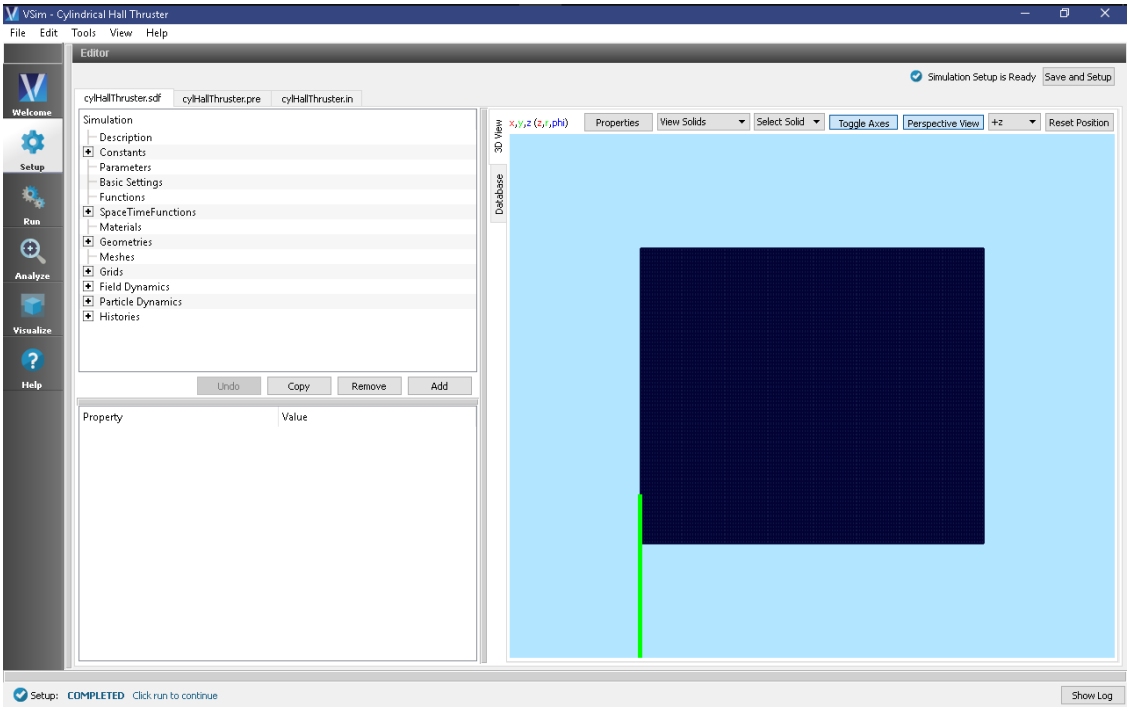


Fig. 6.58: Setup Window for the Cylindrical Hall Thruster Channel example.

Simulation Properties

This example contains many user defined *Constants* which help simplify the setup and make it easy to modify. These include constants such as:

- B0: The amplitude of the background magnetic field
- anodeV: the anode voltage
- innerRad and outerRad: inner and outer cylinder radius
- xeMaxDensity: the maximum density of the background Xe gas

There are also several *SpaceTimeFunctions* that are used to define spatially and/or temporally varying inputs to other properties. These include:

- By: the magnetic field profile
- initialGasDensity: the profile for the background gas density

The self-consistent electric field is solved from Poisson's equation by the electrostatic solver in a cylindrical coordinate system. The simulation is performed in axisymmetric 2-D fashion. The plasma is represented by macro-particles which are moved using the Boris pusher in cylindrical coordinate system. Various types of elastic and inelastic collisions of the particles are also taken into account.

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.

- 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. 6.59.

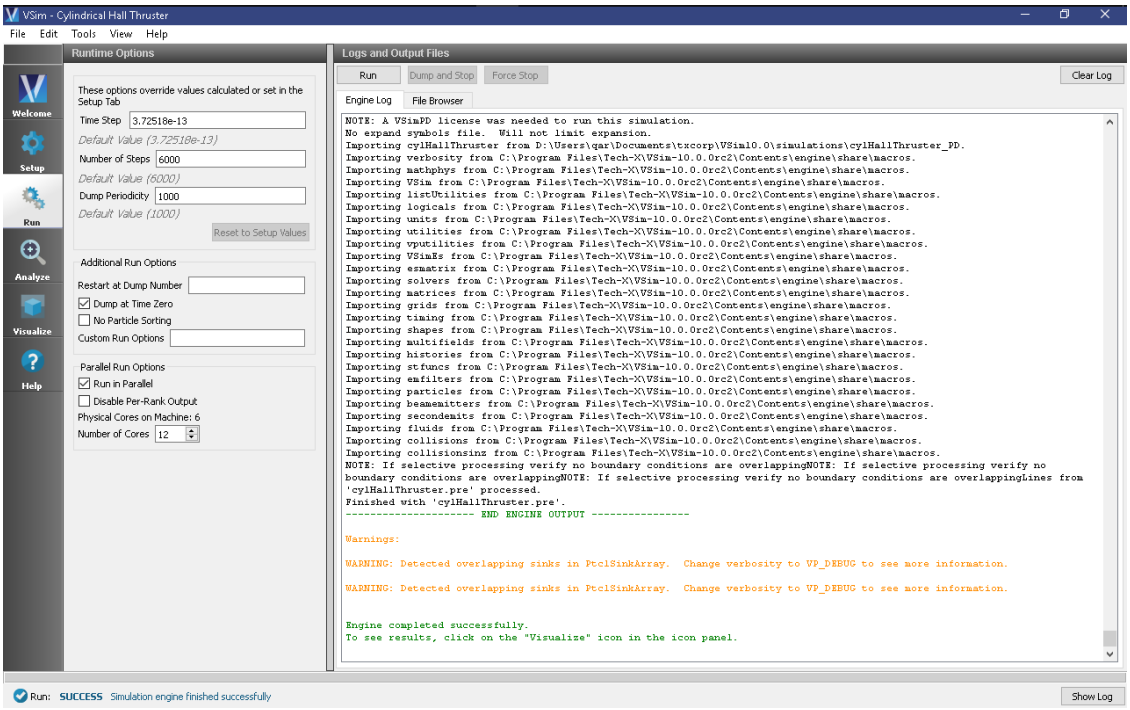


Fig. 6.59: The Run Window at the end of execution.

On an 8 core I7-6000 CPU, it takes about 40 minutes to run for 6,000 time steps. To reach steady state, about 100,000 time steps are required.

Analyzing the Results

If it is desired to calculate the density of the electrons or ions the analysis script *computePtclNumDensity.py* must be used.

- First click on the *Analyze* Tab.
- From the *Available Anaylzers* list, choose *computePtclNumDensity.py*. Then click *Open*.
- This script accepts the *simulationName* (Name of the input file) and *speciesName* to be calculated (species of particles).
- To calculate the density of the electrons, set the simulationName to "cylHallThruster" and the speciesName to "electrons".
- Click on the *Analyze* button at the top right of the *Analysis Results* pane.
- A snapshot of the simulation run completion is shown in Fig. 6.60.

The resulting data will be visualizable as *electronsDensity* under the *Scalar Data* menu in the *Visualize* Tab. The density of other particles such as heavyIons, or Xeplus can also be calculated if those species names are used in place of electrons.

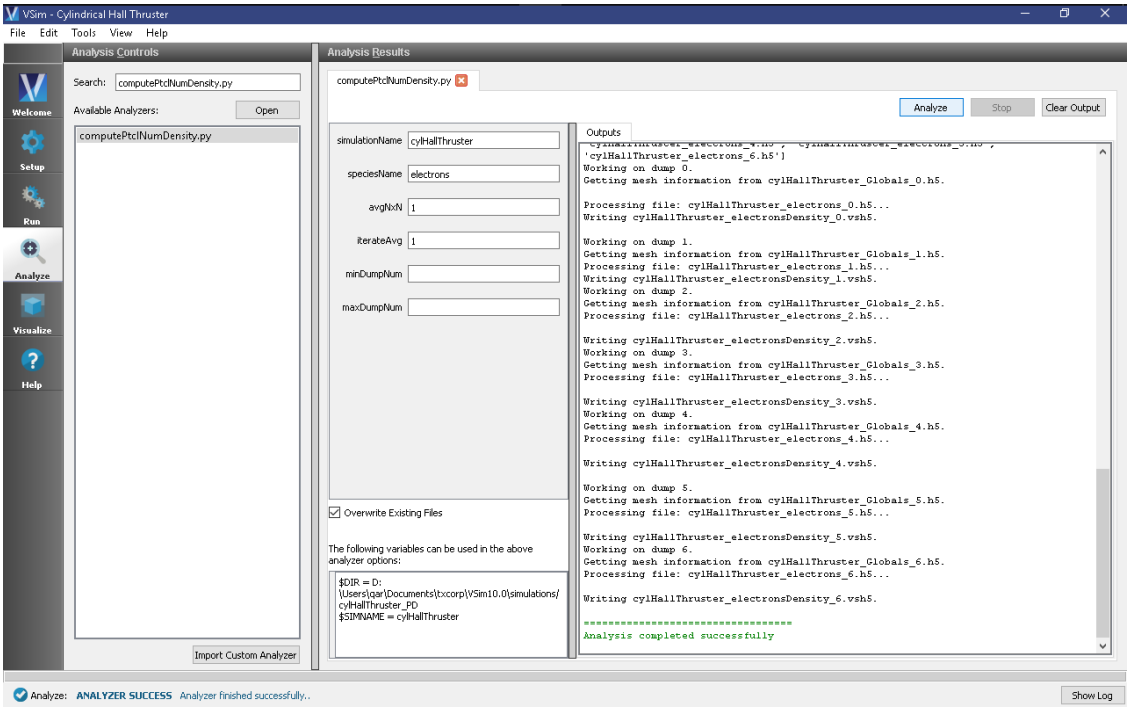


Fig. 6.60: The Analyze Window at the end of execution.

Visualizing the Results

To visualize the results, continue as follows:

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

There are many different fields, particles, and histories that can be visualized in this example. The horizontal axis represents Z direction and the vertical axis represents R direction.

To view the electric potential, switch to the *Data View* to *Data Overview* in the Controls pane. Expand *Scalar Data* and choose *Phi*. Move the *Dump* slider to the right most position. Fig. 6.61 shows the visualization seen for the electric potential of the cylindrical Hall thruster channel and in the exit region.

In the Hall thruster channel plasma, the electrons injected from the right end (i.e., exit of the channel) are accelerated towards the anode biased wall at the left end. To plot the particles, expand *Particle Data* and select "electrons" and "XePlus" check boxes The figure below, Fig. 6.62, shows the distribution of positively charged xenon ions (red dots) and electrons (blue dots).

The static radial magnetic field distribution considered for the SPT-100 Hall thruster channel set up is shown in Fig. 6.63. To reproduce this plot, expand *Scalar Data* then *nodalB* and select *nodalB_r*. The magnetic field is strong near the inner cylinder and has a Gaussian bell-shaped field distribution both inside and at the exit of the channel.

The background xenon neutral gas density distribution (plottable as *XeNeutralFluid* under *Scalar Data*) used in the simulation set up is shown in Fig. 6.64. The maximum neutral gas density is taken at the left end of the channel near the anode wall. A linearly varying neutral gas density is assumed.

Further Experiments

This example can be modified to test different design parameters such as varying anode voltages, varying background neutral gas densities and varying electron emission currents. This will allow users to study high-to-low power and

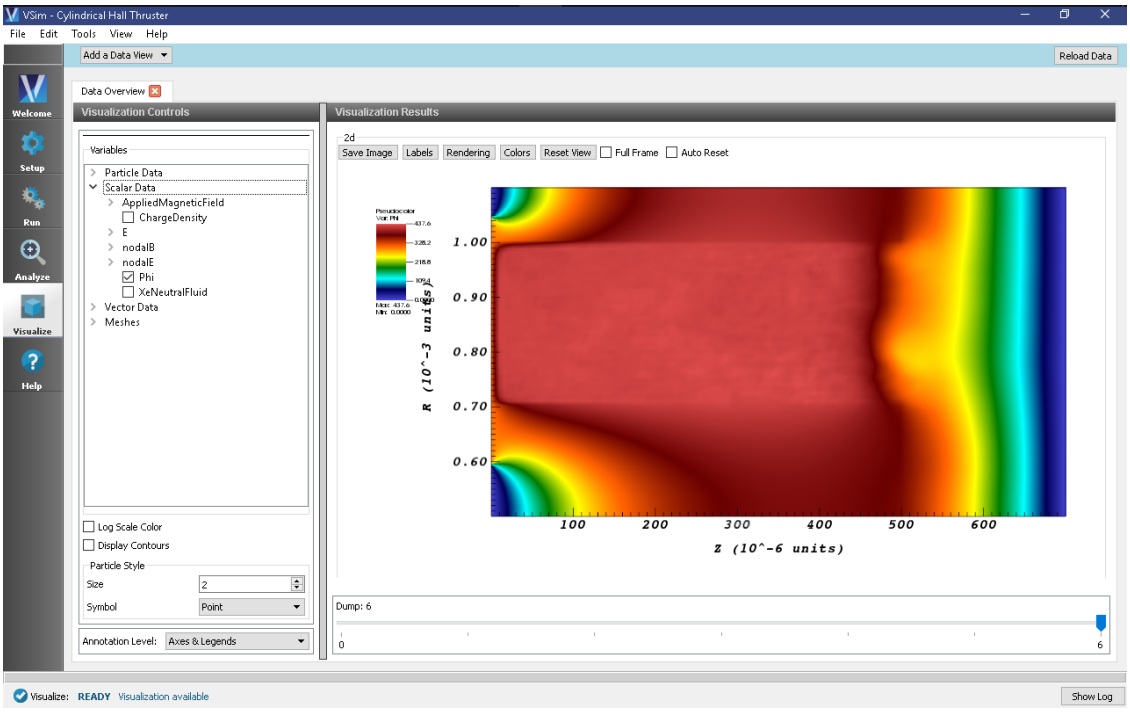


Fig. 6.61: Visualization of Cylindrical Hall thruster channel electric potential results.

