Table of Contents

Your first simulation 1
Basic FDTD algorithm operation example 1
Summary 1
Setup all this with XSvit 1

Your first simulation

Basic FDTD algorithm operation example

Summary

We will use cubic volume with centered point source, similar to a dipole located in center of computation space. Outputs will be different cross-section planes to Gwyddion file and a text file (from single point). Calculation will be performed on CPU or on GPU and material mode selection for speedup will be tested.

What can be tested using this example:

- basic performance of the software (first test to see whether it works after installation)
- performance of Yee algorithm with none material properties (compare with material mode calling full algorithm, resulting in different speed and memory allocation.)
- GPU performance
- GPU performance with none/full material properties (combine GPU and material mode checking)
- scaling of basic calculation speed and memory requirements (varying computational volume size)
- point source properties (varying the orientation or using file input for source).

Setup all this with XSvit

Note that some of the screenshots might be from an older version of XSvit.

First, start the XSvit application. If haven't done this before, setup the location of GSvit and Gwyddion under File->Preferences menu, so you can run the calculation from the GUI and see the results afterwards. Otherwise you will be able to use GUI only for parameter file setup.

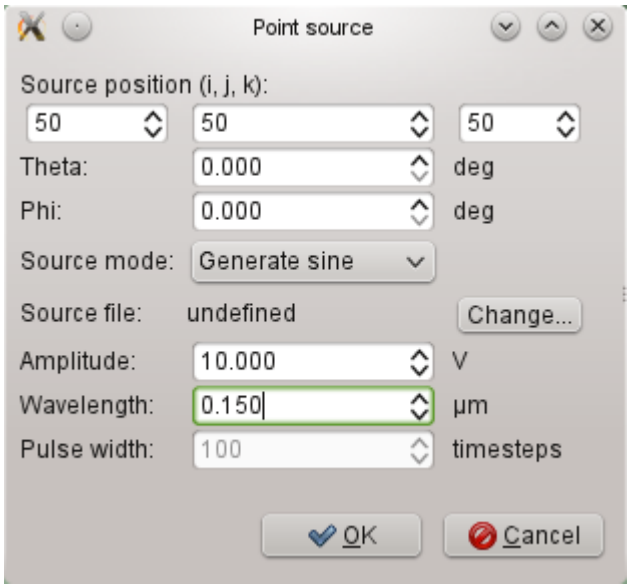
When you start XSvit, there is already a default parameter file loaded, using computational space of size 100x100x100 voxels. We will use this computational space and we will add point souce and point output. In the next figure you can see how the default settings look after XSvit start (you can rotate the 3D projection by mouse, so this was done to get a better view). Note the values of discretisation step (10 nm) in the list of all the entries on the left side of the window; if we work in vacuum (which is the case in this example) we should not use source with wavelength that is smaller than this step multiplied by ten. You can alter the discretisation settings and computational volume size by double clicking on its values in the list.

{{:start:screenshots:first-simulation-init.png?400|}} {{start:screenshots:file-preferences.png?400 |}}

It is reasonable to save your parameter file at this stage using *File->Save* command. We can do it also later, just before running computation, however doing it right now will set the current directory and other input/output files set later can be referred to it (instead of using absolute path). Moreover, by saving parameter file we also set default directory for temporary files (e.g. generated sources), which is also important.

