Second Harmonic Generation of a Gaussian Beam

Introduction

Laser systems are an important application area in modern electronics. There are several ways to generate a laser beam, but they all have one thing in common: The wavelength is determined by the stimulated emission, which depends on material parameters. It is especially difficult to find lasers that generate short wavelengths (for example, ultraviolet light). With nonlinear materials it is possible to generate harmonics that are an even multiple of the frequency of the laser light. With such materials it is straightforward to generate a laser beam with half the wavelength of the original beam. This model shows how to set up a second harmonic generation as a transient wave simulation using nonlinear material properties. A YAG ($\lambda = 1.06$ μm) laser beam is focused on a nonlinear crystal, so that the waist of the beam is inside the crystal.

Model Definition

When a laser beam propagates it travels as an approximate plane wave with a cross-section intensity of Gaussian shape. The waist of the beam is defined as the Gaussian function's characteristic length. A focused laser beam has a minimum waist, w_0, at a certain point along the direction of propagation. The solution of the time-dependent Maxwell's equations gives the following electric field (x-component):

$$\mathbf{E}(x,y,z) = E_{0x}\frac{w_0}{w(z)}e^{-(r/\omega(z))^2}\cos\left(\omega t - kz + \eta(z) - r^2\frac{k}{2R(z)}\right)\mathbf{e}_x ,$$

where

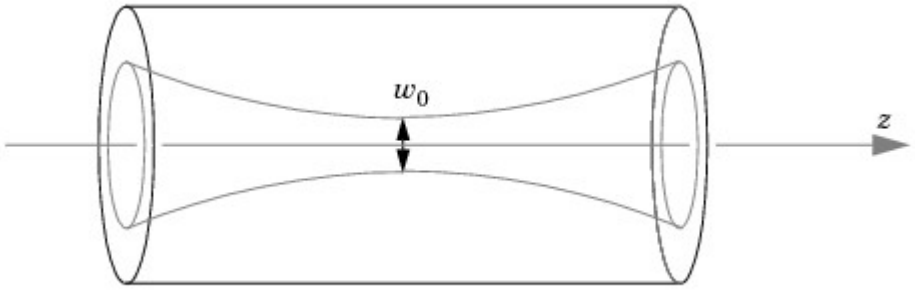
$$w(z) = w_0\sqrt{1+\left(\frac{z}{z_0}\right)^2}$$

$$\eta(z) = \operatorname{atan}\left(\frac{z}{z_0}\right)$$

$$R(z) = z\left(1+\left(\frac{z_0}{z}\right)^2\right) .$$

In these expressions, w_0 is the minimum waist, ω is the angular frequency, r is the radial cylindrical coordinate, and k is the wave number. The wave front of the beam is not exactly

planar; it propagates like a spherical wave with radius $R(z)$. Close to the point of the minimum waist the wave is almost plane. These expressions are used to excite a plane wave at one end of a cylinder.



In this example the problem is solved as a 2D cross section that resembles the axial symmetry of the problem. However, due to the x-polarization of the $\mathbf{E}$-field, true axial symmetry cannot be used (available as an application mode in COMSOL Multiphysics). Here the φ direction is approximated with a standard 2D cross section, neglecting the revolution around the z-axis. So this actually models an infinitely long Gaussian beam plane. The second harmonics generation is probably not affected significantly by this approximation. In the model "Propagation of a 3D Gaussian Beam Laser Pulse" on page 305, propagation of a true 3D Gaussian beam is calculated without any nonlinear effects.

The nonlinear properties for 2nd harmonic generation in a material can be defined with the following matrix,

$$\mathbf{P} = \begin{bmatrix} d_{11} & d_{12} & d_{13} & d_{14} & d_{15} & d_{16} \\ d_{21} & d_{22} & d_{23} & d_{24} & d_{25} & d_{26} \\ d_{31} & d_{32} & d_{33} & d_{34} & d_{35} & d_{36} \end{bmatrix} \cdot \begin{bmatrix} E_x^2 \\ E_y^2 \\ E_z^2 \\ 2E_zE_y \\ 2E_zE_x \\ 2E_xE_y \end{bmatrix},$$

where $\mathbf{P}$ is the polarization. The model only uses the d_{11} parameter for simplicity. To keep the problem size small the nonlinear parameter is magnified by some orders of magnitude. The crystal here has a value of 10^{-17} F/V, when the values for most materials usually are in the 10^{-22} F/V range. Without this magnification, a detectable 2nd harmonics requires the z direction of the geometry to be much longer, resulting in a large problem.