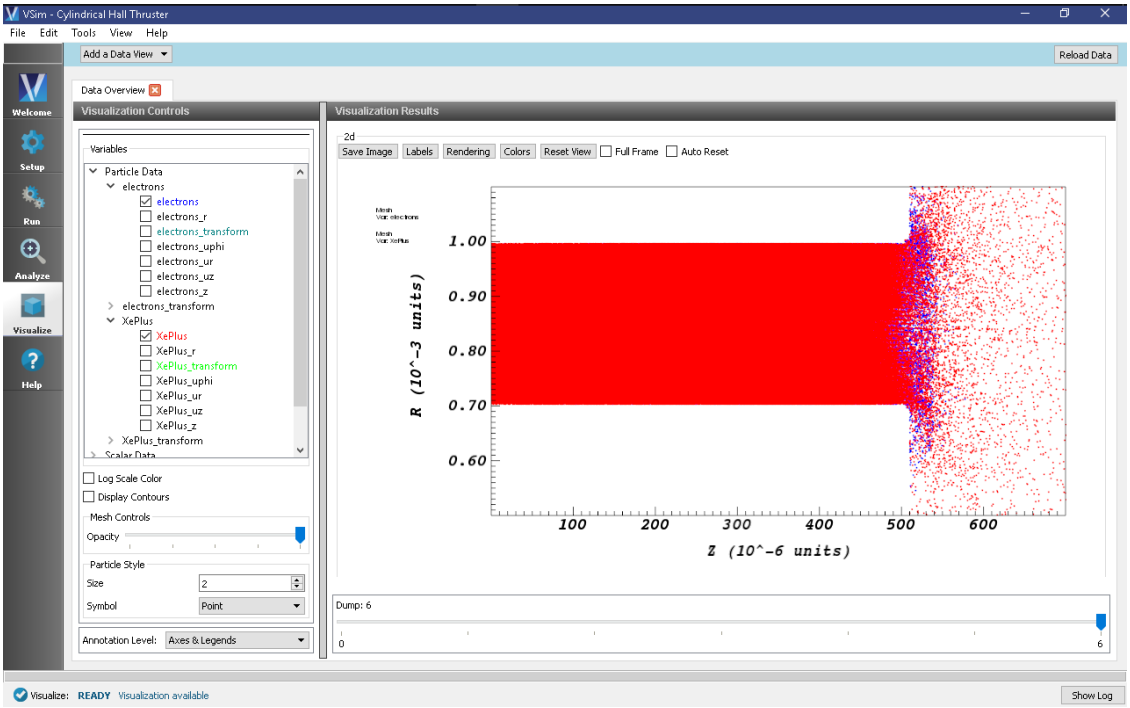


Fig. 6.62: Visualization of Cylindrical Hall thruster channel particle phase-space distribution results.

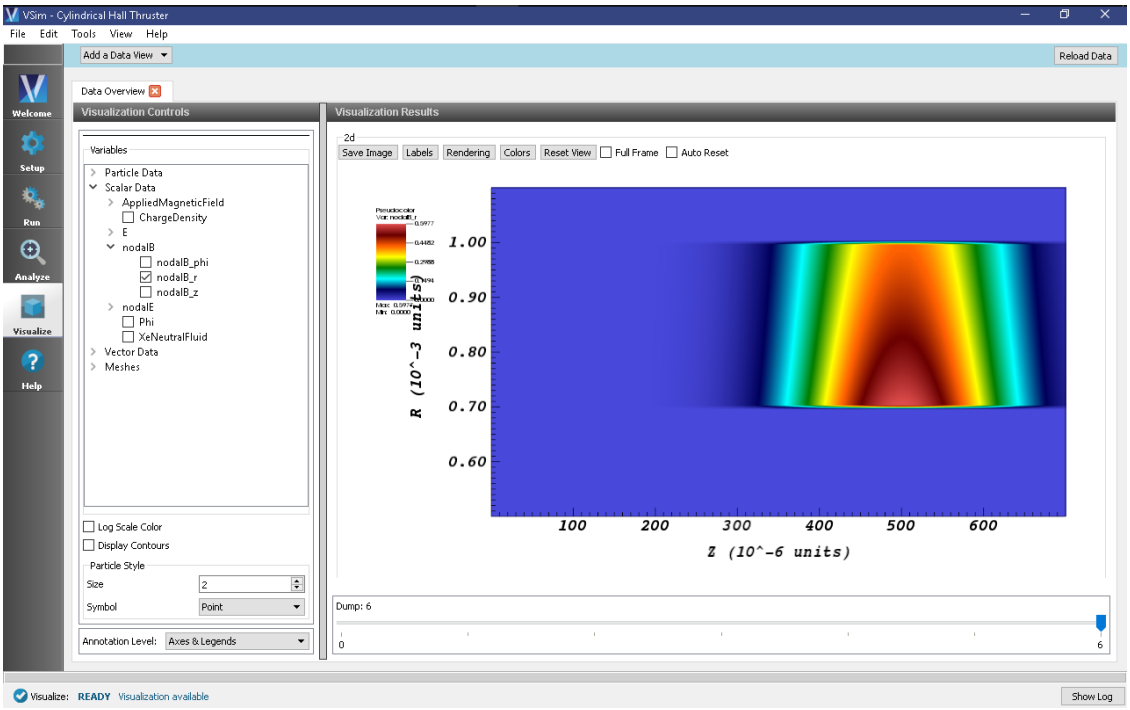


Fig. 6.63: Visualization of radial magnetic field in the Cylindrical Hall thruster channel.

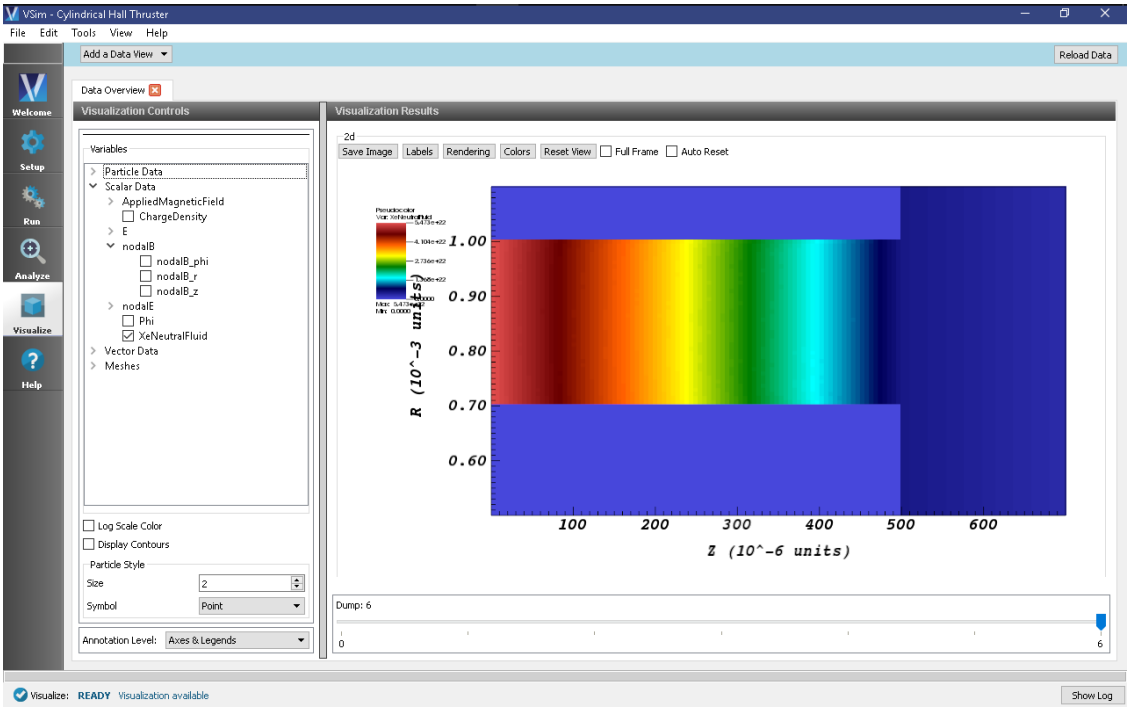


Fig. 6.64: Visualization of xenon neutral gas density in the Cylindrical Hall thruster channel.

high-to-low throttle levels.

Also the background gas type can be changed to investigate other neutral gas kinds in this simulation set up.

6.7 Spacecraft (text-based setup)

6.7.1 Ion Thruster (ionThrusterT.pre)

Keywords:

`electric ion thruster discharge chamber plasma processes in 2D cylindrical system.`

Problem Description

Ion thrusters are electric propulsion devices used for in-space propulsion and satellite station-keeping. In this device, a propellant gas (xenon in this example) is ionized into a plasma state inside a cylindrical discharge chamber. The plasma ions are accelerated out of the chamber through an electrostatic grid optics system to produce thrust. The device consists of the following components: an anode-biased discharge chamber, a discharge hollow cathode assembly, permanent magnet rings, a neutral propellant feed system, grid optics (screen and accelerator plates), and a neutralizing hollow cathode. The discharge hollow cathode is placed in the center of the discharge chamber and emits energetic electrons (primary electrons) into the system. Primary electrons undergo ionizing collisions with the neutral propellant gas inside the chamber to produce plasma ions and secondary electrons. An energetic electron impacting a xenon atom that is already singly ionized may cause another electron to detach, resulting in a doubly ionized xenon atom. The permanent magnetic rings within the discharge chamber confine the electrons, increasing their time of flight in the chamber, and thus their chances of ionizing neutrals before collection at the anode-biased discharge chamber walls. The plasma ions produced in the chamber leave primarily through the screen grid plate with some losses to the cathode biased walls. To ensure long discharge cathode lifetimes, a protective enclosure called a cathode keeper (generally kept between 3 and 5 volts above the discharge cathode voltage) is used to shield the cathode plate from plasma ion collisions. The bombardment of singly charged and doubly charged ions during thruster operation will over time erode the face of the cathode keeper and expose the discharge cathode to energetic ions. Thus, it becomes important to model the ion flux around the cathodes. Recently ion thrusters have been designed to meet high-power and high-thrust-to-power space propulsion requirements. Numerical discharge chamber plasma simulations provide a detailed understanding of the plasma processes that go on inside a discharge chamber and help with the calculation of electron discharge currents, ion beam currents, and ion current losses to the chamber walls.

This example demonstrates the xenon discharge plasma processes inside of a cylindrical discharge chamber with a three-ring magnetic circuit arrangement. One magnetic ring is mounted on the forward wall (seen as the left wall in the geometry of the example) and two magnetic rings are mounted to the exterior wall of the cylindrical discharge chamber (seen as the top wall in the example setup). The radius of the cylindrical chamber is 20 cm and it is 18 cm long. The screen grid plate has a radius of 18 cm and is placed at the aft end of the discharge chamber (far right in the example setup). The discharge hollow cathode assembly is placed at the center of the discharge chamber. The radius of the cathode keeper assembly is taken to be 0.75 cm and its orifice protrudes out 7 cm from the forward wall (from the left wall in the example diagram). An electron particle source is implanted next to the cathode keeper orifice to model the electron emission of the discharge cathode. In this simulation the cathode emission current is taken as 10 A. The same cathode emission source location is also used for modelling the neutral propellant flow from the discharge cathode. The main xenon neutral propellant source is modelled along the exterior wall (top wall in the example diagram). We have taken neutral propellant flow rates of 4.5 sccm and 43.5 sccm for the discharge cathode neutral source and main neutral source respectively. The anode biased discharge chamber walls are kept at 25 V. The discharge cathode keeper is biased at 5 V and the screen grid plate is kept at 0 V. Finally, we enable a self-similar scaling system for the simulation of discharge chamber plasma described by figure 1 in *[MCL+11]*. This is based on earlier work by Taccogna *[TLCS04][TLCS05]*. By default the shrink scale factor is 200, i.e., the thruster dimensions are scaled by 1/200. This scaling ensures that simulations can be performed in a reasonable run time but it requires

use of an inflated permittivity scale factor, i.e. the permittivity of free space is artificially inflated so that numerical parameters like grid spacing and time step values satisfy the smaller plasma frequency and Debye length.

The simulation is initiated with the chamber pre-filled with xenon neutrals. This is because the neutrals are heavy and slow, and it would take a great many time steps at the start of every run to populate an empty chamber. To view the initial distribution of neutrals, the input file can be run with particle sources turned-off. To do this, switch the TURN_THRUSTER_OFF parameter to 1 and run for one time step. Then, in the Visualize window, select XeNeutrals under Particle Data.

This simulation can be performed with a VSimPD license.

Opening the Simulation

The ion thruster example is accessed from within VSimComposer by the following actions:

- Select the *New* → *From Example…* menu item from the *File* menu.
- In the resulting *Examples* window expand the *VSim for Plasma Discharges* option.
- Expand the *Spacecraft (text-based setup)* option.
- Select *Ion Thruster (text-based setup)* and click 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 basic variables of this problem should now be alterable via the text boxes in the left pane of the *Setup Window*, as shown in Fig. 6.65.