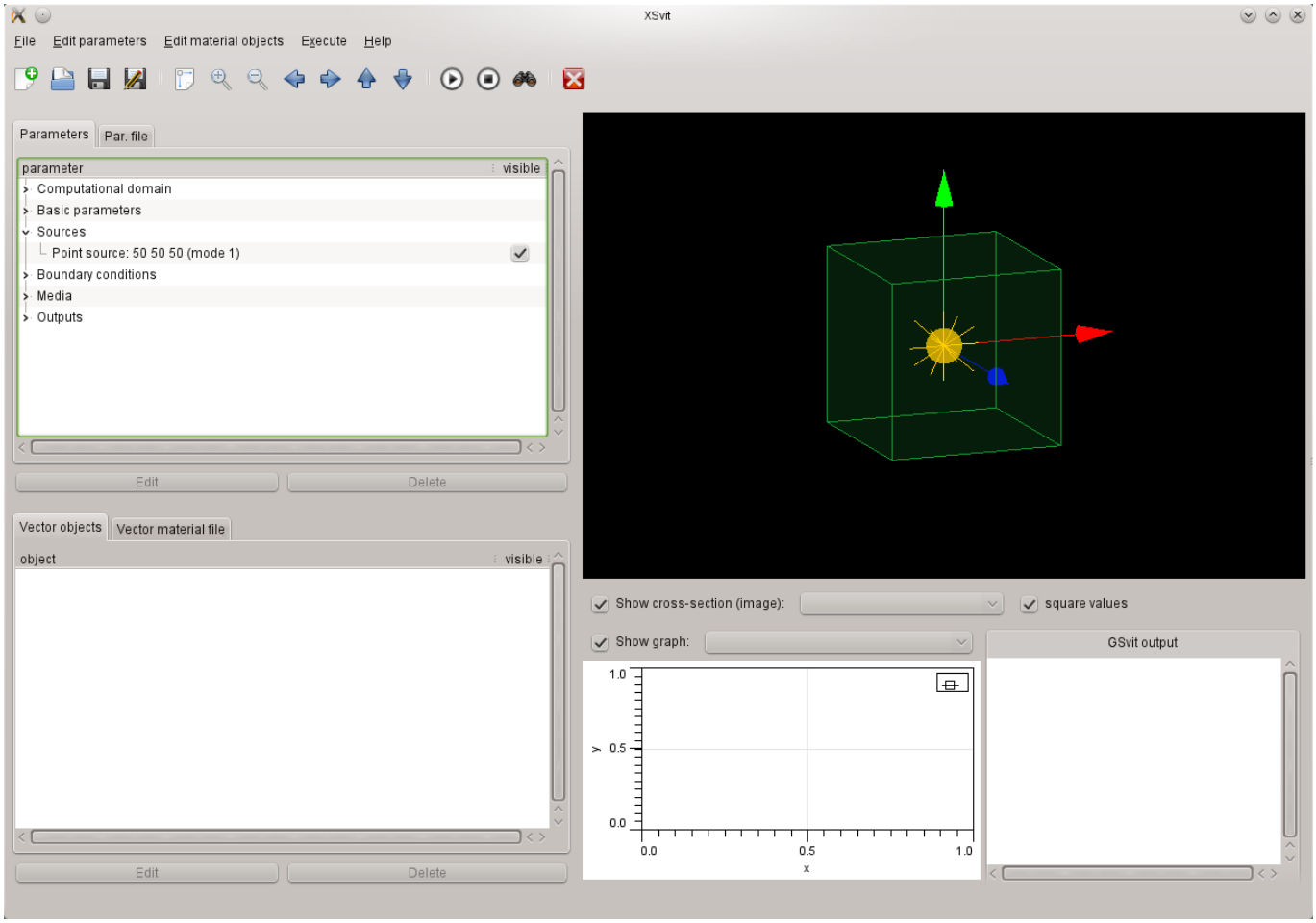
First we will add **point source**. This can be done using menu entry *Edit parameters->Add point source*. The dialogue shown below appears. We set the point source properties, i.e. its position in voxel coordinates, its type (load from file or construct a temporary file for it with sine wave or sine

impulse) and orientation given for temporary file sources by two angles (see the angles for TSF in GSvit documentation). Here we have chosen sine source given by Ez component only.