The Gaussian beam is also excited as a pulse in time, using a Gaussian envelope function. This produces a wave package with a Gaussian frequency spectrum. If you have MATLAB, you can calculate the frequency spectrum from a series of time samples with the fast Fourier transform (FFT) using the function `fft`.

Results and Discussion

The main purpose of this simulation is to calculate the second harmonic generation when the pulse travels along the 20 μm geometry. So you have to solve for the time it takes for the pulse to enter, pass, and disappear from the volume. The pulse has a characteristic time of 10 fs, and below you can see the pulse after it has traveled 61 fs.

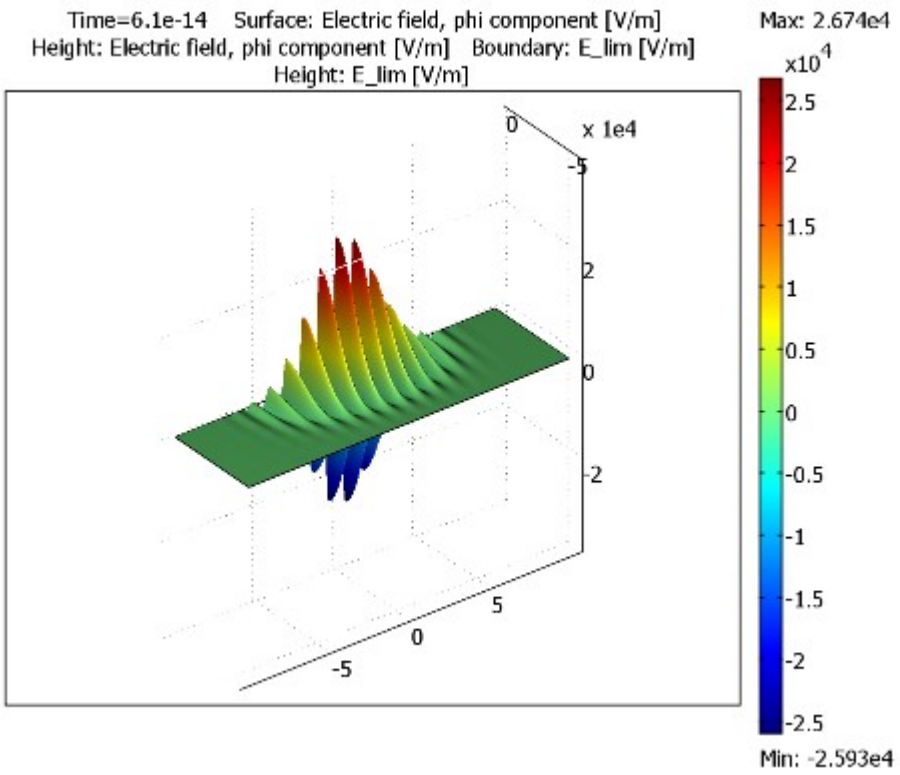
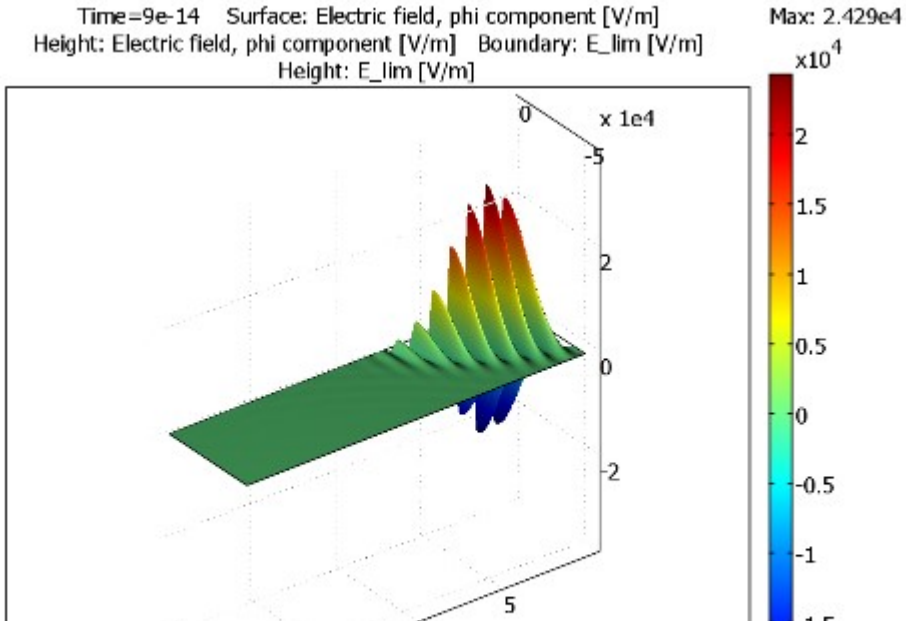