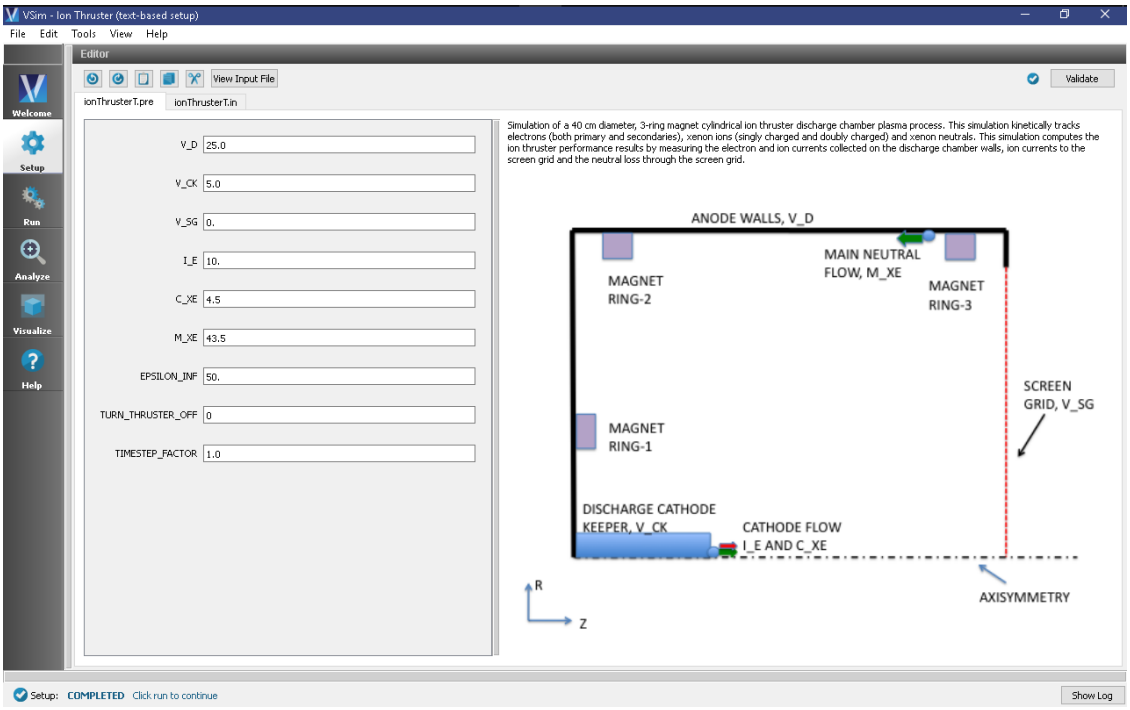


Fig. 6.65: Setup Window for the Ion Thruster example.

Input File Features

The input file allows the user a choice of ion thruster operating parameters such as discharge voltage, cathode keeper voltage, screen grid voltage, discharge cathode electron emission current, cathode neutral flow rate, and main neutral flow rate. Also it gives the user an option to specify the inflated permittivity scale factor by which the real permittivity of free space is scaled.

The self-consistent electric field is solved from Poisson's equation by the electrostatic solver in cylindrical coordinates. Because this simulation is defined in the *r*-*z* plane, the Laplacian uses a cylindrical geometry involving a 1/r term in the derivative with respect to *r*. The simulation is performed in an axisymmetric 2-D (*r*-*z*) domain. The actual thruster dimensions are reduced by the `SHRINK_FACTOR` variable in the input file (default 200). Correspondingly the physical parameters such as electric fields, magnetic fields, and particle densities are scaled by the shrink factor to maintain consistent physical effects (e.g. Larmor radius, Knudsen number).

The plasma is represented by macro-particles which are moved via the Boris pusher in cylindrical coordinates. Various types of elastic and inelastic particle collisions are calculated with the computational engine's Monte Carlo package. In this simulation the propellant xenon neutrals are tracked as kinetic particles and undergo collisions with electrons. The simulation employs variable-weight particle splitting and self-combination via `NullInteraction` blocks to help maintain good particle resolution over orders-of-magnitude variations in density across the domain.

This input file contains an imported magnetic field. The external magnetic field file is in units of Gauss, and is converted into Teslas when imported by VSim.

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 the box labeled "Dump at Time Zero" so that the initial electric potential may be plotted.
- To run the file, click on the *Run* button in the upper right corner of the window. You will see the output of the run in the right pane. The run has completed when you see the output, "Engine completed successfully." A snapshot of the simulation run completion is shown in Fig. 6.66.

The default number of time-steps for this simulation is 5,000, but to approach steady-state, approximately 500,000 time-steps are required. The *Visualizing the Results* section provides a review of the results at 5,000 time-steps, while the *Further Experiments* section is a review of the results after 500,000 time-steps.

Analyzing the Results

If the electron density is desired the analysis script *computePtclNumDensity.py* may be used.

- In the leftmost panel, click the **Analyze** button. Select *computePtclNumDensity.py* from the list of analyzers, then click *Open* at the top of the Analysis Controls pane.
- Enter "electrons" into the *speciesName* field.
- Click the *Analyze* button near the upper right of the Analysis Results pane.
- Repeat with other particle species if desired ("XeIons", "XeNeutrals")

The analysis results are now viewable in the Visualize window, as shown in the following section.

Visualizing the Results

After performing the above actions, continue as follows:

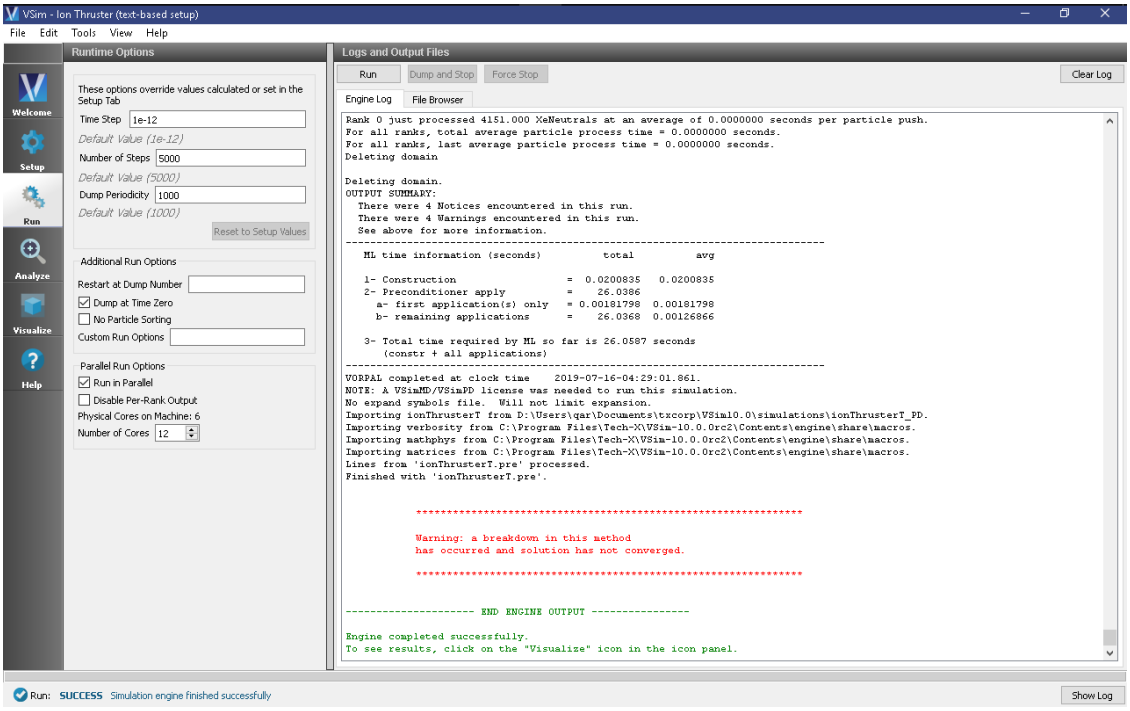


Fig. 6.66: The Run Window at the end of execution.

- Proceed to the Visualize window by clicking the **Visualize** button in the leftmost panel.
- In the top of the *Visualization Controls* pane, click on the *Add a Data View* button then click on *Field Analysis*. This will open a new *Field Analysis* tab next to the existing *Data Overview* tab.
- In the *Field* dropdown menu, select *phi*. A pseudocolor plot of the potential with a radial lineout performed should be displayed as shown in Fig. 6.67.
- To plot the axial potential profile, in the *Lineout Settings* section, select the *Horizontal* tab, change the intercept to `0.00005`, and click *Perform Lineout* to plot the axial accelerating potential as shown in Fig. 6.67. If desired, select the *Advanced* tab to choose arbitrary start and end points for the lineout.
- In the top of the *Visualization Controls* pane, click on the *Add a Data View* button then click on *Phase Space*. This will open a new *Phase Space* tab next to the *Field Analysis* tab.
- In the *Base Variable* dropdown menu, select *electrons*.
- To maintain the same z-r convention as the previous electric potential plot, in the *X-axis* dropdown menu select *electrons_z* and in the *Y-axis* dropdown menu select *electrons_r*.
- Near the bottom of the *Visualization Controls* pane click *DRAW* and at the bottom of the *Visualization Results* pane move the *Dump* slider to the right to dump 5. The z-r phase space should be visible as shown in Fig. 6.68.

Recall that the electron number density distribution was calculated in *Analyzing the Results*. Plot the results of this analyzer as follows:

- In the top of the *Visualization Controls* pane, switch to the *Data Overview* tab.
- In the *Variables* section, expand *Scalar Data*.
- Select *electronDensity*. A plot of the electron number density distribution should be displayed, though due to the large variation in densities, only a small portion of the domain will appear to be non-zero
- At the bottom of the variables section of the Visualization Controls pane, select the *Log Scale Color* checkbox.

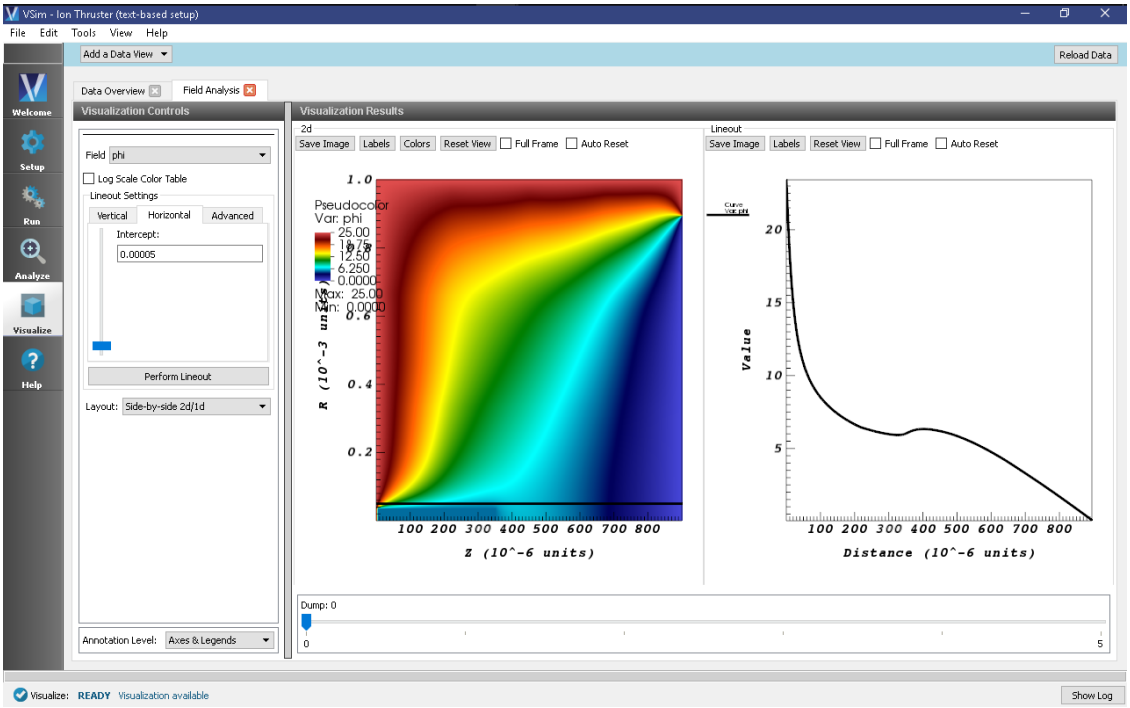


Fig. 6.67: Visualization of the *Field Analysis* result for the electric potential inside the ion thruster discharge chamber at timestep 0. Move the slider to the right with your mouse to view advanced timesteps.

