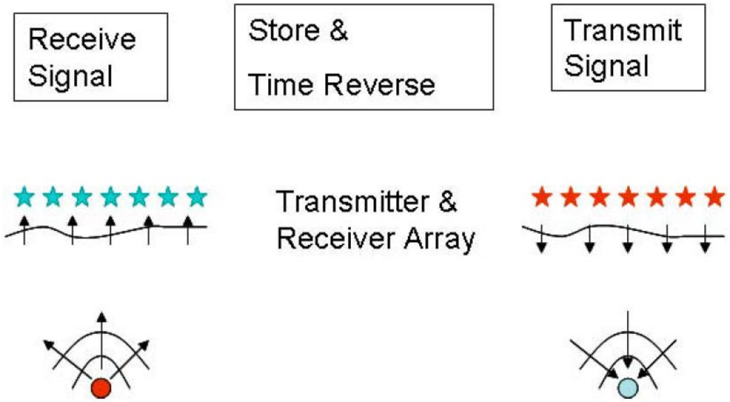
Tutorial for the Time Reversal Simulation

Wave Computation Tech.

Nov., 2017

Introduction

The time reversal method utilizes the reciprocity of wave propagation in a time-invariant medium to find the shape and the location of a object through the field focusing.



The focusing quality in the time-reversal method is determined by the effective aperture of transmitter-receiver array. This effective aperture includes the physical size of the transmitter-receiver array and the effect of the environment. A complicated background will create so-called multipath effect and can significantly increase the aperture of transmitter-receiver array.

In this tutorial, we will demonstrate how to simulate time-reversal cases in Wavenology EM-EL package.

Table of Content

- Demo Cases
    1) **EM**: detect a PEC sphere behind walls by the scattered electrical field
    2) **EM**: detect a object in a chamber by antenna array by the scattered voltage
    3) **EM**: detect a source in a 2D human skull slice by the direct wave
    4) **EL**: detect elastic monopole sources in multiple layers background by the direct wave

Basic Setup Scheme
(1) Backward Propagate the Direct Wave

The purpose of this scheme is to find out the field focusing on the sources. In this scheme, the source may be introduced by other physics. For example, a EL wave is excited by the thermal effect on the tissue by a EM field.

A time-reversal simulation will include following steps:

1) Obtain the received signals on sensors
2) Set up a WCT project, with transmitters at sensor's position
   - Set up source directly for a new case
   - Or, convert the receiver to source in GUI, if the user has a similar project with the same physics
3) Define the simulation time window with a fixed time
4) Load the received signals to source as the excitation pulse
   - User can reverse the time sequence of signal by themselves and load
   - WCT GUI already provide a function to reverse the time sequence
5) Define snapshot to record the EM/EL wave propagates
6) Run the simulation and check the result
   - transient snapshot result
   - WCT RTI image result

Basic Setup Scheme

(2) Backward Propagate the Scattered Field

The purpose of this scheme is to find out the field focusing on the induced sources. In general, this scheme is for the scattered signal from the same physics.

This time-reversal simulation scheme will be almost the same as previous one, the change is the source signal for imaging will be the scattered field instead of the direct wave.

Obtain the scattered signals

a) it can be from the measurement
b) Or, it can be from some kinds of simulation tools. In WCT, it can be obtained by
   - Simulate two cases to obtain the transient fields on receivers. These two cases are almost the same. The only difference is one case has the target (**total field case**). Another has not the target (**incident field case**).
     - Setup transmitter in two cases.
     - Setup sensor array in two cases.
     - Calculate the scattered field from the data obtained from these two cases.

Transmitters & Receivers

EM Solver

For a time-reversal project,

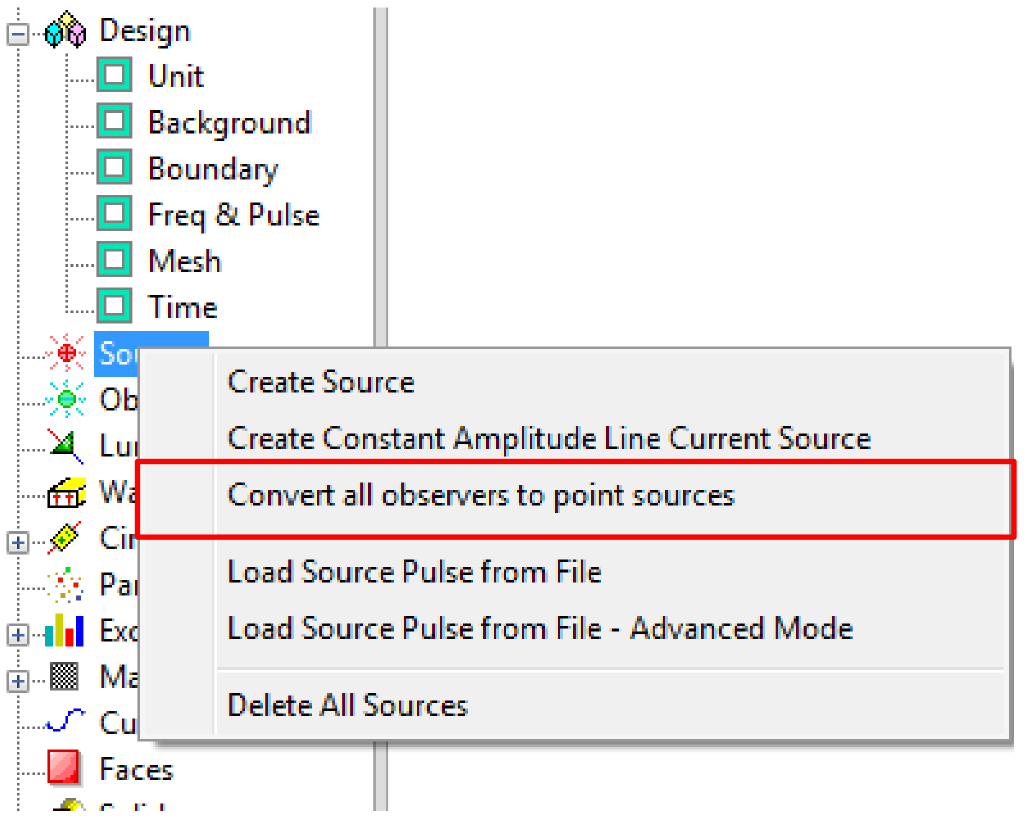
- the transmitter can be
  - ideal point dipole source
  - or, antenna
- the receiver can be
  - ideal point observer, the received signal is the field
  - or, antenna, the received signal is the voltage

EL Solver

For a time-reversal project,

- the transmitter can be
  - ideal dipole source
  - ideal monopole source
  - ideal moment tensor source
- the receiver is
  - ideal point observer, to capture particle velocity or force

Tools to Convert Point Receiver to PointTransmitter



In Wavenology GUI, with menu item "***Convert all observers to point sources***", a point observer in the project will be converted to one or several ideal point sources at the same position.

But for the port on an antenna to record voltage, user need to manually convert it from the receiving mode to the transmitting mode.

The conversion rule

EM Project

- if all receivers to capture **single** field component, the ideal E dipole source converted from receiver will be
  - ❑ at the same position
  - ❑ with the same name
  - ❑ the dipole source will be set with a polarization along the field component direction. For example, Ey field receiver will be converted to a point E dipole with polarization as (0, 1, 0).

- if there is any one receiver to capture **multiple** field components, for each component on a receiver, there will be a converted ideal E dipole source
  - ❑ at the same position
  - ❑ with the same name as prefix, following by component name
  - ❑ the dipole source will be set with a polarization along the field component direction.
  - ❑ for example, a (Ex + Ez) field receiver 'A' will be converted to
    - one point E dipole with a polarization as (1, 0, 0), name by "A_x"
    - another point E dipole with a polarization as (0, 1, 0), name by "A_y"

EL Project

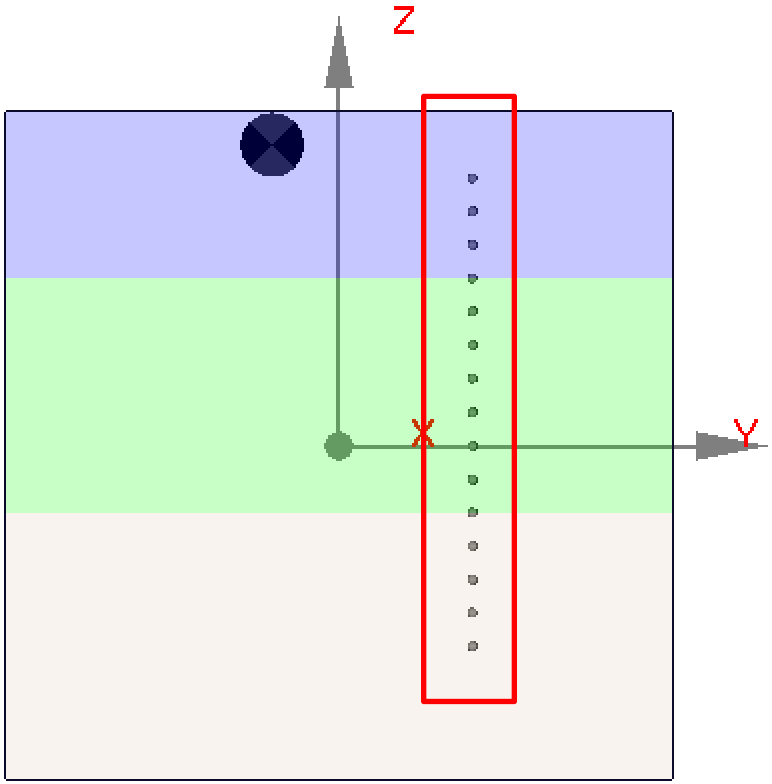
Cont.

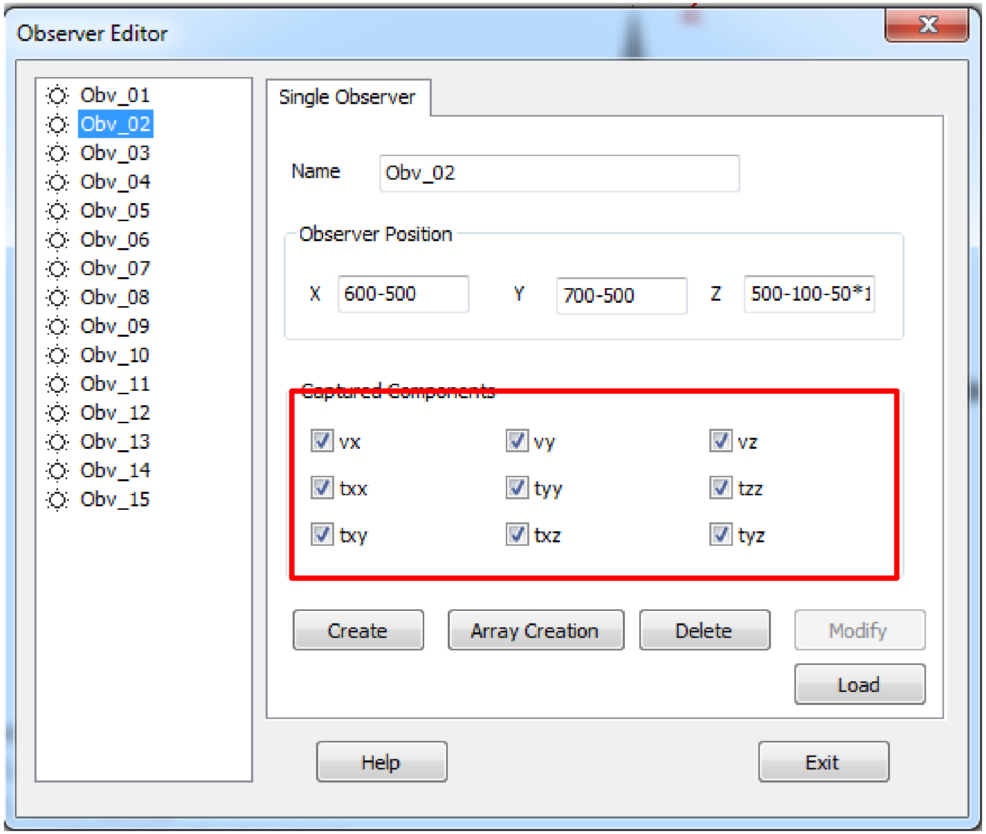
- if all receivers to capture single field component, the ideal point source converted from receiver will be
  - at the same position
  - with the same name
  - the source will be determined by the recorded field type
    - velocity receiver will be converted to point dipole source, with a polarization along the field component direction.
    - compress force receiver (T_{au_xx}, T_{au_yy}, T_{au_zz}) will be converted to a point monopole source
    - shear force receiver ($_{Tau\_xy}$, T_{au_xz}, T_{au_yz}) will be converted to a point moment tensor source

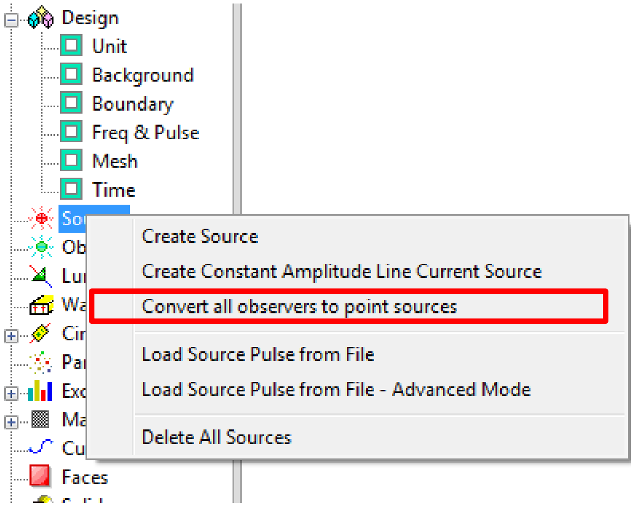
- if there is any one receiver to capture multiple field components, for each component on a receiver, there will be a converted ideal point source with above rules except the name rule
  - the name will be the same name as prefix, following by component name
  - for example, a (V_y + T_{au_xx}) field receiver 'A' will be converted to
    - one point dipole source with a polarization as (1, 0, 0), name by "A_vx"
    - another point monopole source with the name by "A_tau_xx"

Following is a demo in a EL project to convert receiver to source.

There are multiple receivers in the project,
and capture all 9 field components

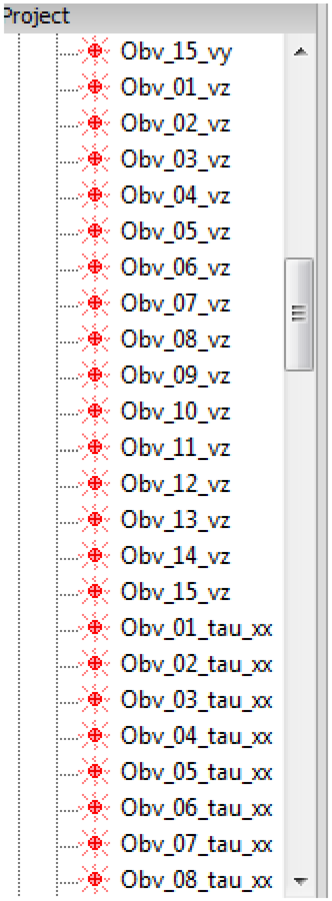


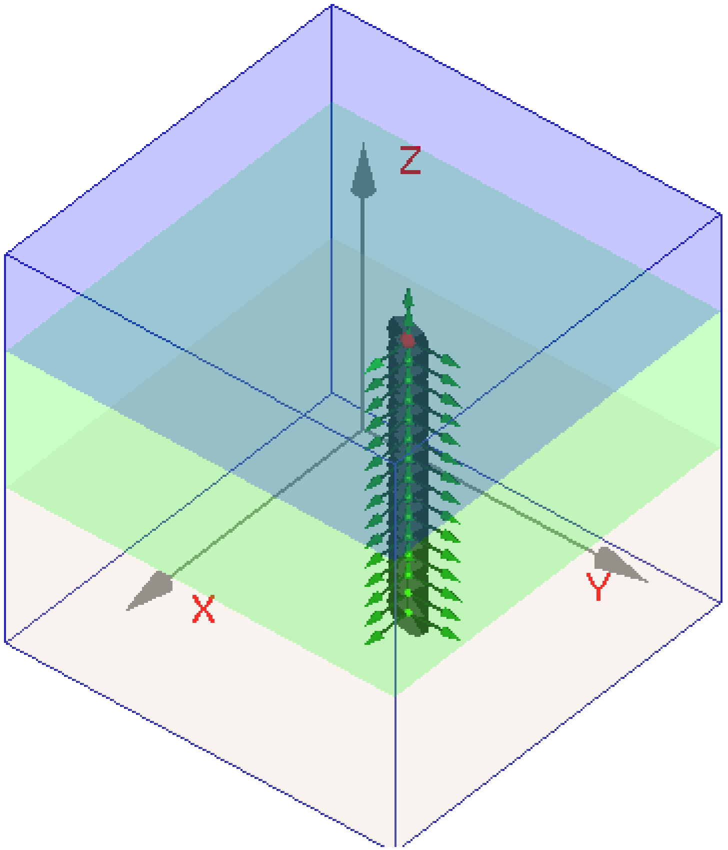




With conversion menu, we get following sources.

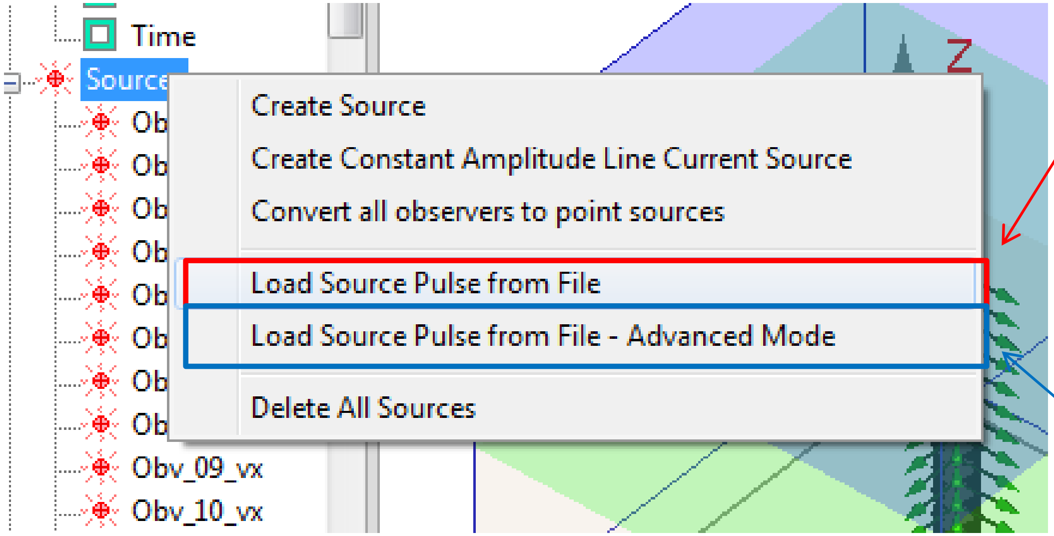
Each one receiver is converted to 9 sources, with corresponding name subfix and source type





Loading Source Pulse with GUI Tools

➢ WCT GUI provide two tools to load pulses to multiple sources in one operation, as following,



Load pulse to all sources with name-matching method without reverse the signal

Load pulse to sources with options

Load pulse to all sources with the name-matching method without reverse the signal

This method load a ASCII file with following format:

Header

```
%Wave Computation Technologies simulation waveform data, version 3.0
%created at 10/30/17 22:32:16 in GMT
%==================================
%Title      Observer txx transient
%X-Lin      0.001    Unit:ms
%Y-Lin      1    Unit:Pa
```

Block for trace 1

```
%Sub-Title  Obv_01
731  731  %Size Nx=Ny
0.00000000000e+000  0.00000000000e+000
2.73481699210e+000  0.00000000000e+000
…..
```

Block for trace 2

```
%Sub-Title  Obv_02
731 731  %Size Nx=Ny
0.00000000000e+000  0.00000000000e+000
2.73481699210e+000  0.00000000000e+000
…..
```

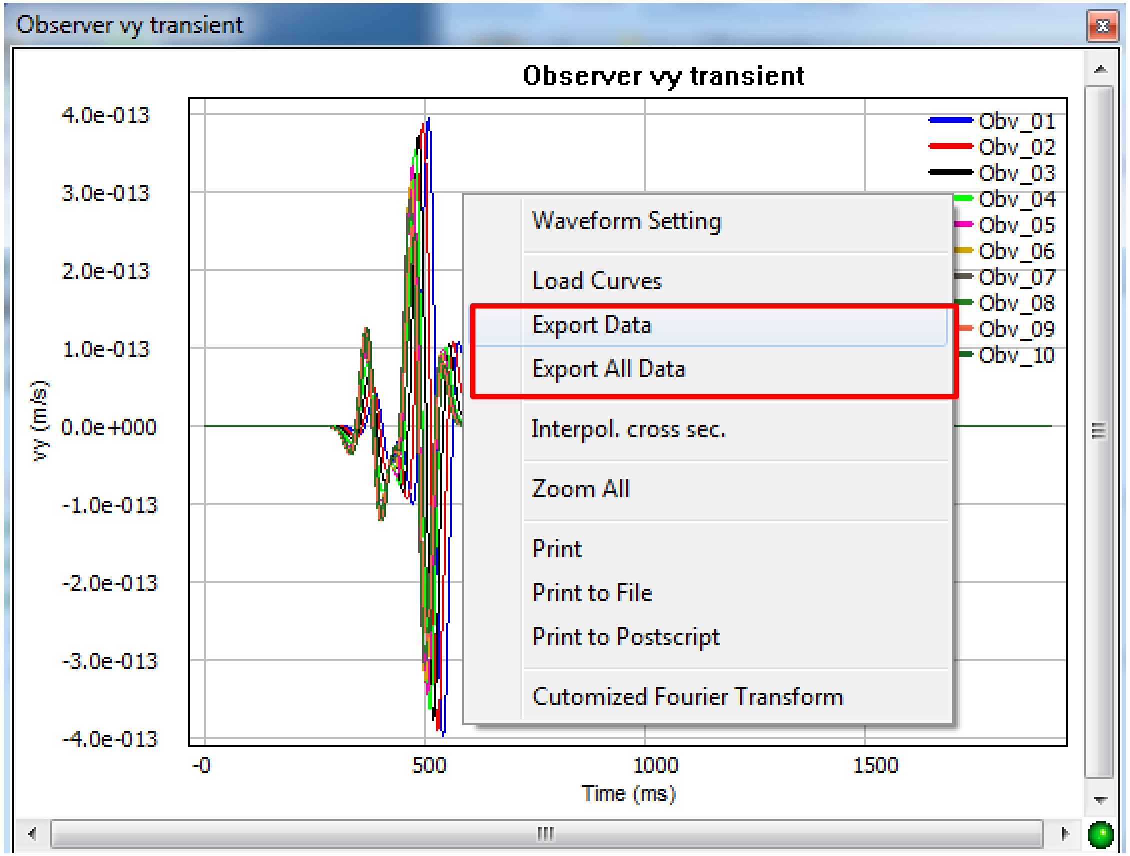
```
%Sub-Title  Obv_03
731 731  %Size Nx=Ny
0.00000000000e+000  0.00000000000e+000
2.73481699210e+000  0.00000000000e+000
…..
```

Trace name

Trace length

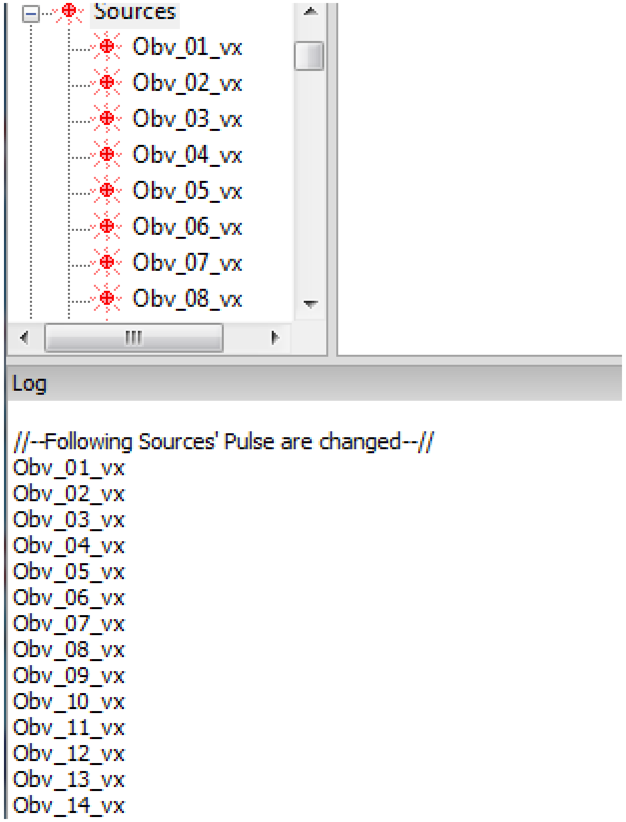
Trace data, 1st column is the time, 2nd column is the value

This format is the same as the data file format exported from WCT 2D canvas



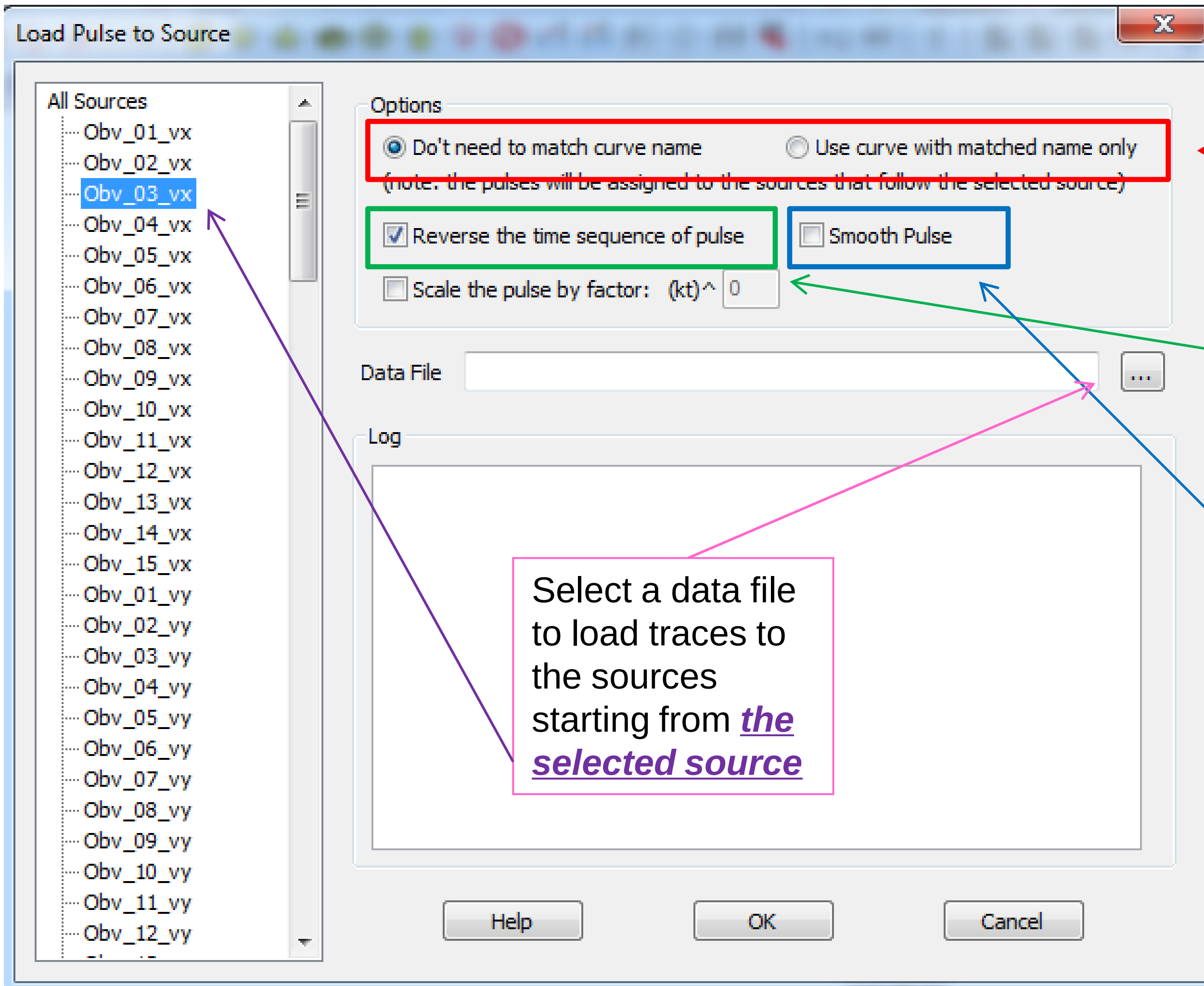
In this method, only the traces which have a matching name to a source will be assigned to that source correctly, other traces will be skipped.

After loaded, the name of source which load pulse successfully will be reported in GUI log, as following



Load pulse to sources with options

This method has a control dialog as following



Select a data file to load traces to the sources starting from ***the selected source***

Whether need to use naming matching rule in loading traces

Whether need to reverse trace in loading

Whether need to remove the noise in traces

This trace loading method support 3 kinds of format, the code will automatically detecting the file format and load.

a. Pure ASCII data file with 2 columns data
b. Wavenology trace data, ASCII format, version 3.0
c. Wavenology simulation result ASCII file, version 1.0

a. 2 columns, pure ASCII data file

- One trace per file
- Two columns data. 1st column is time, 2nd is value
- MKSU as unit
- The line start with “%” or “#” will be considered as comment line and be skipped

b. Wavenology trace data, ASCII format, version 3.0

- Please refer to this page

c. Wavenology simulation result ASCII file, version 1.0

- It is the Wavenology simulation result file stored in folder
  - Projectname_res\observers
- Format as

%Wave Computation Technologies simulation waveform data, version 1.0 ::
%Time (ns)
%frames number
12
%frame start
0
%frame end
3.0005e-008
%frame step
1.7e-011
%frame length
1766
0.0000000e+000
0.0000000e+000
0.0000000e+000
…
0.0000000e+000
0.0000000e+000
0.0000000e+000
0.0000000e+000
…
0.0000000e+000

Information for all curves.
All curves has the same length with the same uniform interval.

1766 rows for the first curve.

1766 rows for the second curve.

Note: the curve sequence is the ACSII string index for all observers in the project. It may be different from the sequence shown in the project tree.