Figure 4-11: The pulse after 61 fs.

After 90 fs the pulse has reached the output boundary (see Figure 4-12). The simulation has to continue for another 30 fs so the pulse completely disappears.



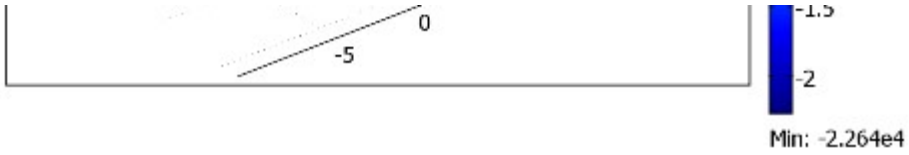


Figure 4-12: The pulse after 90 fs. It has now reached the output boundary.

The simulation stores the times between 60 fs and 120 fs, which is when the pulse passes the output boundary. The electric field at this boundary has a second harmonic component that can be extracted using FFT. Perform the extraction and transformation at the MATLAB command line. The result appears in Figure 4-13 on page 296. There are two groups of peaks in the figure, where the peaks on the right are mirrors of the ones to the left. This mirror effect is due to the sampling of the harmonic signals, which always creates a mirror spectrum around the sampling frequency.

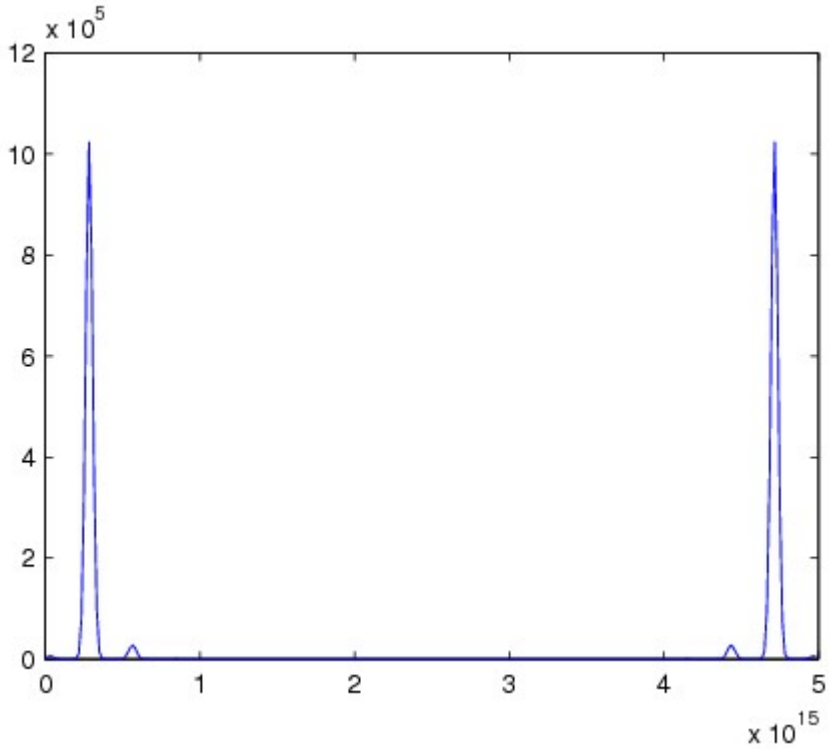


Figure 4-13: The result from the FFT on the beam at the output boundary. The two small peaks between the large peaks are the second harmonic generation.