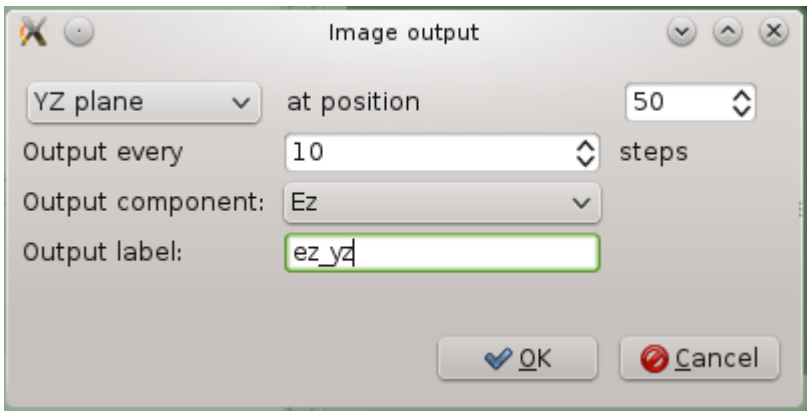
After adding the source we can see it in the 3D window as shown below and also on the list of entries. If we are not satisfied with its position we can double-click on the entry and correct it.

Hint: if we switch to "par. file" tab we can edit the source in native GSvit format file much faster and see the position changes immediately.

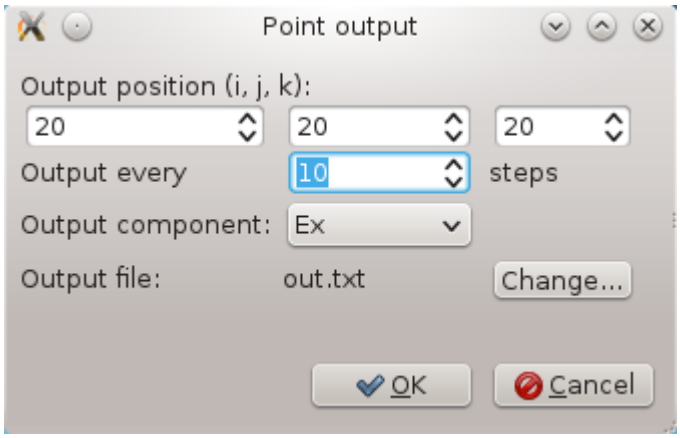


Next we will add **point and image outputs**. This is done by menu entry *Edit parameters->Add point*

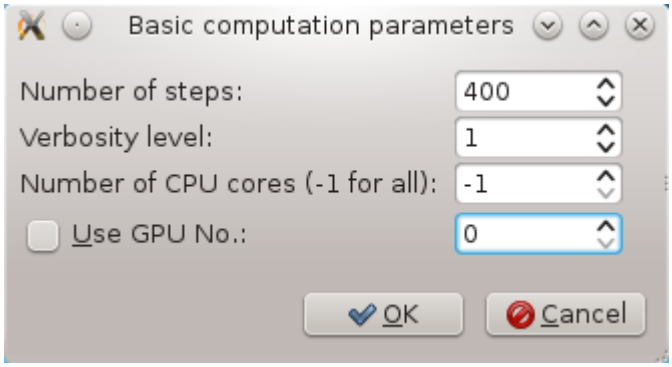
output and *Edit parameters->Add image output* that again open dialogues for specifying how often (parameter "skip") and which component should be output. Point output is collected from a single point within computational volume and is stored into a text file. Image output comes to general output file that can be opened by Gwyddion open source software. A cross section in certain normal direction is output, selected as shown below. We will add a cross section in every direction, so we will call the dialog three times.



Similarily we will add point output as shown below.