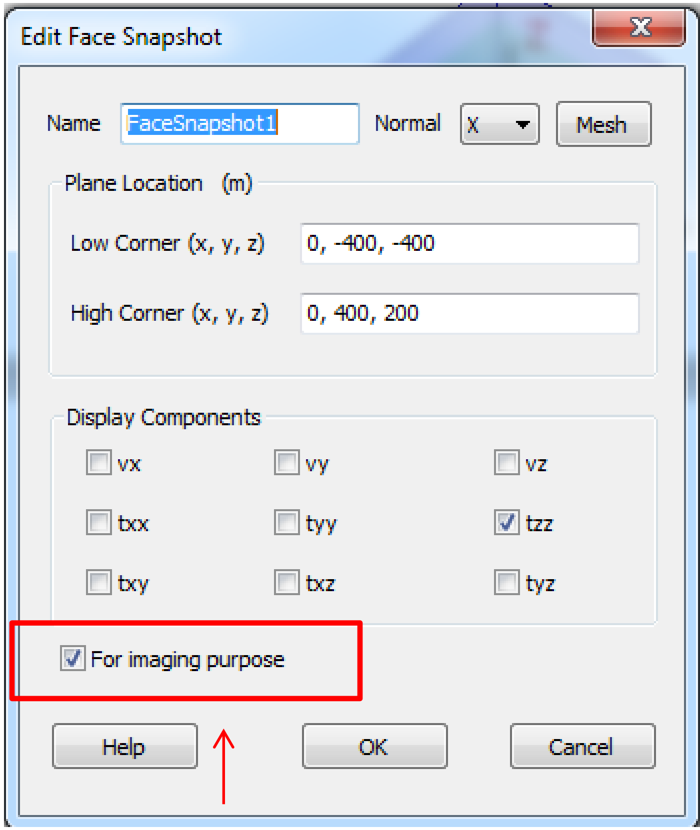
Define Snapshot for Imaging Purpose

Wavenology GUI provide two methods to generate images

A. snapshot to record field propagation, user can export any frame of a snapshot as an image

B. define a special snapshot for let Wavenology Imaging solver to find out a frame with a strongest energy point in the whole snapshot history

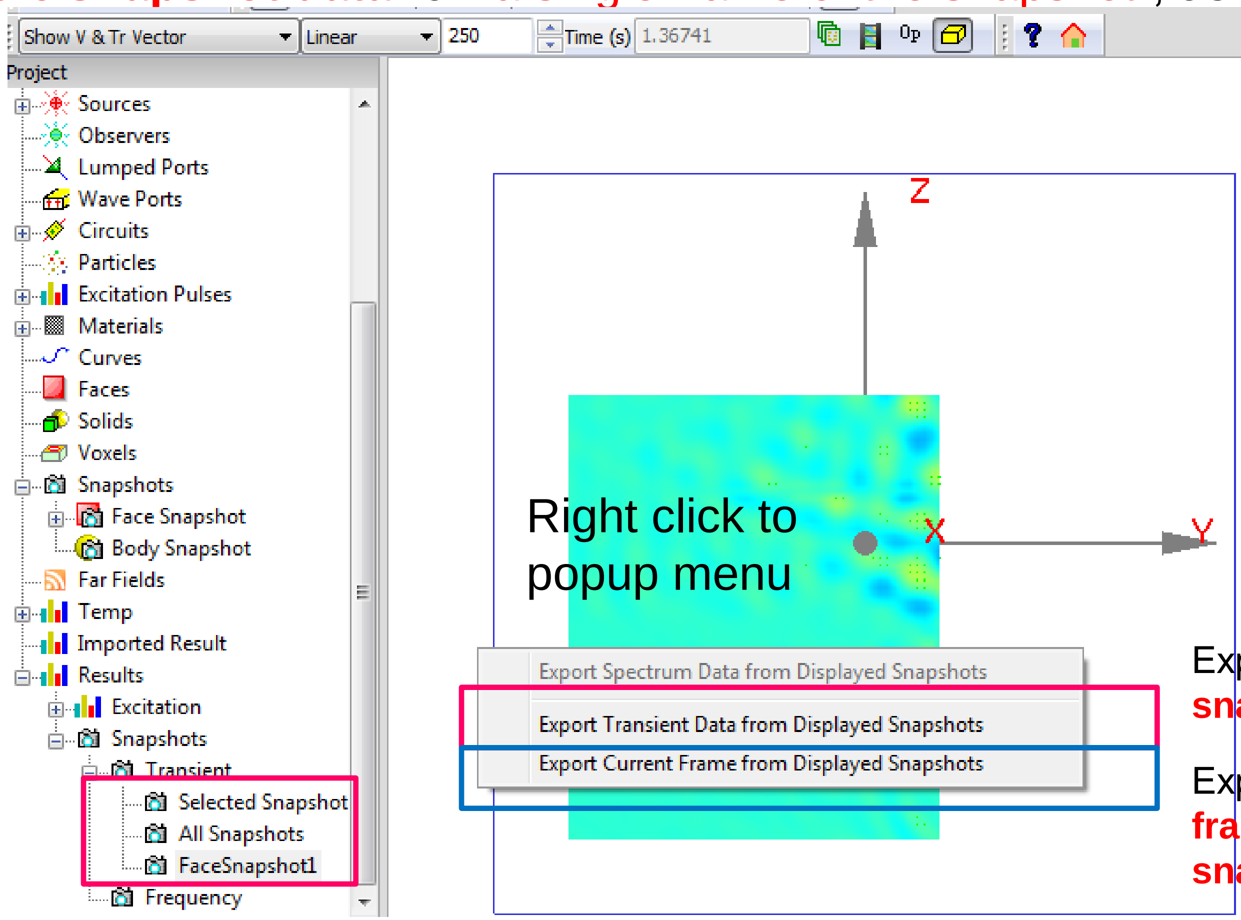
- ❑ this snapshot will generate two sets of data after simulation
    1) The regular transient field on this snapshot
    2) One image with a strongest energy point



Without check this item, it is a regular type of snapshot (type A). With checked item, it is a type B snapshot.

Display the Field Propagation in the Snapshot & Export Image

➢ in the transient snapshot displaying mode, there are two menu items to export “**whole snapshot data**” or “a single frame of the snapshot”, as following



Double click to enter snapshot displaying mode

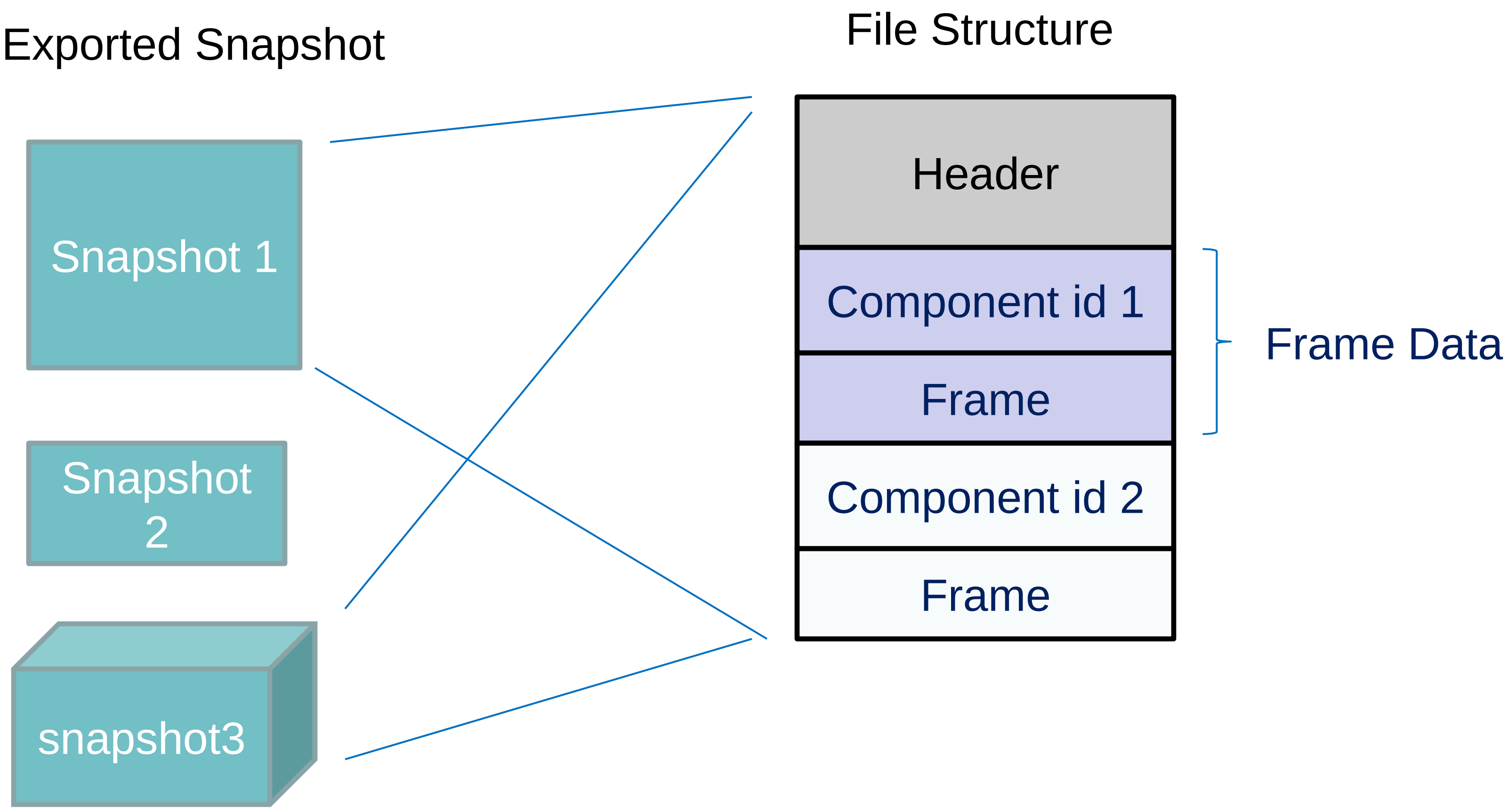
Export “**whole snapshot data**”

Export “**a single frame of the snapshot**”

Exported Transient Data File Format

Single Frame

Rule: one snapshot one file



Header

- 128 Bytes Text
- Number of components[int32]
- 32 Bytes Text (name of component 1)
- 32 Bytes Text (name of component k)
- Δt [float]
- Frame length[int32]
- Number of capture points in this snapshot[int32]
- Array of 3D position (x,y,z) [float]

Frame Data

- Component index[int32]
- Frame index[float]
- Value on capture points in this snapshot[float]

Following matlab code show how to load a single component from a Single Frame file

```
clear all;

% open file
fid = fopen( 'fs.bin', 'rb' ); % target file
if( fid == -1 )    return;  end;

%%% read header
sHeader = fread( fid, 128, '*char' );
nComp = fread( fid, 1, '*int32' );
for k = 1 : nComp;
    sCompName = fread( fid, 32, '*char' );
    sComps(k, :) = sCompName';
end;
dt = fread( fid, 1, '*float32' );
nFrameLength = fread( fid, 1, '*int32' );
nPt = fread( fid, 1, '*int32' );
pos = fread( fid, double(nPt*3), '*float32' );
pos = reshape( pos, 3, length(pos)/3 );
x_pos = pos(1,:);
y_pos = pos(2,:);
z_pos = pos(3,:);

%%% read one componet, if there are several
components in this file
%%% change this part
CompId = fread( fid, 1, '*int32' );

FrameId = fread( fid, 1, '*float32' );

val = fread( fid, double(nPt), '*float32' );

%%%%% close file  %%%%%%%%%%%
fclose( fid );
```

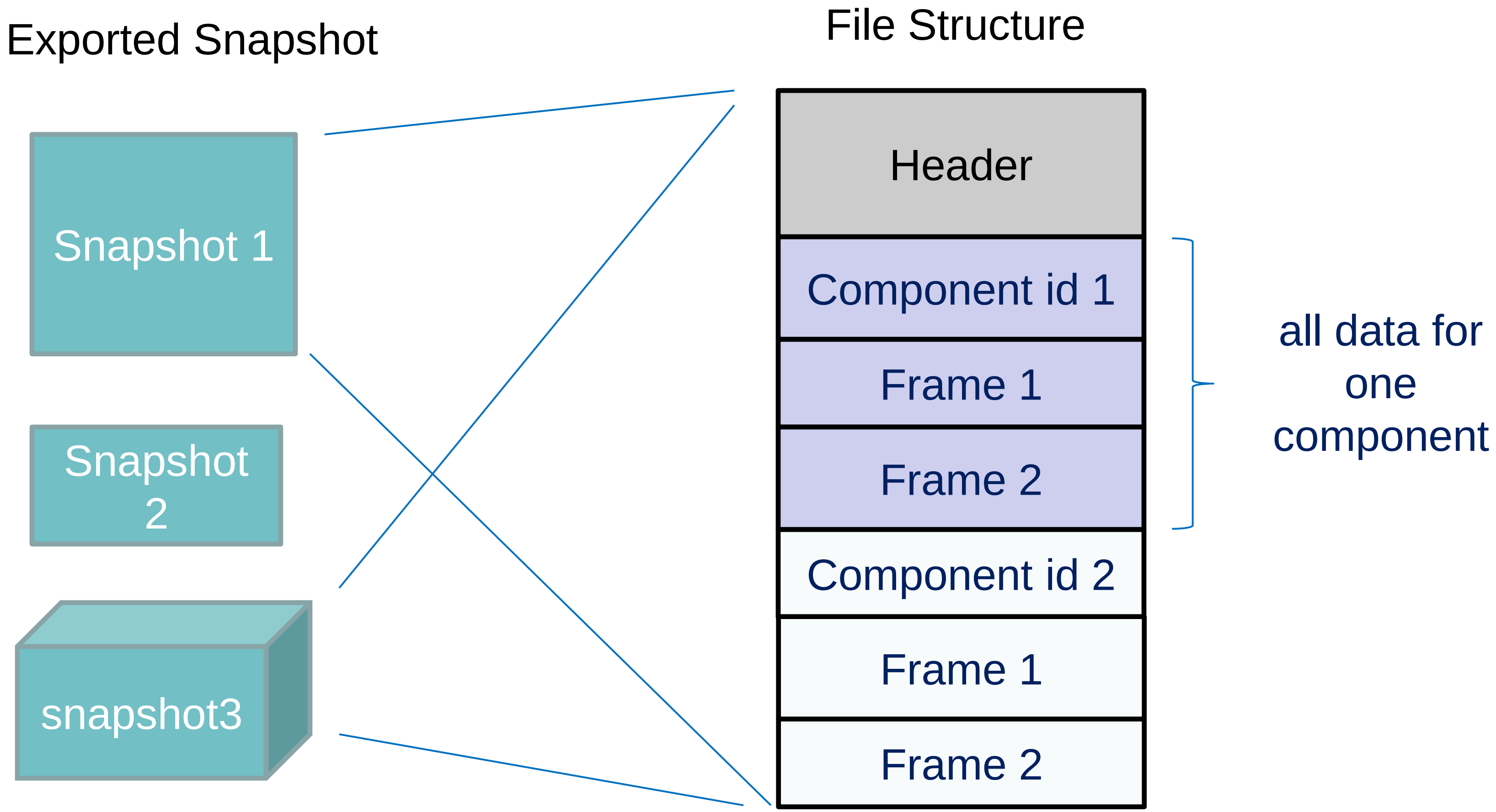
For a 2D face snapshot or a 3D rectangular volume snapshot, the mesh point in the snapshot is generated by following loop:

```
for x
    for y
        for z
```

Exported Transient Data File Format

Whole Snapshot

Rule: one snapshot one file



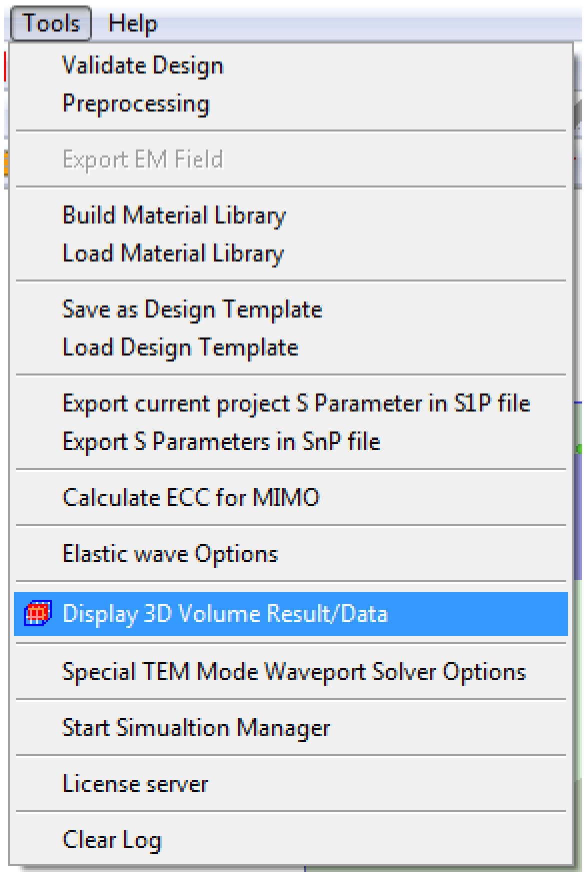
Header

128 Bytes Text
Number of components[int32]
32 Bytes Text (name of component 1)
32 Bytes Text (name of component k)
Δt [float]
Frame length[int32]
Number of capture points in this snapshot[int32]
Array of 3D position (x,y,z) [float]

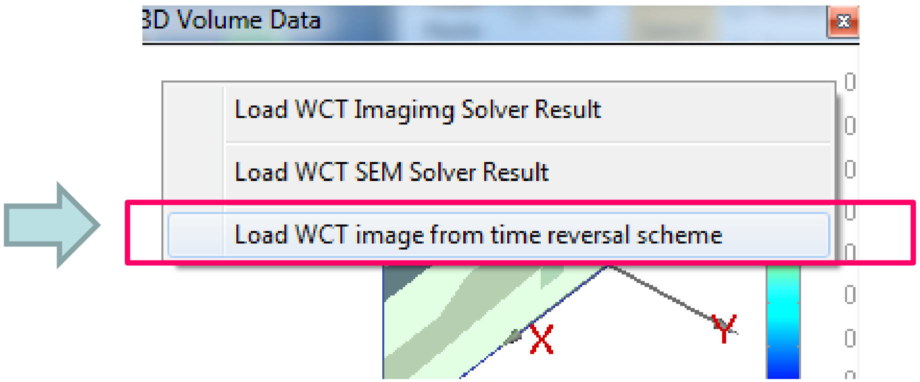
Data for One Component

Component index[int32]	
Frame index[float]	1st frame
Value on capture points in this snapshot[float]	
Frame index[float]	2nd frame
Value on capture points in this snapshot[float]	

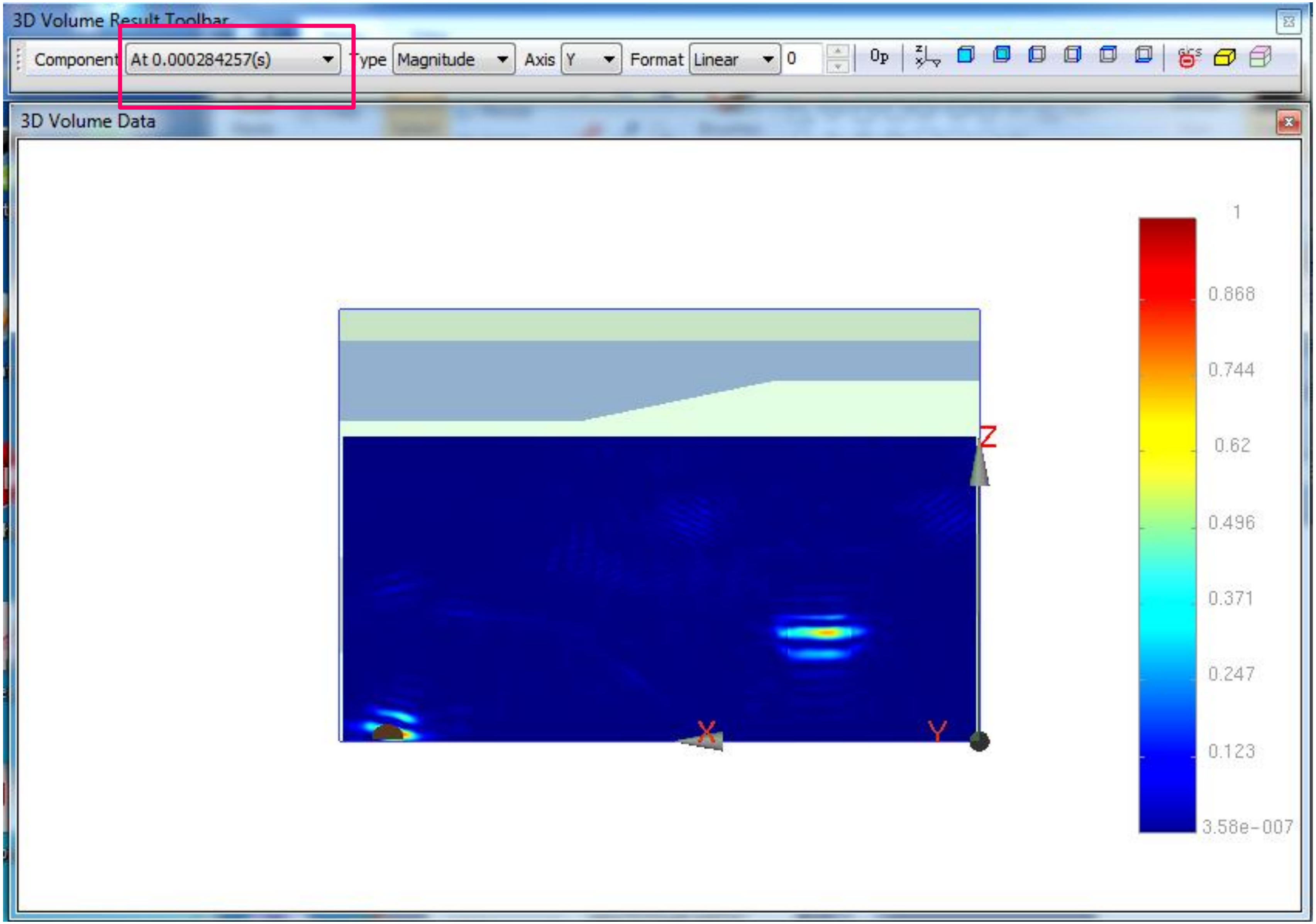
Display the Image with a Strongest Energy Point



Follow these steps to load a image on a snapshot with a strongest energy point in the snapshot history.



The time of this frame



This image file is in: ***project_name_res \tri\snapshot_name.img***

Format as,

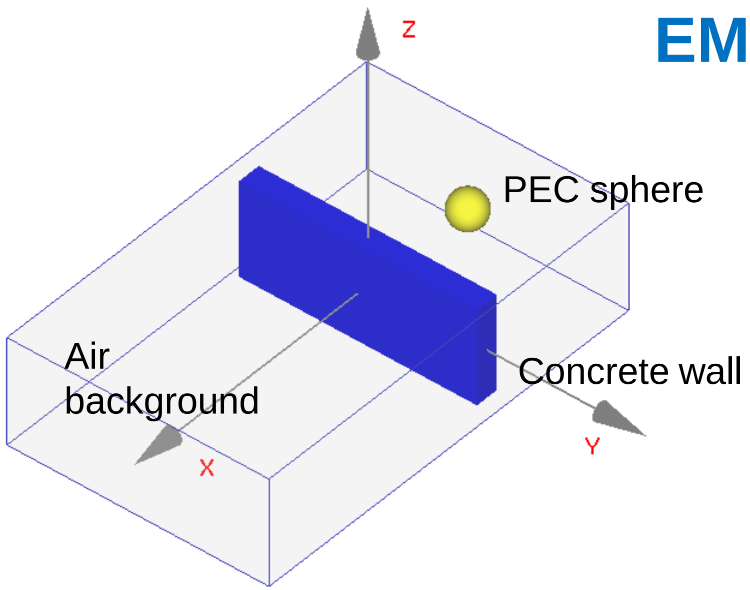
Data Type	Length in Byte	Meaning
char	128	comment
int32	4	Version number, value is 3
double	8	Frame time (MKSU)
char	120	reserved
int32	4	Number of cell in X, **nx**
double	8*nx	Cell positions in x
int32	4	Number of cell in Y, **ny**
double	8*ny	Cell positions in y
int32	4	Number of cell in Y, **nz**
double	8*nz	Cell positions in z
double	nx*ny*nz	Value for cells

Following matlab code show how to load a image file which has the strongest energy point in the snapshot history

```
clear all;

% open file
fid = fopen( 'FaceSnapshot1.img', 'rb' ); %
target file
if( fid == -1 )    return;  end;

%%% read header
sHeader = fread( fid, 128, '*char' );
nVer = fread( fid, 1, '*int32' );
dt = fread( fid, 1, '*double' );
sExt = fread( fid, 128-8, '*char' );

nx = fread( fid, 1, '*int32' );
xpos = fread( fid, double(nx), '*double' );

ny = fread( fid, 1, '*int32' );
ypos = fread( fid, double(ny), '*double' );

nz = fread( fid, 1, '*int32' );
zpos = fread( fid, double(nz), '*double' );

val = fread( fid, double(nx*ny*nz), '*double' );

%%%%%% close file  %%%%%%%%%%%%
fclose( fid );

%%%%%% display image %%%%%%%%%%%
img = reshape( val, nz, ny, nx );
img = squeeze( img );

figure;
pcolor( img );

colorbar;
shading flat;
```

Case (1): Detect a PEC Sphere behind a Wall

EM Solver



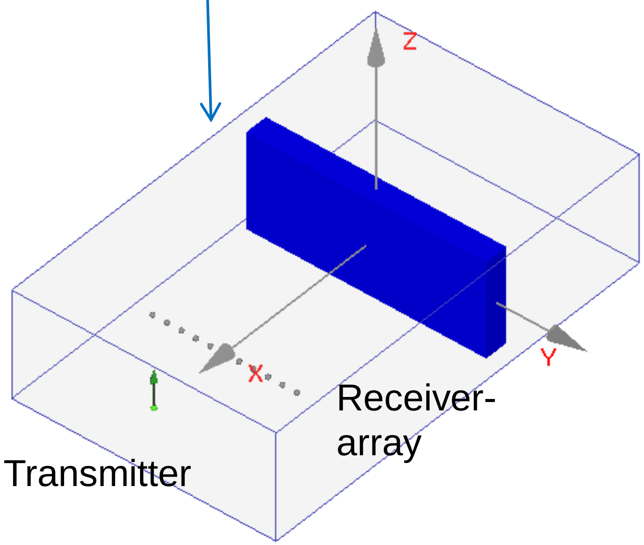
This case shows how to setup a time-reversal project to detect a object by the scattered signal.

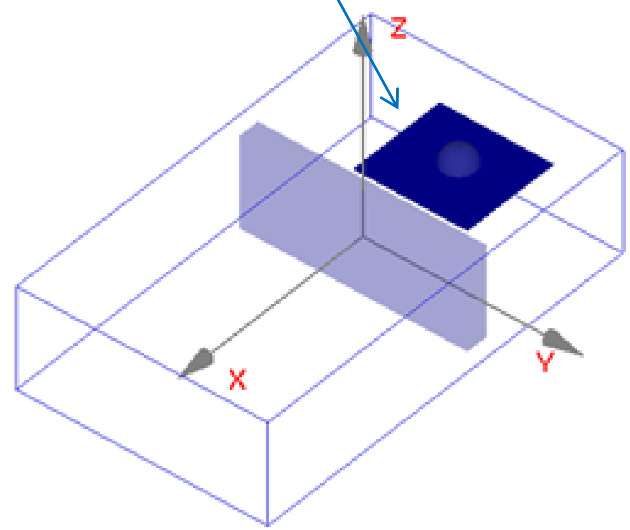
Note:

1. Due to the time-reversal is an approximate method, it cannot focus on a small target. It is suggested that the target size is larger than one wavelength.
2. The focus quality of the time-reversal method depends on the aperture size of the re-radiation transmitter array, i.e., larger size of the array can obtain better focus.
3. The clutter between the re-radiation transmitter array and the target will increase the aperture and significantly improve the quality of focus. If there is not clutter between the re-radiation transmitter array and the target, in order to get a good focus, the size of array should be much larger.

The basic idea of this case is

- we have a T-R (Transmitter-Receiver Array) system in front a wall
- we know the source excitation and the measurement on the receiver array
- we would like to find out whether there is any object behind the wall and where it is, through checking the focus of field in face snapshot

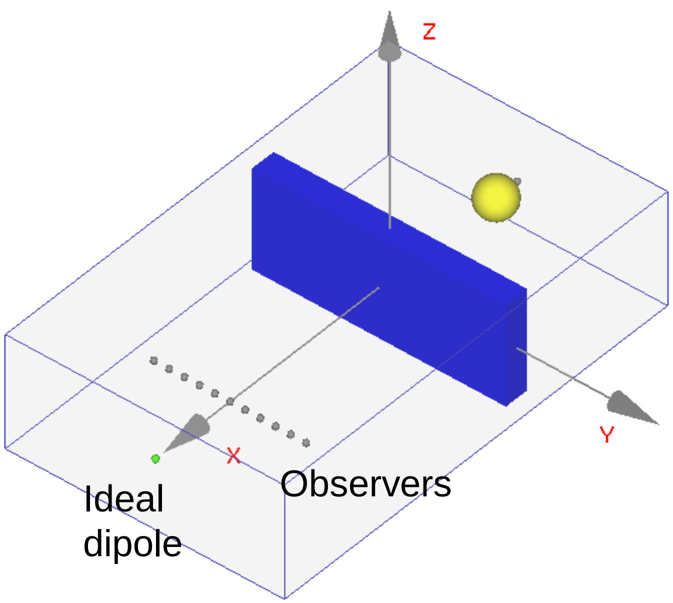




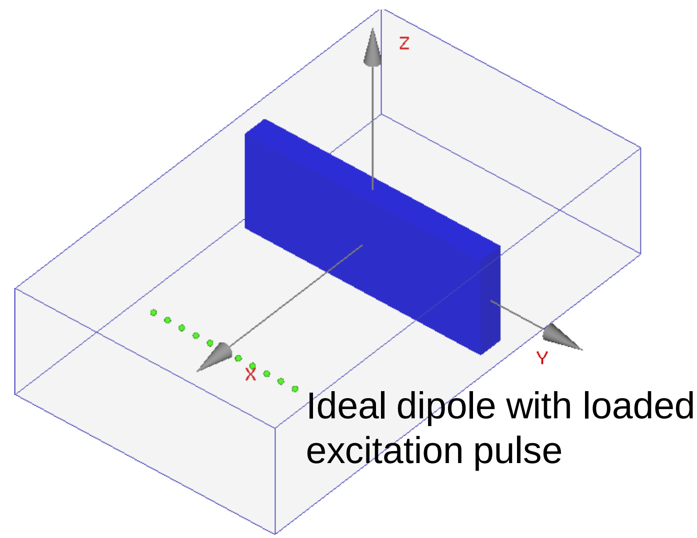
Following, we will demonstrate how to

1) obtain the scattered signal from a PEC sphere behind a wall for this T-R system
2) Set up a time-reversal simulation to check whether there is object after wall through the scattered signal

Project to obtain the scattered signal



The time-reversal project

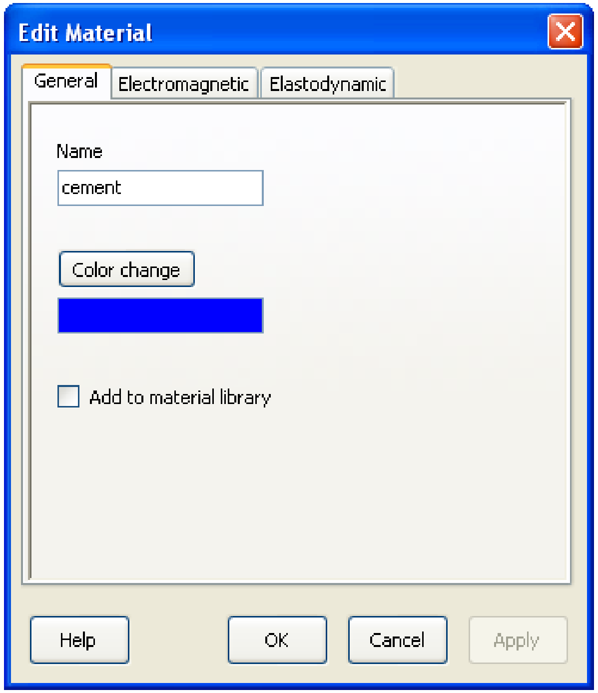


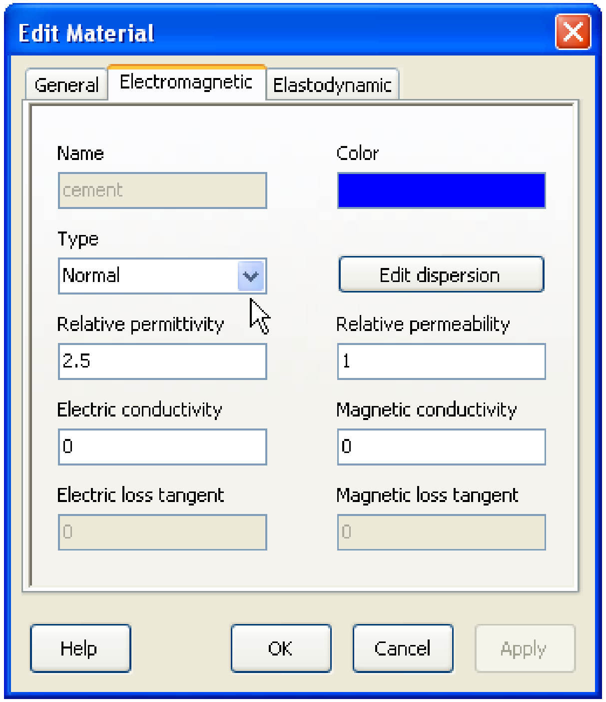
Part 1: the project to obtain the scattered signal from a PEC sphere behind a wall

Steps:

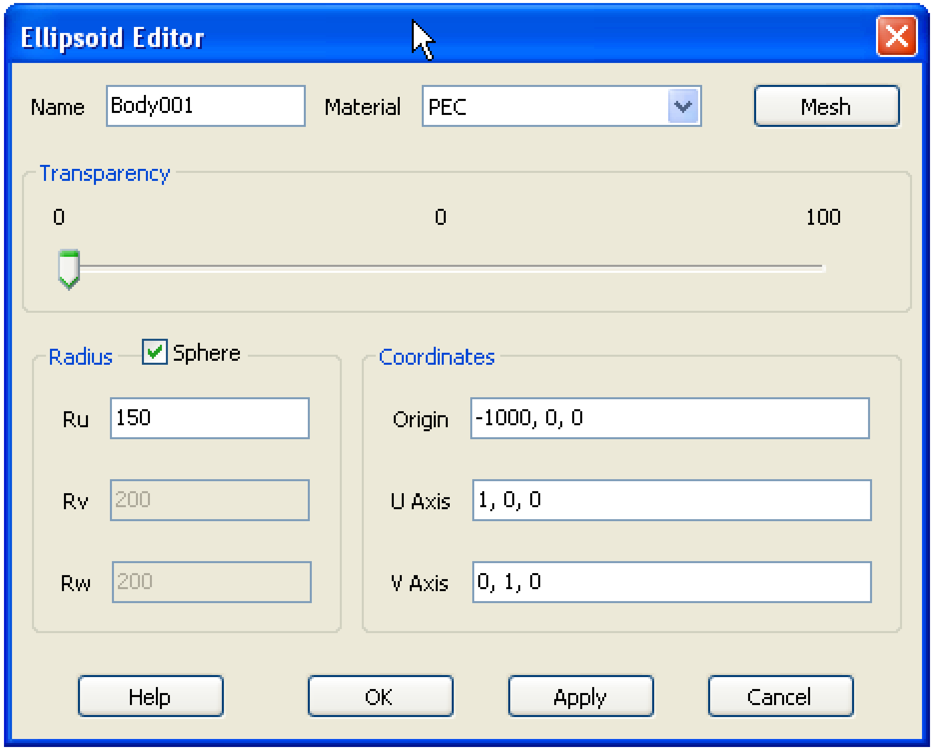
1. Create a new folder as single_wall_single_target. Then use WCT GUI create a new project, save it under the sub-folder tot with name 1.