Reference

1. A. Yariv, *Quantum Electronics*, 3rd Edition, John Wiley & Sons, 1988.

Model Library path:
`RF_Module/Optics_and_Photonics/second_harmonic_generation`

Modeling Using the Graphical User Interface

MODEL NAVIGATOR

1 In the **Model Navigator**, select **2D** in the **Space dimension** list.

2 Click the **Multiphysics** button, and then Click the **Add Geometry** button.

3 In the **Add Geometry** dialog box, type `z r phi` in the **Independent variables** edit field. Click **OK**.

4 Select the **RF Module>In-Plane Waves>TE Waves>Transient propagation** application mode.

5 Type `Aphi` in the **Dependent variables** edit field, and click **Add**.

6 Click **OK**.

OPTIONS AND SETTINGS

1 From the **Options** menu, choose **Constants**.

2 In the **Constants** dialog box, define the following constants with names, expressions, and descriptions (optional):

NAME	EXPRESSION	DESCRIPTION
w0	2[um]	Minimum waist of laser beam
lambda0	1.06[um]	Wavelength of laser beam
E0	3e4[V/m]	Peak electric field
z0	pi*w0^2/lambda0	Peak electric field z position
k0	2*pi/lambda0	Wave number
t0	25[fs]	Pulse time delay
dt	10[fs]	Pulse width
d11	1e-17[F/V]	Matrix element for 2nd generation

3 Click **OK**.

GEOMETRY MODELING

All the dialog boxes for specifying the primitive objects are accessed by choosing **Specify Objects** from the **Draw** menu. The first column in the table below contains the labels of the geometric objects. These are automatically generated by COMSOL Multiphysics, and you do not have to enter them. Just check that you get the correct label for the objects that you create.

1 Draw a rectangle with the properties according to the table below.

LABEL	WIDTH	HEIGHT	BASE	CORNER
R1	20e-6	6e-6	Corner	(-1e-5,-6e-6)

2 Click the **Zoom Extents** toolbar button.

PHYSICS SETTINGS

1 From the **Options** menu, choose **Expressions>Scalar Expressions**.

2 In the **Scalar Expressions** dialog box, define the following variables with names, expressions, and descriptions (optional):

NAME	VALUE	DESCRIPTION
w	w0*sqrt(1+(z/z0)^2)	Analytical waist function along z
eta	atan(z/z0)	Analytical angle along z
R	z*(1+(z0/z)^2)	Analytical radius along z
c0	1/sqrt(epsilon0_rfwe* mu0_rfwe)	Speed of light
omega0	2*pi*c0/lambda0	Angular frequency

3 Click **OK**.

4 From the **Options** menu, choose **Expressions>Boundary Expressions**.

5 In the **Boundary Expressions** dialog box, define the following variables with names and expressions:

SETTINGS	BOUNDARY 1	BOUNDARY 2	BOUNDARY 3
E_bnd	w0/w*exp(-r^2/w_bnd^2)* cos (omega0*t-k0*z+eta_bnd-r^2*k0/(2*R_bnd))		
E_pulse	exp(-(t-t0)^2/dt^2)		
E_lim		E0*1.2	-E0*1.2

Because of the use of an integration coupling variable (w_bnd), COMSOL Multiphysics cannot determine the unit for E_bnd and warns for an inconsistent unit here and in the specification of the scattering boundary condition. You can disregard these warnings.

6 Click **OK**.

7 From the **Options** menu, point to **Integration Coupling Variables** and then click **Point Variables**.

8 In the **Point Integration Variables** dialog box, define the following variables with names and expressions. Use **Global destination** for all variables. When done, click **OK**.