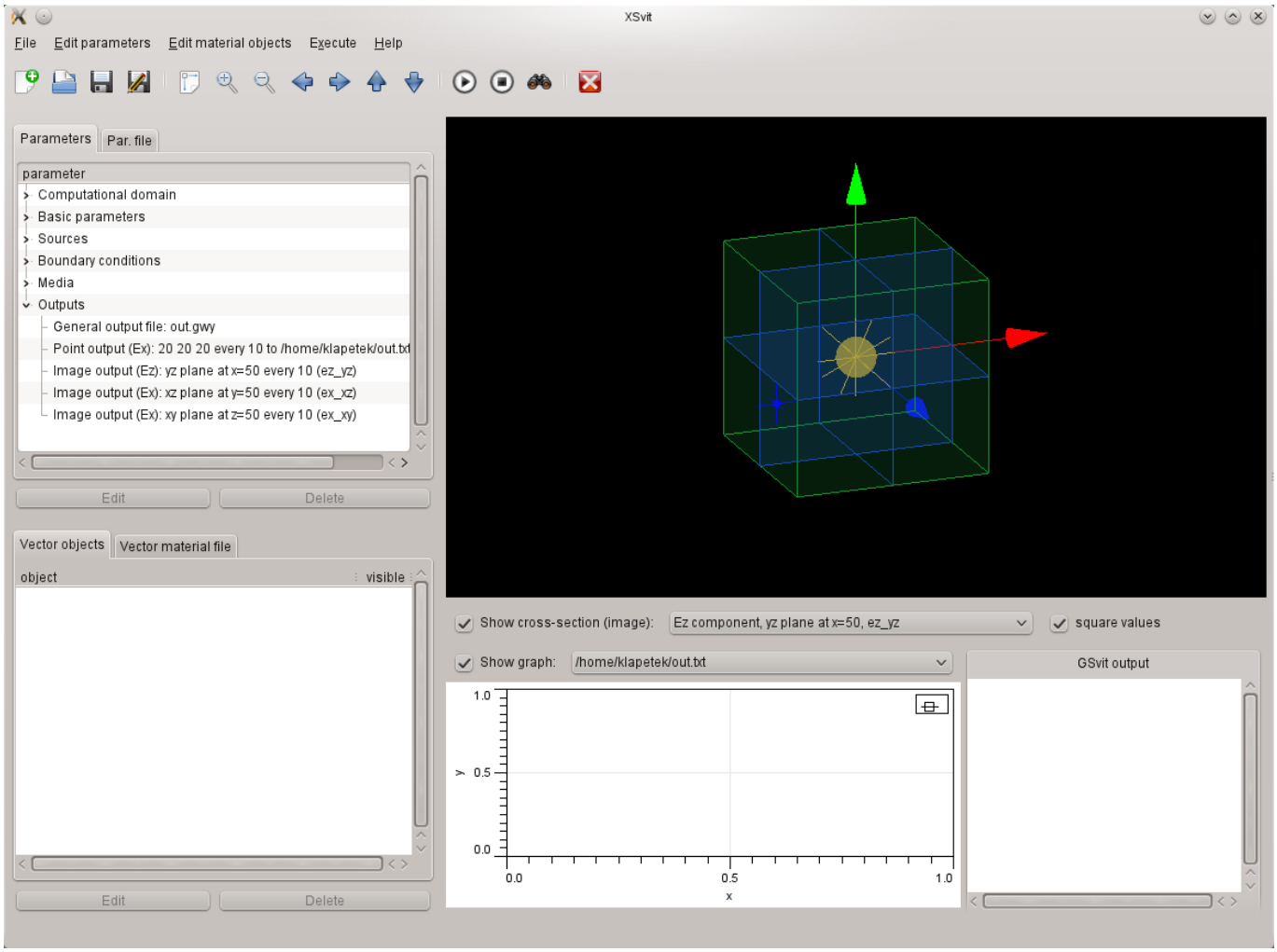
Note that all the outputs can be also seen in 3D view window. We can adjust some more computational parameters, here namely we will increase **number of steps** to be calculated to 300 (clicking to any of the entries in "Basic parameters" group). Physical length of the time step is calculated automatically on the basis of space discretisation.