2. Define a new material: cement

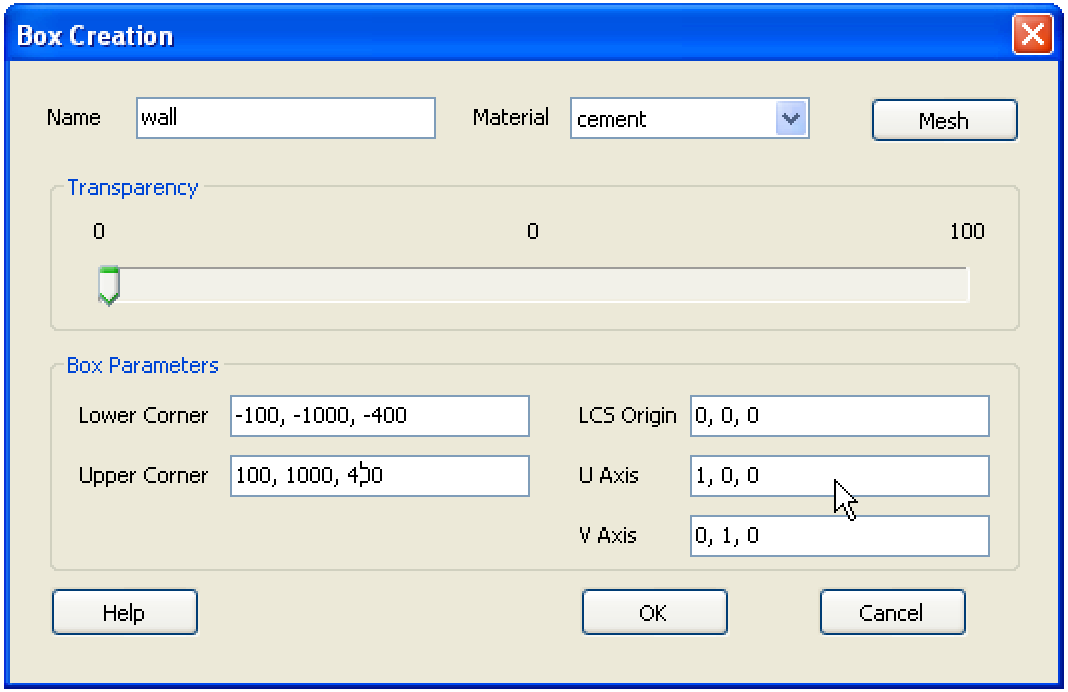




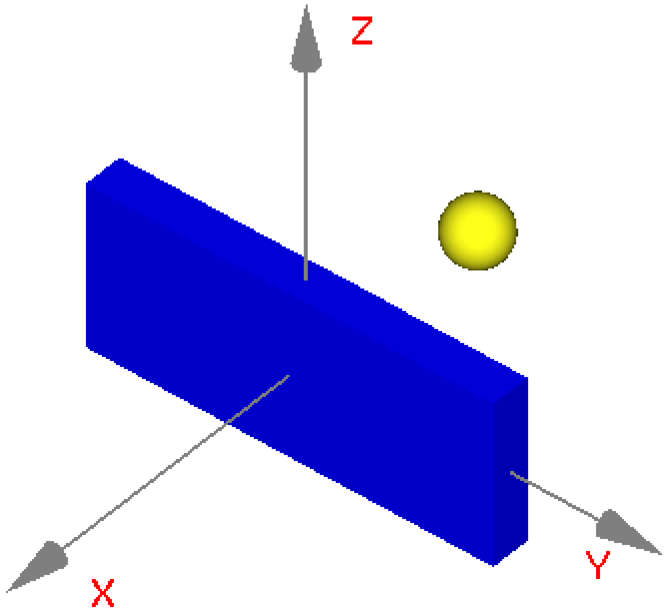
3. Define the target: PEC sphere (unit: mm)



4. Define clutter: wall (unit: mm)

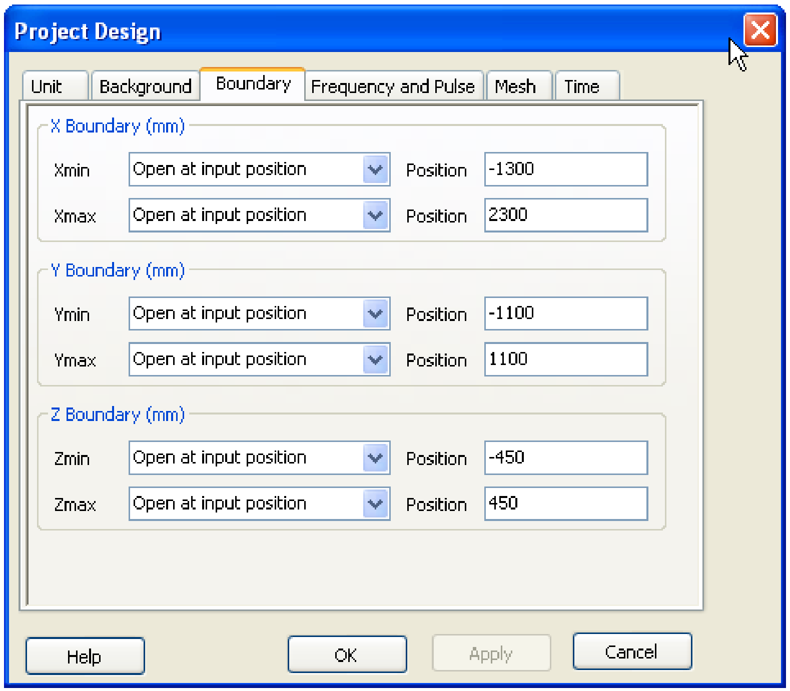


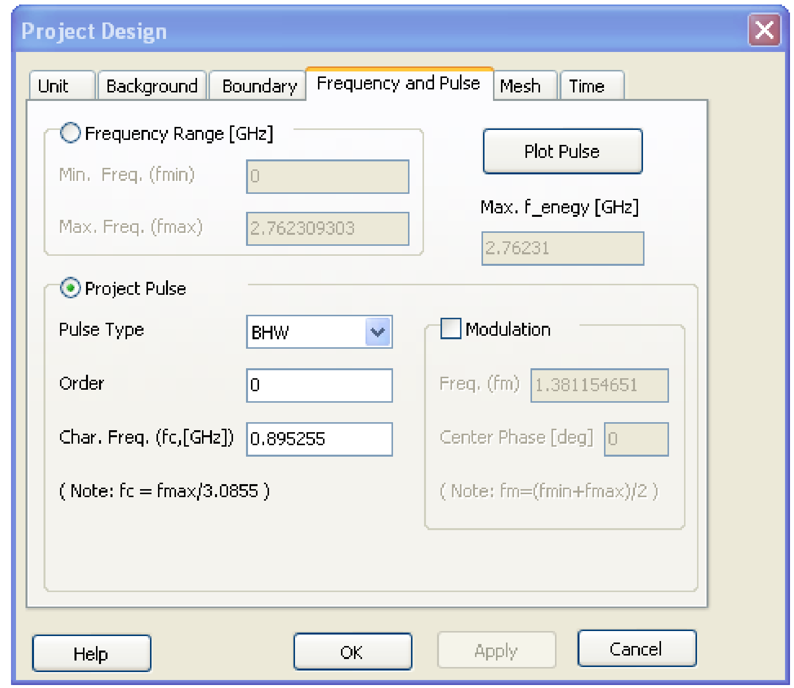
The position of Sphere & Wall is shown in the following Figure.



5. Setup project unit, background, boundary, excitation pulse, mesh and simulation time.

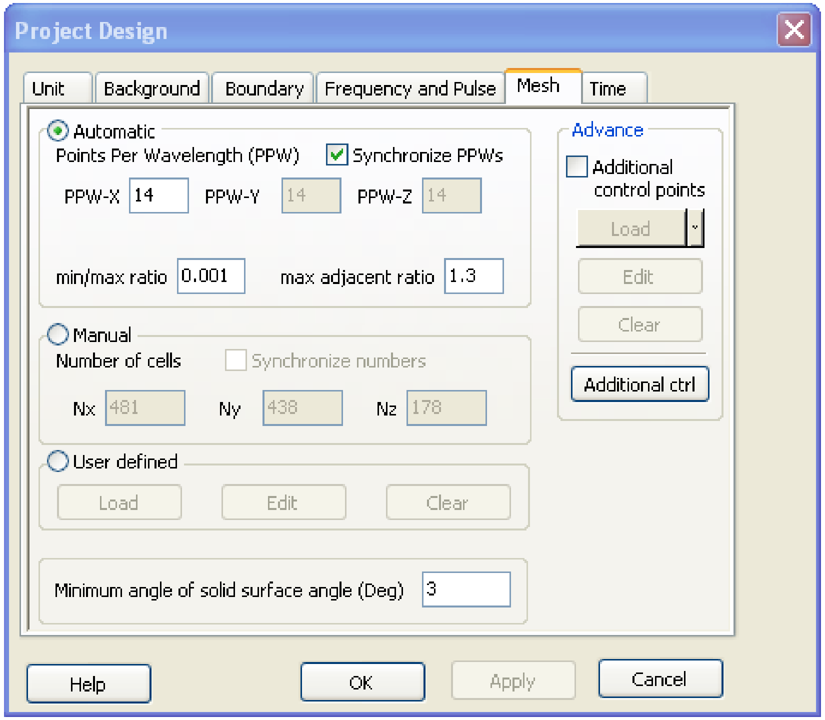
For the unit, we use default setting. The background material is default material, air.

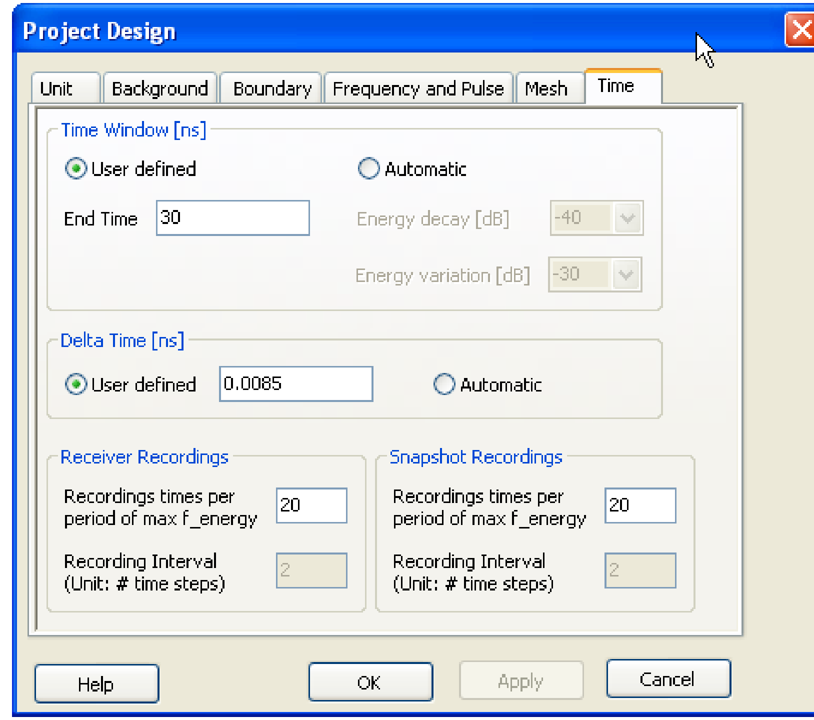




In this case, we use 0th order BHW wideband signal. And we set up the size of sphere about one wavelength of the frequency which has the maximum energy in this wideband excitation. 30 cm sphere -> 1λ≈30 cm->f_{max_enrgy}≈1 GHz.

In our setup, we set f_{max}=2.76 GHz.





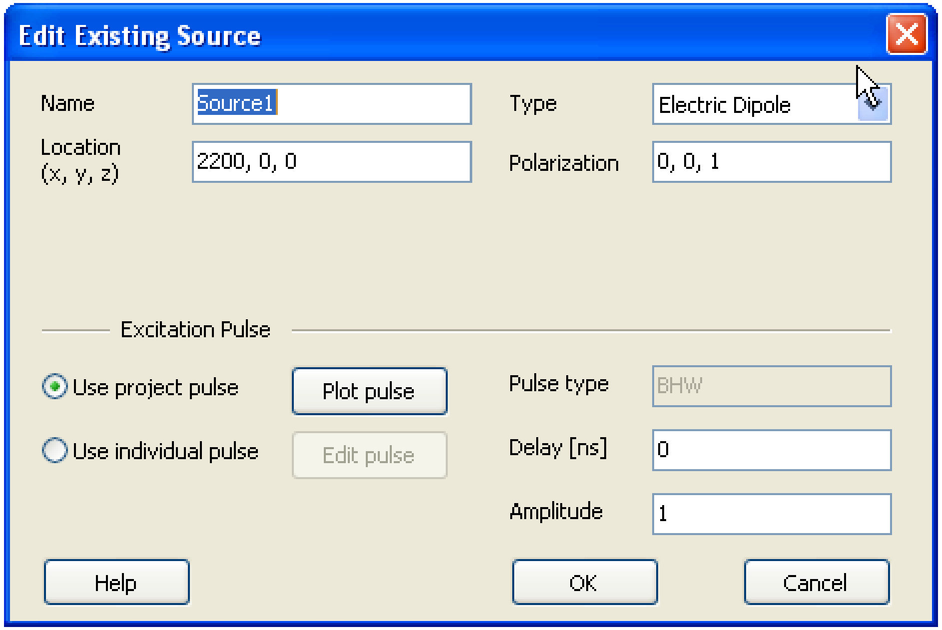
Sampling density is set to be 14 points per wavelength

Because for time-reversal case, we only want to check the transient result. So, we manually set up the simulation time window as 30 ns.

For the FDTD method, a high accurate scattered field can be obtained by **incident & total field cases** if these two cases use the same Δt and mesh.

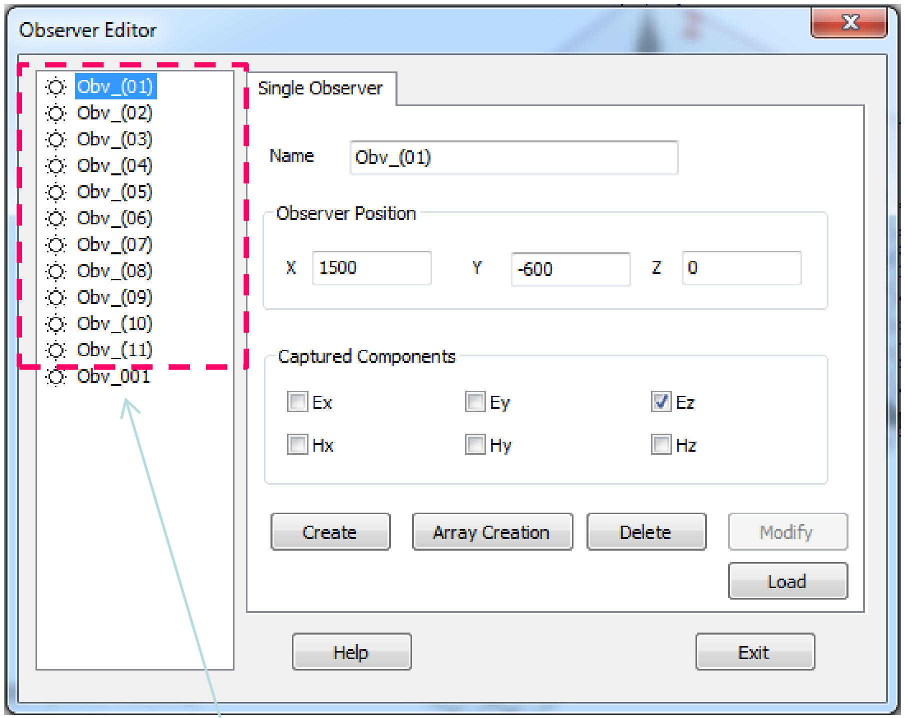
Here, we set Δt as 0.0085 ns.

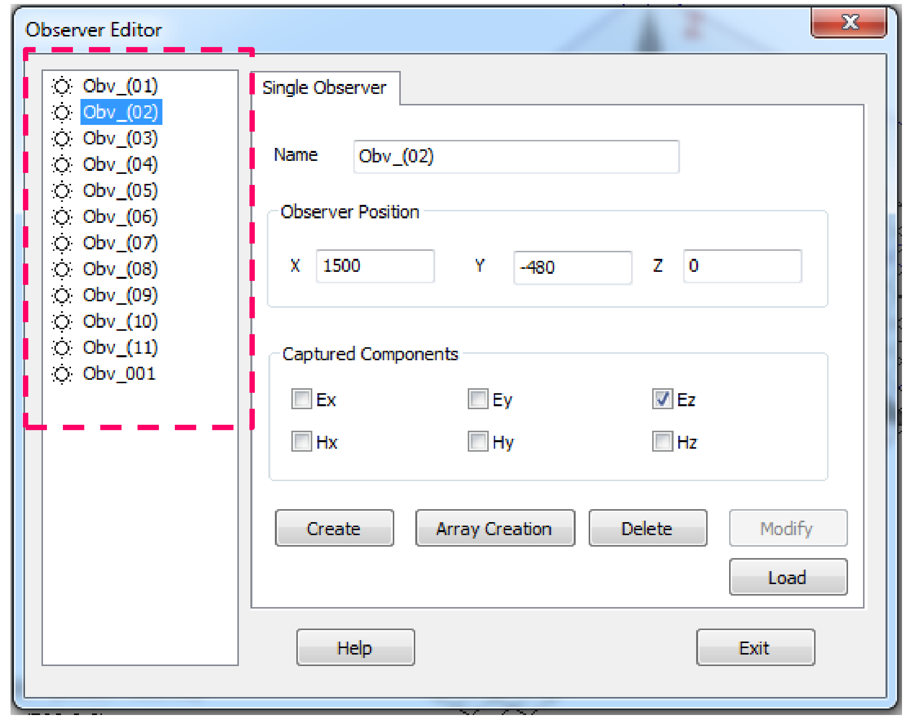
6. Setup an ideal dipole source.



7. Setup point observers.

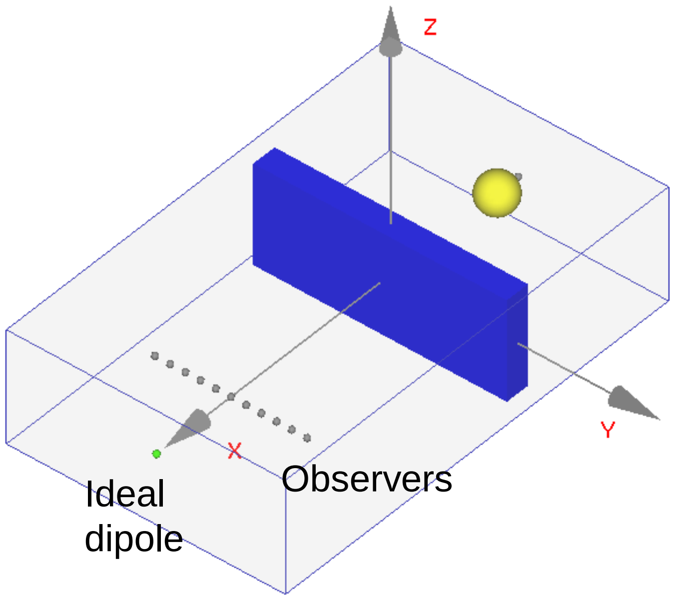
Eleven observers in a line with a distance of 120 mm. Recording Ez field only.





Obv_001 is for monitoring only. It is not a part of reverse-radiation array.

8. Simulate the case to obtain the total field on observers.

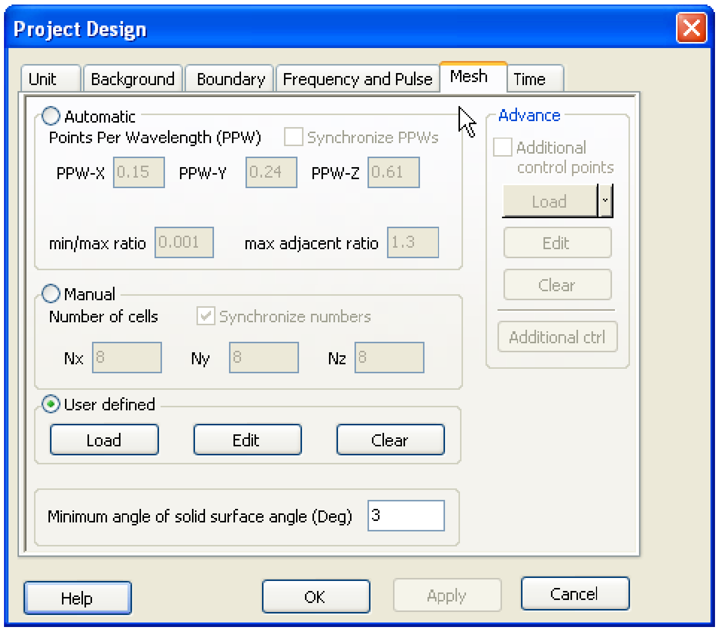


9. Setup a case to obtain the incident field on observers.

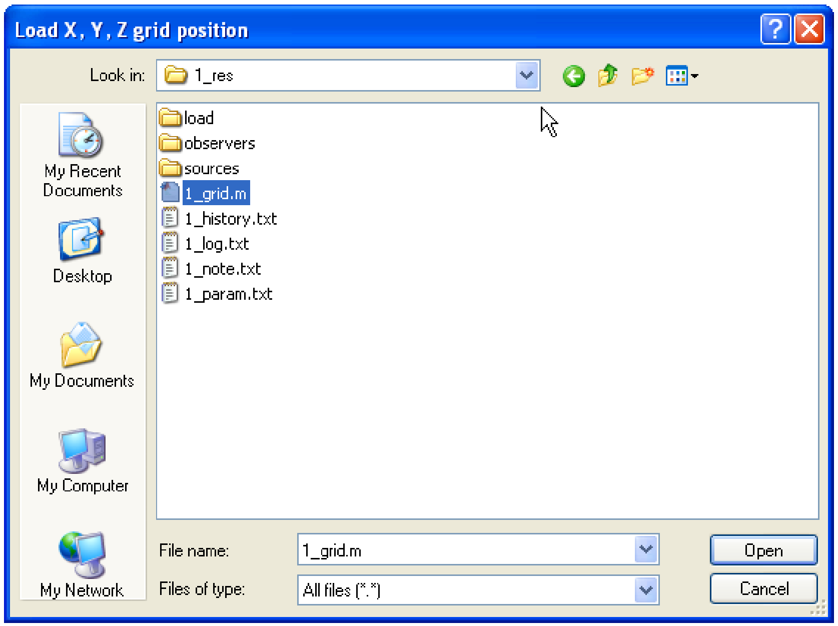
9.1 Create a new sub-folder under single_wall_single_target as inc. Use SaveAs function to save the total field case to inc sub-folder with name 1.

9.2 Remove PEC sphere, or change the material of sphere as background material air.

9.3 We need to let this case has the same mesh as the total field case. But due to the sphere does not exist. If we use Automatic meshing method, we cannot generate the same mesh as the total field case. Thus, we need to load the mesh directly from the total field case.



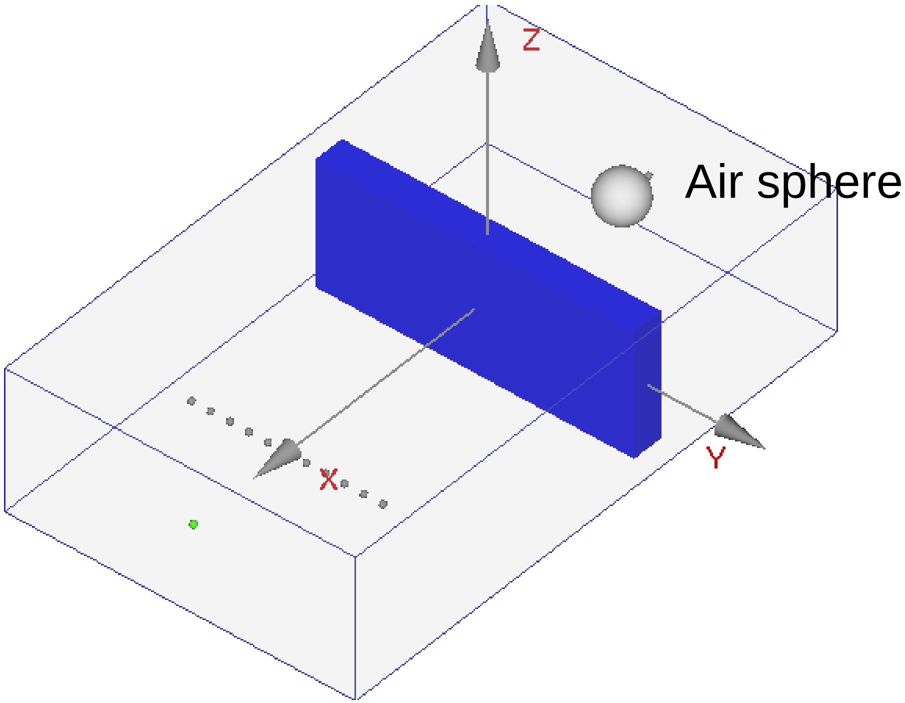
Switch to “User defined” mesh option, press “Load” button



In file dialog, goto single_wall_single_target\tot\1_res\, there is a file “1_grid.m”, which has the mesh information of the total field case, load this mesh into this incident field case.

Note: WCT GUI generate the case mesh automatically after the simulation or preprocessing.

10. Simulate this incident field case to obtain the incident field on observers.



11. Calculate the scattered field from the total field case and the incident field case.

$E_{sct}=E_{tot} – E_{inc}$

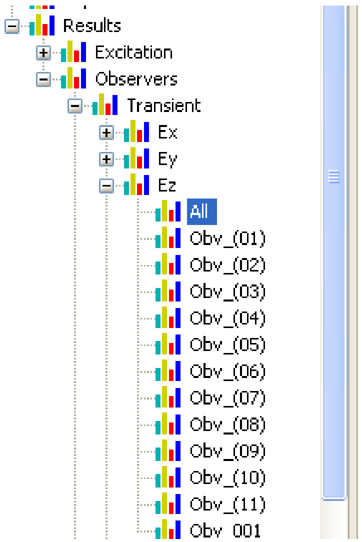
E_{tot} is the total field, from the result on the observers in the total field case. E_{inc} is the incident field, from the result on the observers in the incident field case.

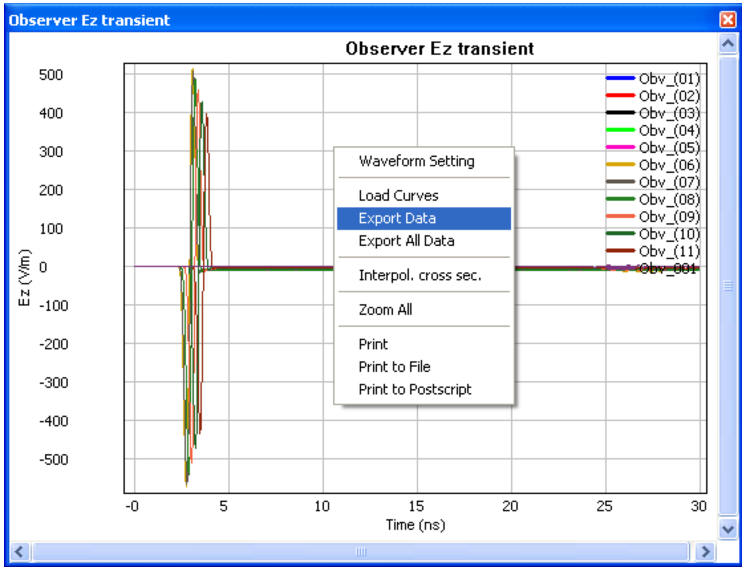
In this case, the ***z*** polarized dipole source makes the ***z*** component in the case is the dominant component. Therefore, in our time-reversal case, we will use Ez to reverse-radiate only.

The transient received signal on receiver can be obtained by two methods:

1) Export from 2D result canvas.
2) Directly use the simulation result data files.

A. Export Data from 2D result canvas.
For example, we will export Ez on all receivers.





Double click this tree-node

On 2D canvas, right click mouse to get this menu. Export all data curves.