NAME	POINT 2	POINT 4	ALL OTHERS
w_bnd	w		

eta_bnd	eta		
R_bnd	R		
Eout		Ephi_rfwe	

Boundary Conditions

1 From the **Physics** menu, open the **Boundary Settings** dialog box and enter the settings according the following two tables (leave all fields not specified at their default values):

SETTINGS	BOUNDARY 1	BOUNDARY 4
Boundary condition	Scattering boundary	Scattering boundary
$E_{0,\varphi}$	E0*E_pulse*E_bnd	0

SETTINGS	BOUNDARY 2	BOUNDARY 3
Boundary condition	Perfect electric conductor	Perfect magnetic conductor

2 Click **OK**.

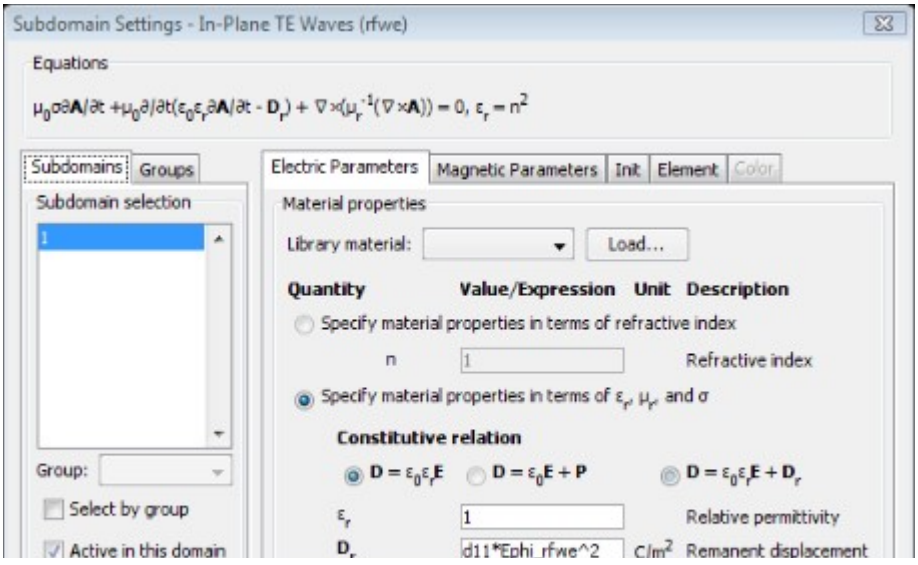
Subdomain Settings

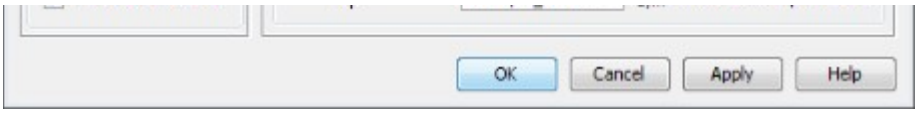
1 From the **Physics** menu, open the **Subdomain Settings** dialog box, select Subdomain 1, and click the **D=e0erE+Dr** button for the constitutive relation.