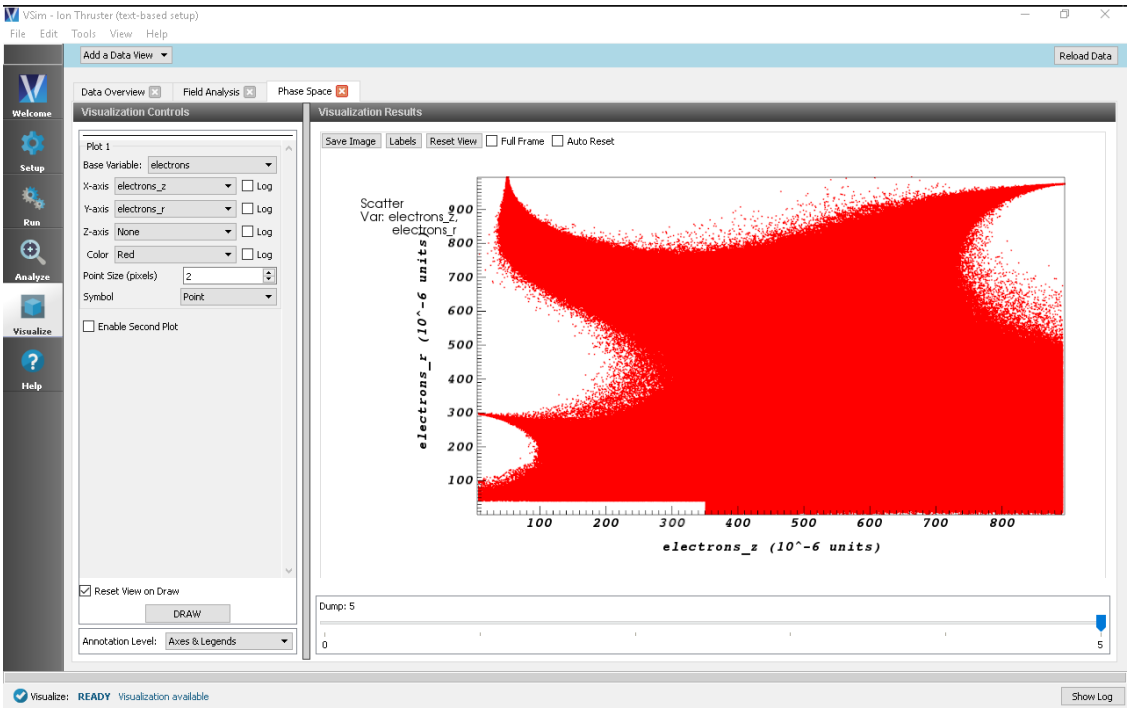


Fig. 6.68: Electron phase-space distribution results after 5,000 steps.

- At the top of the Visualization Results pane click the *Colors* button, and in the resulting dialog set the limits to a minimum of 1e16 and maximum of 1e22, or experiment with limits as desired. The electron density on a logarithmic color scale should now displayed as shown in Fig. 6.69.

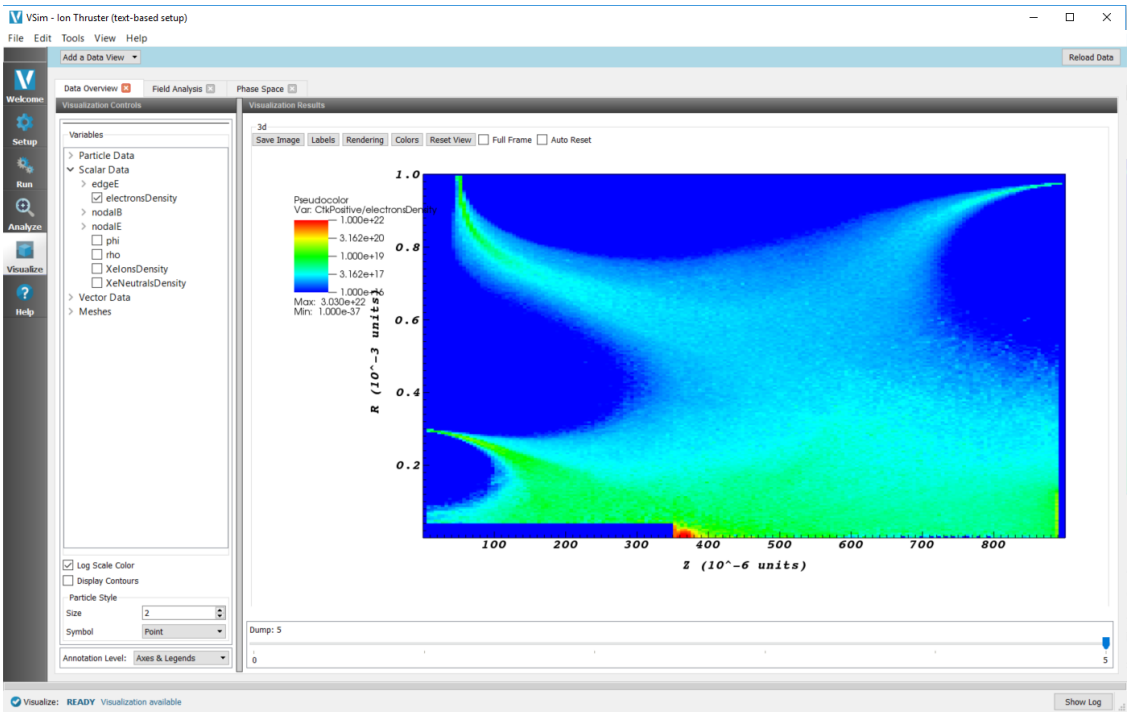


Fig. 6.69: Electron number density distribution results inside the discharge chamber after 5,000 steps.

Further Experiments

Return to the Run window by clicking on the **Run** button in the leftmost panel, change *Number of Steps* to 500,000. To reduce the number of output files, change the *Dump Periodicity* to 5000 and then click the *Run* button at the top of the Logs and Output Files pane. A high-performance computing cluster is recommended for this run, which will require approximately 6 days running on 64 cores. When the run has completed, take the following steps.

- Plot the potential at the final data dump similar to the steps taken in *Visualizing the Results*.
- In the Visualization Results pane, in the *2d* section, click the *Colors* button
- Set the minimum to 0 and the maximum to 25 (Volts). The resulting plot is shown in Fig. 6.70

It can be seen that the ions experience most of their acceleration in the sheath near the right-side boundary of the plasma chamber. Plot the electron and ion densities by taking the following steps:

- Following once again the steps taken in *Visualizing the Results*, run the *computePtclNumDensity.py* analyzer on both *electrons* and *XeIons*.
- Plot the electrons density using the color log scale and the same limits as previous, as shown in Fig. 6.71
- Plot the ion density using the color log scale with a minimum of 1e18 and a maximum of 1e23 to get the image shown in Fig. 6.72 or experiment with the limits as desired.

It can be seen from Fig. 6.71 and Fig. 6.72 that the electrons are confined by the magnetic field lines while the much heavier ions are not, allowing a more uniform acceleration of ions out the right side of the chamber, resulting in thrust.

Plot the electron and ion macroparticle positions with the following steps:

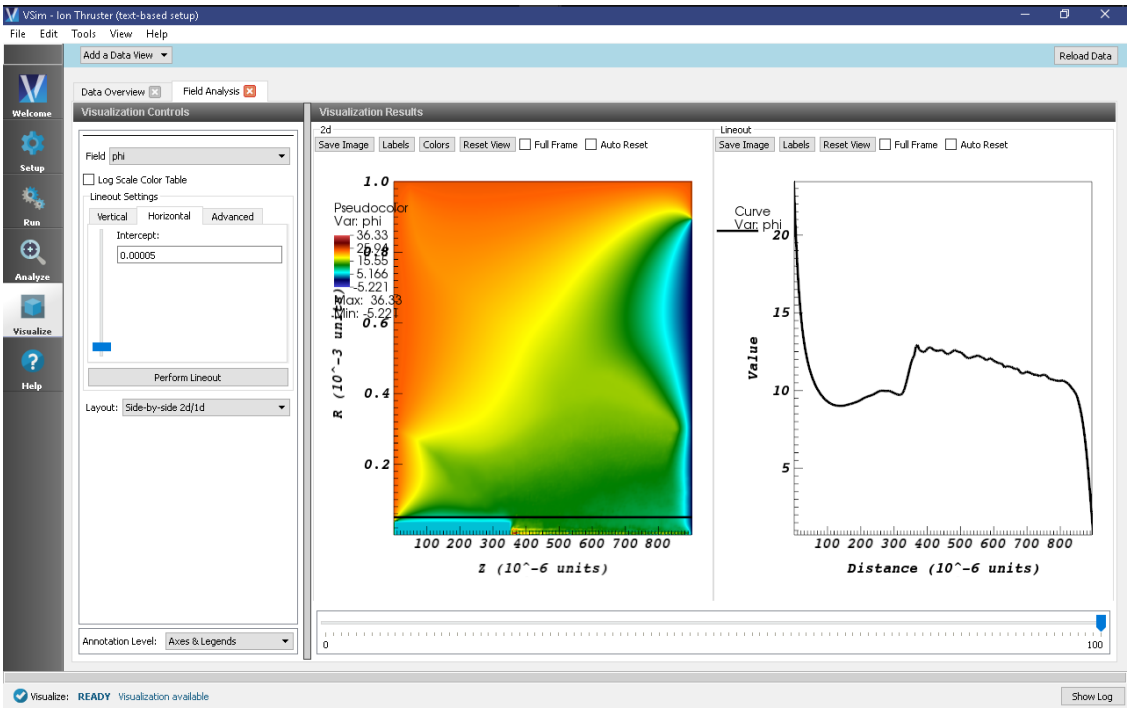


Fig. 6.70: Electric potential of the plasma inside the ion thruster discharge chamber after 500,000 time-steps.

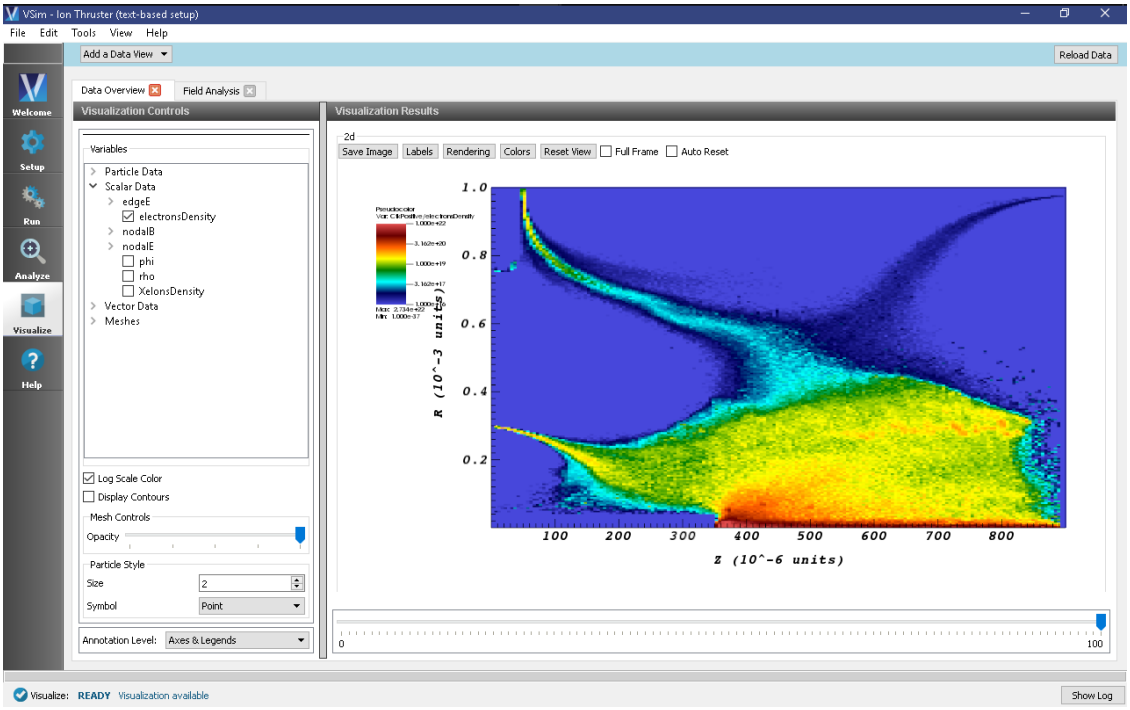


Fig. 6.71: Electron number density distribution results inside the discharge chamber after 500,000 steps.

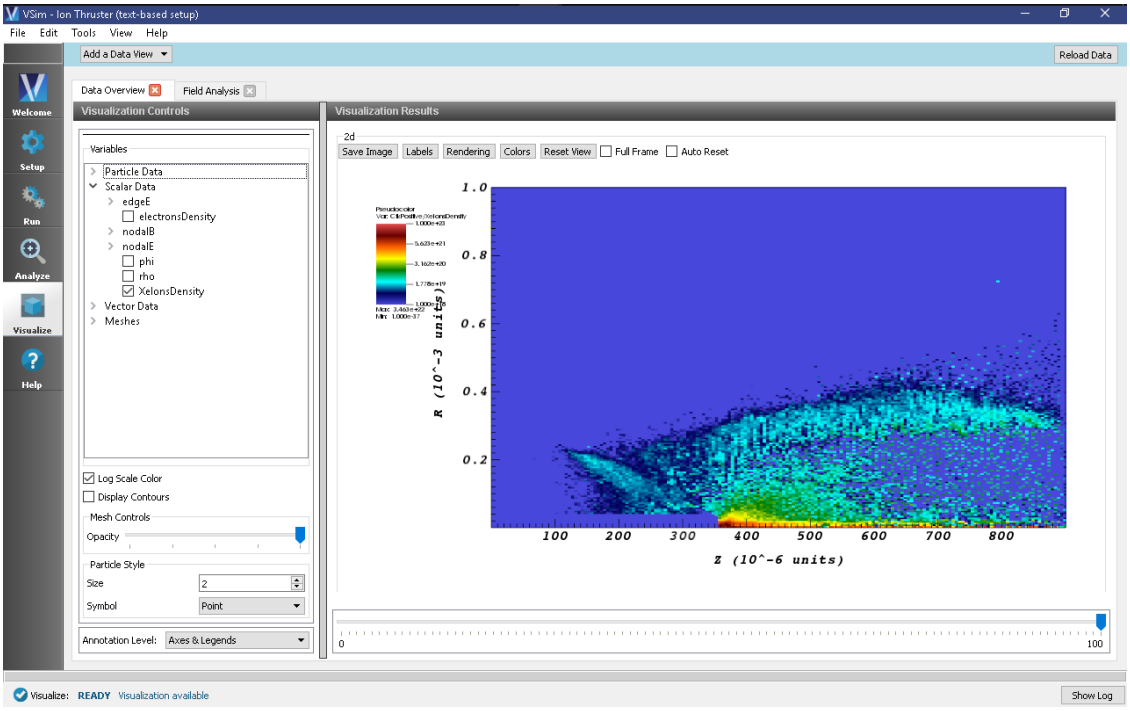


Fig. 6.72: Ion number density distribution results inside the discharge chamber after 500,000 steps.