Data format of export data file, it is compatible for ***Matlab*** ASCII data file.

```
%Wave Computation Technologies simulation waveform data, version 3.0
%created at 08/23/12 21:55:00 in GMT
%==================================
%Title      Observer Ez transient
%X-Lin      1e-009    Unit:ns
%Y-Lin      1    Unit:V/m
%Sub-Title  Obv_(01)
1766  1766  %Size Nx=Ny
0.0000000e+000  0.0000000e+000
1.7000000e-002  0.0000000e+000
3.4000000e-002  0.0000000e+000
5.1000000e-002  0.0000000e+000
6.8000000e-002  0.0000000e+000
8.5000000e-002  0.0000000e+000
1.0200000e-001  0.0000000e+000
1.1900000e-001  0.0000000e+000
1.3600000e-001  0.0000000e+000
1.5300000e-001  0.0000000e+000
1.7000000e-001  0.0000000e+000
…
%Sub-Title  Obv_(02)
1766 1766  %Size Nx=Ny
0.0000000e+000  0.0000000e+000
1.7000000e-002  0.0000000e+000
…
```

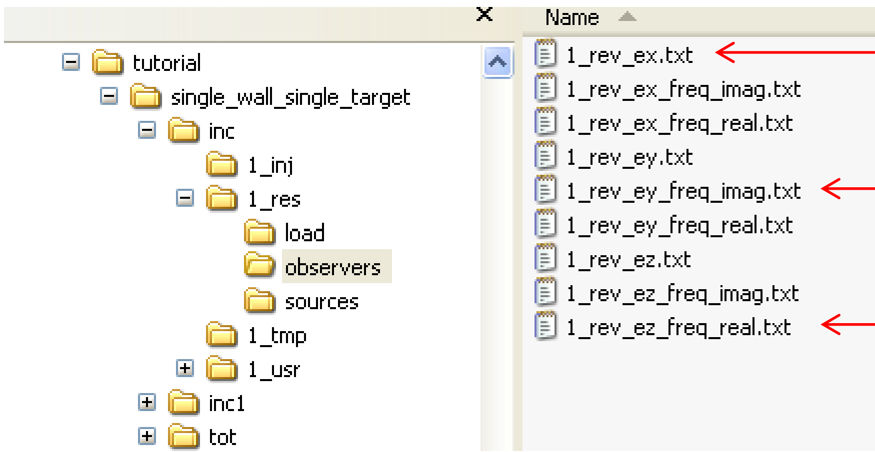
Header

1st curve name

How many data (rows) in this curve + real data. 1st column is time, 2nd column is magnitude.
For this curve, there are 1766 recorded data. It means the 1766 rows after the line “1766 1766 %Size Nx=Ny“ is the real data for curve 1.

2nd curve, same format as curve 1

B. Directly use the simulation result data files.
The simulation result for each case has following structure:



Ex transient result for all observers

Imaginary part of Ey freq. result for all observers

Real part of Ez freq. result for all observers

x*_res** is simulation result folder. (x*** is project name)

The sub-folder ***"observers"*** has all results for observers

Data format of simulation data file, it is compatible for ***Matlab*** ASCII data file.

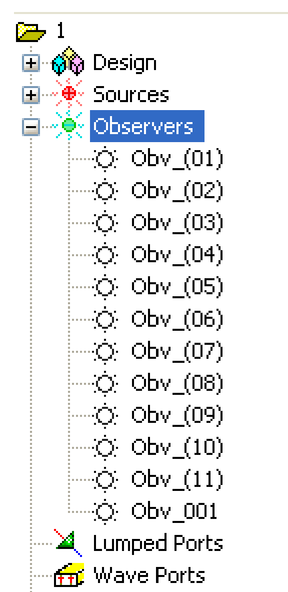
```
%Wave Computation Technologies simulation waveform data, version 1.0 ::
%Time (ns)
%frames number
12
%frame start
0
%frame end
3.0005e-008
%frame step
1.7e-011
%frame length
1766
0.0000000e+000
0.0000000e+000
0.0000000e+000
…
0.0000000e+000
0.0000000e+000
0.0000000e+000
0.0000000e+000
…
0.0000000e+000
```

Information for all curves.
All curves has the same length with the same uniform interval.

1766 rows for the first curve.

1766 rows for the second curve.

Note: the curve index is the same as that shown in the project tree.



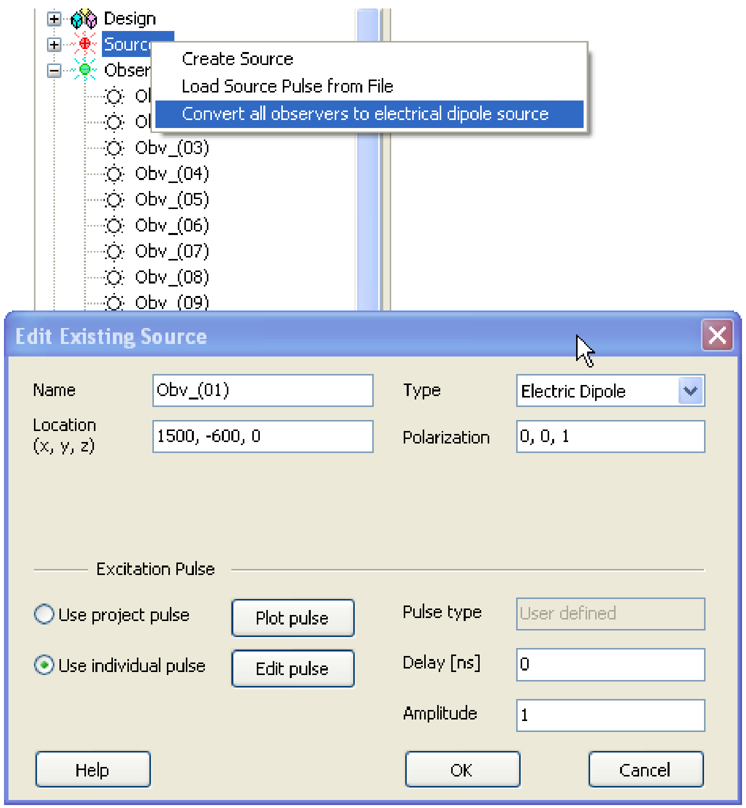
Following matlab code shows how to calculate the Ez_sct from the simulation data files directly, then reverse the time of the Ez_sct. (This matlab file for this code is single_wall_single_target\ sct_reverse_time.m)

```
clear all;

%%% load total field
Etot_tx = load( '.\tot\1_res\observers\1_rev_ez.txt' );
nFrame = Etot_tx( 1 );
t0 = Etot_tx( 2 );
t1 = Etot_tx( 3 );
dt = Etot_tx( 4 );
nLen = Etot_tx( 5 );
Etot_tx = reshape( Etot_tx(6:end), nLen, nFrame );

%%% load incident field
Einc_tx = load( '.\inc\1_res\observers\1_rev_ez.txt' );
Einc_tx = reshape( Einc_tx(6:end), nLen, nFrame );

%%% resampling
ND = 1;
Etot_tx = Etot_tx( 1:ND:end, : );
Einc_tx = Einc_tx( 1:ND:end, : );

dt = dt * ND;
nLen = length( Etot_tx( :, 1 ) );
t = [0:(nLen-1)] * dt;
t = t';
```

```
%%% make transient scattered field
Esct_tx = (Etot_tx - Einc_tx);
Esct_tx_1 = flipud( Esct_tx );   % reverse the result time

%% assume you will create Z field only
%%%%%%%%%%%%%%%%%%%%
fid = fopen( 'Z_field.txt', 'w' );

fprintf( fid, '%%Wave Computation Technologies simulation waveform data, version 3.0\n' );
fprintf( fid, '%%created at 08/21/12\n' );
fprintf( fid, '%%==================================\n' );
fprintf( fid, '%%Title      User defined pulse\n' );
fprintf( fid, '%%X-Lin      1    Unit: \n' );
fprintf( fid, '%%Y-Lin      1    Unit: \n' );

for k = 1 : 11,
   tt = [ t Esct_tx_1(:,k) ];

   fprintf( fid, '%%Sub-Title  Obv_(%02d)\n', k );           %% assign trace name to match a
source
   fprintf( fid, '%d\t %d\t %%Size Nx=Ny\n',  nLen, nLen );    %% this pulse length

   for j = 1 : nLen,
      fprintf( fid, '%g\t %g\t\n',  tt( j, 1 ), tt( j, 2 ) );
   end;
end;

fclose( fid );
```

Part 2: Project for the time-reversal imaging

1. Set up the reverse-radiation case.
   1) SaveAs the incident case to sub-folder “inc1” with name 1.
   2) delete the original source “source1”
   3) convert all observer s to dipole source (except the observer Obv_001)

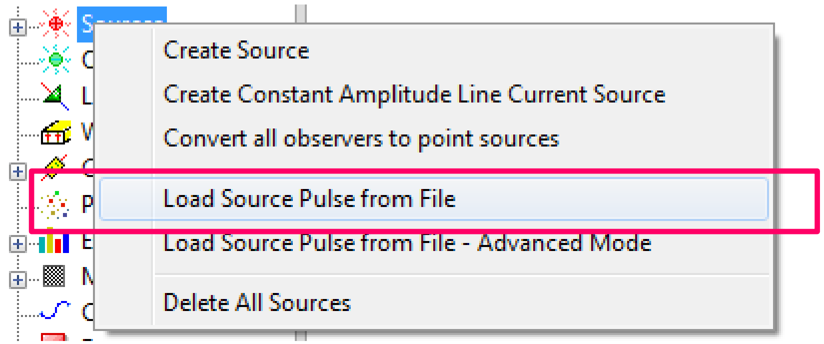


WCT GUI provides a menu to simplify this operation.
Right click mouse button on tree-node “Source”, there is menu item “Convert all observers to electrical dipole source”.

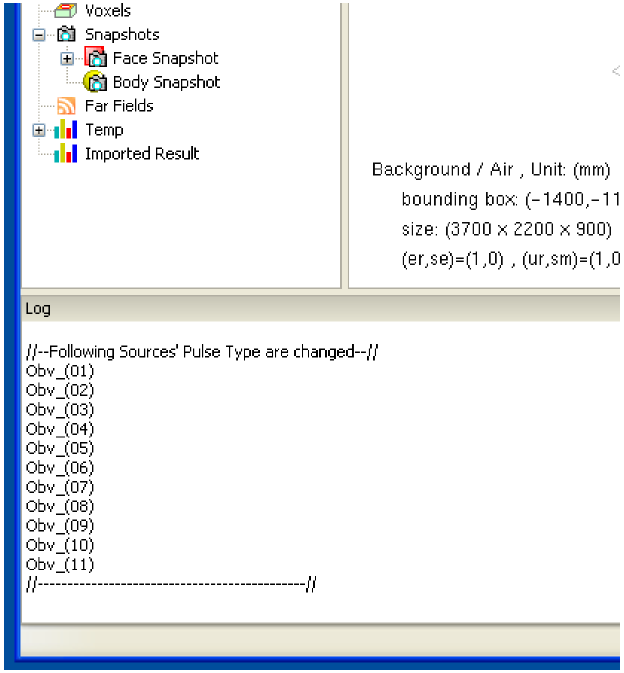
This is the converted source with z polarization.

Due to the receiver is capturing single component only, the conversion make the new sources have the same name of the original receivers.

4) Load excitations for all new sources.



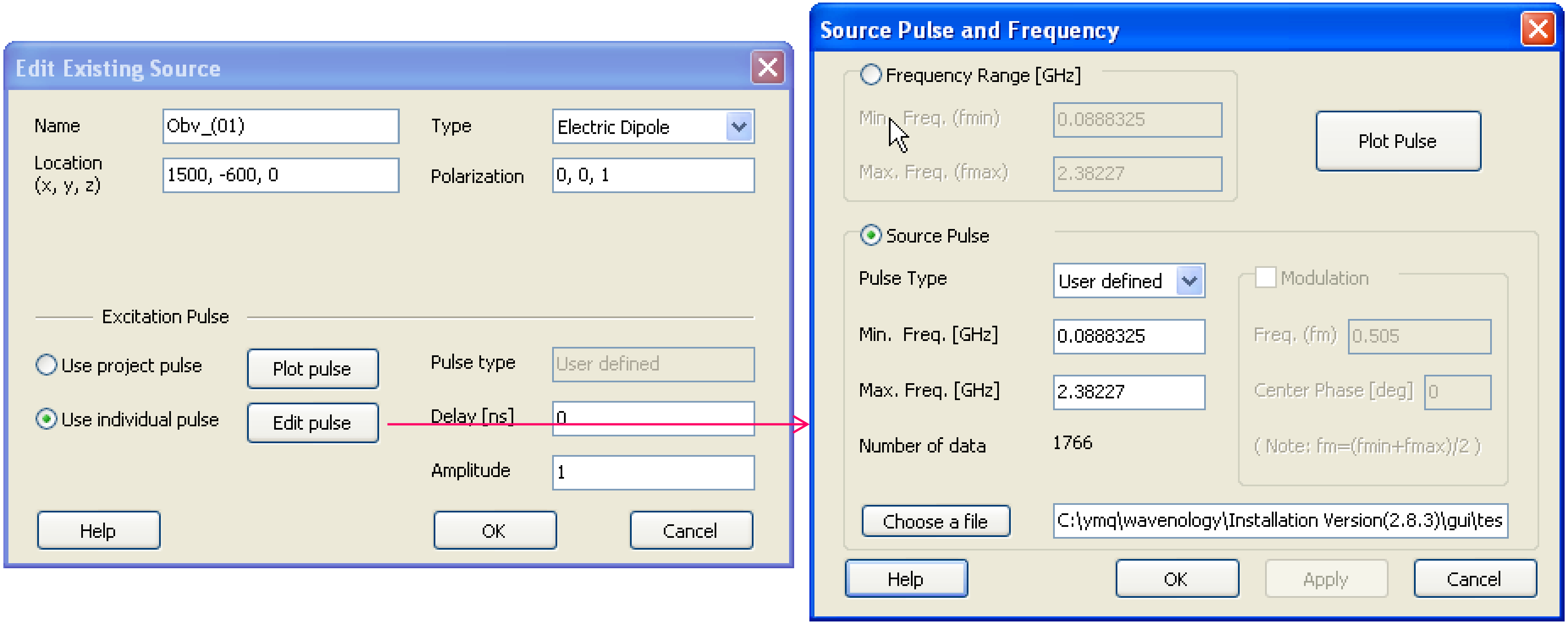
WCT GUI provides a menu to simplify this operation.
Right click mouse button on tree-node "Source", and use menu item "Load Source Pulse from File" load the reversed-time Ez_sct as the source pulse.



Select the file we create in the previous page, load the pulse for all sources.

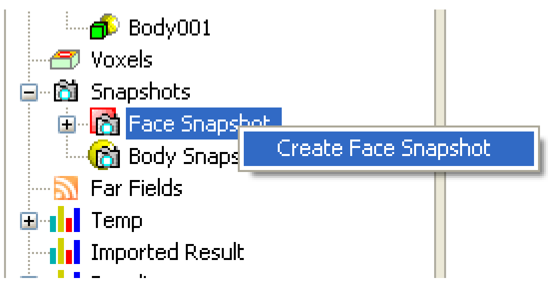
This function uses the name of curve in the file to match the source name. WCT GUI will report how many curve has been successfully loaded.

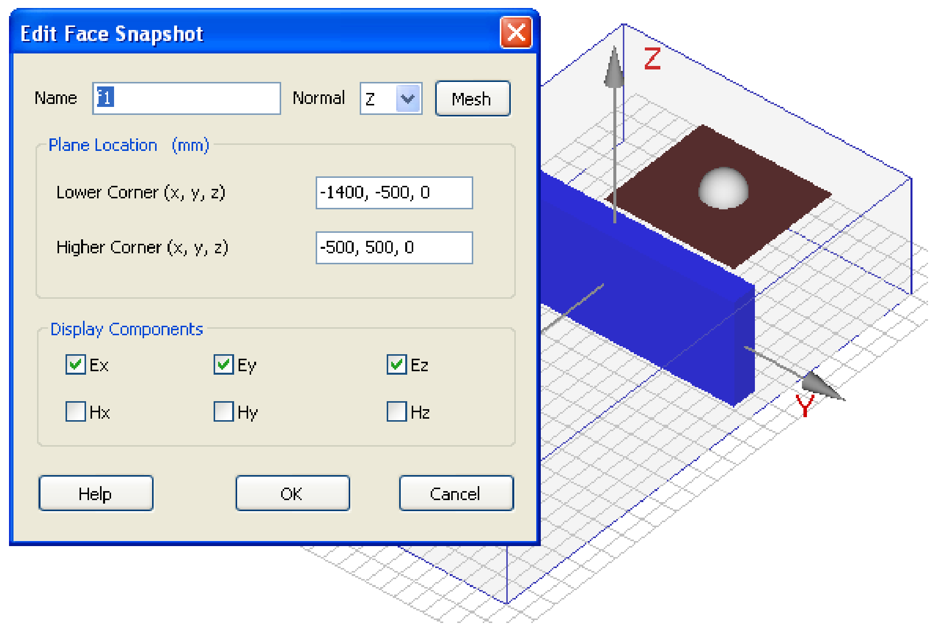
By checking the source pulse, all source pulse become "User defined"



2. User can modify the computation size, mesh and time to reduce simulation time. But this is a optional setup, user can still keep all setups.

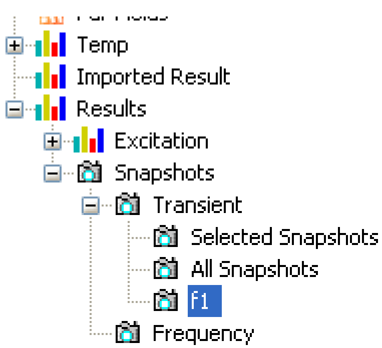
3. Define 2D face snapshot to record the result.

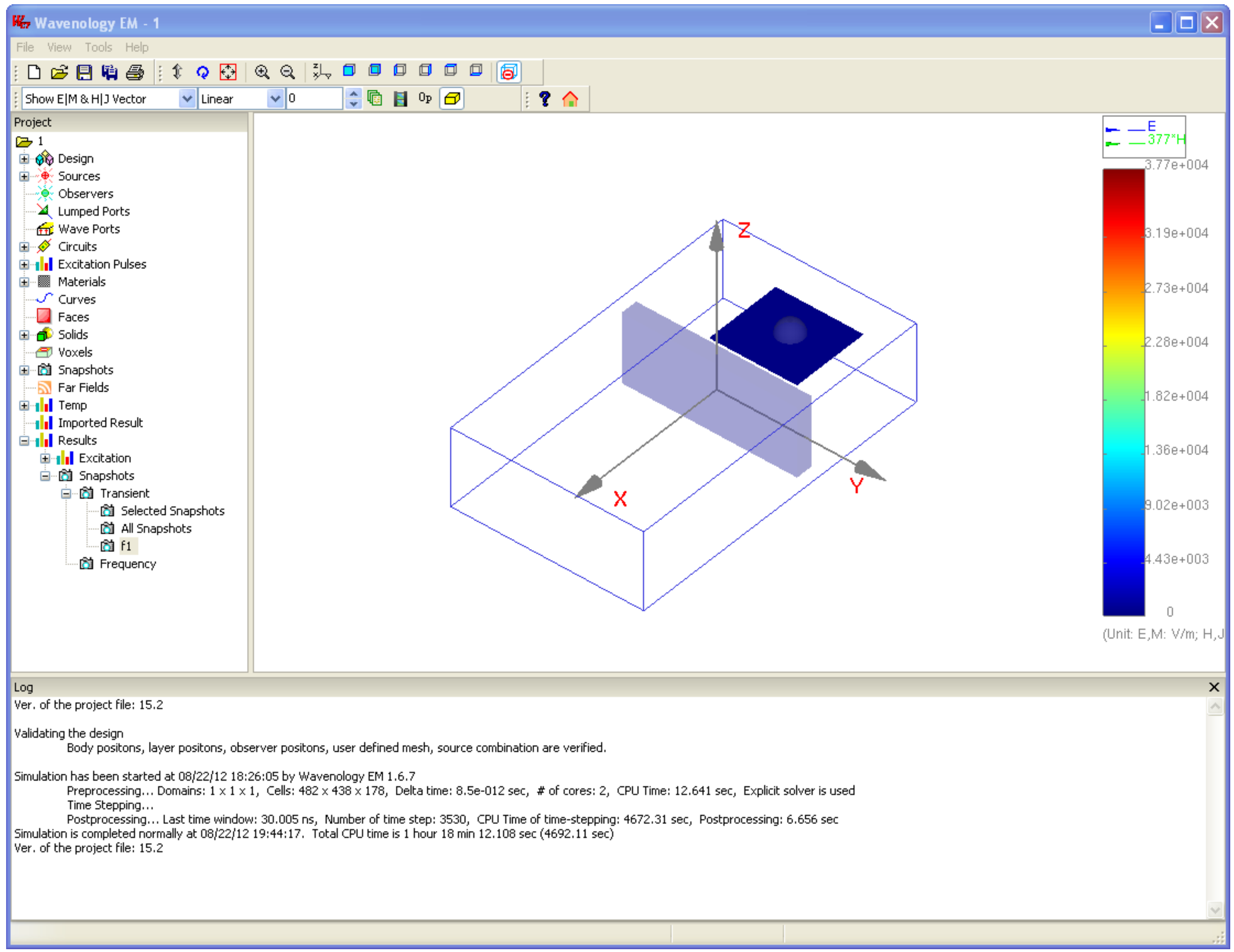




15. Simulate the case.

4. Check the result by watching the snapshot.

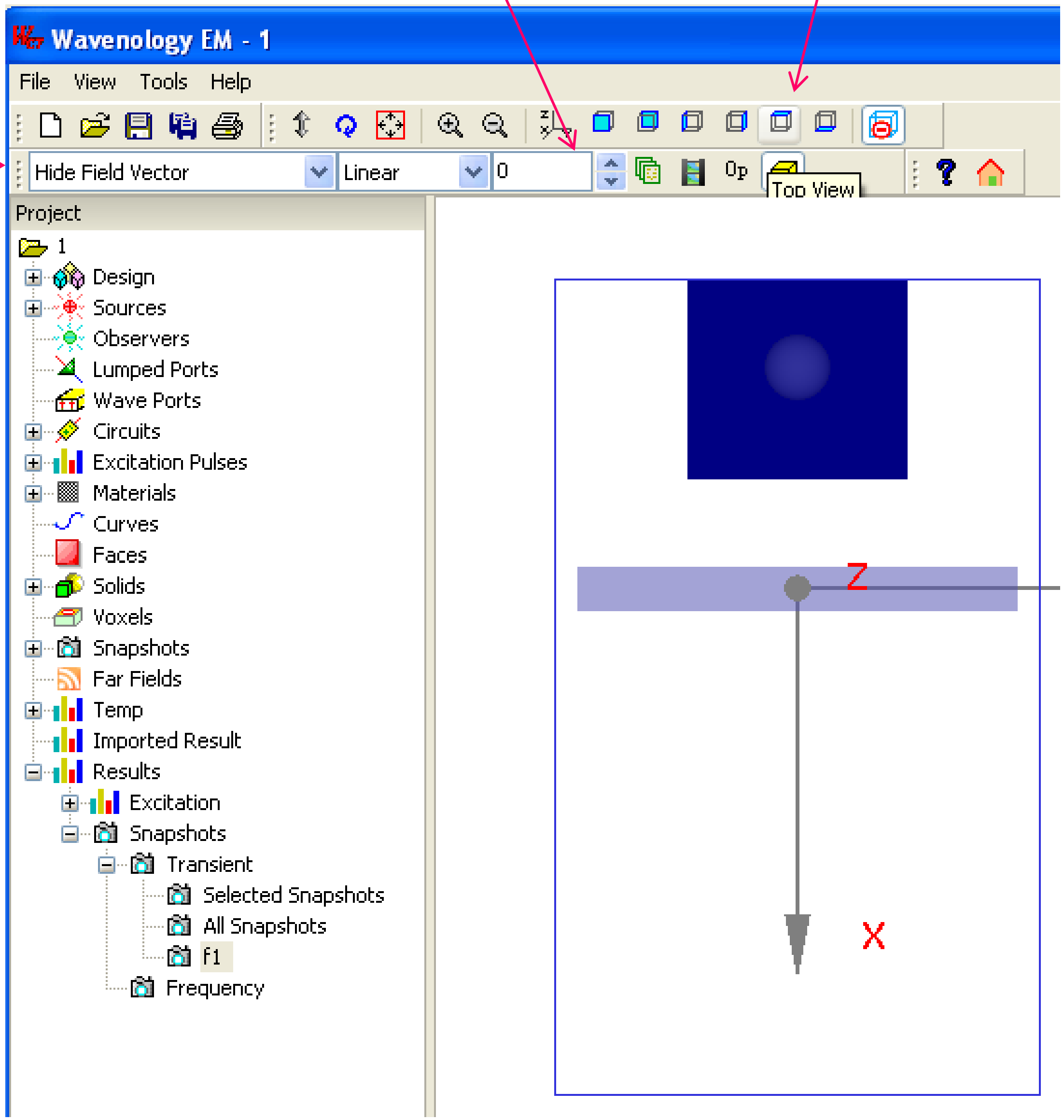


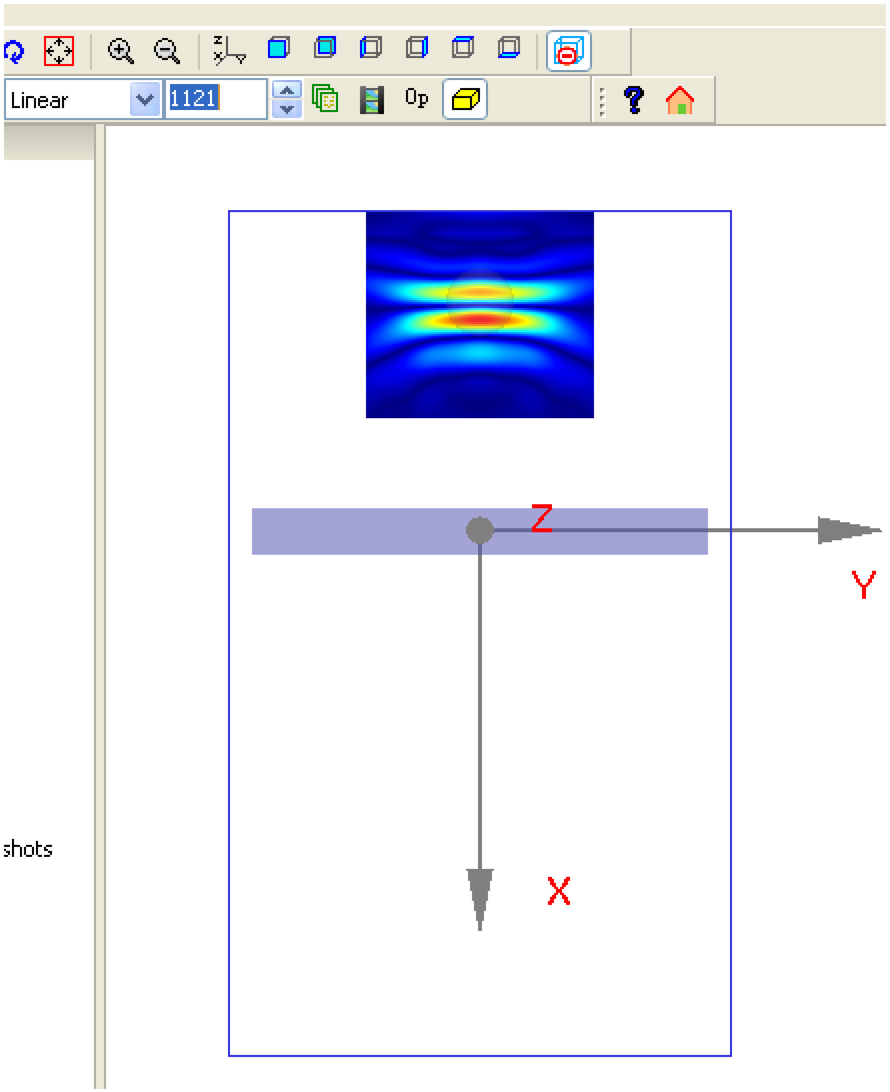


Change frame index to watch how wave propagate

Top view

Hide 3D field vector



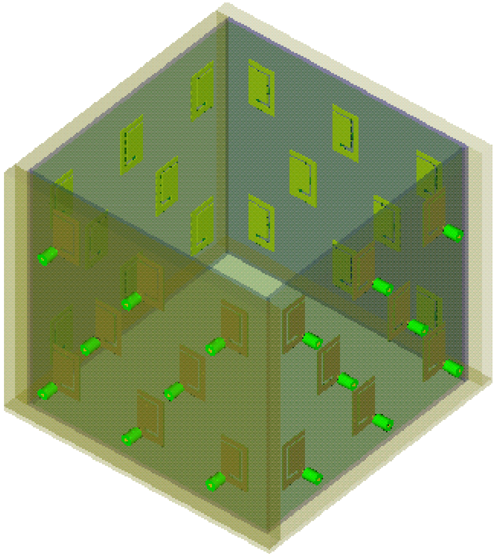


A strong focus on sphere can be seen at frame #1121

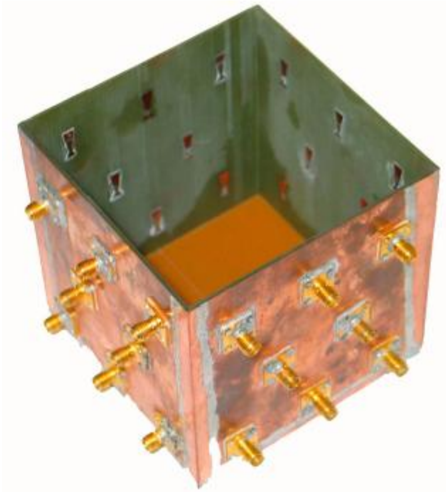
Case (2): Detect a Object in a Chamber by an Antenna Array

EM Solver

This case shows how to setup a time-reversal case with an antenna array.
We will demo how to locate a target in the chamber by the time-reversal method.



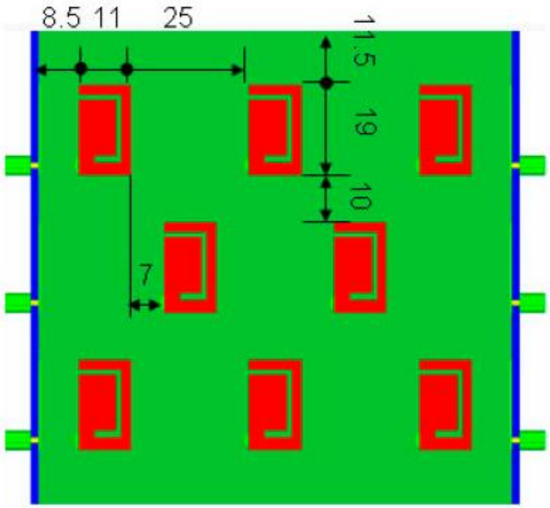
The simulated chamber model composed by an PIFA Antenna Array



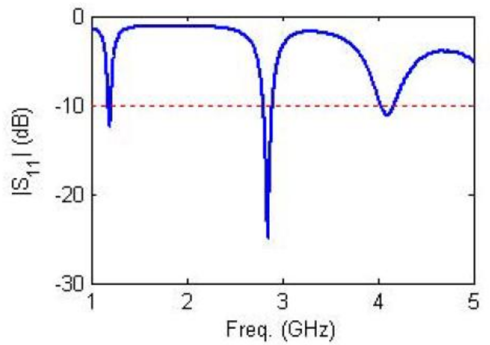
An reference physical Chamber composed by an Bowtie Antenna Array

The antenna array is built on a rectangular chamber with 5 PCB panels (the material of PCB board is FR4). The chamber is open at the +*z* direction.

The chamber size is 10×10 ×10 cm^3, filled by a fluid(ε_r=5, σ=0.01 S/m). There are 8 Planar Inverted F Antennas (PIFA) fabricated on each panel, totally 32 antennas in the chamber.