2 Define the subdomain settings according to the following table:

SETTINGS	SUBDOMAIN 1
ε_r	1
D_r	d11*Ephi_rfwe^2

3 Click **OK**.





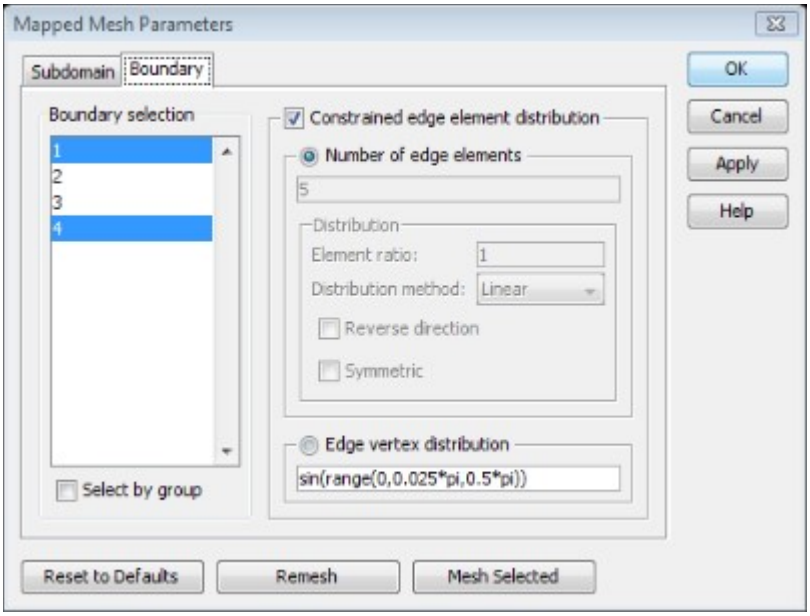
MESH GENERATION

1 From the **Mesh** menu, choose **Mapped Mesh Parameters**.

2 On the **Boundary** page, select Boundaries 1 and 4 from the **Boundary selection** list.

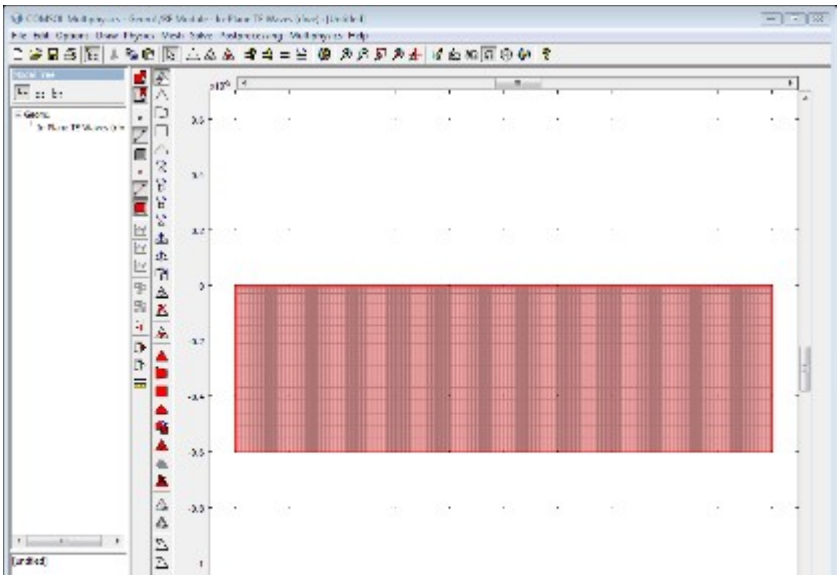
3 Select the **Constrained edge element distribution** check box.

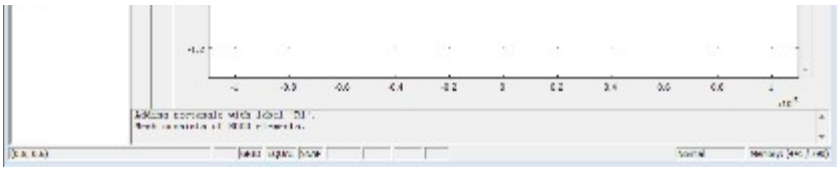
4 Click the **Edge vertex distribution** button. In the edit field below, type `sin(range(0,0.025*pi,0.5*pi))`. This creates a denser mesh closer to the upper boundary.



5 Select Boundary 2, click the **Edge vertex distribution** button, and type `range(0,5e-8,20e-6)` in the edit field.

6 Click the **Remesh** button; when the mesher has finished, click **OK**.





COMPUTING THE SOLUTION

To perform an FFT analysis in MATLAB, the number of time steps that have to be saved is very large, and to store all solutions of the $\mathbf{A}$-field results in a huge model file. However, for the FFT, it is only interesting to look at the field at the output boundary. You can take advantage of this fact by first solving the model with a large number of output time steps using a single-step parametric sweep to store the quantity of interest in a global variables log file. Once the solver has finished, you reset the model, change the output time steps, and solve the model again (this time without using a parametric sweep). The following instructions describe the relevant modeling procedure to take whether you want to skip or perform the FFT analysis.

To be able to view the time-evolution of the electric field at the input and output boundaries while the solver computes the solution, first set up a probe plot.

1. From the **Postprocessing** menu, choose **Probe Plot Parameters**.
2. In the **Probe Plot Parameters** dialog box, click the **New** button.
3. In the dialog box that appears, choose **Point probe** from the **Plot type** list, and enter `Eout` in the **Plot name** edit field. Click **OK**.
4. Select Point 4, and choose **Electric field, phi component** from the **Predefined quantities** list.
5. Click the **New** button again and repeat the steps above to define a point probe plot named `Ein` for Point 2.
6. Select the **Plot all plots in the same axis** check box. Click **OK**.