- In the top of the *Visualization Controls* pane, open a new *Phase Space* tab under the *Add a Data View* button.
- Under *Plot 1* click the *Base Variable* drop-down menu and select *electrons*
- Change *X-axis* to *electrons_z* and *Y-axis* to *electrons_r*, change *Point Size* to `1`, and at the bottom of the Visualization Controls pane, click *DRAW* to see the electron macro-particle positions.
- Check the *Enable Second Plot* button.
- Under *Plot 2* change *Base Variable* to *XeIons*, change *X-axis* to *XeIons_z* and *Y-axis* to *XeIons_r*, change *Point Size* to `1`, and click *DRAW* once again to see the electron and singly-ionized xenon macro-particle positions.
- Check the *Enable Third Plot* button.
- Under *Plot 3* change *Base Variable* to *XeDblIons*, change *X-axis* to *XeDblIons_z* and *Y-axis* to *XeDblIons_r*, change *Point Size* to `3`, and click *DRAW* once again. The ion and electron positions should be displayed as shown in Fig. 6.73.

The electrons appear well confined by the 3-ring magnetic circuit arrangement, and move along the magnetic cusp regions formed between the magnets. Most of the electrons are lost to the discharge chamber walls through the magnetic cusps and are absorbed at the walls in 3 small areas. Only a few electrons are able to cross the strong magnetic field lines and reach the top wall between the cusps.

Singly and doubly ionized xenon are generated inside the discharge chamber through ionizing collisions of electrons with xenon neutrals. Only electrons with energies above the ionization thresholds (12.1 eV for the first ionization level and 21.25 eV for the second) can ionize neutrals.

This input file can be modified to test different design parameters covering a range of anode voltages, xenon flow rates, and electron emission currents, to allow study of high-to-low power and high-to-low throttle levels.

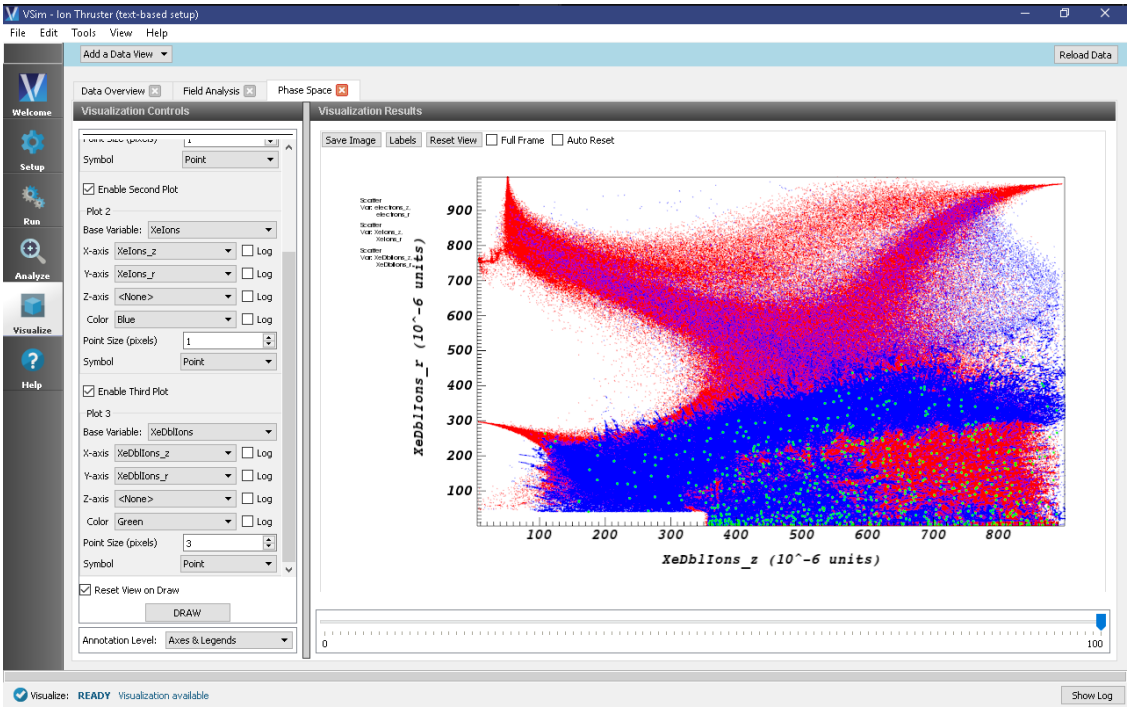


Fig. 6.73: Electron and ion phase-space distribution results after 500,000 steps.

6.7.2 Satellite Surface Charging (satelliteSurfaceChargeT.pre)

Keywords:

`electrostatics, surface charges`

Problem description

Satellites and other spacecraft operating in the space environment often suffer arcing and breakdown problems due to surface charging. Charged particles buildup on the spacecraft surfaces (such as solar panel arrays and other components) leading to localized arcing/breakdown discharges that can critically fail a component or the entire unit. This problem is made worse as the demand for high power space missions in both satellite and deep-space applications rises. These high-power spacecraft are outfitted with high-voltage solar panels. These panels minimize the overall payload requirements and offer other advantages over more massive, low-voltage arrays. However, they are also more vulnerable to surface charge related arcing. It therefore becomes important to predict the surface charge buildup on spacecraft bodies operating in different space environments, where the ion sources may be natural solar wind or human-made space plasma resulting from electric thruster plasma plumes.

This example demonstrates a satellite body operating in the solar wind environment where the space plasma consists of ions and electrons. The simulation box is set up with dimensions of 15 m x 30 m x 15 m. The satellite system is placed in the middle of the domain. It has a 3 m radius x 5 m long cylindrical central unit connected to solar panels at either end. Each solar panel has a total span length of 7.8 m and a width of 5 m. The satellite central unit has a 5-volt equipotential circular body with radius 2 m and length 3 m. The satellite system is treated as a conductor floating in free space. The system domain boundaries are assumed to have zero perpendicular electric field, i.e. Neumann boundary conditions. The solar wind plasma is introduced in the simulation domain from the positive z direction. The solar wind density is set to $1 \times 10^7 \mathrm{m}^{-3}$ with a temperature of 10 eV. The number of physical particles per macroparticle is set to 5000. Both electrons and ions are introduced from the source based on the solar wind density and temperature. To maintain plasma uniformity within finite bounds, the electron source rate is inflated slightly because

electrons are lighter and leave the system more quickly than do ions. At the same time, the positive ions are imbued with a lighter mass value to speed up the simulation. All simulation boundaries are set up to absorb particles. The charges collected in the satellite system are counted by emitting a heavy electron or heavy ion at the point where an electron or ion was absorbed. The heavy electron/ion is not a physical concept, it is a computational trick whereby any charged particle striking the satellite gets converted to a new species, one with equivalent charge but drastically swollen mass (1 kg in this case) and suppressed energy (suppressed by a factor of 10 billion). In this way, the heavy particles do not propagate from their point of origin, effectively sticking to the satellite surface. To limit the number of macro heavy particles tracked we apply a particle combining algorithm which limits the number of macro particle per cell to one. The collected electron and ion currents on the satellite surfaces are output as histories.

This simulation can be performed with a VSimPD license.

Opening the Simulation

The satellite surface charging example is accessed from within VSimComposer by the following actions:

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim for Plasma Discharges* option.
- Expand the *Spacecraft (text-based setup)* option.
- Select "Satellite Surface Charging (text-based setup)" 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 basic variables of this problem should now be alterable via the text boxes in the left pane of the Setup Window, as shown in Fig. 6.74.

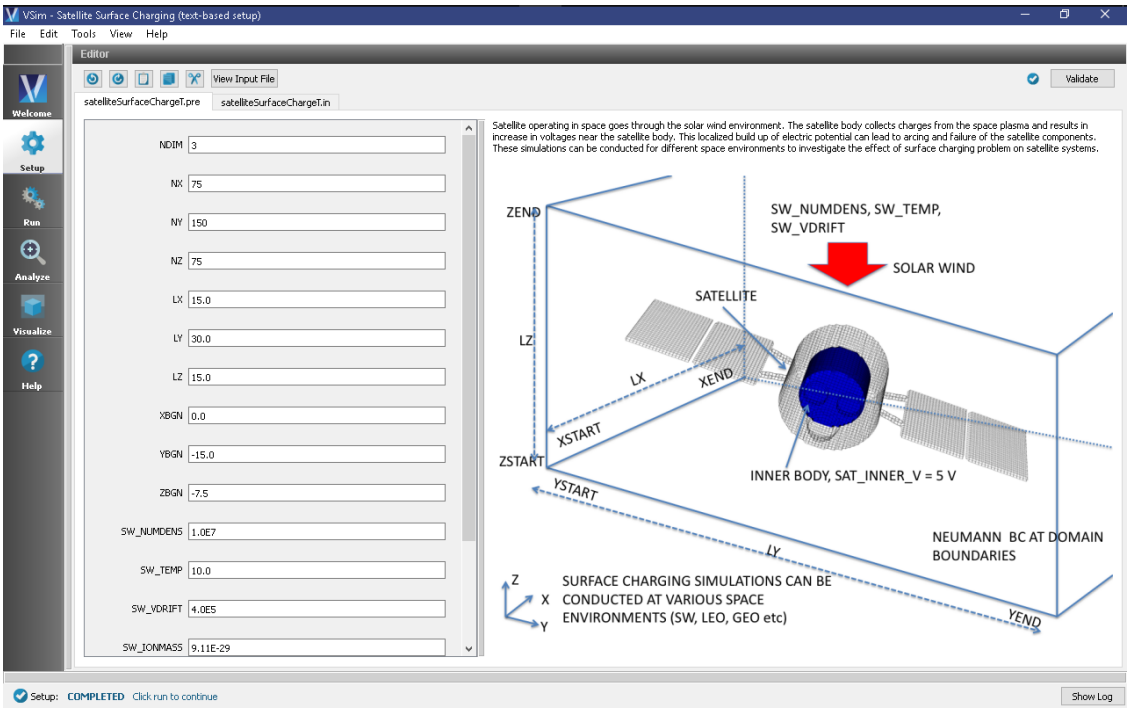


Fig. 6.74: Setup Window for the Satellite Surface Charging example.

Input File Features

The input file allows a choice of space environment parameters (number density, plasma temperature, drift speed), satellite body voltage, simulation domain size, and resolution (number of cells in each direction).

The self-consistent electric field is solved from Poisson's equation by the electrostatic solver. The far-field space boundaries are handled with Neumann boundary conditions. The satellite inner body is set up with an equipotential boundary. The surface charges collected on the satellite system make the satellite body float at a slightly higher voltage than the space plasma.

This is a large domain, 3-D problem, and its resolution is aided by several numerical methods. The plasma is represented by macro-particles which are moved according to the Boris pusher. Variable weight particle treatment is employed on all simulated species, reducing the overall number of macro-particles in the computation. Additionally, null interactions are considered as part of the Monte Carlo analysis to limit the number of macro-particles per cell; macro-particles are eliminated in overcrowded cells by means of inelastic combination.

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.
- To run the file, click on the *Run* button in the upper left corner of the right panel. 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 is shown in Fig. 6.75.

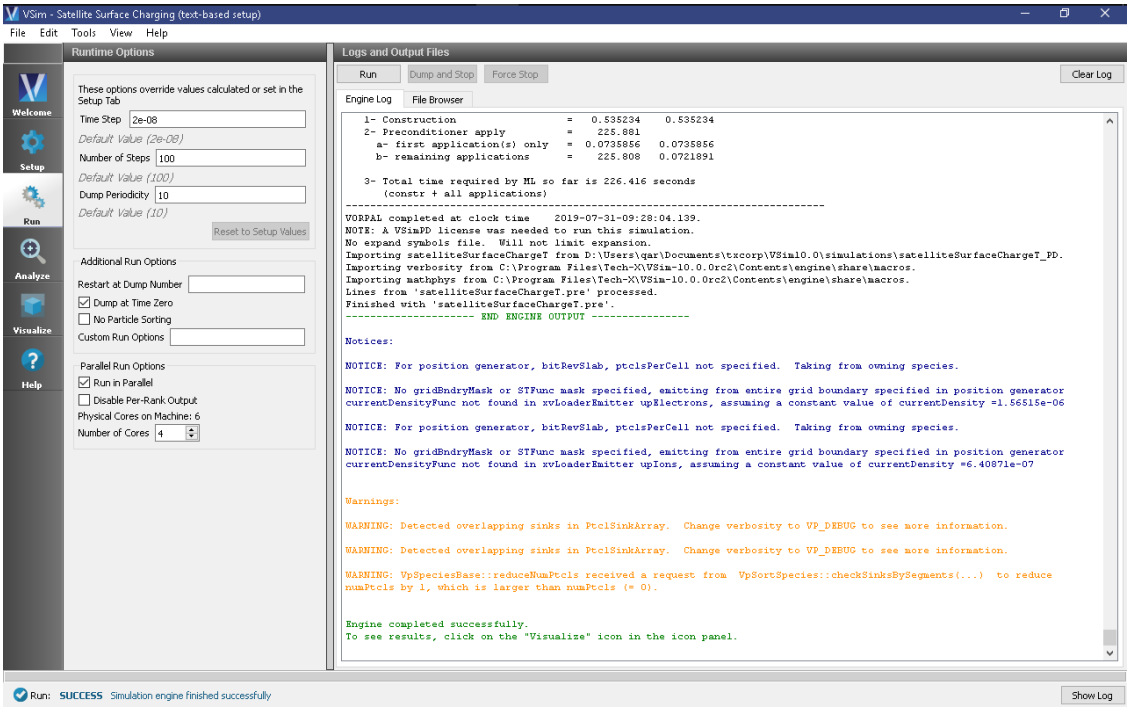


Fig. 6.75: The Run Window at the end of execution.