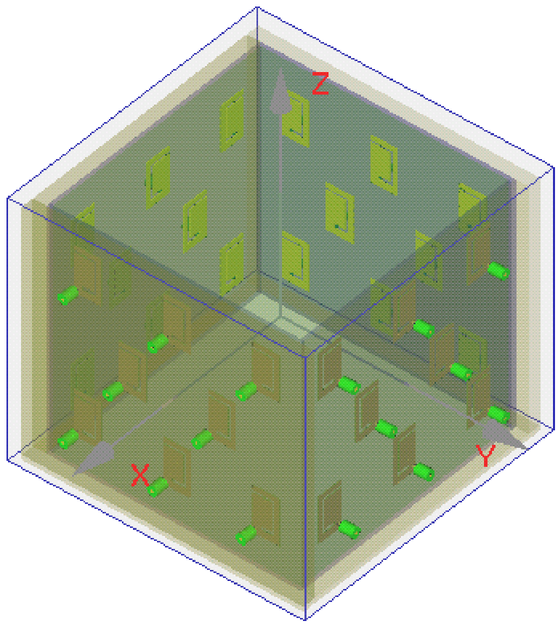
Antennas layout in a panel



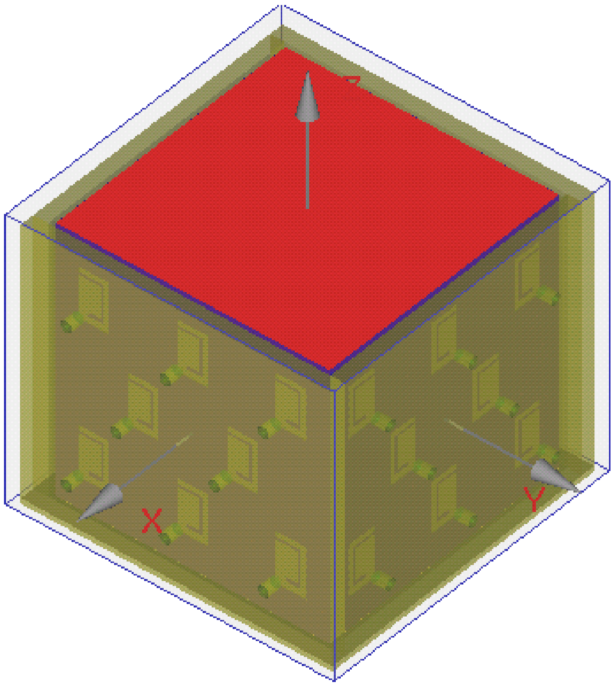
S_{11} of PIFA antenna

Because this chamber model is complicated. Thus, in this demo, we will not discuss how to build this model and just use this model.

Assume that we already have the chamber filled with fluid, we will show how to modify this model to simulate a time reversal-case.



Chamber with fluid (Fluid is set as transparent).



Fluid in the chamber (Fluid is highlighted by selection).

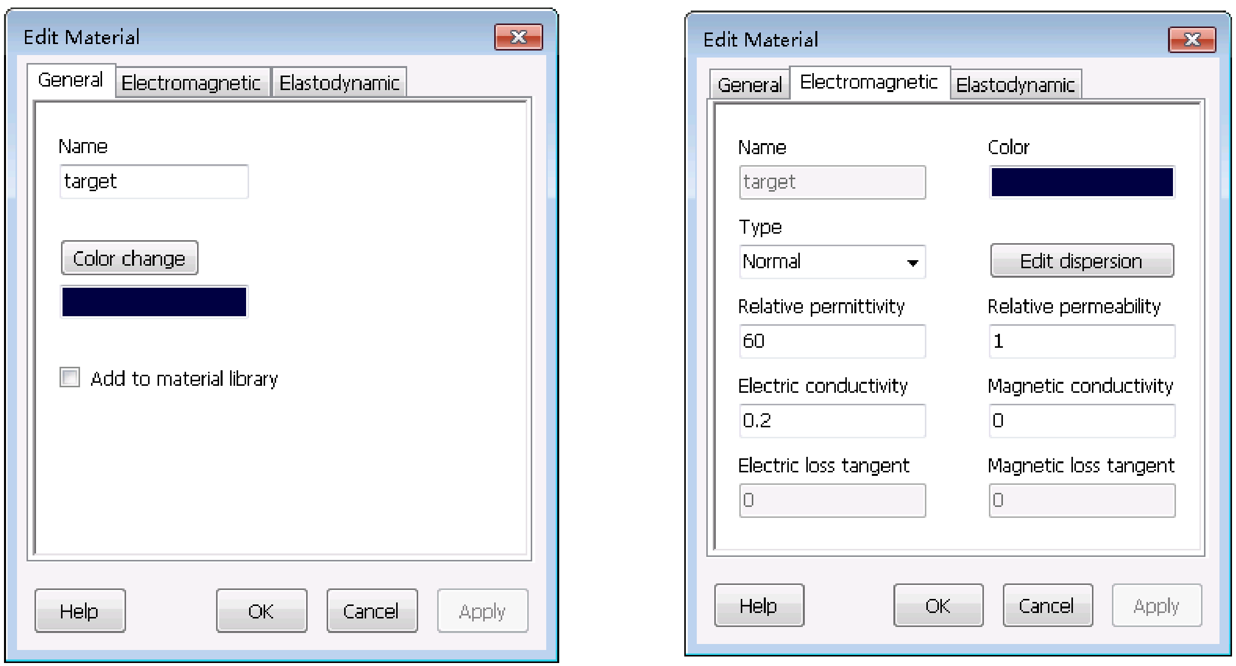
Similar to Case (1), we need to build three cases: the total field case, the incident field case and the reverse-radiation case.

Steps:

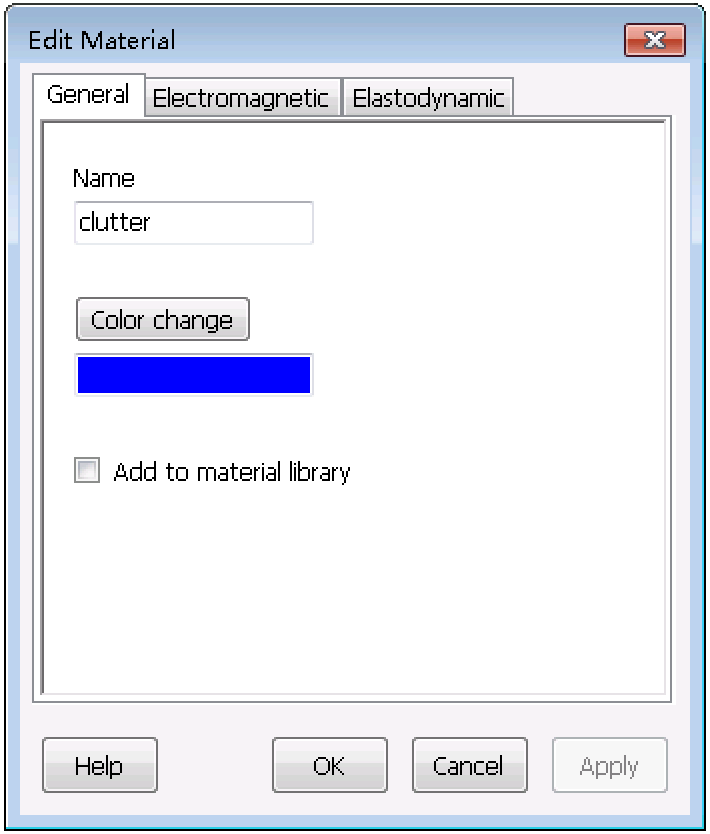
1. Assuming we have the chamber model case, we will build the total field case firstly.

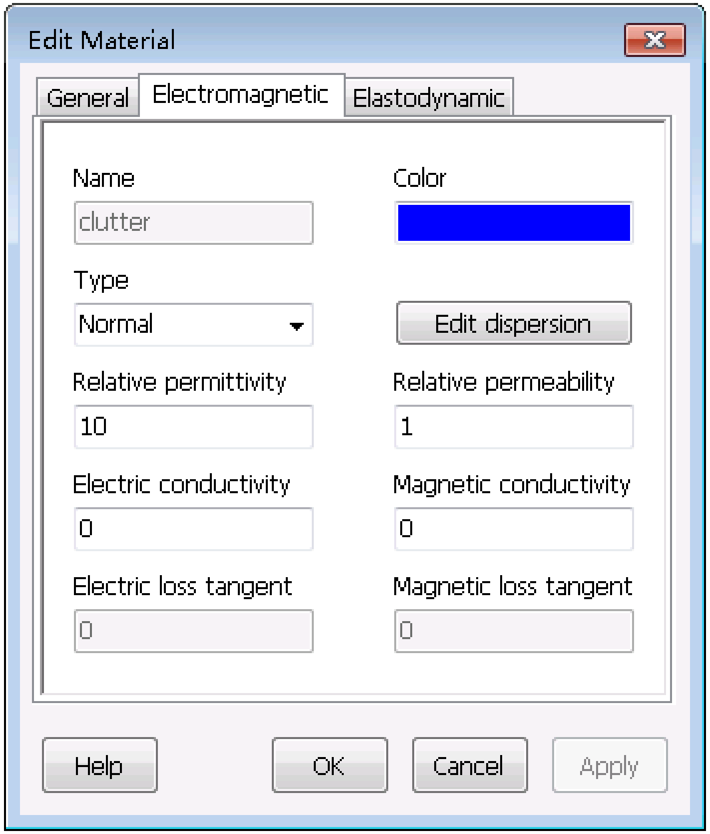
SaveAs the chamber case to folder “time-reversal-chamber\tot\” with name “pifa_chamber”.

2. Define material for the target and the clutter.



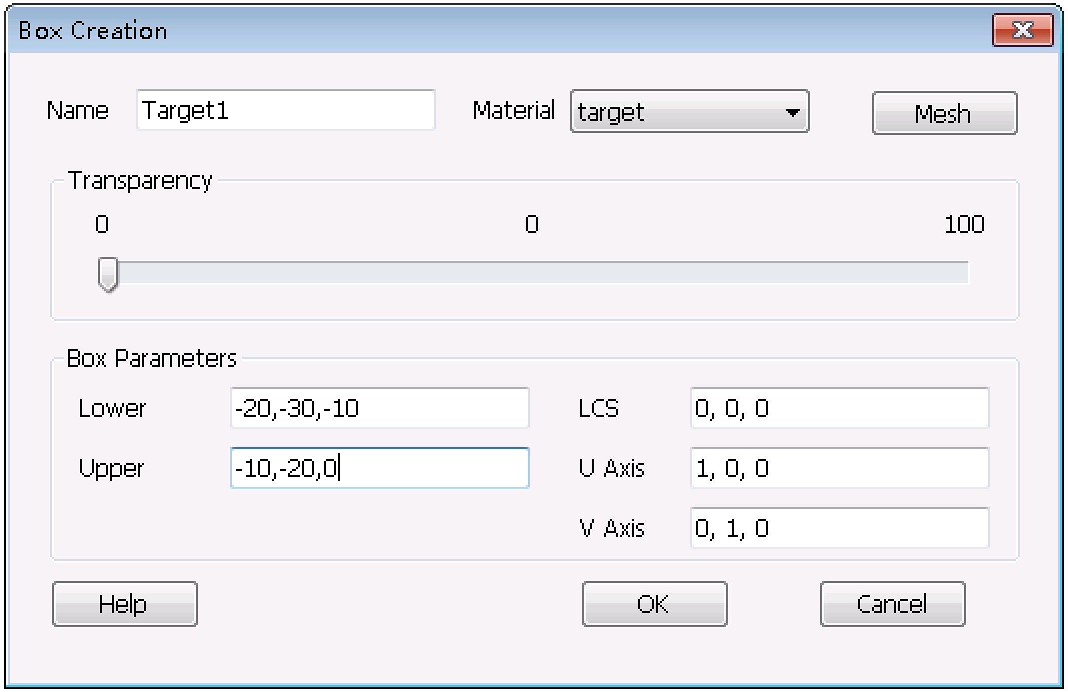
Material for target

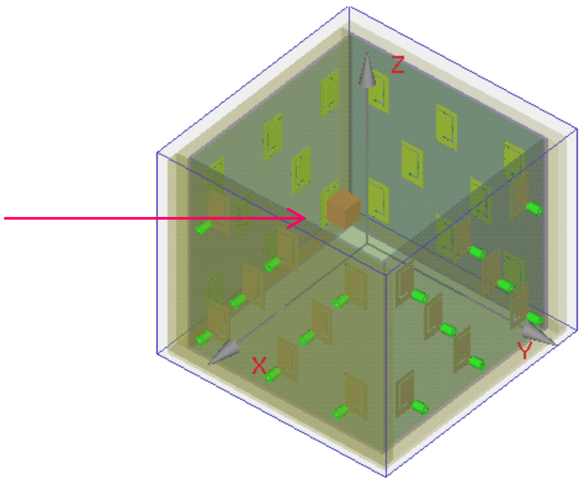




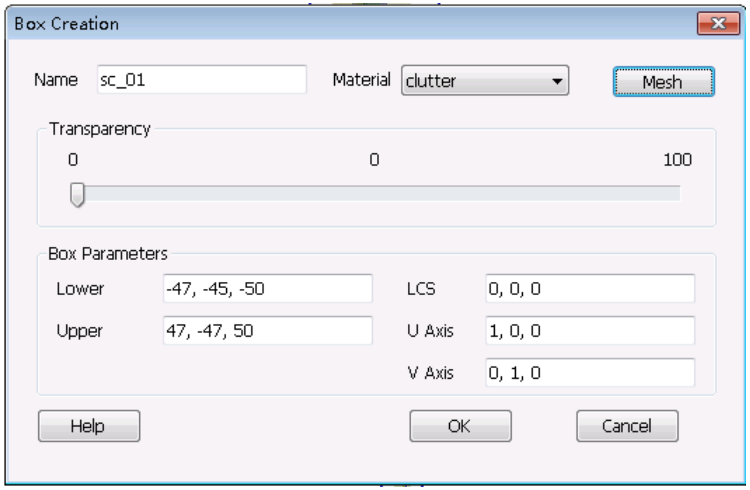
Material for clutter

3. Insert a cubic target in the chamber.

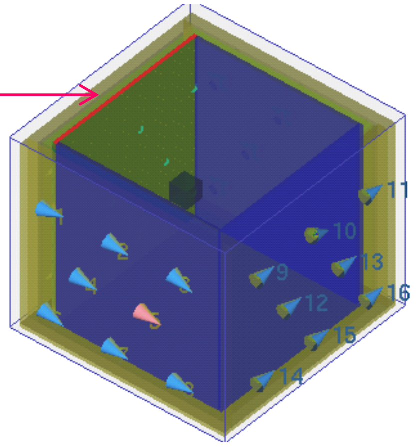




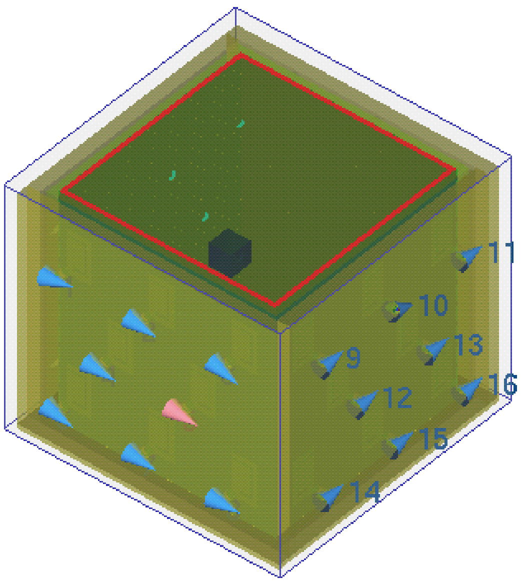
4. Insert four dielectric planes to work as clutter.



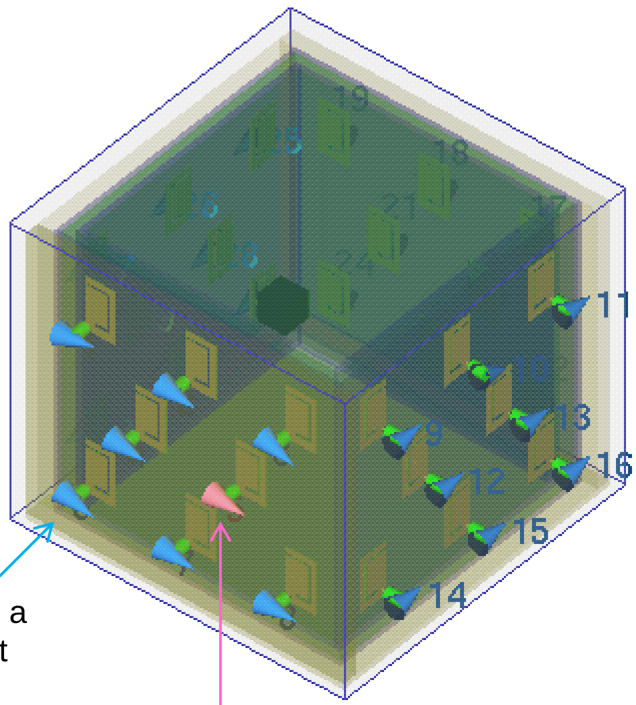
Plane 1



Four planes have been added.



5. Define the transmitting and receiving antennas. In this case, we use lumped port as the feeding port and receiving ports. Thus, there is one transmitting port and 31 receiving port.



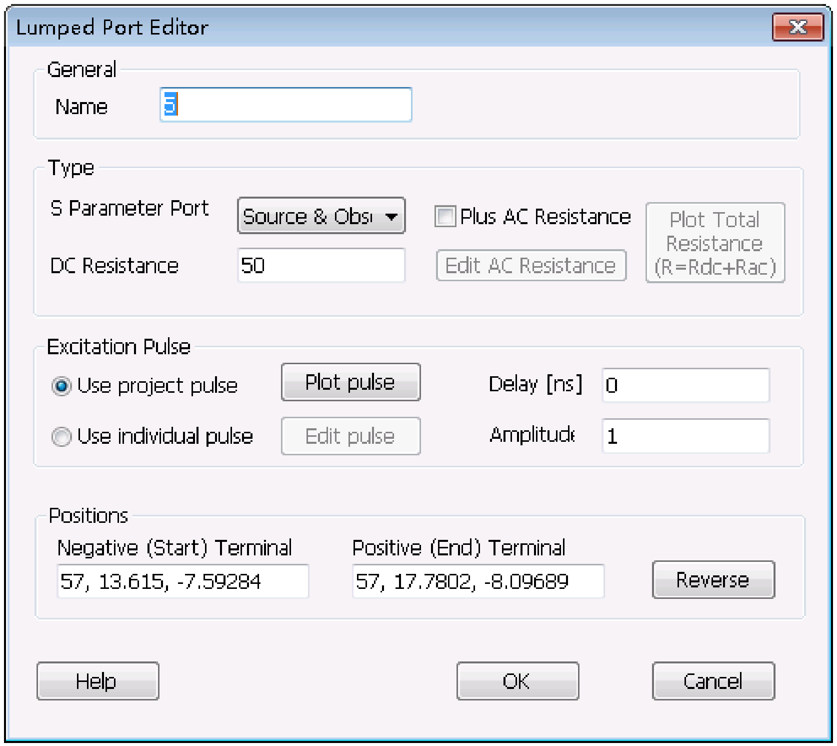
Blue arrow is a receiving port

Purple arrow is a transmitting port

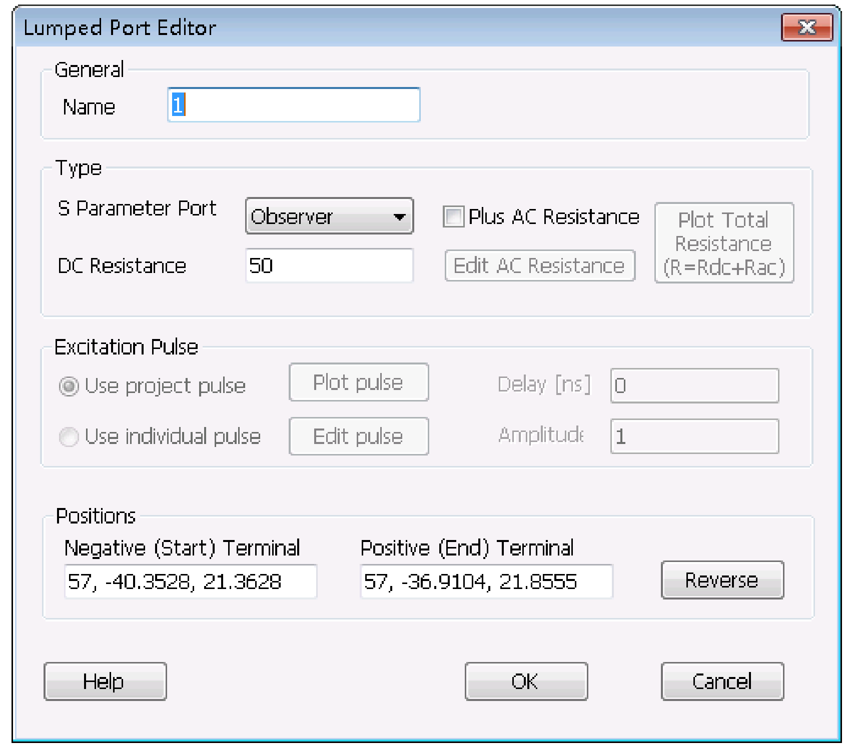
One terminal of the port is on the coax pin.

Another terminal of the port is on the PEC ground.

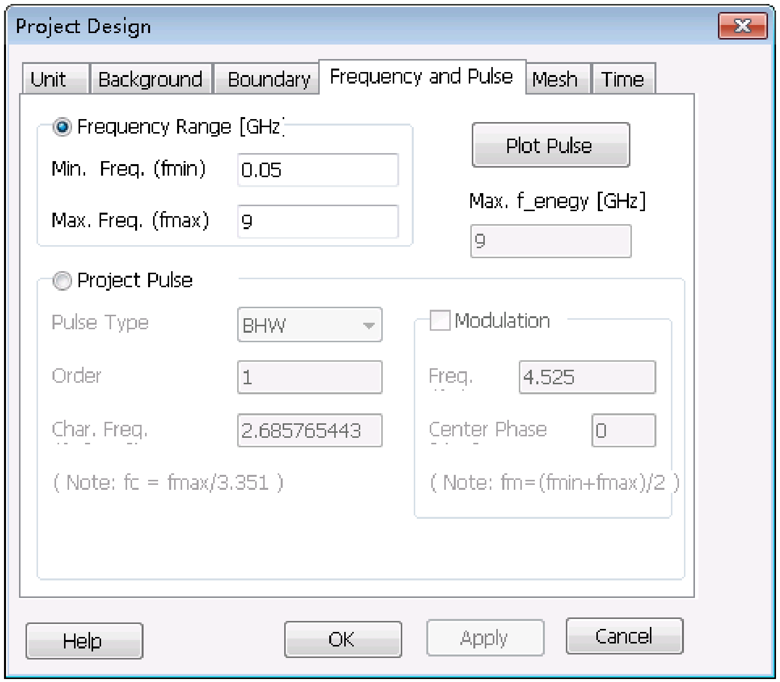
transmitting port

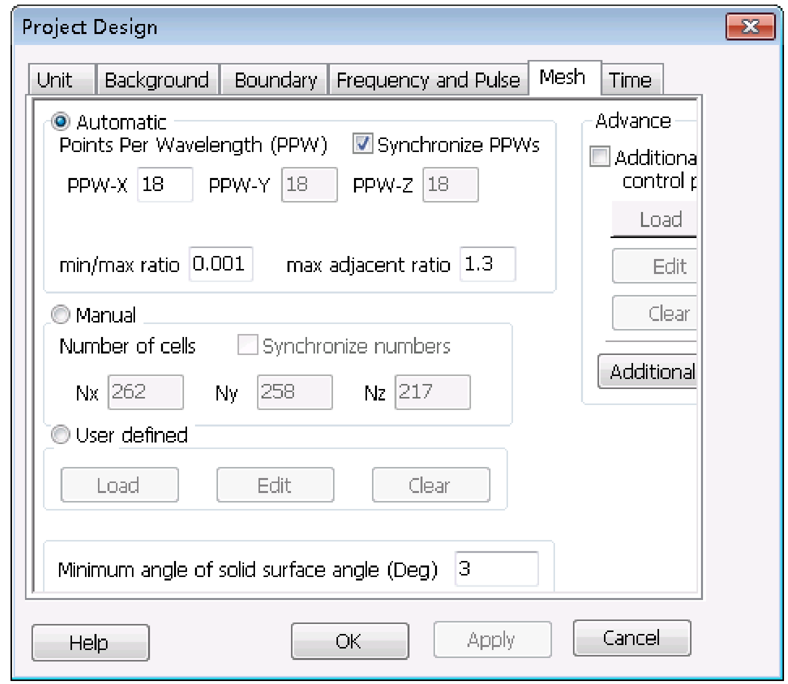


receiving port

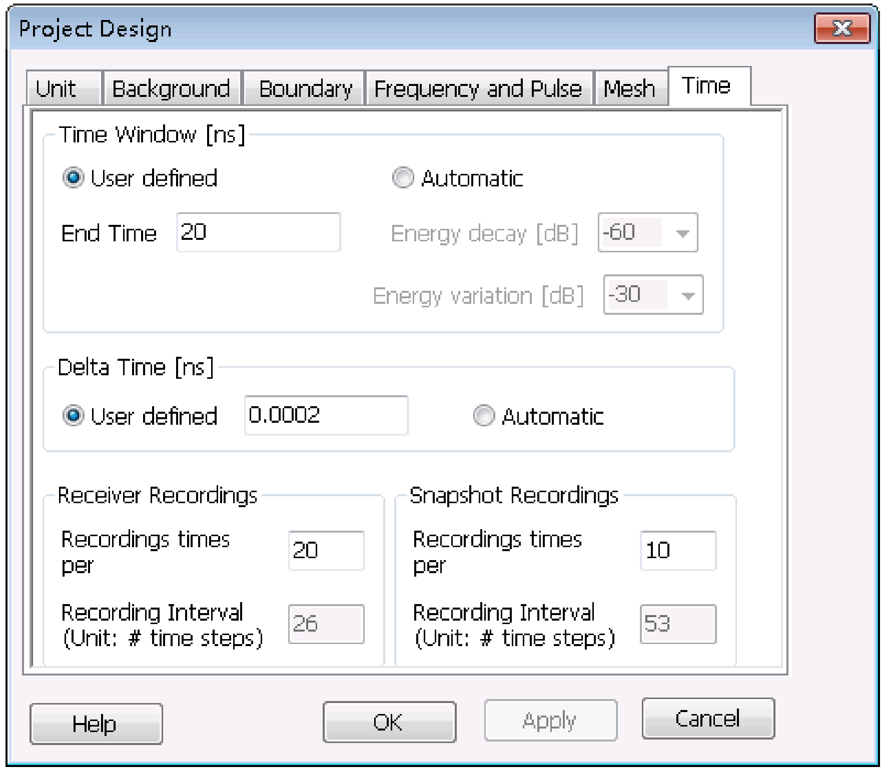


6. Set up simulation pulse, mesh density and time.

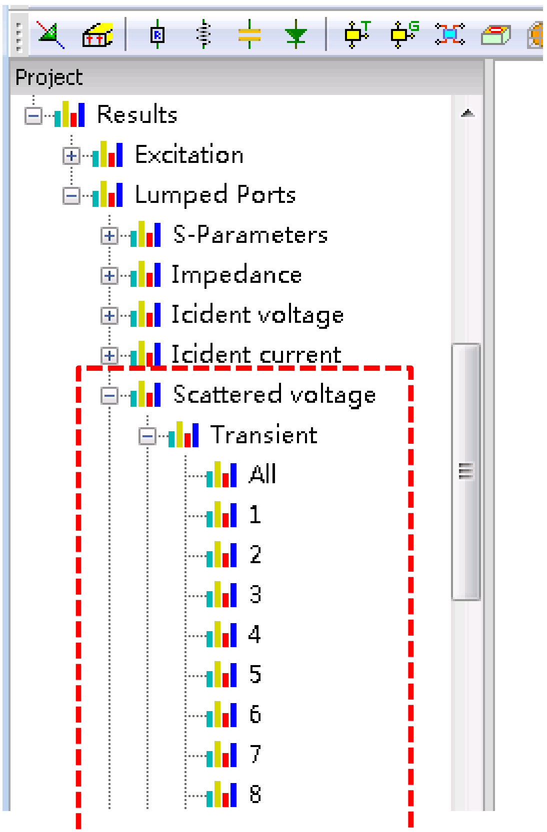




Here, we use 1st BHW wideband pulse as the source pulse.
Due to the antenna radiation freq. is about 2.7 GHz, we need to make the excitation pulse has a characteristic freq. around 2.7 GHz.


Project Design
Unit
Background
Boundary
Frequency and Pulse
Mesh
Time
Time Window [ns]
User defined
Automatic
End Time
20
Energy decay [dB]
-60
Energy variation [dB]
-30
Delta Time [ns]
User defined
0.0002
Automatic
Receiver Recordings
Recordings times per
20
Recording Interval (Unit: # time steps)
26
Snapshot Recordings
Recordings times per
10
Recording Interval (Unit: # time steps)
53
Help
OK
Apply
Cancel

8. Simulate the case to obtain the transient total voltage on the receiving antennas.



Note: in WCT GUI, there is scattered voltage on the transmitting & receiving ports.

For receiving port, it has not excitation. Thus, the total voltage on port is the same as the scattered voltage.

For this time-reversal case, in order to distinguish the voltage on the receiving port in the two cases (with-target & without-target), we call the scattered voltage for with-target case is the total voltage, the scattered voltage for with-target case is the incident voltage, in this demo.

9. Setup a case to obtain the incident voltage on receiving antennas.

9.1 Create a new sub-folder under “time-reversal-chamber” as inc. Use SaveAs function to save the total voltage case to inc sub-folder with name “pifa_chamber”.

9.2 Change the material of the target as fluid (oil).

9.3 We set this case having the same mesh as the total voltage case, through loading the mesh directly from the total voltage case. Detail information please refer to Page 18.

10. Simulate this incident voltage case to obtain the incident voltage on the receiving antennas.

11. Calculate the scattered voltage from the total voltage case and the incident voltage case.

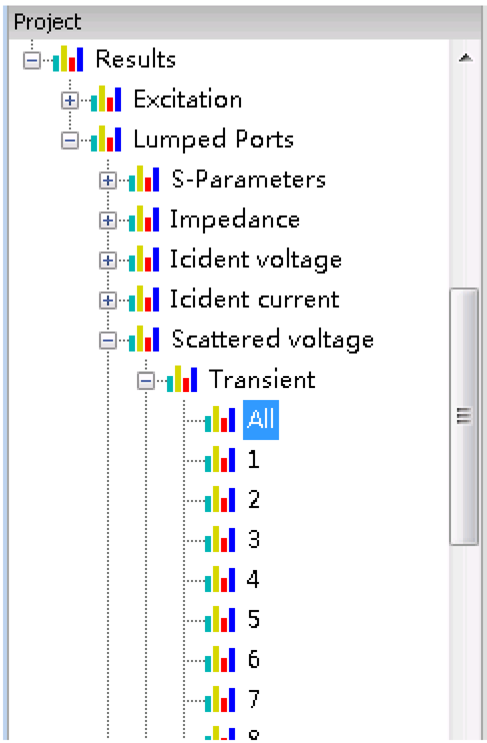
$V_{sct}=V_{tot} - V_{inc}$

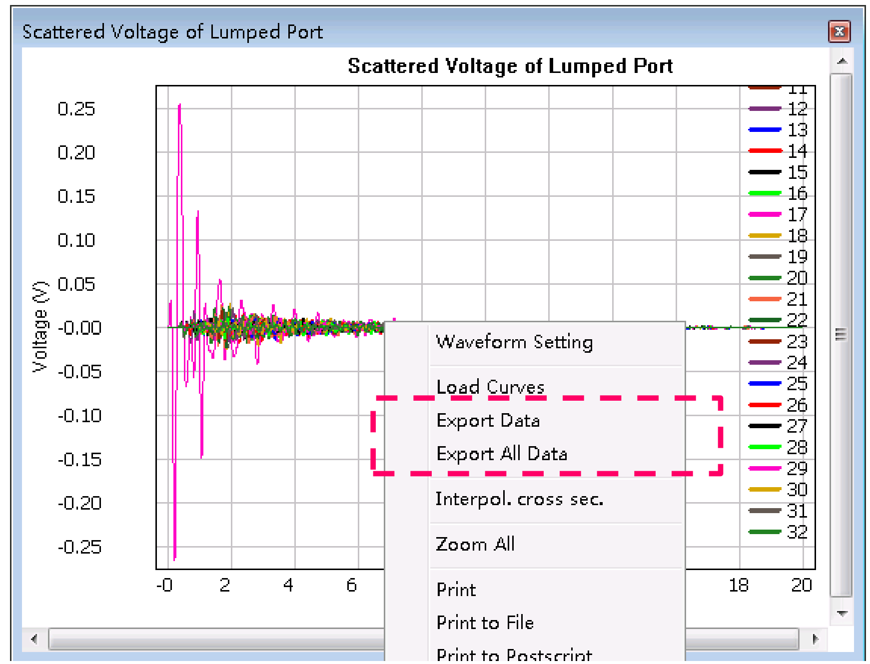
V_{tot} is the total voltage, from the result on the receiving antennas in the total voltage case. V_{inc} is the incident voltage, from the result on the receiving antennas in the incident voltage case.