Proceed to set up the solver.

1. From the **Solve** menu, choose **Solver Parameters**.
2. In the **Times** edit field type `0 6.1e-14 9e-14 1.2e-13`.
3. On the **Time Stepping** page, select **Manual** from the **Time steps taken by solver** list. Set the **Time step** to `2e-16`.
4. If you do not intend to perform an FFT analysis or want to use a presaved data file for this purpose, skip directly to Step 17; otherwise continue with the next step.
5. Change the **Times to store in output** to **Time steps from solver**.
6. Click **Apply**.

Next, set up and solve a single-step parametric sweep to generate a log file.

7 From the **Solve** menu, choose **Parametric Sweep**.

8 In the **Parameter names** edit field, enter a dummy name, for example, `dummyparam`.

9 In the **Parameter values** edit field, enter a single value, for example, `0`.

10 In the **Global variables to evaluate** edit field, type `Eout`.

11 In the **Log file name** edit field, type `second_harmonic_generation_outdata.txt` to save the log in the current directory or click **Browse** and use the **Log File** dialog box to save the file in a suitable directory, preferably one on you MATLAB path.

12 Click the **Solve** button in the **Parametric Sweep** dialog box.

13 When the solver has finished, click **OK** to close the **Parametric Sweep** dialog box.

14 From the **File** menu, select **Reset Model**. In the dialog box that appears, click **OK** to confirm.

15 Return to the **Solver Parameters** dialog box.

16 On the **Time Stepping** page, set the **Times to store in output** to **Specified times**.

17 Click **OK** to close the **Solver Parameters** dialog box.

18 Click the **Solve** button on the Main toolbar.

POSTPROCESSING AND VISUALIZATION

1 Select **Plot Parameters** from the **Postprocessing** menu.

2 On the **General** page, make sure that the **Surface**, **Boundary**, and **Geometry edges** check boxes are selected.

3 From the **Solution at time** list, select **6.1e-14**.

4 On the **Surface** page, type `Ephi_rfwe` in the **Expression** edit field.

5 Click the **Height Data** tab, select the **Height data** check box, and type `Ephi_rfwe` in the **Expression** edit field.

6 On the **Boundary** page, type `E_lim` in the **Expression** edit field.

7 Click the **Height Data** tab, select the **Height data** check box, and type `E_lim` in the **Expression** edit field.

8 Click the **Uniform color** option button.

9 Click the **Color** button and select white from the palette in the dialog box that appears. Click **OK** to close the dialog box.

10 Click **Apply** to get the plot in Figure 4-11.

11 On the **General** page, select **9e-14** from the **Solution at time** list. Click **OK**. You should now see the plot in Figure 4-12.

FFT ANALYSIS USING MATLAB

If you have MATLAB, you can go on to compute and plot the Fourier transform of the electric field at the output boundary that you stored in the parametric sweep log file. This data is also provided in the text file `second_harmonic_generation_outdata.txt`, located in the model's Model Library folder.

Now switch to the MATLAB workspace.

1 If you stored the data file in a directory that is not on your MATLAB path, use the command `cd` to navigate to that directory. Open the file in a text editor and remove the initial header before saving it again.

2 At the MATLAB prompt, enter the following commands to fetch the output field data from the file:

```
S = load('second_harmonic_generation_data.txt','-ascii');
xdata = S(:,2);
ydata = S(:,3);
```

(The first column in the data text file contains the values for the dummy parameter used in the parametric sweep and can thus be discarded.)

3 Now you can proceed to perform the FFT analysis. Because the output boundary field is effectively zero for the first half of the time steps in the file, you can restrict the time interval for the analysis by discarding the first 300 time steps.

```
T = xdata(301:end);
Eout = ydata(301:end);
freq = ((1:length(T))-1)/(length(T)*mean(diff(T)));
xEout = fft(Eout);
plot(freq,abs(xEout));
```

This produces the plot in Figure 4-13 on page 296.