Analyzing the Results

If the electron density is desired, then proceed as follows:

- In the leftmost panel, click the *Analyze* button and then select *computePtclNumDensity.py* from the list of analyzers, then click *Open* at the top of the Analysis Controls pane.
- Enter the following parameters in the appropriate fields:
  - simulationName = satelliteSurfaceChargeT
  - speciesName = solarElectrons
  - avgNxN = 1
  - iterateAvg = 1
- Click the *Analyze* button in the upper right corner of the window.

See Fig. 6.76.

The resulting data will be visualizable as *solarElectronsDensity* under the *Scalar Data* menu in the *Visualize* Tab. The density of solarIons can be calculated in the same way by substituting that species name in place of *solarElectrons*.

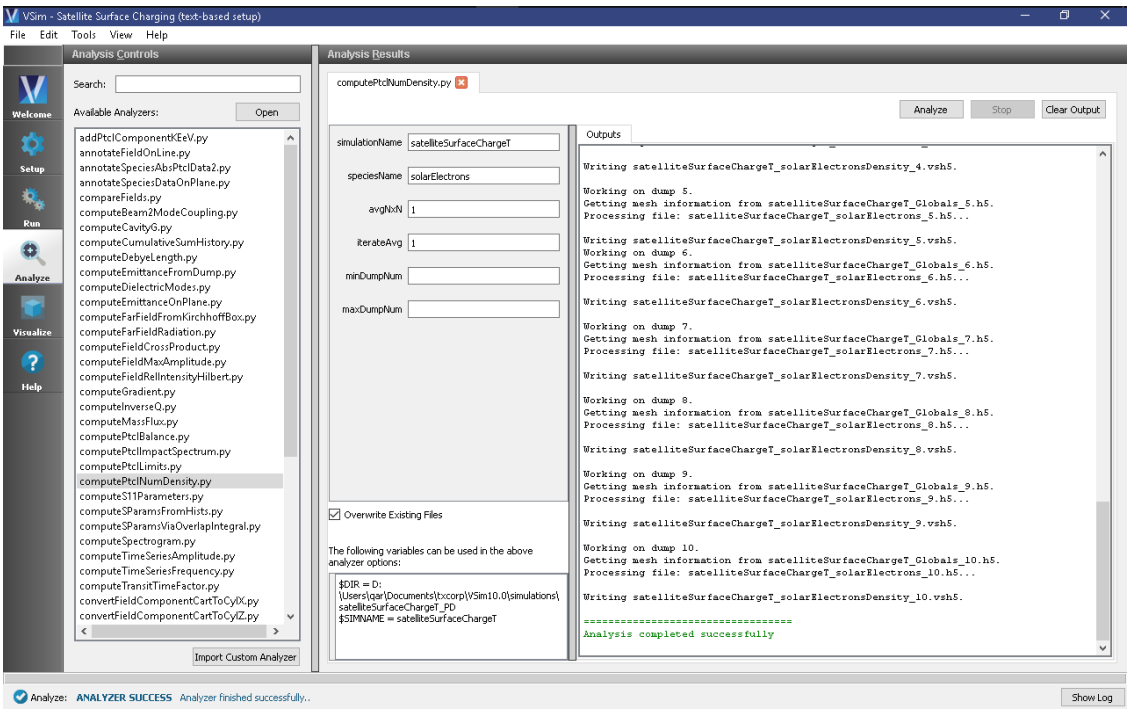


Fig. 6.76: The Run Window at the end of execution.

Visualizing the results

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

To visualize the satellite geometry with electrons Fig. 6.77, proceed as follows:

- Expand *Particle Data*.
- Expand *heavyElectrons*.
- Select “heavyElectrons” in red.
- Expand *solarElectrons*.

- Select “solarElectrons” in green.
- Expand *Geometries*.
- Select “poly_surface (Satellite)”.
- Move the Dump slider to dump 7.

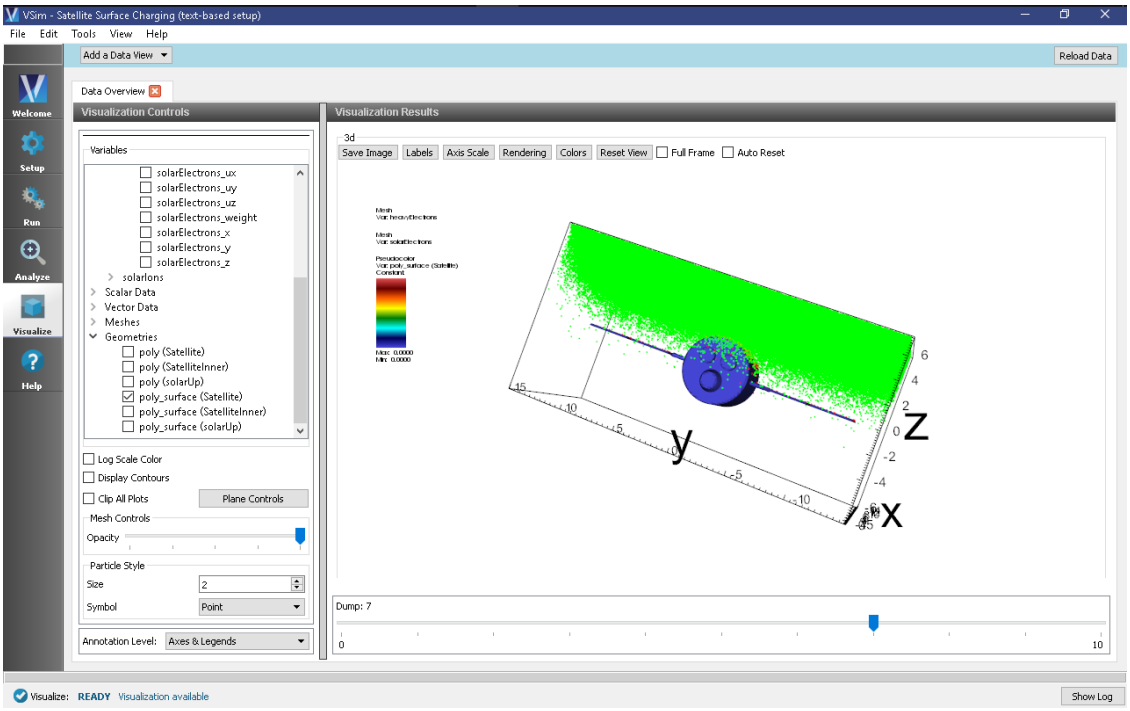


Fig. 6.77: Visualization plot of satellite system with solar wind electrons in green and the electrons that stick to the satellite surface in red.

Here are some things to try:

- Under *Data Overview* you can access plots of the electric field, charge density (rho), and electric potential (phi). Select the *Display Contours* check box for viewing these.
- To view the phase space distribution for the electrons and ions, click on the *Data View* drop down menu and select *Phase Space*. Click the *Draw* button to generate a plot.
- Also from the *Data View* menu select *History* to observe the satellite currents and the time history results for the number of macro-particles broken down by species.

To generate Fig. 6.78, that shows the satellite system with the inner equipotential cylindrical body, proceed as follows:

- In the *Data View* pane on the left side select “Data Overview” from the drop-down menu.
- Expand *Geometries*
- Select “poly (Satellite)”
- Select “poly_surface (SatelliteInner)”.

The phase-space distribution of the positive ions (solarIons species) surrounding the satellite system is shown in Fig. 6.79 which is obtained after running the simulation for 500 time steps. Solar wind plasma enters into the simulation system from the top z boundary, i.e. above the satellite body.

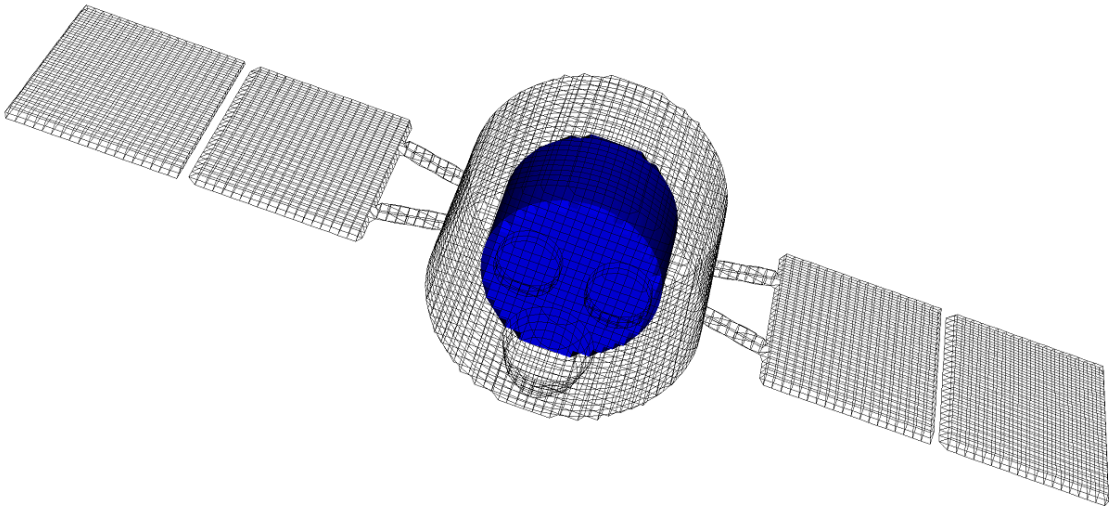


Fig. 6.78: Visualization of the inner body inside the satellite system.

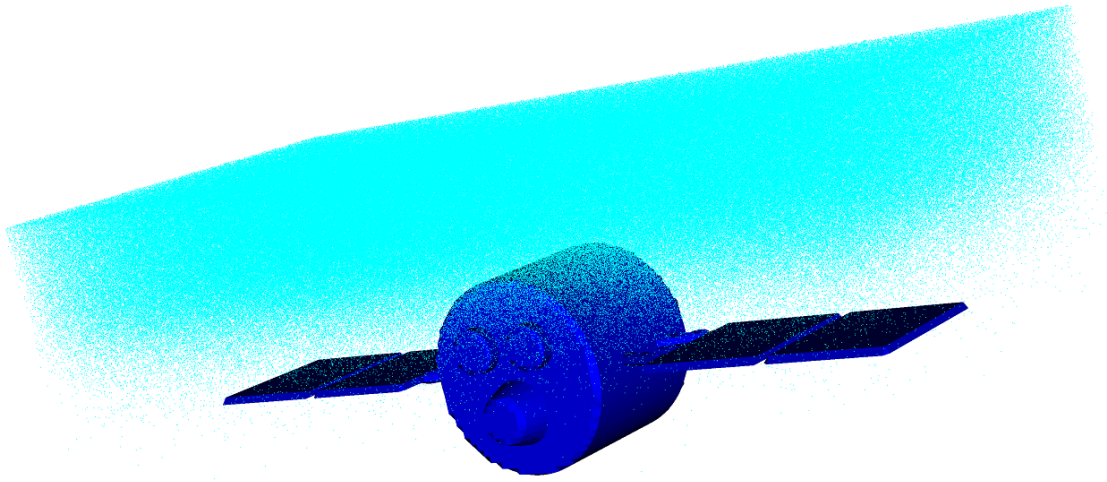


Fig. 6.79: Visualization of the satellite system with solar ions.

Surface charge accumulation on the satellite body after 93,000 time steps is shown in Fig. 6.80. The red dots indicate electrons and the green dots ions. The surface charges on the satellite body can be viewed in VSimComposer by turning on Particle Data –> heavyElectrons and ParticleData -> heavyIons under the Data Overview pane.

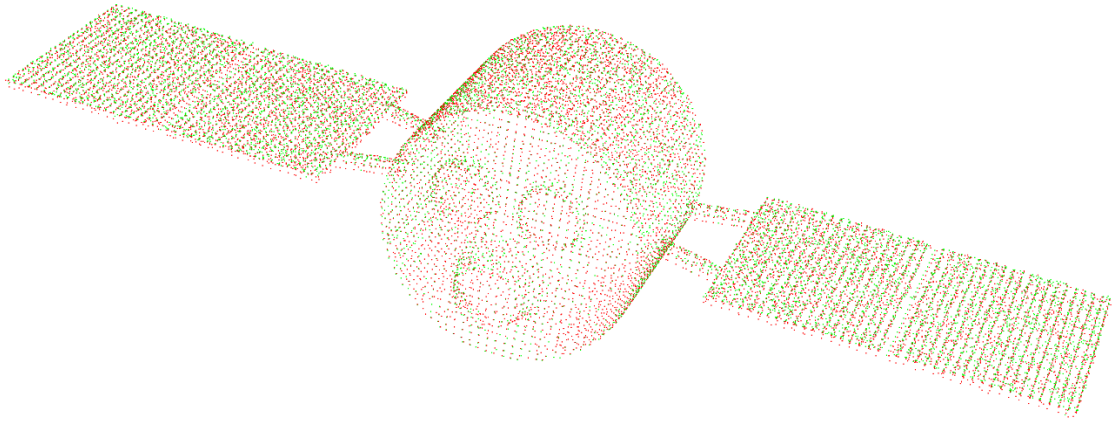


Fig. 6.80: Visualization of surface charge buildup on the satellite system after 93,000 time steps.