The transient voltage on the receiving antennas can be obtained by two methods:

1) Export from 2D result canvas.
2) Directly use the simulation result data files.

A. Export Data from 2D result canvas.

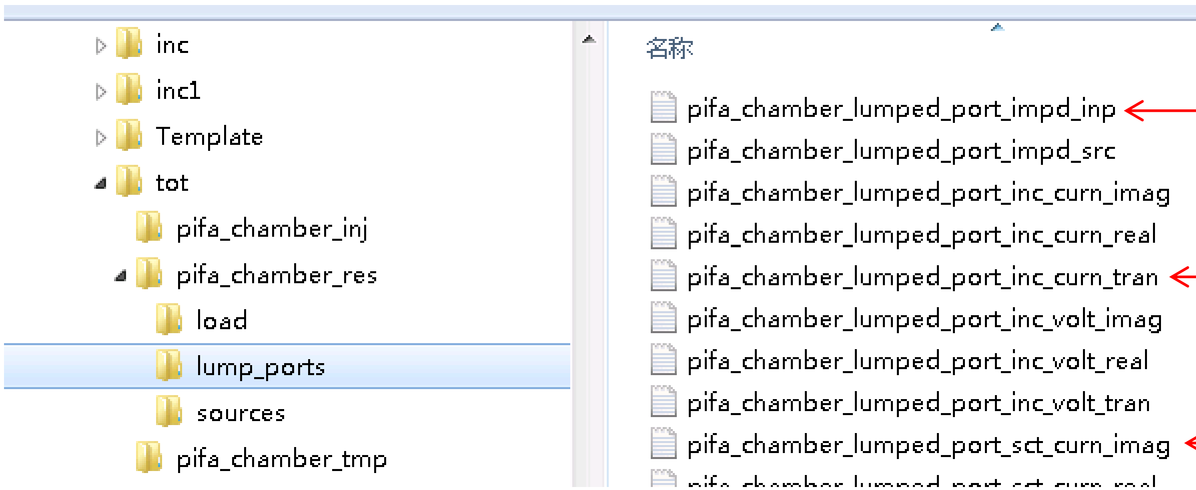




Double click this tree-node and export the data

Exported data file format is descript on page 22.

B. Directly use the simulation result data files.
The simulation result for each case has following structure:



Lumped ports' input impedance

Lumped ports' transient incident current

Lumped ports' scattered current, imaginary part

x*_res** is simulation result folder. (x*** is project name)

The sub-folder ***"Lump_ports"*** has all results for lumped ports, the scattered voltage has the file name "***x***_lumped_port_sct_volt_tran.txt".

Data file format is descript on page 24.

Following matlab code shows how to calculate the Vsct from the simulation data files directly. (This matlab file for this code is time-reversal-chamber\sct_reverse_time.m)

```
 clear all;

%%% load incident field
Etot = load( '.\tot\pifa_chamber_res\lump_ports\pifa_chamber_lumped_port_sct_volt_tran.txt' );
nFrame = Etot( 1 );
t0 = Etot( 2 );
t1 = Etot( 3 );
dt = Etot( 4 );
nLen = Etot( 5 );
Etot = reshape( Etot(6:end), nLen, nFrame );

%%% load total field
Einc = load( '.\inc\pifa_chamber_res\lump_ports\pifa_chamber_lumped_port_sct_volt_tran.txt' );
Einc = reshape( Einc(6:end), nLen, nFrame );

%%% resampling
ND = 1;
Etot = Etot( 1:ND:end, : );
Einc = Einc( 1:ND:end, : );

dt = dt * ND;
nLen = length( Etot( :, 1 ) );
t = [0:(nLen-1)] * dt;
t = t';

Esct = Etot - Einc; %%% make transient scattered field

ids = [ 1 10 11 12 13 14 15 16 17 18 19 2 20 21 22 23 24 25 26 27 28 29 3 30 31 32 4 5 6 7 8 9 ];
% need mapping, because the export function of WCT use ASCII sequence. (Till current version 1.6)

for k = 1 : 32,
   tt = [ t flipud(Esct(:,k)*1e4) ];
   str = sprintf( 'pifa_chamber_user_lport_pulse_%0d.txt', ids(k) );

   fid = fopen( str, 'w' );
   fprintf( fid, '%%Wave Computation Technologies simulation waveform data, version 3.0\n' );
   fprintf( fid, '%%created at 11/06/10 04:21:37 in GMT\n' );
   fprintf( fid, '%%=================================\n' );
   fprintf( fid, '%%Title      User defined pulse\n' );
   fprintf( fid, '%%X-Lin      1    Unit: \n' );
   fprintf( fid, '%%Y-Lin      1    Unit: \n' );
   fprintf( fid, '%%Sub-Title  User defined pulse\n' );
   fprintf( fid, '%d\t %d\t %%Size Nx=Ny\n',  nLen, nLen );

   for j = 1 : nLen,
      fprintf( fid, '%g\t %g\t\n',  tt( j, 1 ), tt( j, 2 ) );
   end;

   fclose( fid );
End;
```

Note: This code generate 32 data files. Each file is corresponding to the scattered voltage of one port.

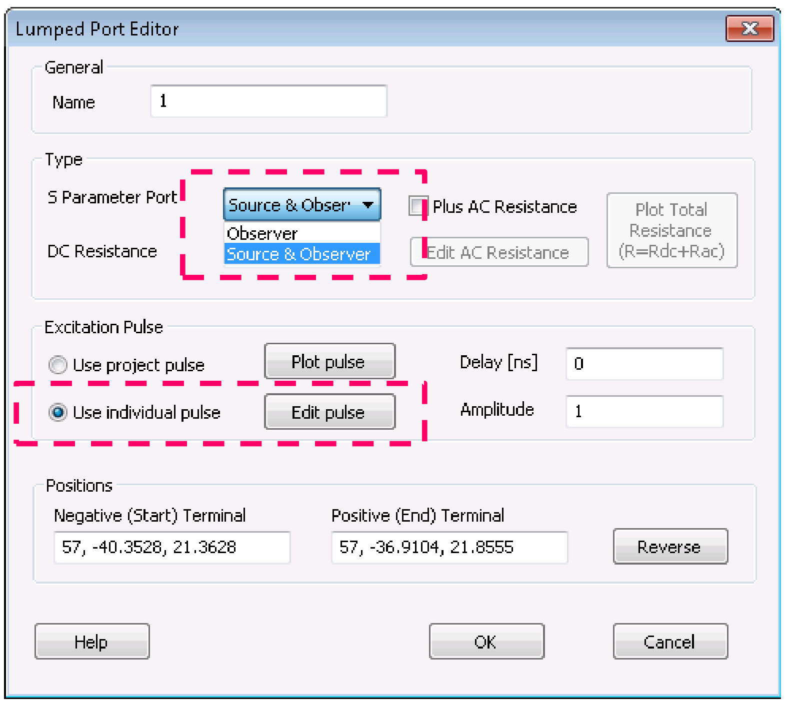
12. Set up the reverse-radiation case.

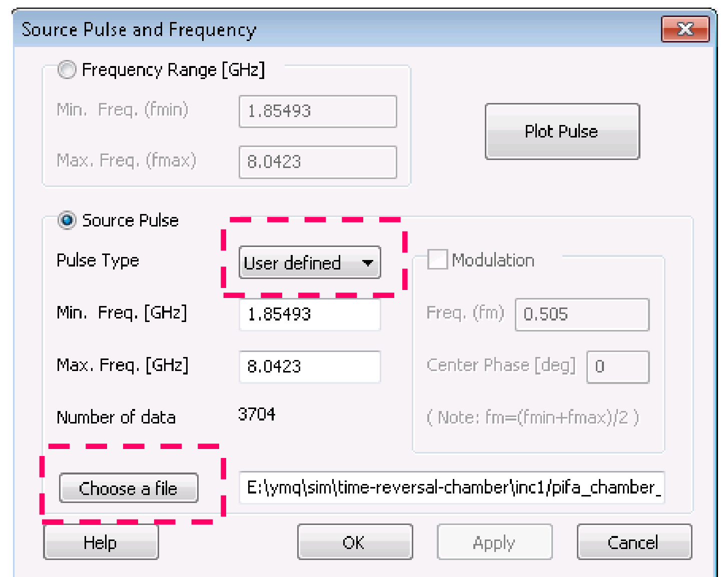
1) SaveAs the incident case to sub-folder “inc1” with name “pifa_chamber”.

2) delete the original excitation port “5”

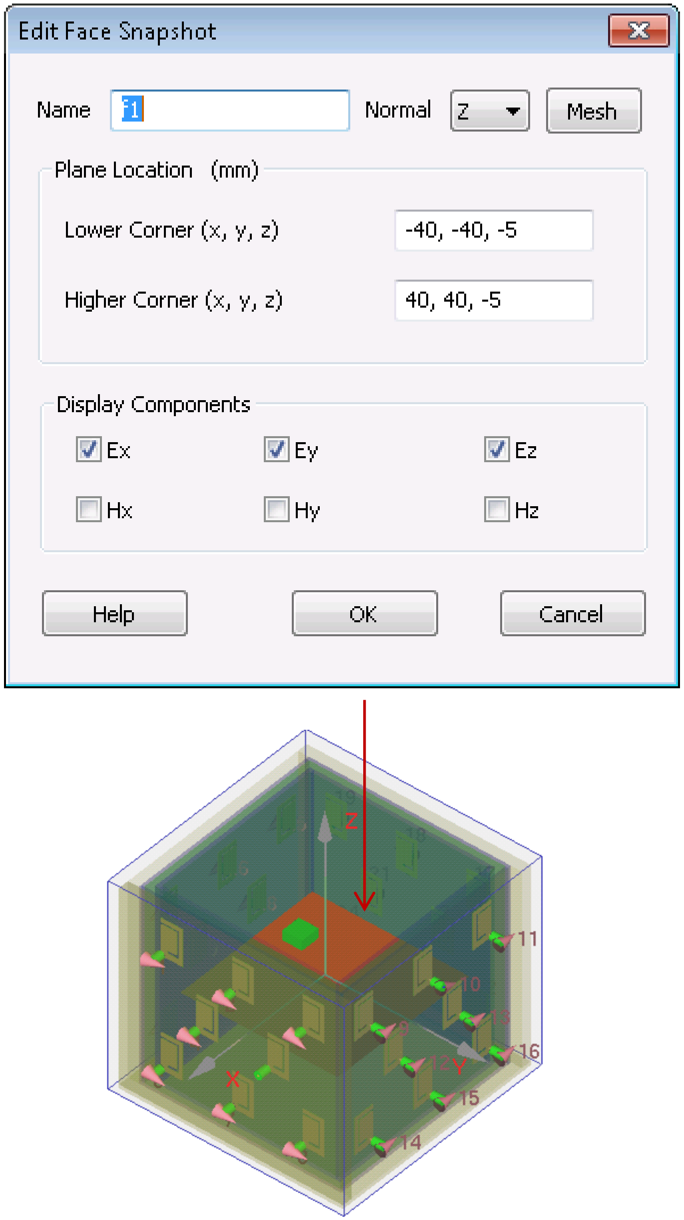
3) convert all receiving ports to excitation port, set up the excitation pulse of port as “User defined” and load it from the data file.

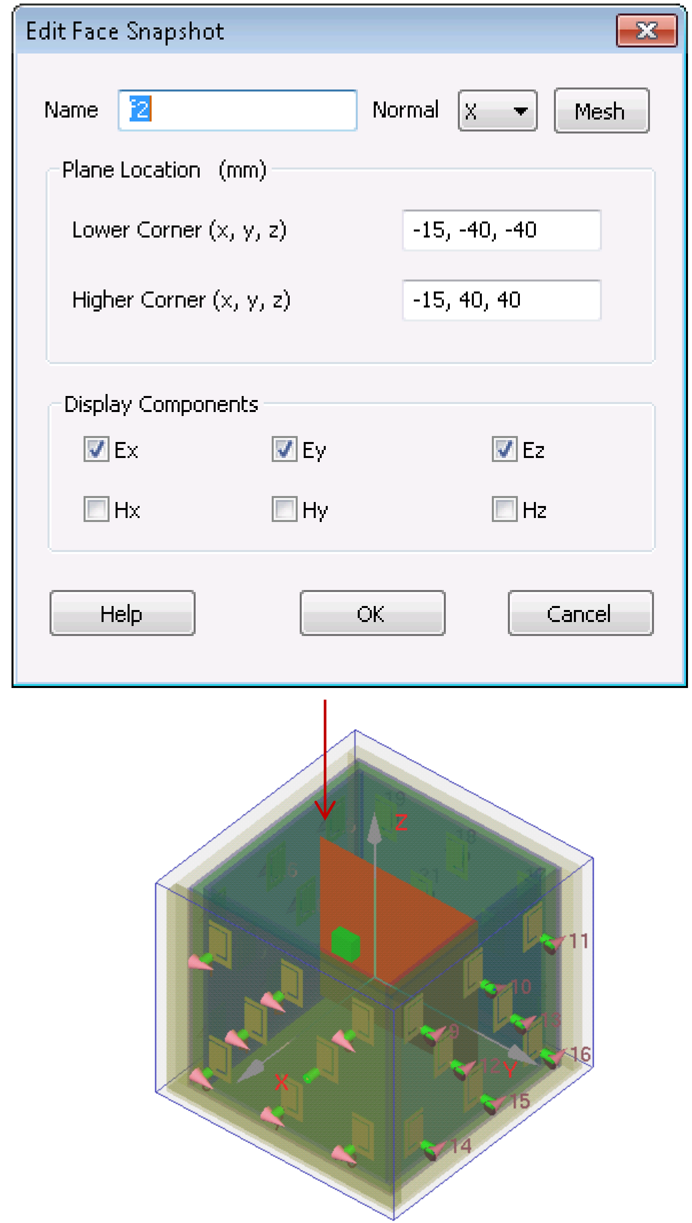
(The code in page 48 generates the scattered voltage for each port, named as “'pifa_chamber_user_lport_pulse_**xx**.txt”. **xx** is the name of port.)

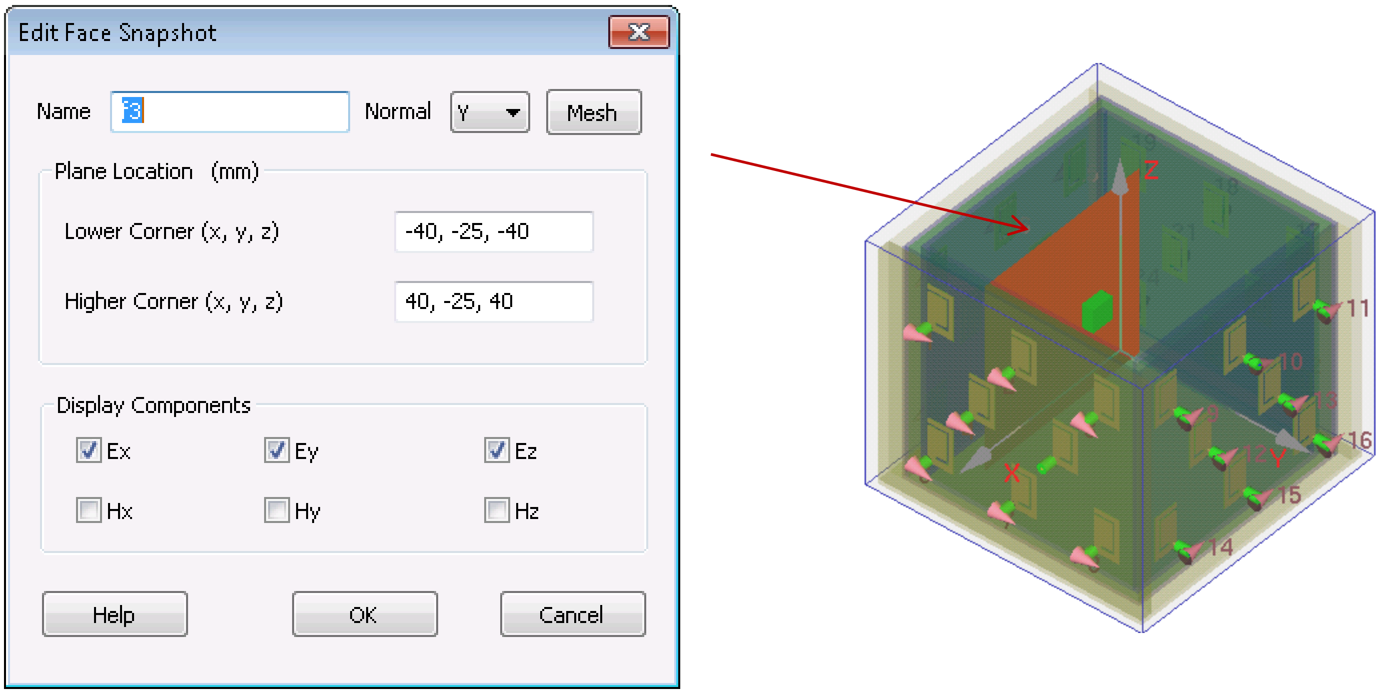




13. Define 2D face snapshots to record the result.

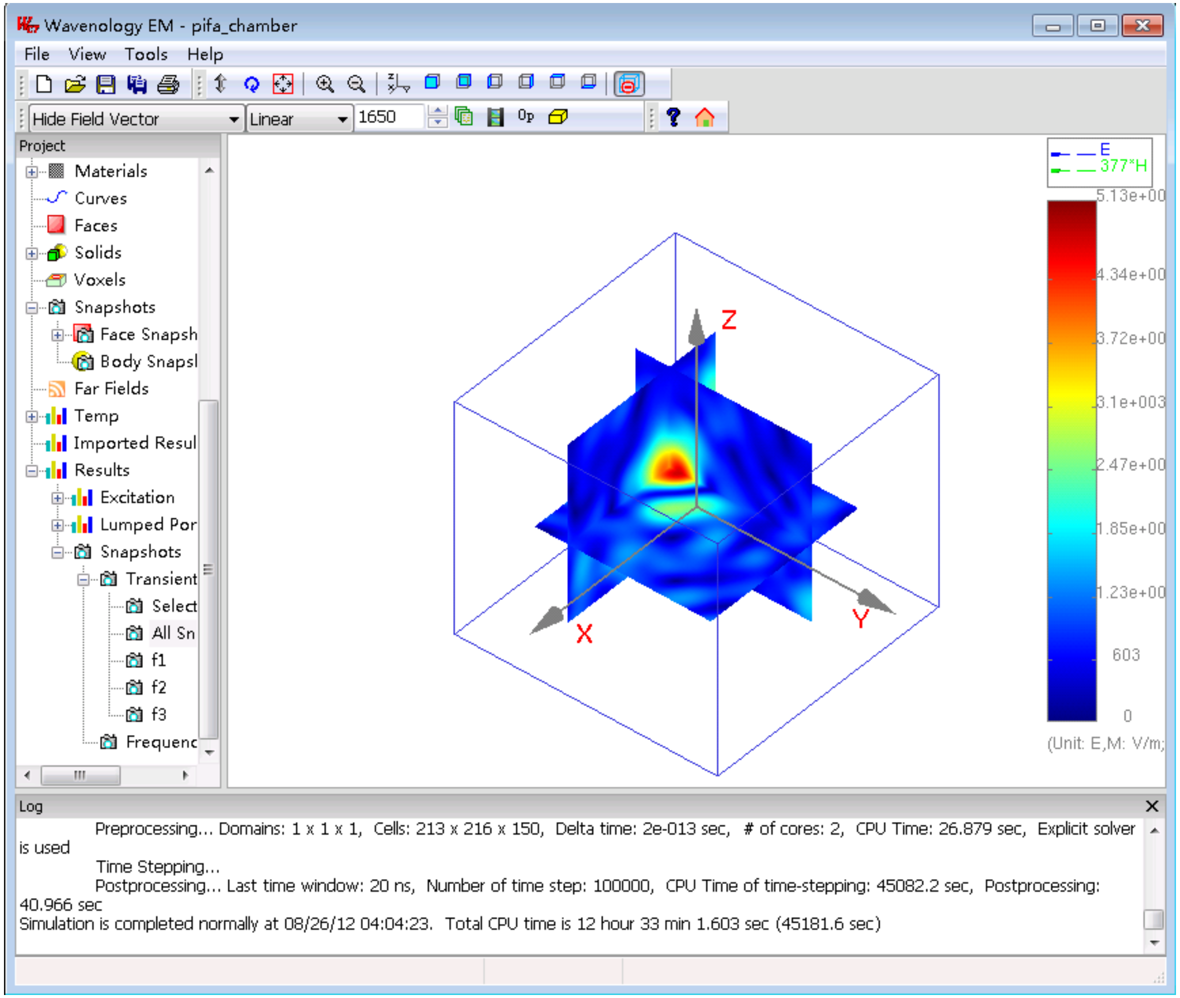





Edit Face Snapshot
Name
f3
Normal
Y
Mesh
Plane Location (mm)
Lower Corner (x, y, z)
-40, -25, -40
Higher Corner (x, y, z)
40, -25, 40
Display Components
Ex
Ey
Ez
Hx
Hy
Hz
Help
OK
Cancel
Z
X
Y
11
10
16
15
14

14. Simulate the case.

15. Check the result by watching the snapshot.

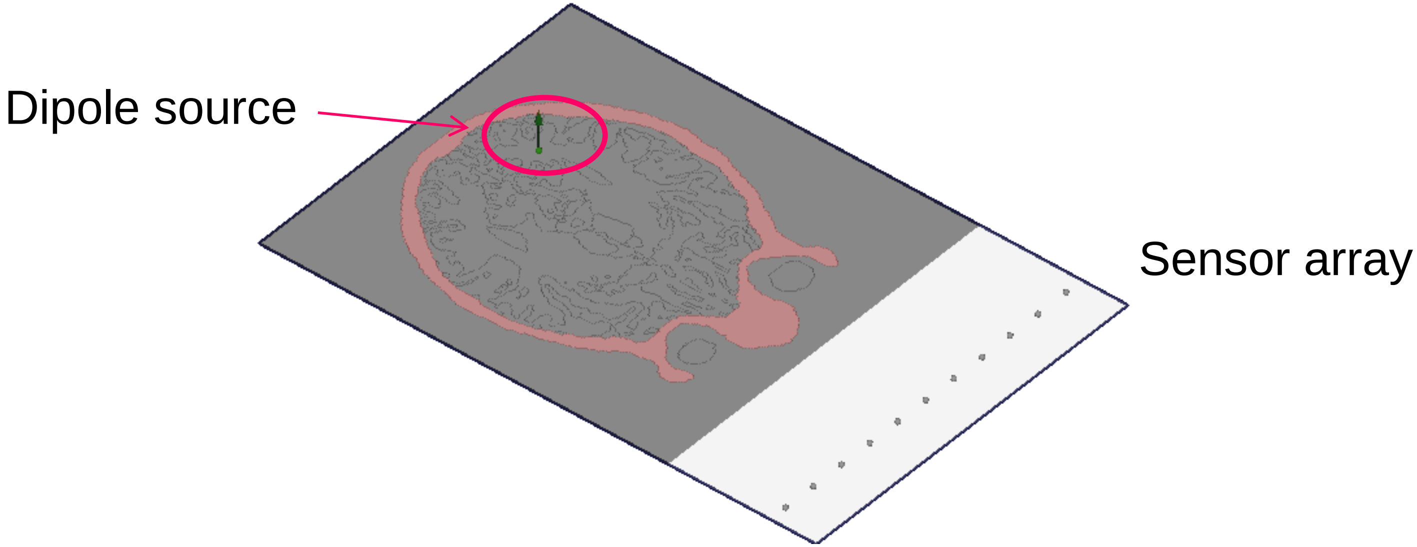


It can be seen, there is a strongest focus (frame 1650) at the position of target.

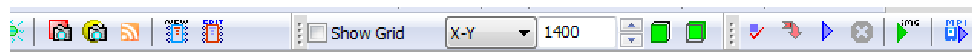
Case (3): Detect a source in a 2D human skull slice by the direct wave

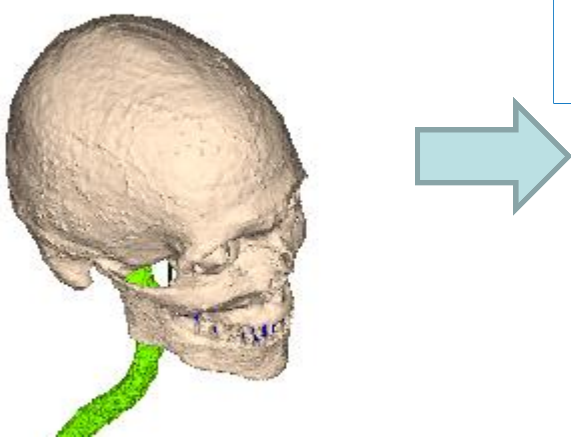
EM Solver

This case shows how to setup a time-reversal case to detect an electrical dipole source in a 2D human skull slice by the direct wave



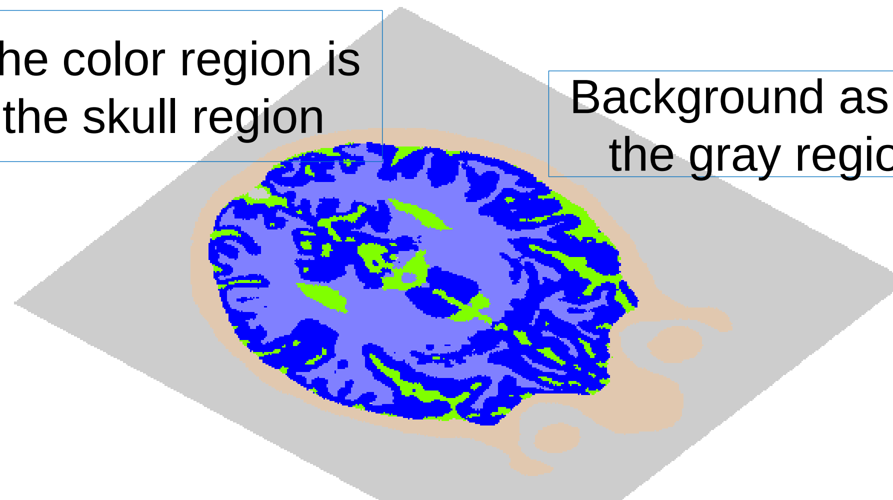
2D human skull slice is generate by WCT Cartesian mesher

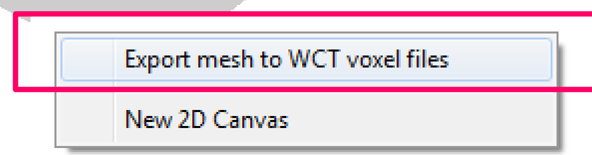




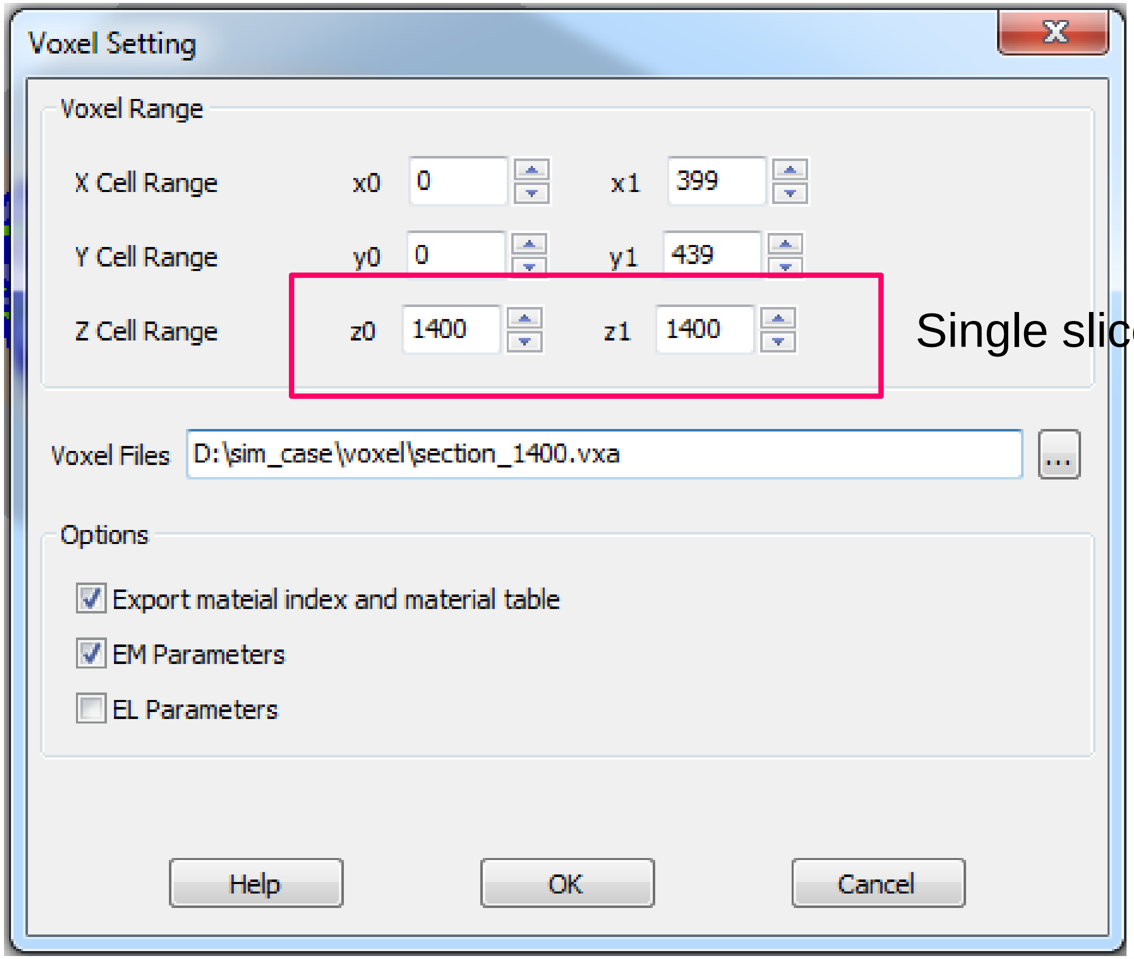
The color region is the skull region

Background as air, the gray region



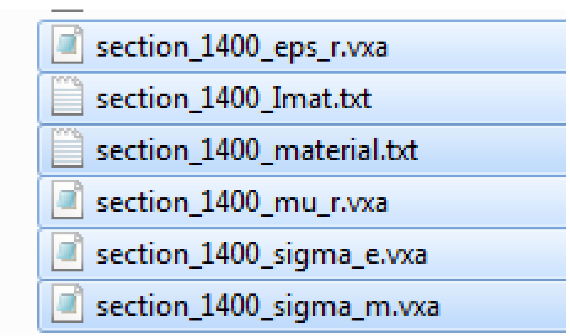


More detail about how to use WCT Cartesian mesher and voxel data please refer to Wavenology Voxel Manual.



Single slice

The exported voxel data files for 2D human skull slice



Note: in this model, we set up the EM parameter for the human skull model as: different ε_r for different tissues. μ_r=1, σ_e=0, σ_m=0 for all tissues.