Finally we need to save the parameter file if we haven't done it at beginning, as since now we worked only with a temporary file or at least modified file (it is important to save it always before running computation which is not done automatically). This is done by *File->Save* command as usual. Then we can **start the computation** clicking on the arrow in the toolbar or using *Execute->Run* menu

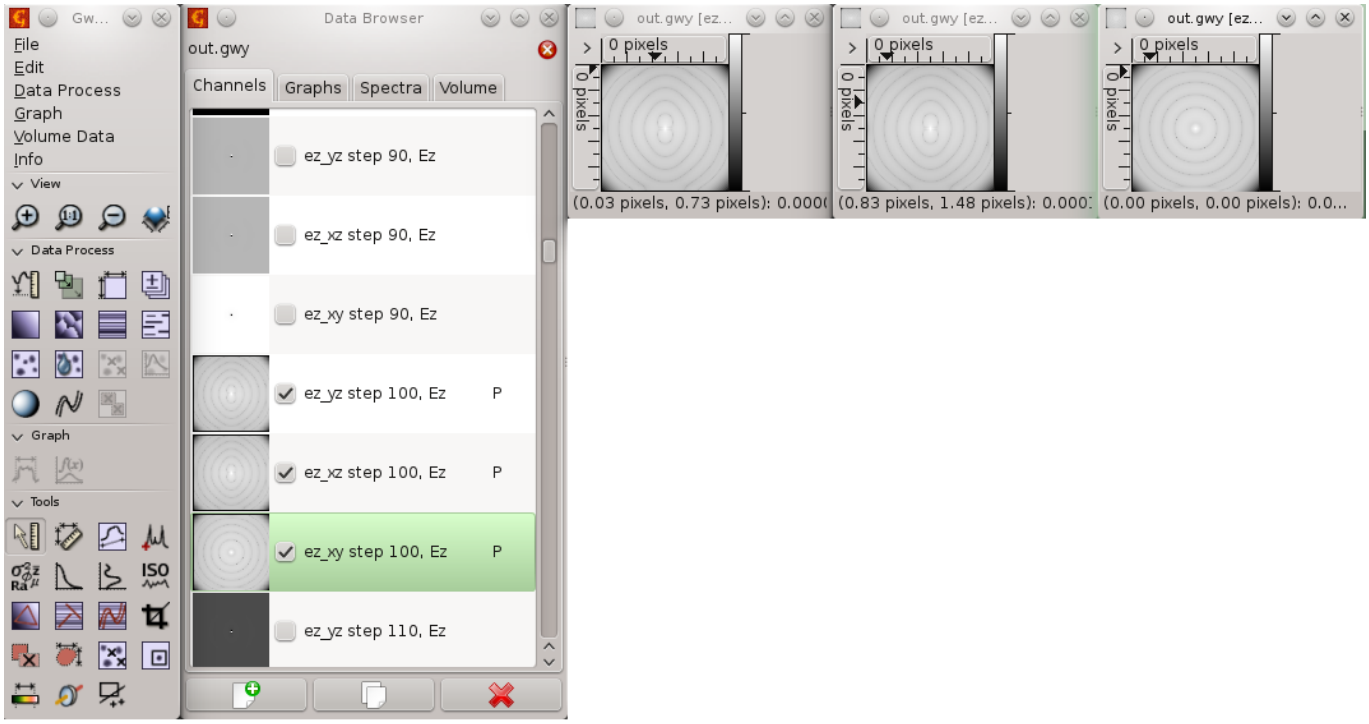
command.

Hint: if we don't want to use the GUI for starting the computation we can always run GSvit manually on our parameter file.