The charge density built-up on the satellite system is shown in Fig. 6.81 after running for 93,000 time steps. To view the charge density in the simulation domain, turn on the Scalar Data -> rho field in the *Data Overview* pane. In this figure the satellite body is also included by turning on the Geometries -> poly_surface(Satellite) option in the *Data Overview* pane. The charge density appears net positive in most regions of the solar panels.

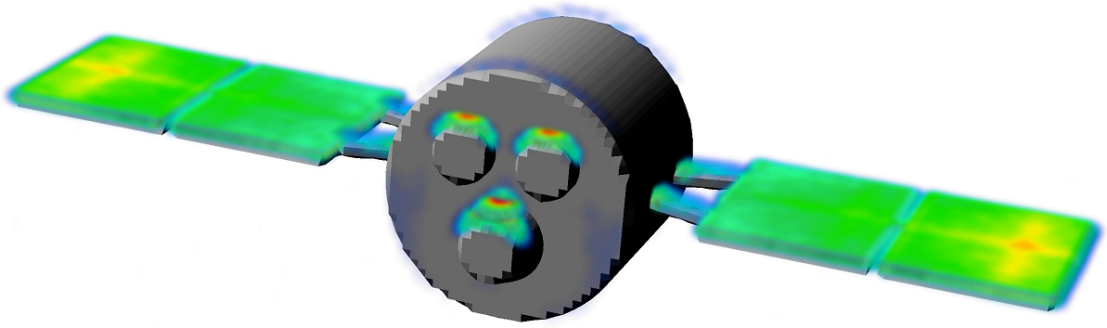


Fig. 6.81: Visualization of the charge density on the satellite system.

To view the electrostatic potential, turn on phi under the Scalar Data in the *Data Overview* pane. The electrostatic potential of the satellite system simulated is shown in Fig. 6.82 after running for 93,000 time steps. The electrostatic potential is plotted in X-Y-Z with domain clipping in the X and Z directions. The bulk of the plasma potential in the space region is close to 0 volts (blue contours). The surface charge built-up on the solar panels raises the surface potential by up to 4 to 5 volts above the bulk space plasma.

The magnitude of the electric field distribution on the satellite surface after 93,000 time steps is shown in Fig. 6.83. The peak of the distribution coincides with regions on the solar array where there is net positive charge buildup. The magnitude of the electric field was computed using the *Expressions* function in the Visit interface. Should you wish to get to those calculations, right-click on the plot and select *Open GUI*. (You must have the *Enable VisIt context menu* box check-marked in Visualization Options. Go to Tools -> Settings -> Visualization Options, to enable this.) This will launch the VisIt control panel. From there, go to Controls -> Expressions. You can select any plottable variable to view its mathematical definition.

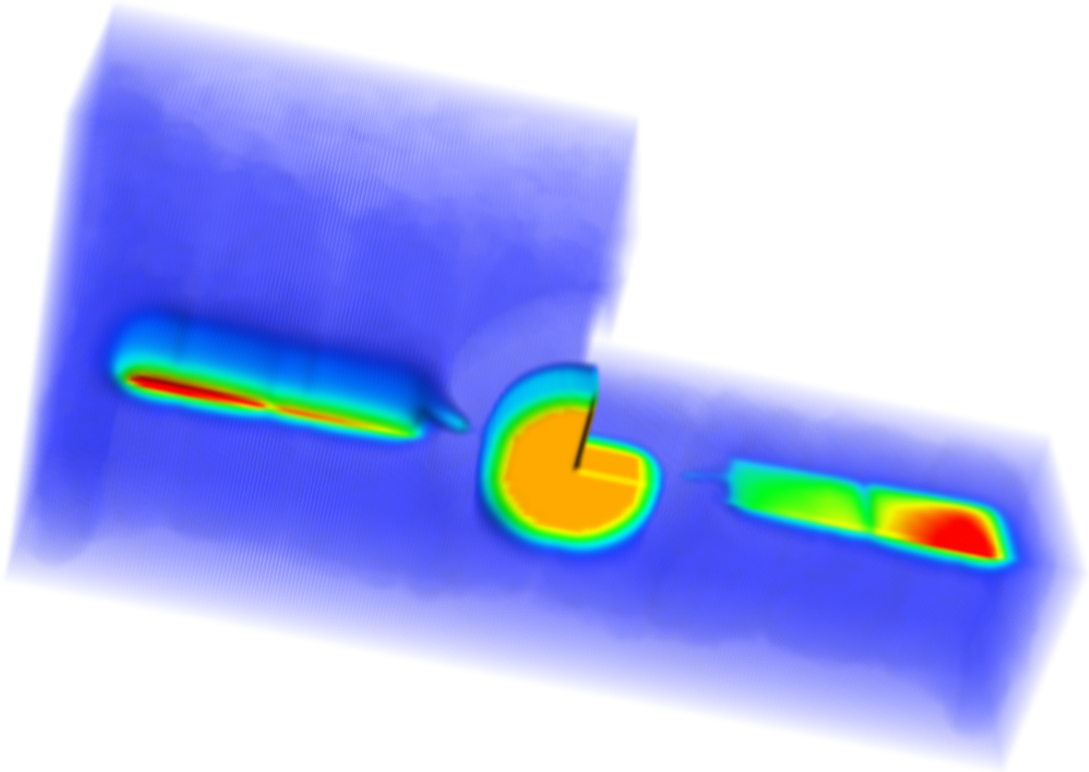


Fig. 6.82: Visualization of the electric potential surrounding the satellite system.

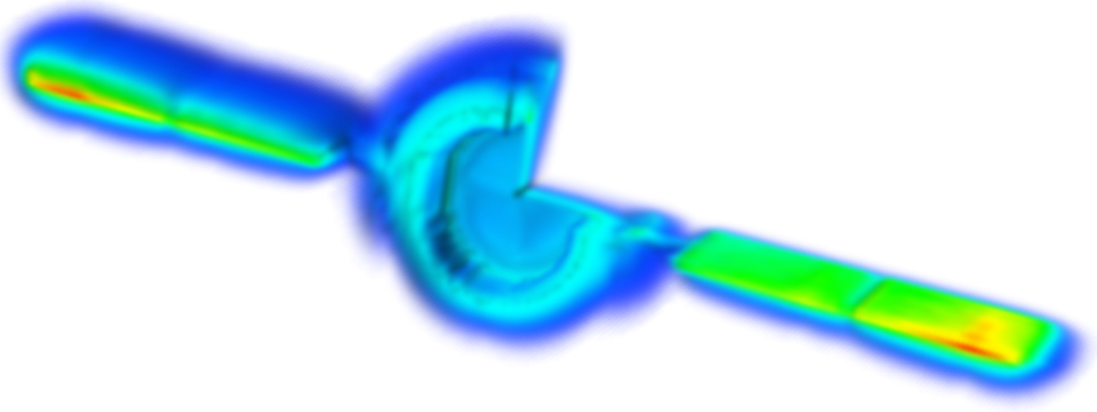


Fig. 6.83: Visualization of the magnitude of electric field surrounding the satellite system.

Further Experiments

The geometry and background space plasma parameters of this input file can be modified to test satellite inner body voltages and satellite surface charge collection in a variety of different space environments.

VSim allows the use of "open" boundary conditions to represent the far-field boundaries in the space environment.

6.8 Sputtering

6.8.1 Ion Beam Sputtering (ionBeamSputtering.sdf)

Keywords:

`sputtering, ion beam, electrostatics`

Problem Description

In this simulation, a 450 eV beam of positively charged argon ions strikes a copper plate (cathode) at -25 volts with respect to the walls on the upper and lower x boundaries of the simulation. A Neumann boundary condition where the slope of the electric potential is zero is placed on the upper and lower y walls of the simulation.

The argon ion beam travels through a vacuum and sputter off neutral copper atoms from the cathode.

This demonstrates how to set up a sputtering interaction through the Visual Setup in VSimComposer.

This simulation can be performed with a VSimPD license.

Opening the Simulation

The Ion Beam Sputtering example can be accessed from within VSimComposer by the following actions:

- Select the *New* → *From Example...* menu item in the *File* menu.
- In the resulting *Examples* window expand the *VSim for Plasma Discharges* option.
- Expand the *Sputtering* option.
- Select "Ion Beam Sputtering" 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.

You can expand the tree elements and navigate through the various properties, making any changes you desire, but the run here is using the setup as is. The right pane shows a 3D view of the geometry, if any, as well as the grid, if actively shown. The setup window, with the *Particle Dynamics* → *Kinetic Particles* → *ArgonIons*, and *neutralCopper* elements expanded is shown in Fig. 6.84.

Simulation Properties

The Ion Beam Sputtering example includes some constants for easy adjustment of simulation properties:

- **`BEAM_RADIUS`**: sets the radius of the argon ion beam
- **`BEAM_ENERGY`**: sets the energy (and speed) of the ion beam
- **`BEAM_CURRENT`**: sets the emitted current. *Note*: the total current emitted will be less than this value if a mask is applied to the particle emitter (which is the case in this simulation).

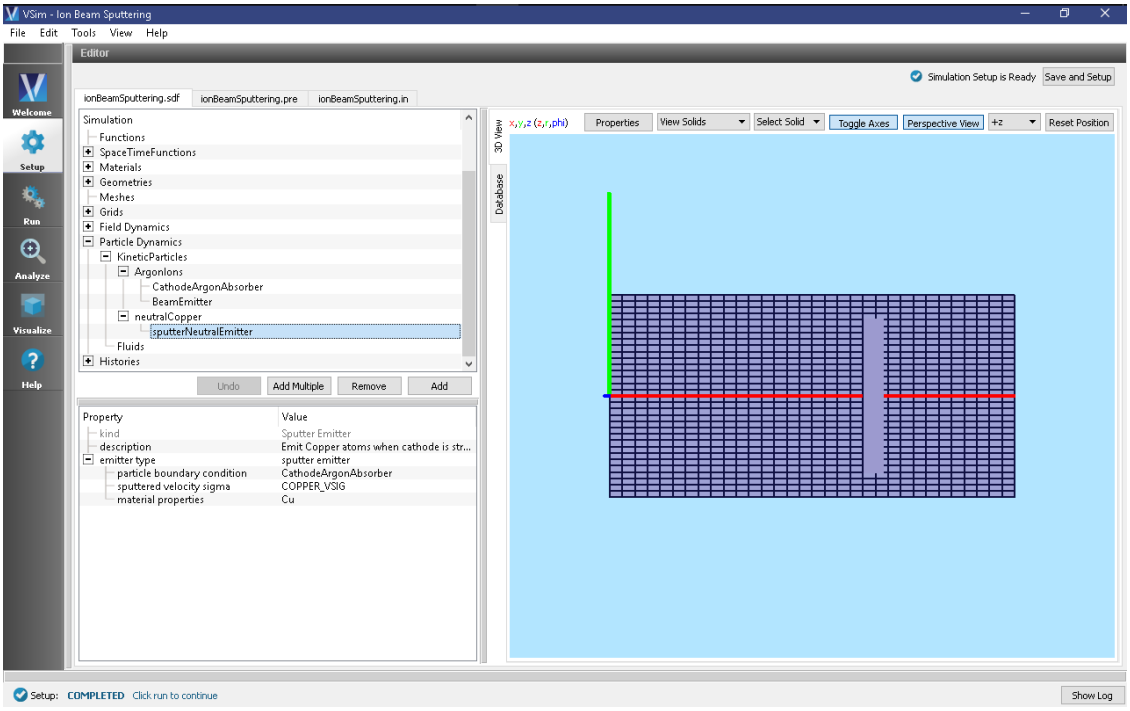


Fig. 6.84: Setup window for the Ion Beam Sputtering example.

- **`CATHODE_VOLTAGE`**: the negative bias for the cathode.

Running the Simulation

After performing the above actions, continue as follows:

successfully." This is shown in Fig. 6.85.

Analyzing the Results

This simulation includes particle species which enter the simulation at different dump times. When this is the case, the *createMissingPtclDumps.py* analyzer must be run. If no particles of a particular species exist in a simulation when data is written (dumped), then there will be no dump files for that particular species. This will cause particle data to not be lined up to the same dump in the *Visualize Tab*.

To fill in these missing files, navigate to the *Analyze* Tab. From the *Available Analyzers* list, select *createMissingPtclDumps.py* from the list and click "Open". Ensure the following is entered into each field:

- **`simulationName:`** "ionBeamSputtering"
- **`speciesName:`** "neutralCopper"

Click *Analyze* in upper right corner of the window. When the analysis is finished, you should see a window similar to Fig. 6.8.1.