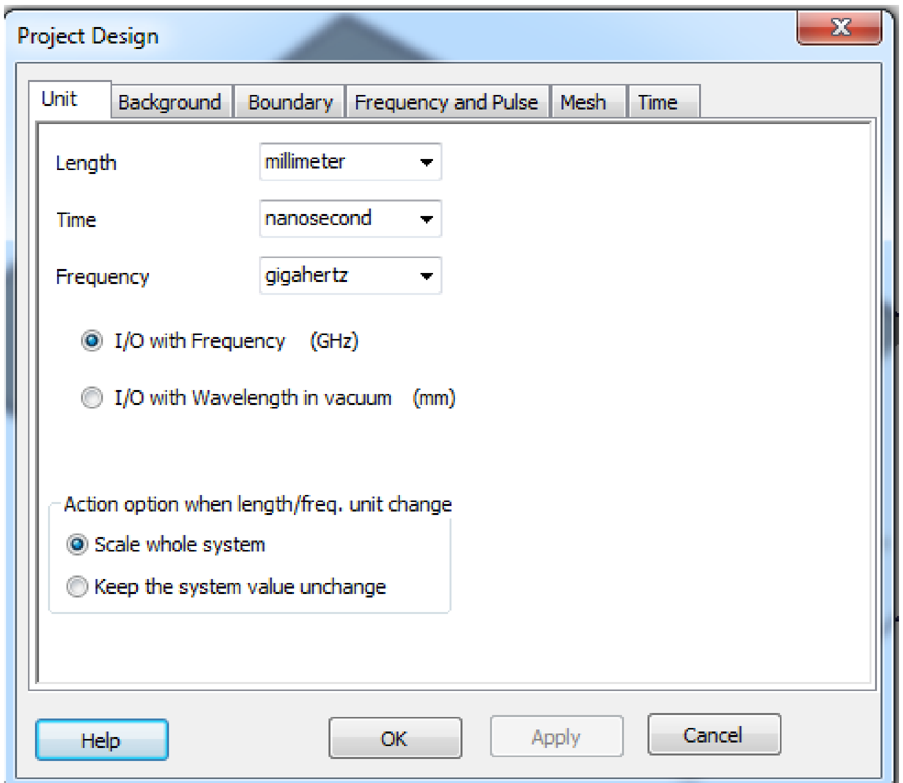
Part 1: the project to obtain the signal from a electrical dipole source in 2D human skull slice

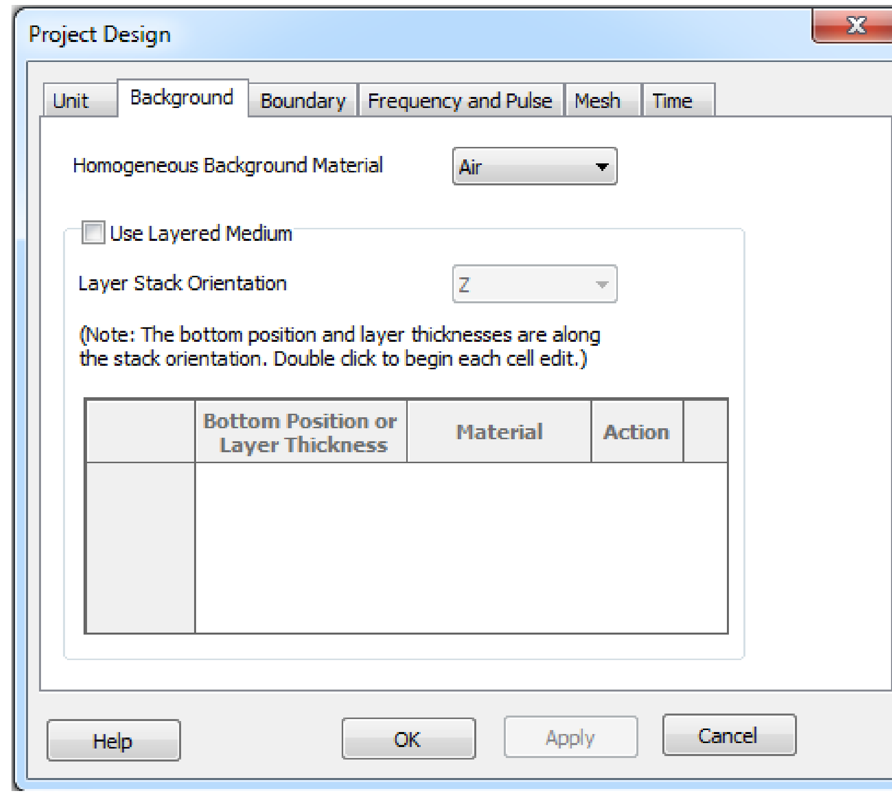
- before we build this case, we has following information
  - the 2D human skull slice data file is WCT "vxa" file for relative permittivity: **section_1400_eps_r.vxa**
  - the 2D human skull slice has a cell number of: 400x440x1 in X, Y and Z axis, respectively
  - the cell resolution of the slice is – (0.5, 0.5, 0.5) mm^3, the total size is (200, 220, 1) mm^3
  - the background outside the skull region is air

Define the project name as: **2_1.wnt**

Length unit is mm

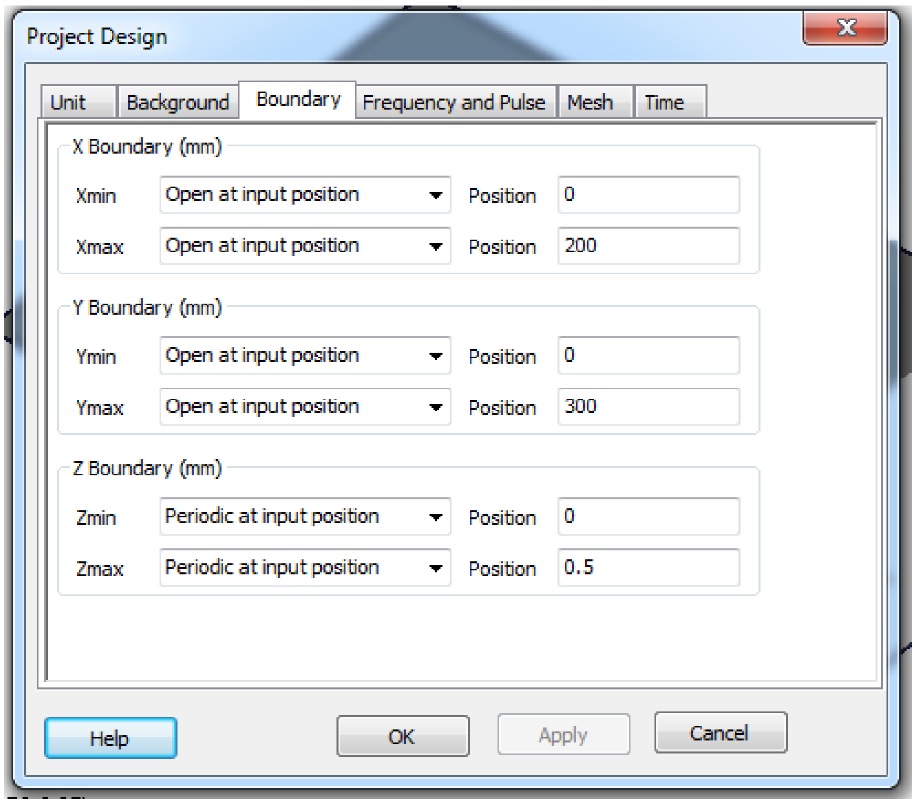
Background as air

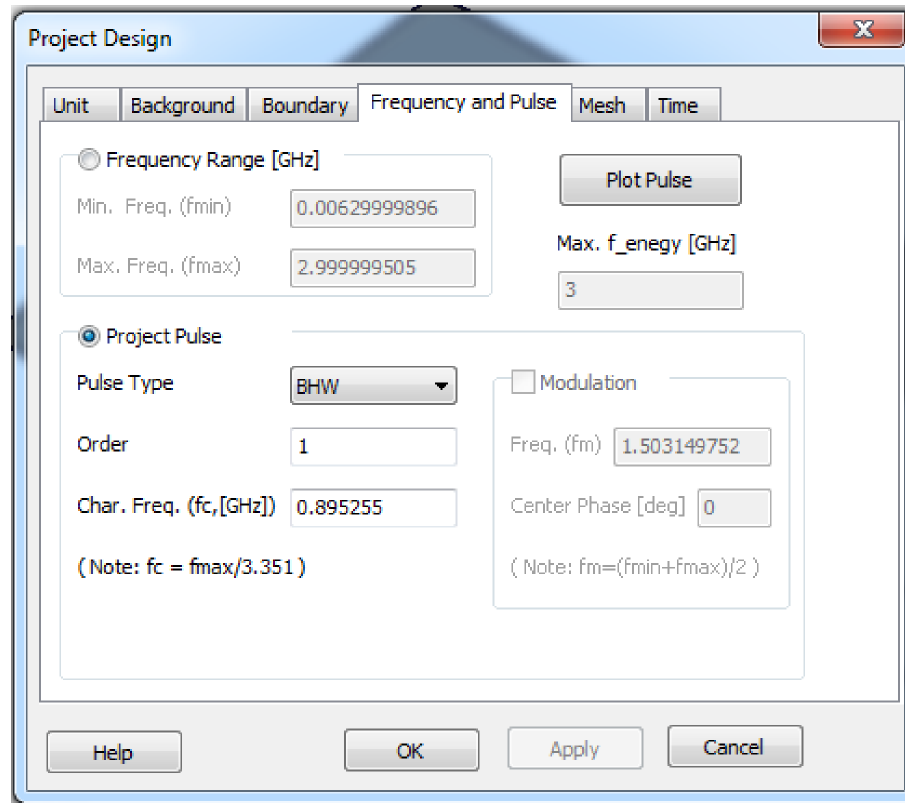




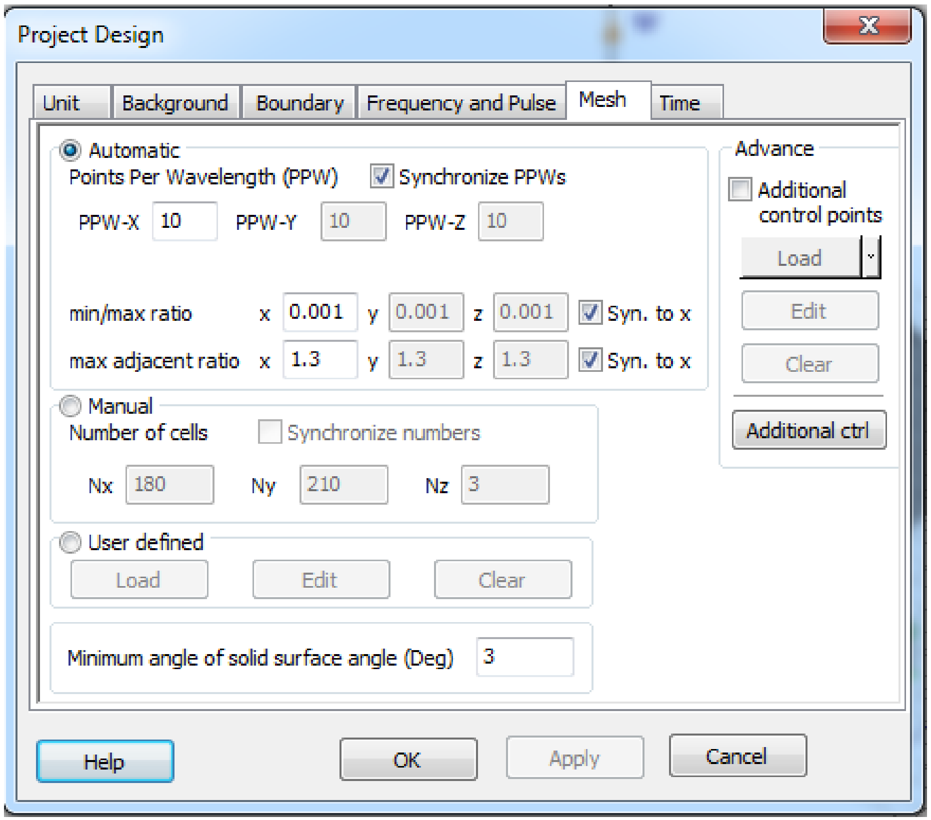
B.C. in Z is periodic for this 2D case

Source pulse is an order=1 BHW waveform with f_{max}=3 Ghz

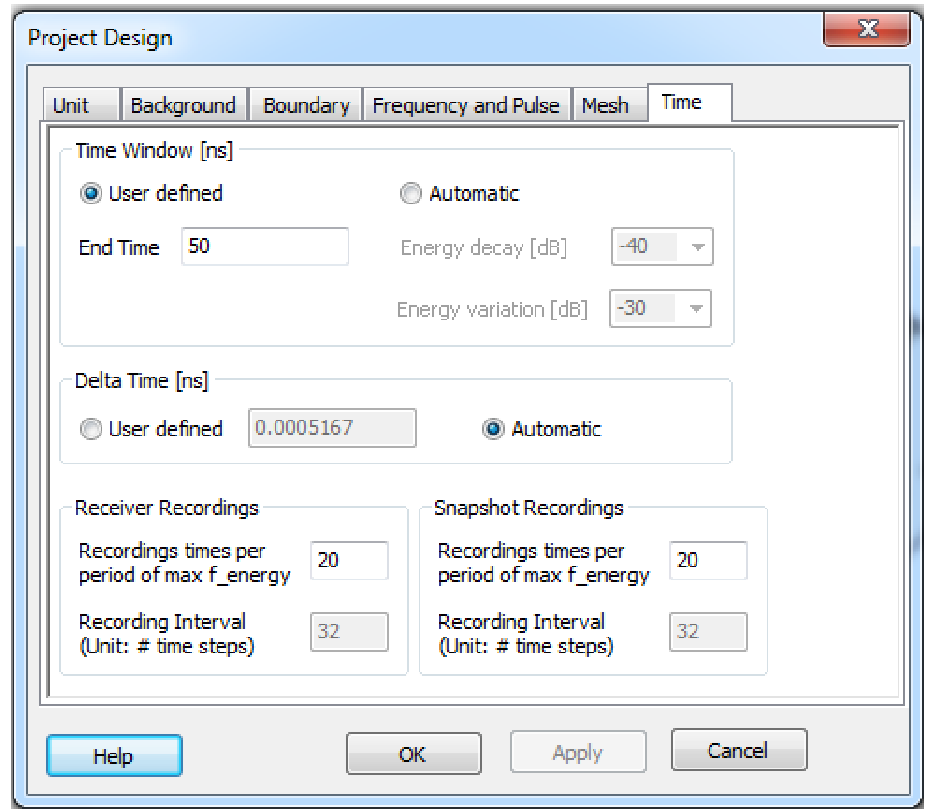




Automatic mesh with ppw=10

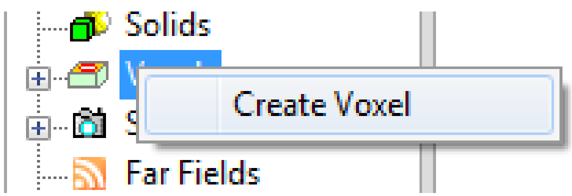


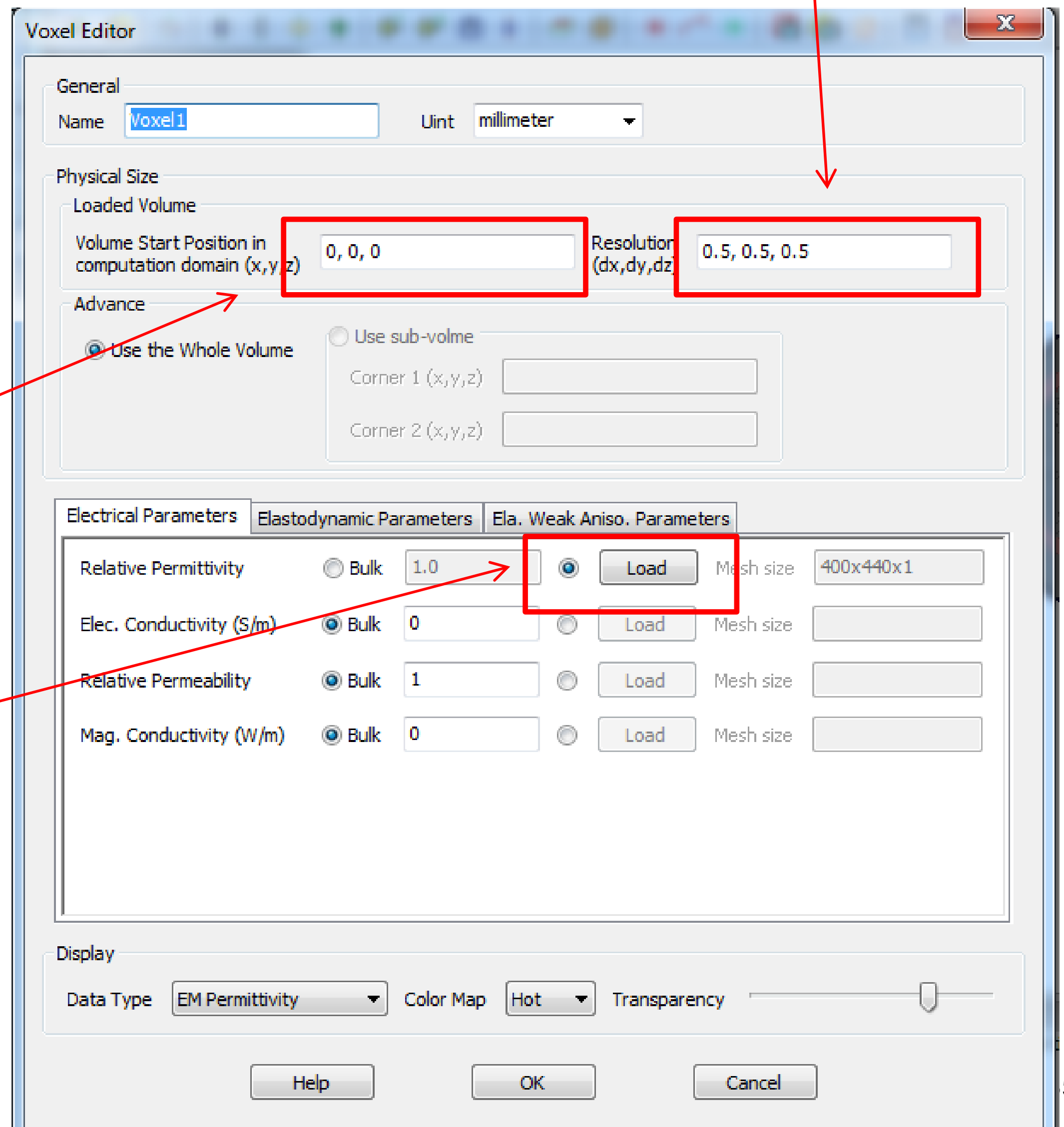
With a fixed time window as 50 ns



Then we create voxel to load 2D human skull slice

Voxel resolution

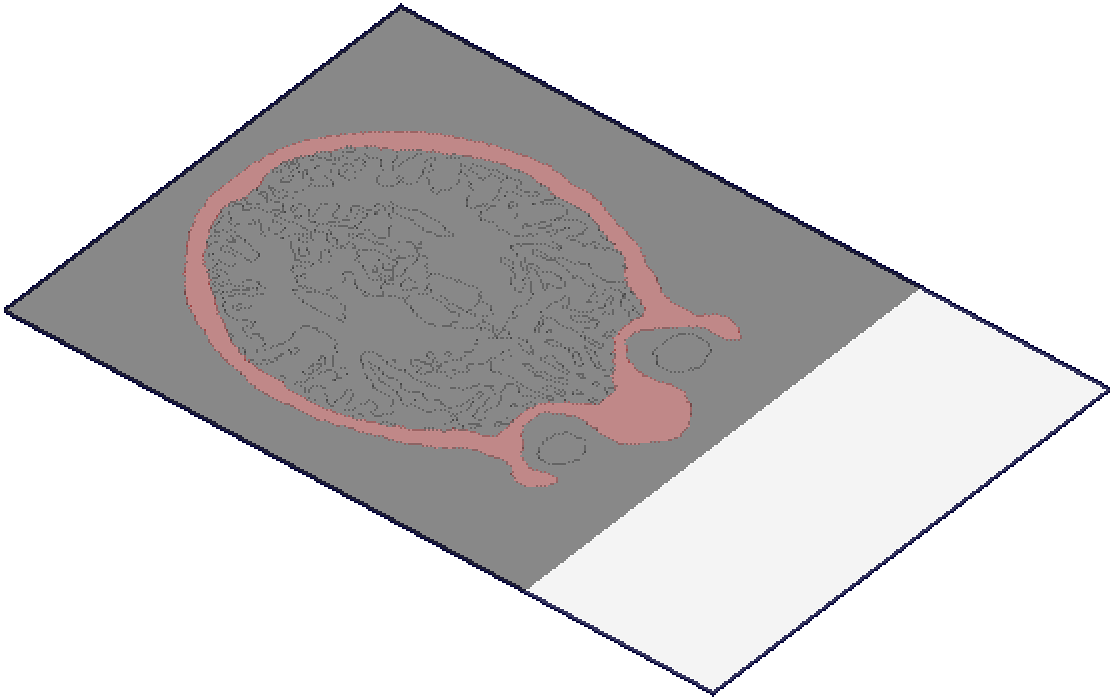




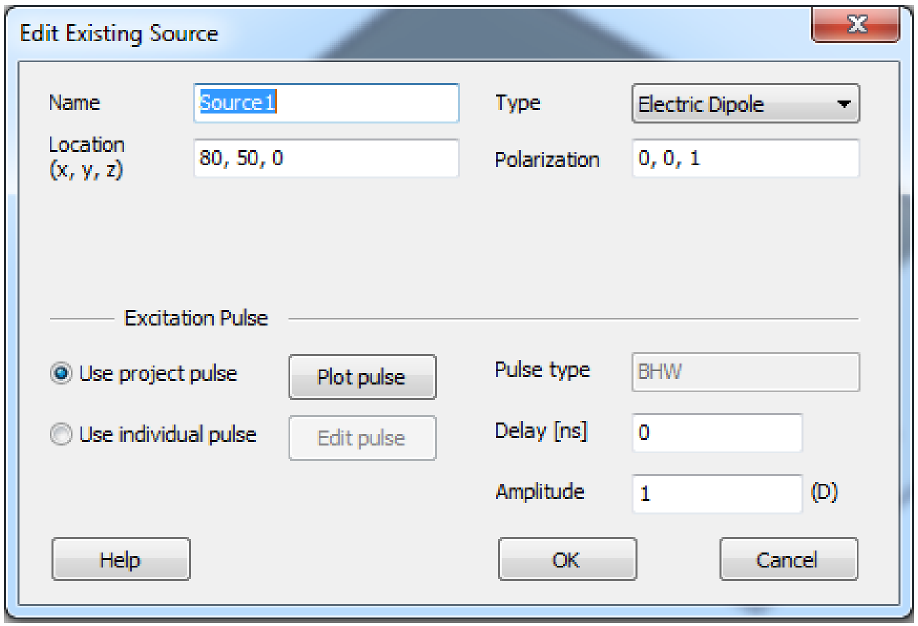
Slice start position in the project

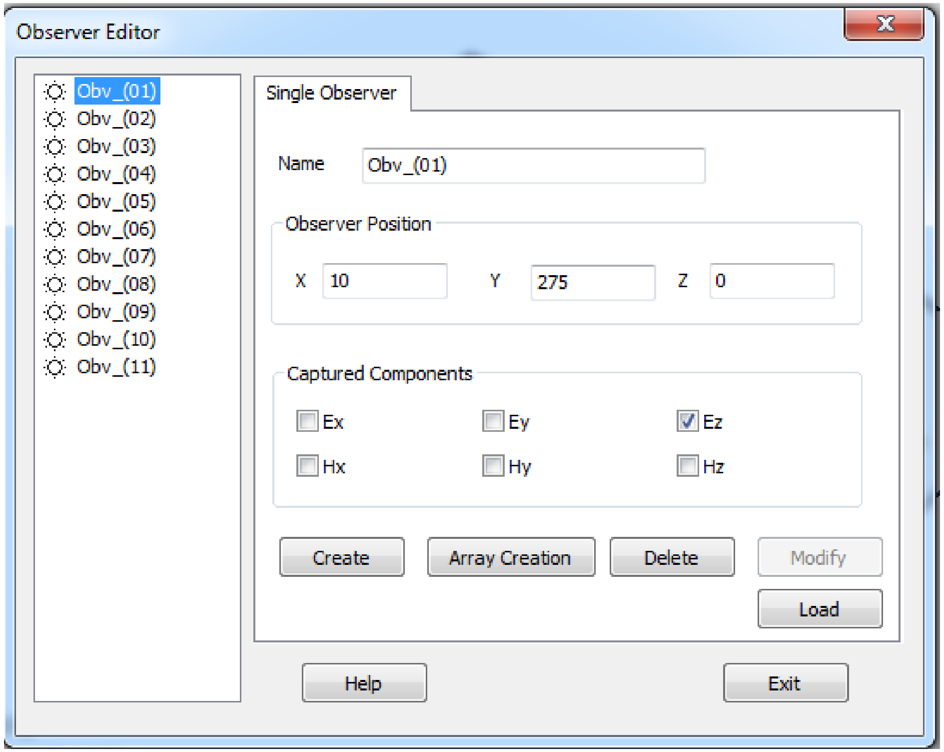
Load **section_1400_eps_r.vxa**

Then we can get project as figure

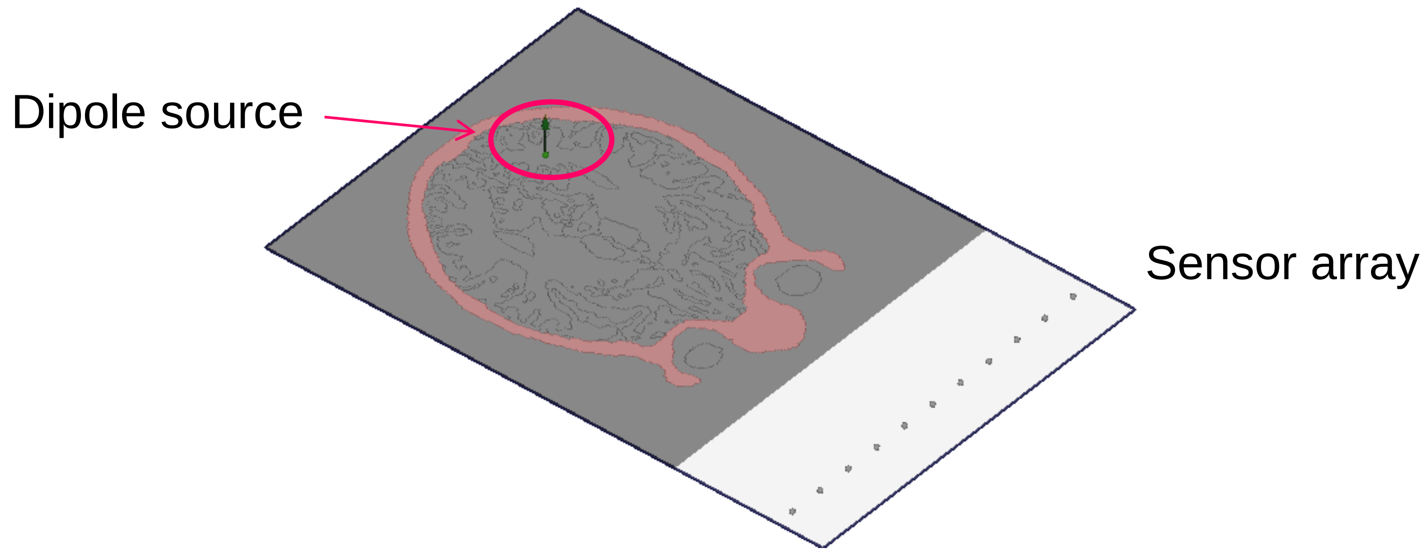


Set up a Z polarized E dipole at (80, 50, 0), an array of observers at (10:18:190, 275, 0) to record Ez field.





The project layout is



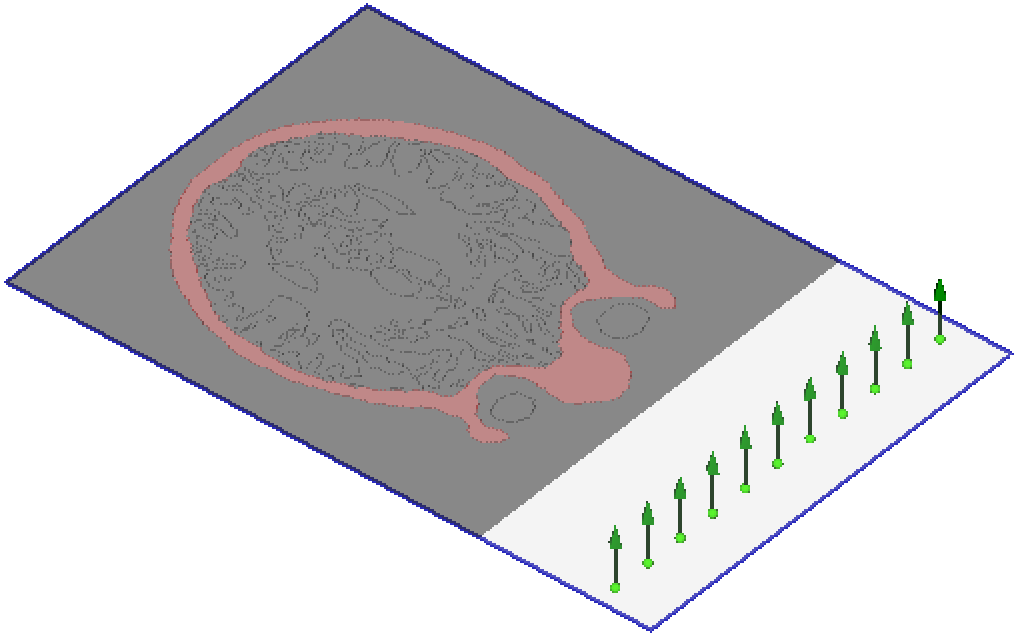
Simulate the project, we get the transient Ez field on the sensor.
The data file for it is: **2_1_res\observers\ 2_1_rev_ez.txt**

This data will be considered as the signal for the time reversal imaging case.