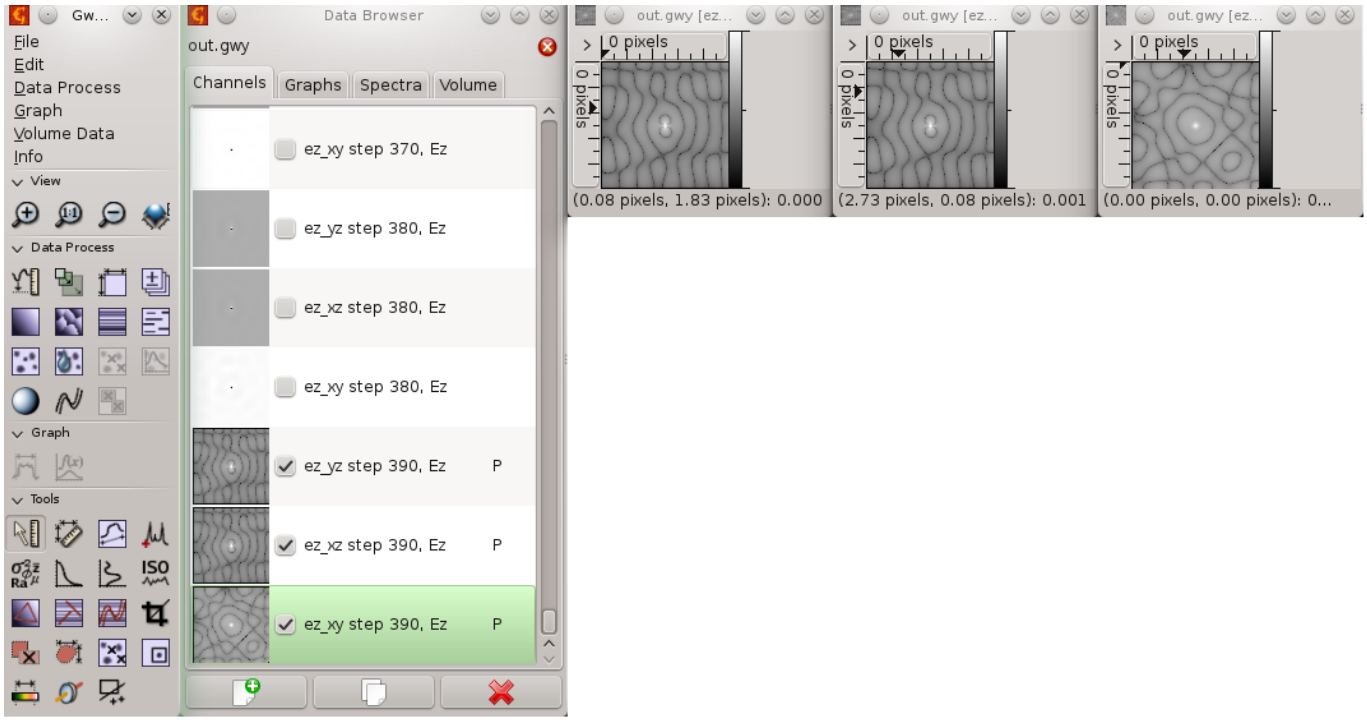
While computation is running we can see the GSvit output (if the verbose level is not 0) and we can choose any text file output to be visualised in the graph window. You can also watch preview of last saved images - slices through the computational volume - directly in 3D view. The GSvit output is directly what you would see if you run it at console, with no GUI. Note that graph is updated at maximum as often as the data are saved, so if we skip every 100 points while setting the output we will see the graph updated only e.g. every few seconds (depending on computational volume size, etc.). Similarily, the images in the 3D view are updated when these are saved by the computational core (so you can affect the update frequency by image output skipping parameter).

When computation is finished or killed by the Stop command (again via toolbar or menu), we can **open the output** file. As most of the data are output in Gwyddion file we can directly open it using Gwyddion called from the menu or toolbar. Gwyddion is a powerfull 2D/3D data processing software, however we will only use it to see the images in this example. As image outputs are saved succesively as the GSvit is running, we will be probably having more channels to see in Gwyddion, so we pick what is interesting for us in the Gwyddion Data Browser.