Visualizing the results

After performing the above actions, continue as follows:

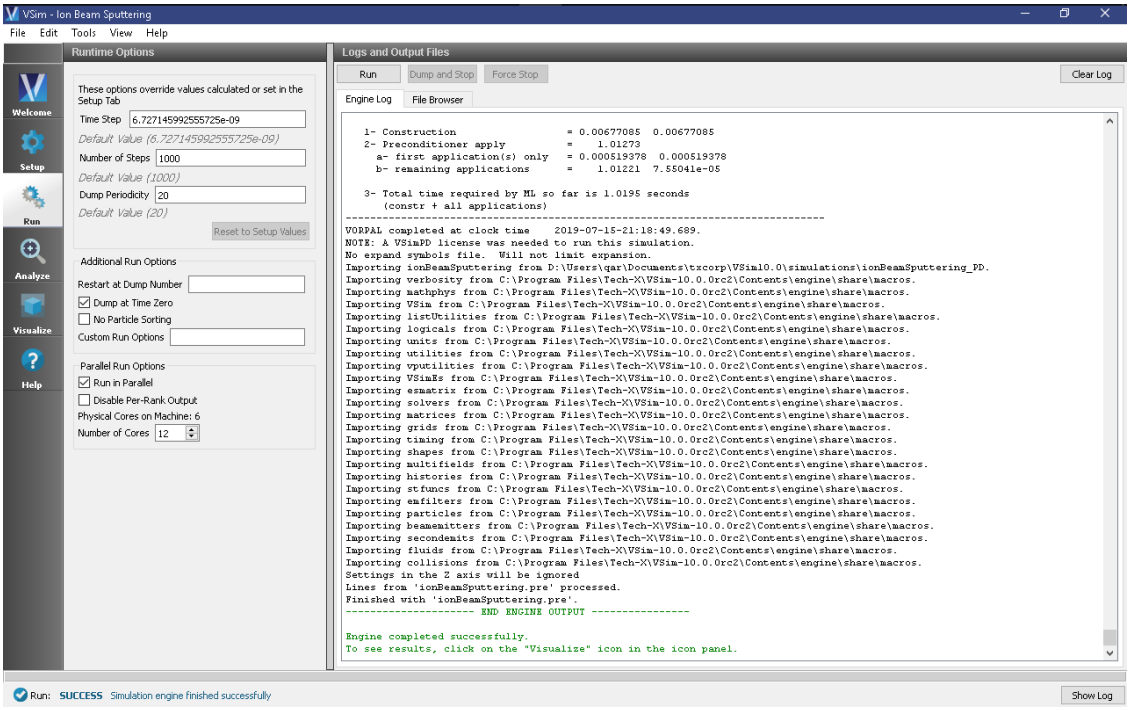
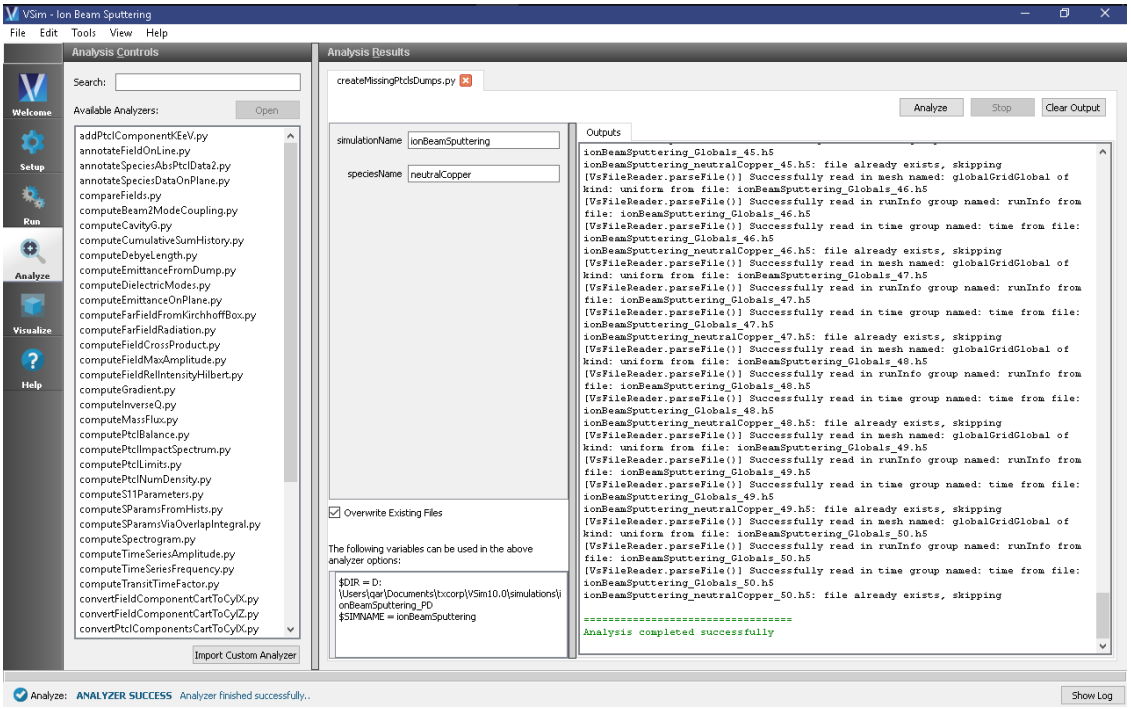


Fig. 6.85: The Run window at the end of execution.



1. Proceed to the *Visualize* window by pressing the Visualize button in the left column of buttons. Be sure to press the "Reload Data" button at the bottom of the window if you have previously navigated to the *Visualize* window.
2. Expand "Particle Data" then expand "ArgonIons" and check the red "ArgonIons" box.
3. Expand "neutralCopper" and check the green "neutralCopper" box.
4. Expand "Scalar Data" and check "Phi". Then check the "Display Contours" box.
5. Expand "Geometries" and select poly (cathode).

Scrolling through the dumps, you should see the argon beam expand as it travels towards the cathode, as in Fig. 6.86. Because we ran the *createMissingPtclDumps.py* analyzer, copper atoms won't appear until after the argon beam strikes the cathode.

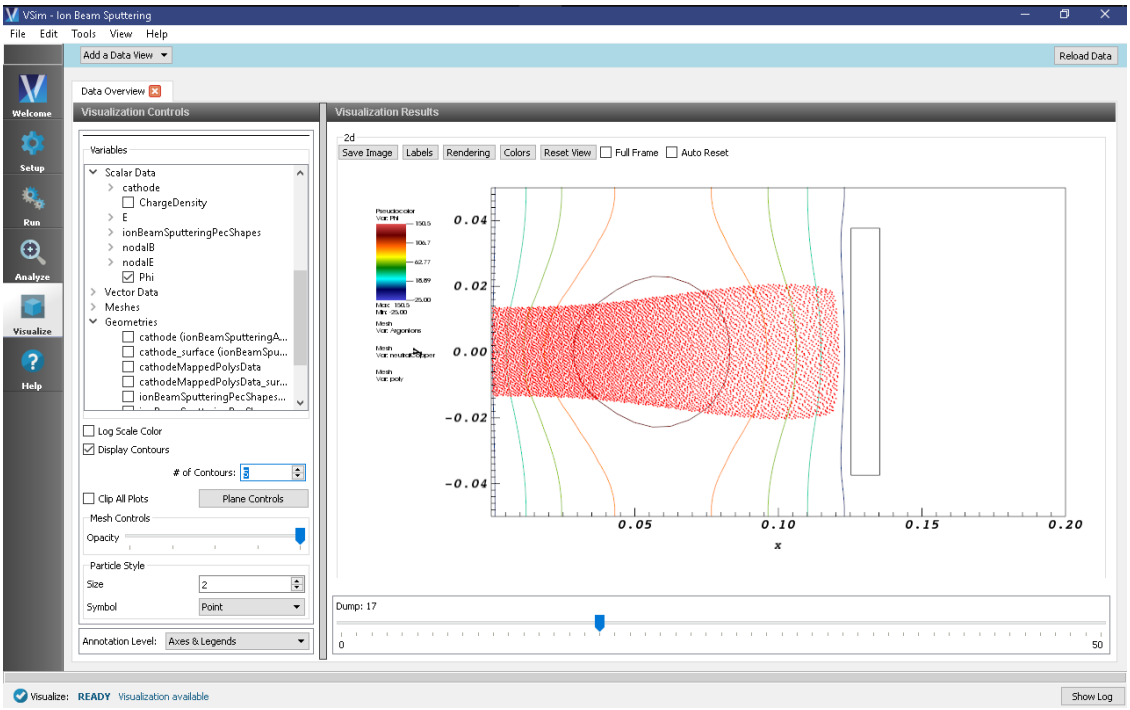


Fig. 6.86: Ion Beam Sputtering an instant before the argon beam strikes the wall.

Next, in the *Data View* drop down menu at the top left of the window, switch from "Data Overview" to "History". In *Graph 2*, change from "emittedArgonCurrent" to "<None>" to see the plots shown in Fig. 6.87

These plots show the energy deposited onto the cathode from the argon ion species as well as the number of neutral copper macro particles. From these plots its clear that the ion beam strikes the cathode after about 2.5 microseconds. A history for the total number of physical copper atoms is also available to be added to the simulation.

Further Experiments

1. Vary the *BEAM_ENERGY* constant, the species of ions in the beam, and the target material to see how the sputter yield changes in response (see the "numMacroCopper" history available in the *Visualize* Tab).

   To change the material of the target to see the effect on the sputtering yield, expand the *Particle Dynamics* element, then *neutralCopper*. Select the *sputterNeutralEmitter* and choose a new material from the "material properties" dropdown menu.

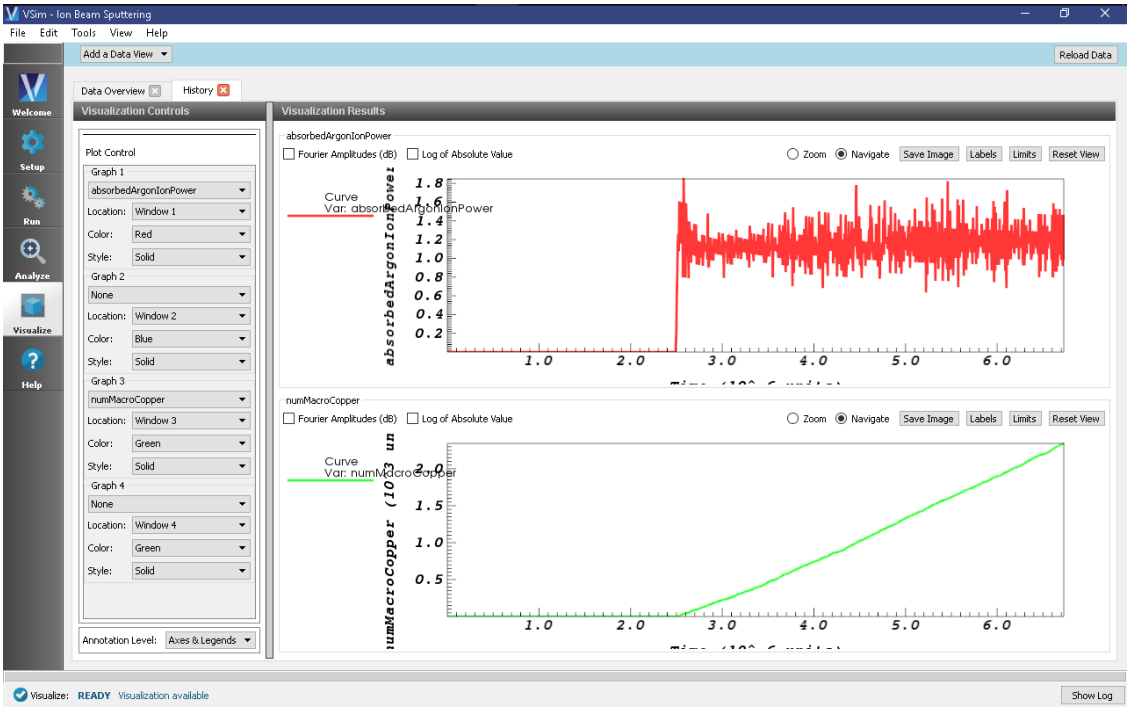


Fig. 6.87: Plots of the absorbed power and neutralCopper macro particle count.

Note: Savvy users may notice that the material of the cathode is also set in the *Geometries* element under *CSG → cathode*. This assignment of "PEC" sets the _electromagnetic_ properties of the geometry, not the particle properties.

---

2. Add electrons, secondary electron emission, an RF oscillating frequency on the cathode, and possibly an external magnetic field. Add a copperIons species and an electron impact ionization process to create copperIons from neutralCoppers. Add a second copper sputter emitter to the cathode that creates copper neutrals from copperIons to simulate self-sputtering.

3. Modify the beam current to account for the loss in current due to the mask. See the Negative Ion Beam example (*Negative Ion Beam (negativeIonBeam.sdf)*) for an example of how this is accomplished.

CHAPTER

SEVEN

TRADEMARKS AND LICENSING

- Vorpal™ © 1999-2002 University of Colorado. All rights reserved.
- Vorpal™ © 2002-2018 University of Colorado and Tech-X Corporation. All rights reserved.
- VSim™ except for Vorpal™ is © 2012-2018 Tech-X Corporation. All rights reserved.

For VSim™ licensing details please email sales@txcorp.com. All trademarks are the property of their respective owners. Redistribution of any VSim™ input files from the VSim™ installation or the VSim™ document set, including *VSim Installation*, *VSim Examples*, *VSim User Guide*, *VSim Reference*, and *VSim Customization*, is allowed provided that this Copyright statement is also included with the redistribution.

INDEX

A

B

C

D

E

F

G

H

I

K

L

M

N

O

P

R

S

T

U

V

W

X

Y