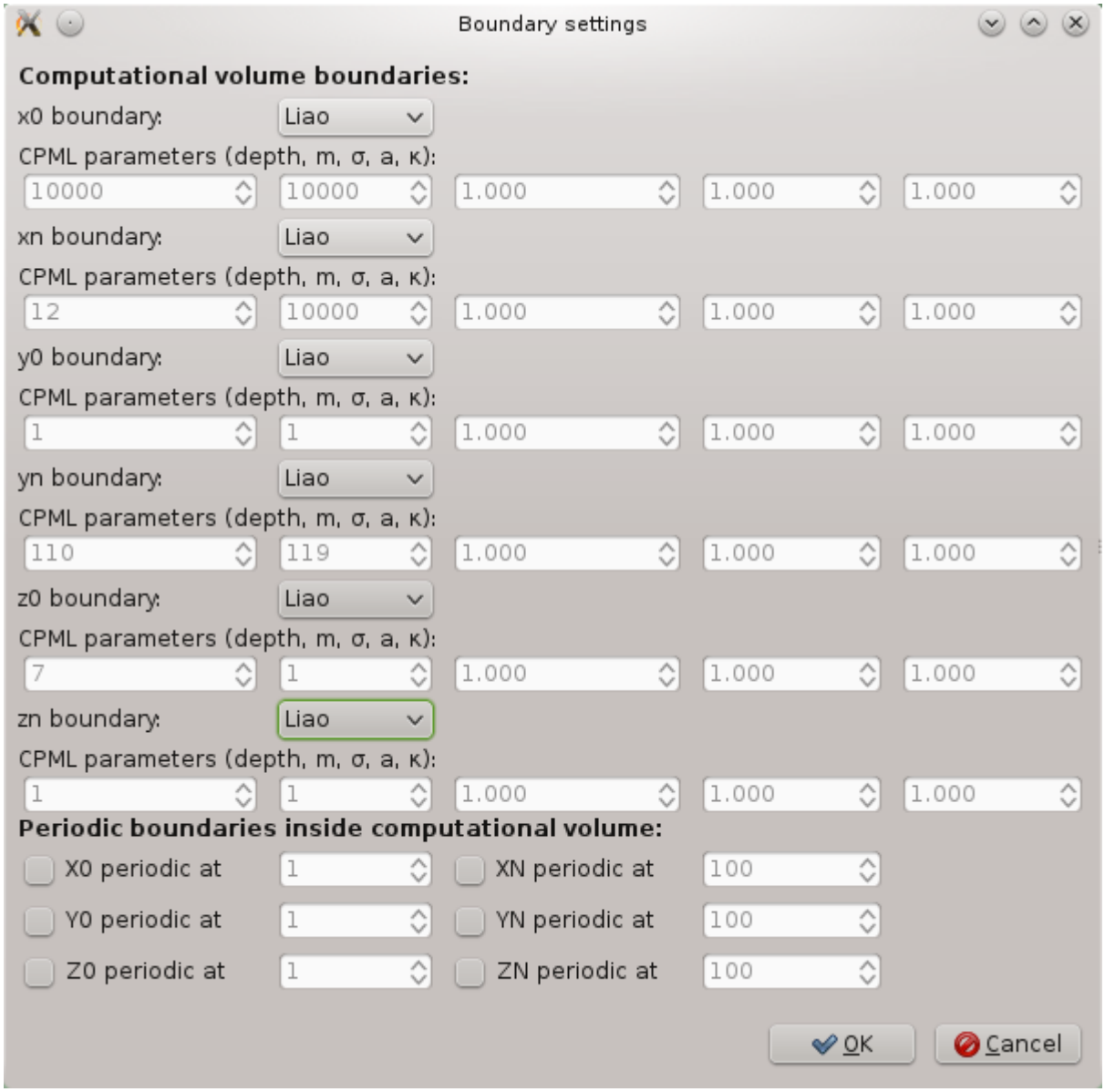


Note that in this example we have clicked on *Data Process->Presentation->Logscale* for every displayed output to see the data in better scale. We can see that in different cross-sections of the computational space there is a wave similar to a dipole output.

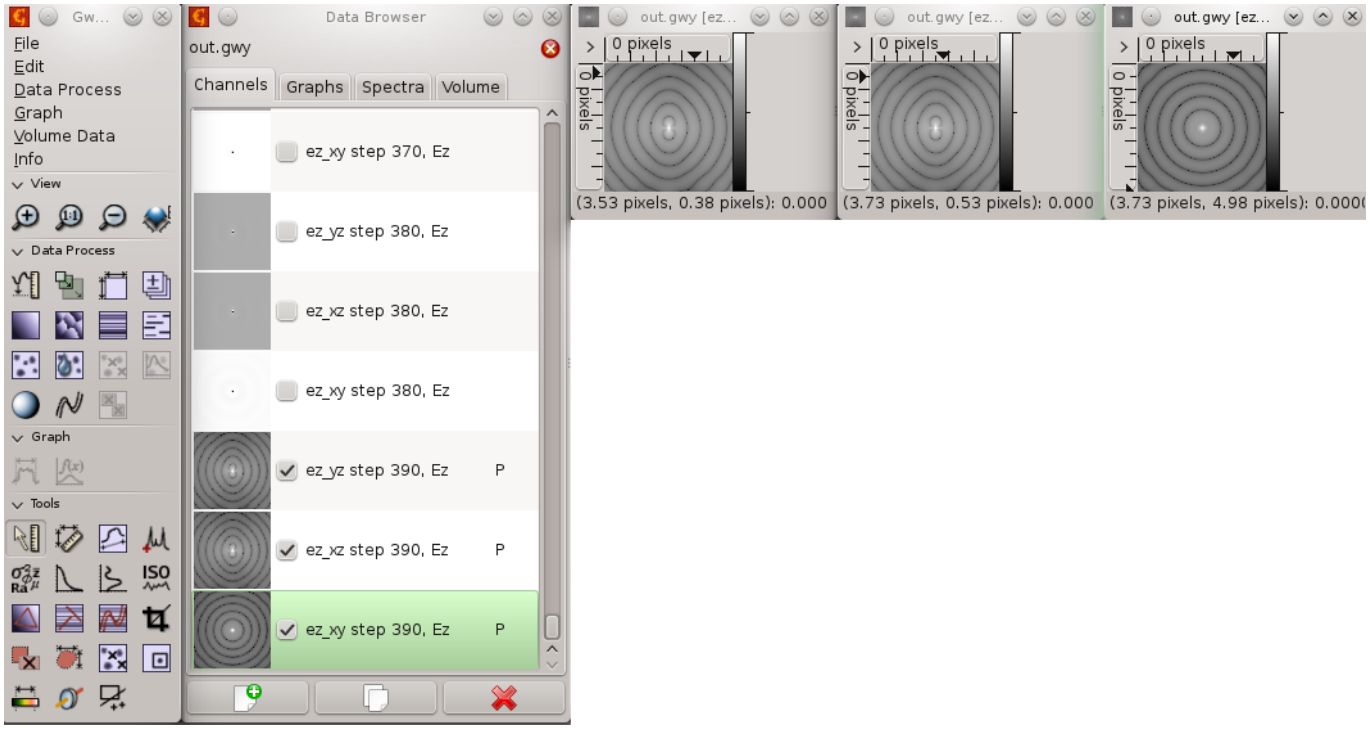
If we check the data in further time steps, we can see that they look much worse.



This behavior is caused by unset **boundary conditions** that behave like mirrors here. In order to let the wave go out of the computational space we need to apply an absorbing boundary condition, so we click on any entry in "Boundary conditions" set to change the boundary conditions on all the faces to a simple absorbing condition (Liao).



After that (and saving the file, running computation and redisplaying data again) we can see that the output is already propagating freely outside of the computation volume.



From:
http://gsvit.net/wiki/ - **GSvit documentation**

Permanent link:
http://gsvit.net/wiki/doku.php/start:first_simulation?rev=1536064907

Last update: **2018/09/04 14:41**