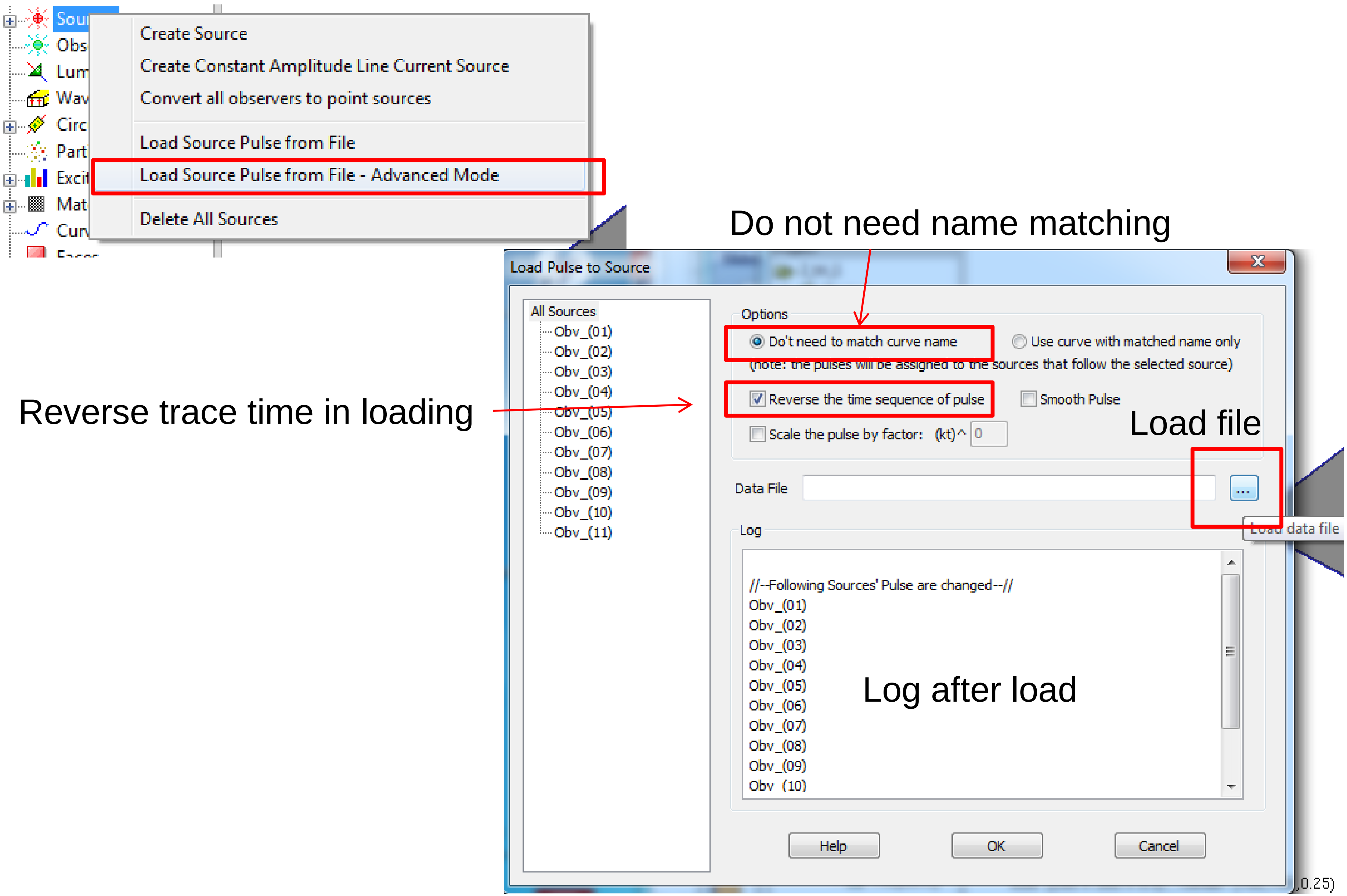
Part 2: Project for the time-reversal imaging

- we can “**SaveAs**” project “**2_1.wnt**” as “**2_tri_1.wnt**”
- delete the original dipole source
- convert all observers to source

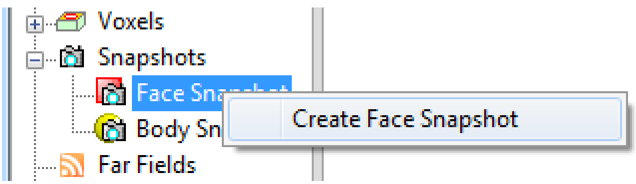
Then we get the project as

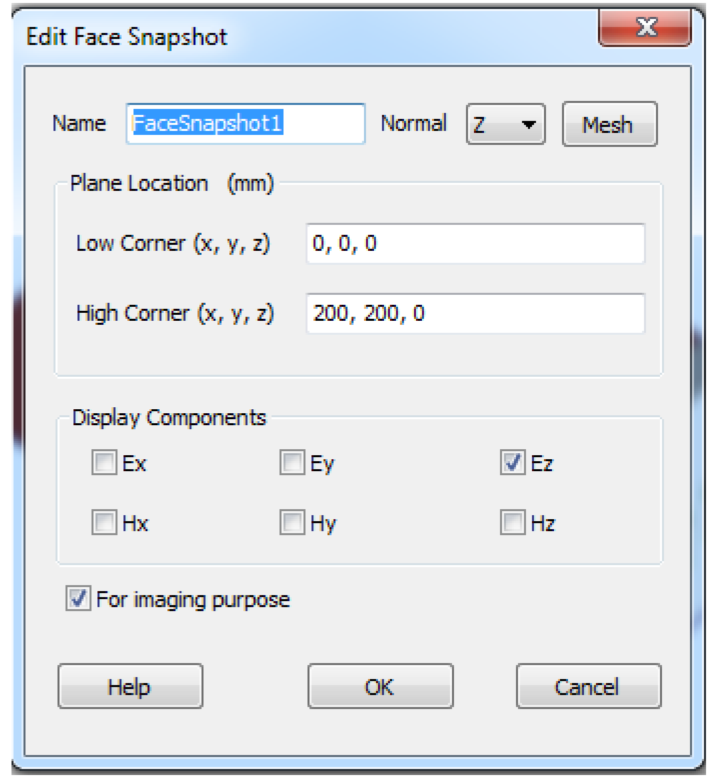


➢ Load signal to sources as the excitation pulse

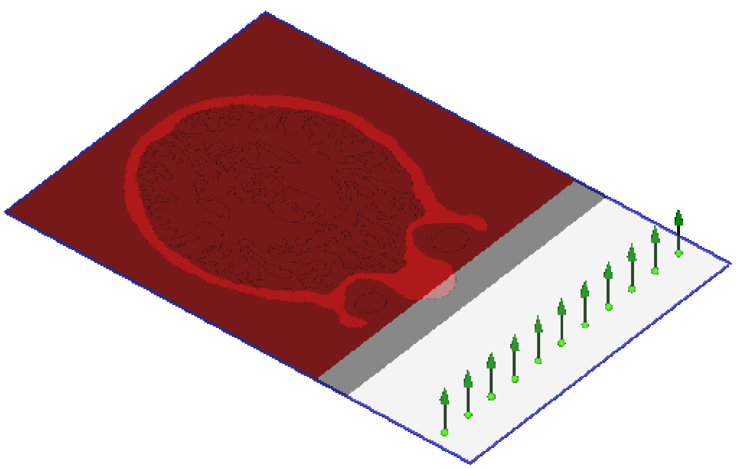


- define a 2D snapshot for imaging

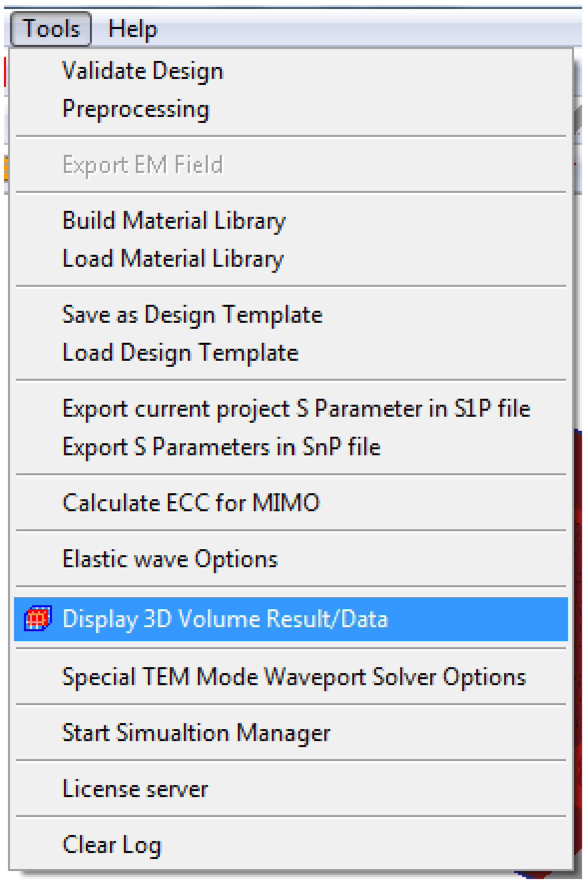


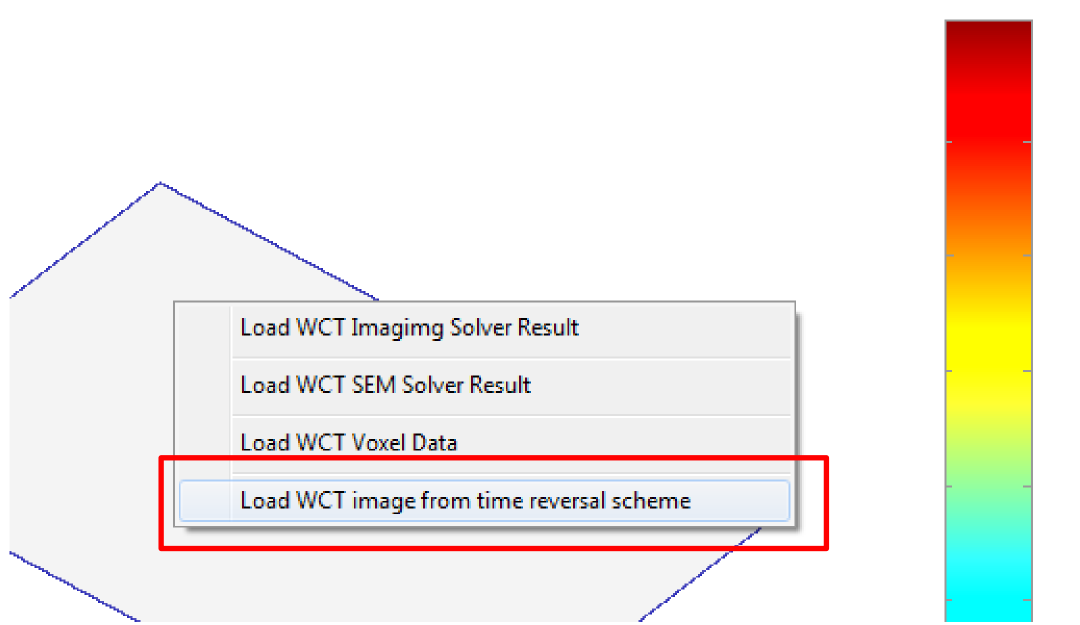


Snapshot area

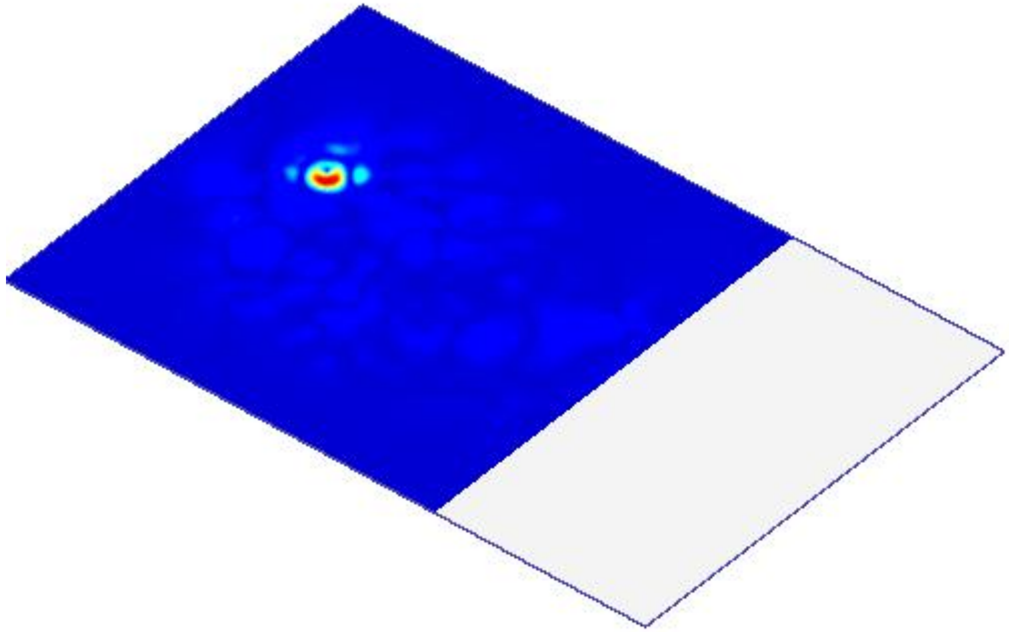


Simulate the project. After finish, we load the image with the maximum energy.





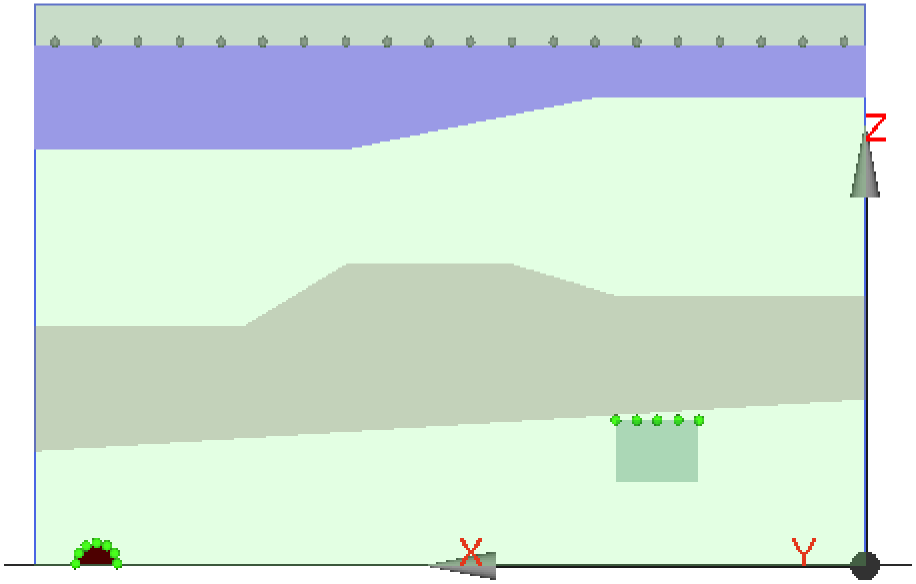
We can see the maximum energy position match the source position.



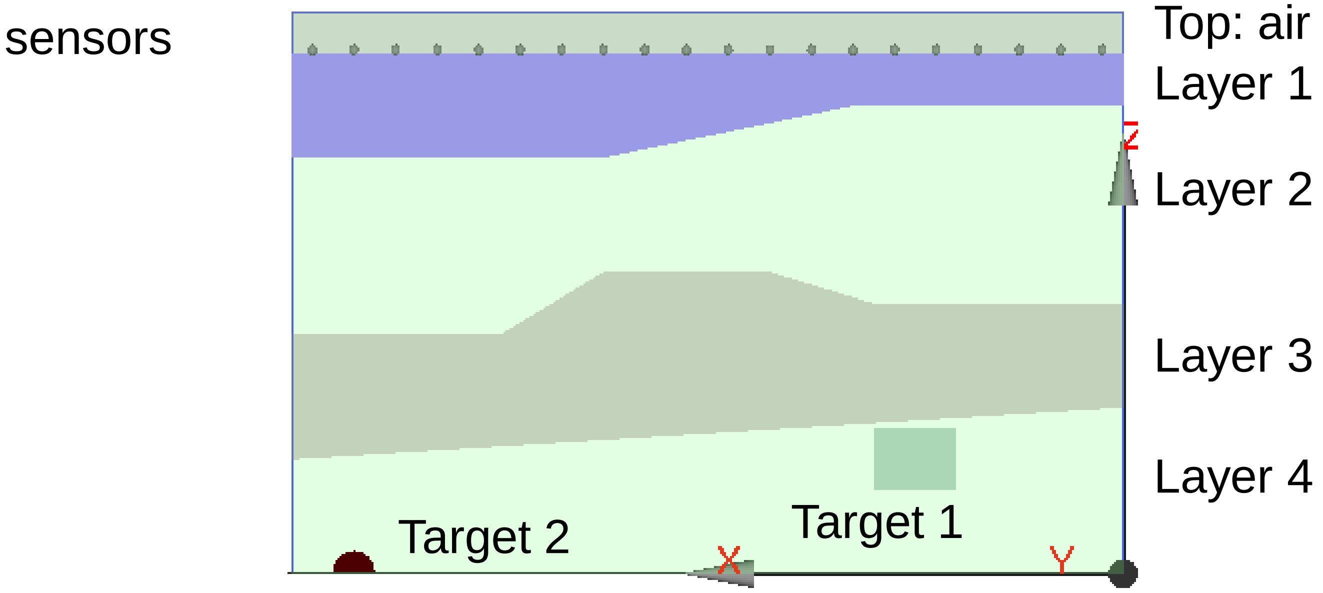
Case (4): Detect array of sources in multiple layers background by the direct wave

EL Solver

This case shows how to setup a time-reversal case to detect two source array in multiple layers background by the direct wave,

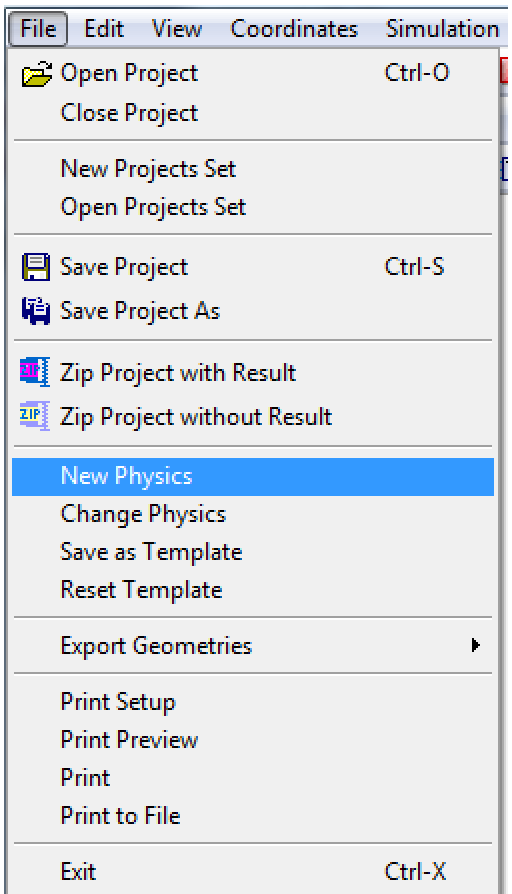


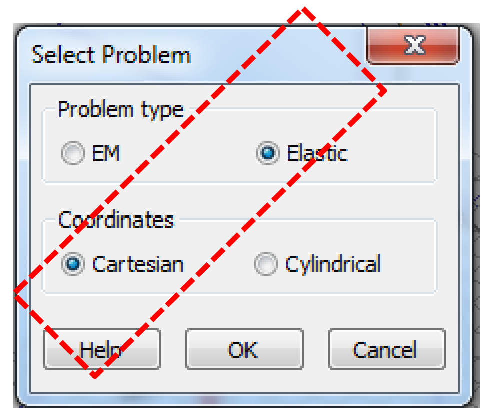
This case is a elastic wave project. It assumes that there are two objects embedded in a multiple layers background. Through some kind of operation in other physics, the edge of object become elastic wave sources. Then, we will use the signal on sensors to detect the edge of these two object.



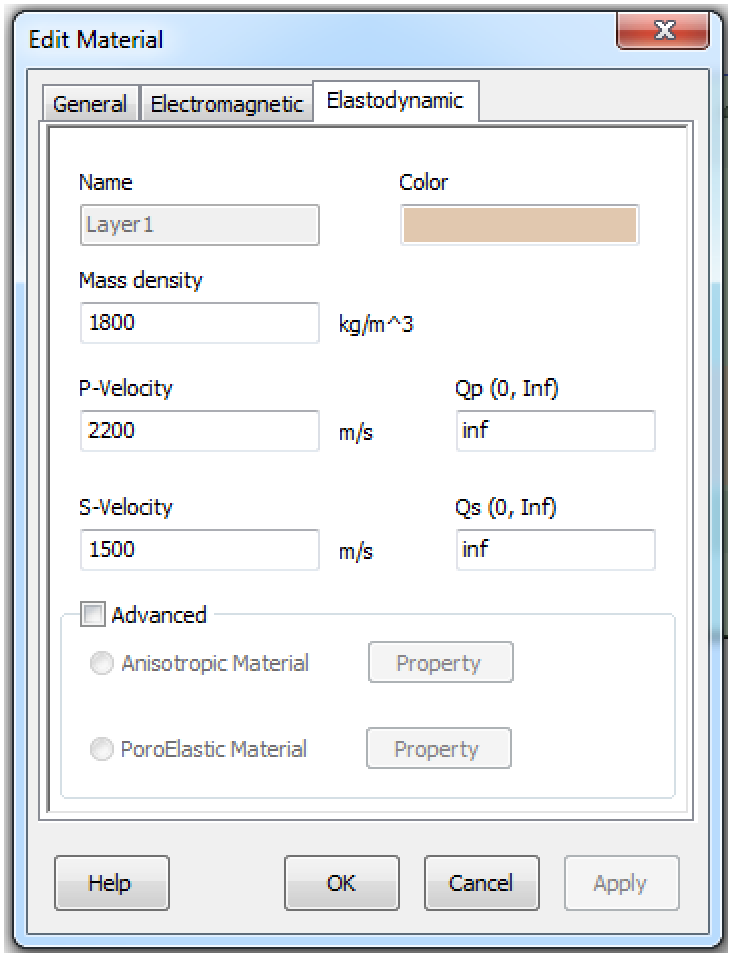
Part 1: the project to obtain the signal from equivalent array of monopole sources in 2D multiple layers background

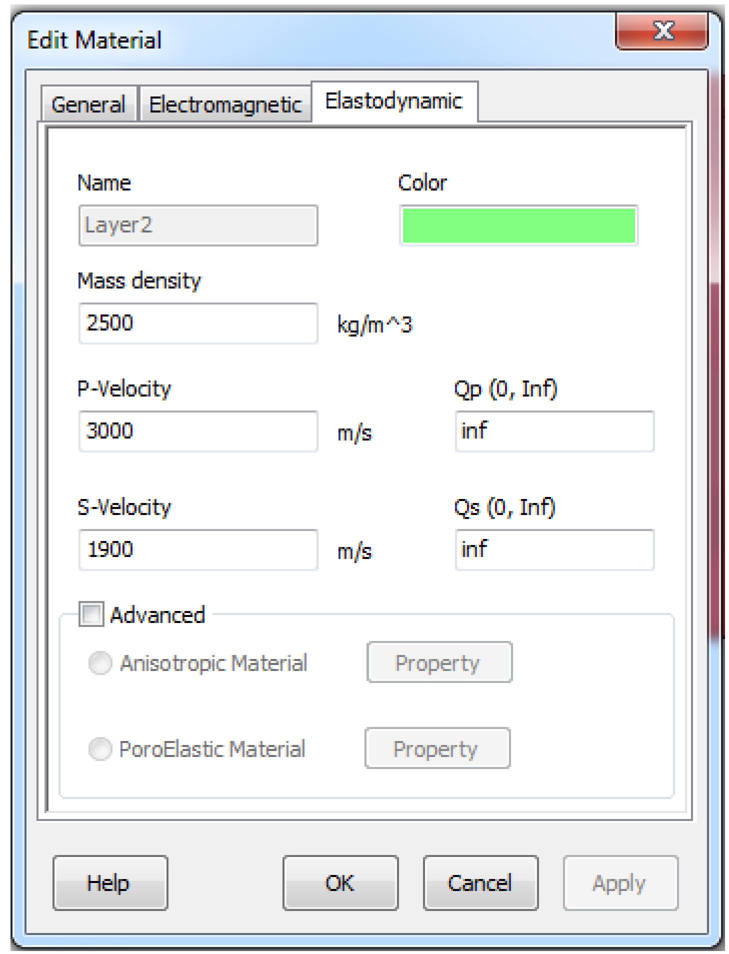
1. create a Cartesian EL project, named as "**Layered_2D_01.wnt**"

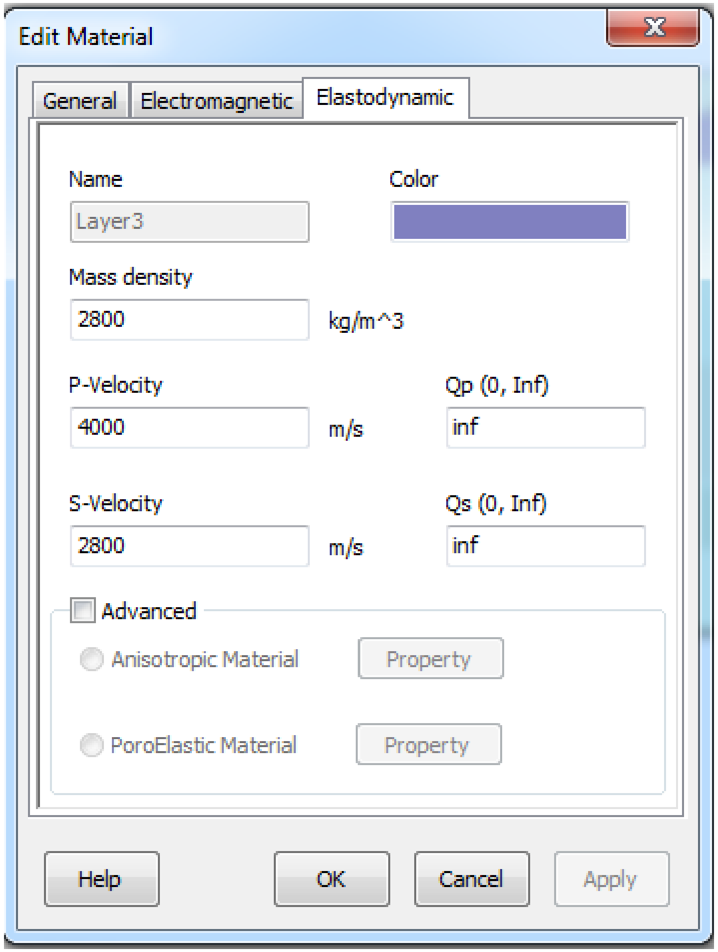


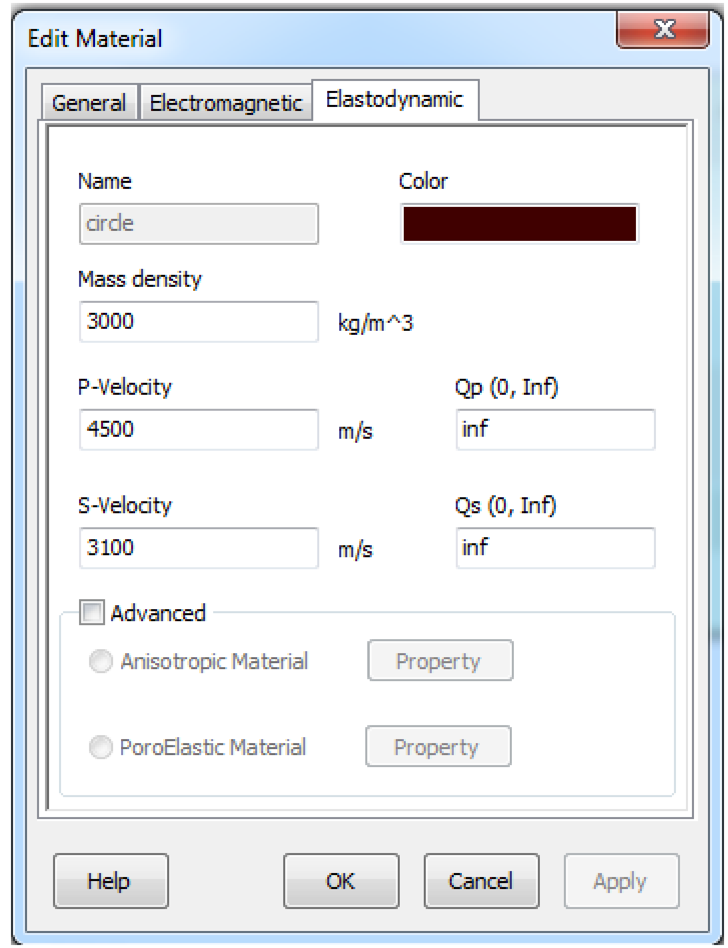


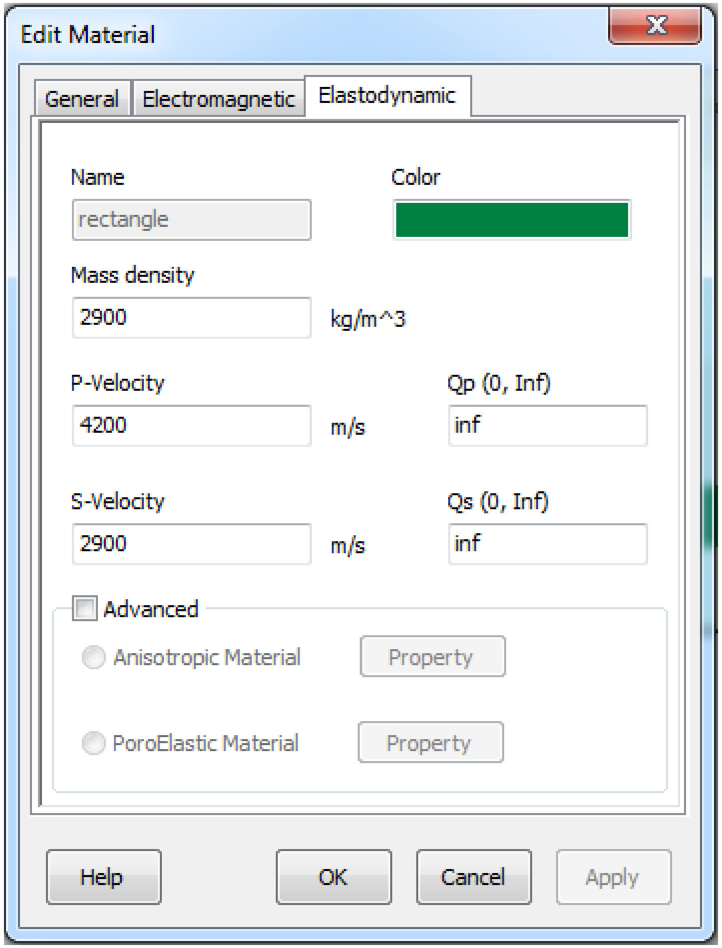
2. Create new materials for background and objects






Edit Material
General
Electromagnetic
Elastodynamic
Name
Color
Layer3
Mass density
2800
kg/m^3
P-Velocity
4000
m/s
Qp (0, Inf)
inf
S-Velocity
2800
m/s
Qs (0, Inf)
inf
Advanced
Anisotropic Material
Property
PoroElastic Material
Property
Help
OK
Cancel
Apply

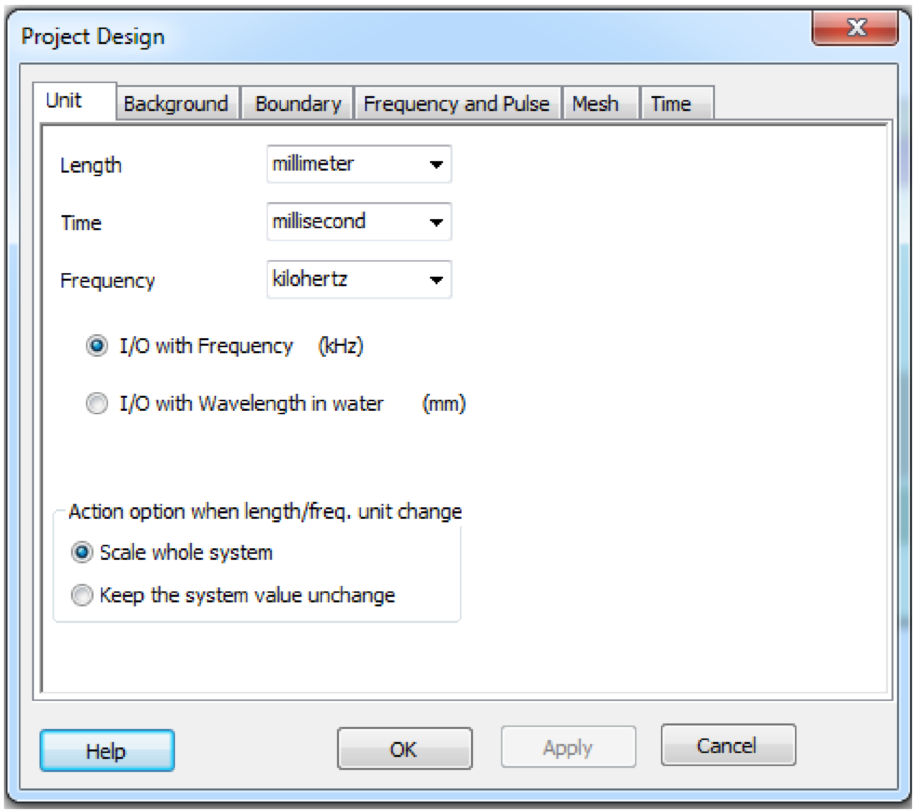

Edit Material
General
Electromagnetic
Elastodynamic
Name
Color
circle
Mass density
3000
kg/m^3
P-Velocity
4500
m/s
Qp (0, Inf)
inf
S-Velocity
3100
m/s
Qs (0, Inf)
inf
Advanced
Anisotropic Material
Property
PoroElastic Material
Property
Help
OK
Cancel
Apply


Edit Material
General
Electromagnetic
Elastodynamic
Name
Color
rectangle
Mass density
2900
kg/m^3
P-Velocity
Qp (0, Inf)
4200
m/s
inf
S-Velocity
Qs (0, Inf)
2900
m/s
inf
Advanced
Anisotropic Material
Property
PoroElastic Material
Property
Help
OK
Cancel
Apply

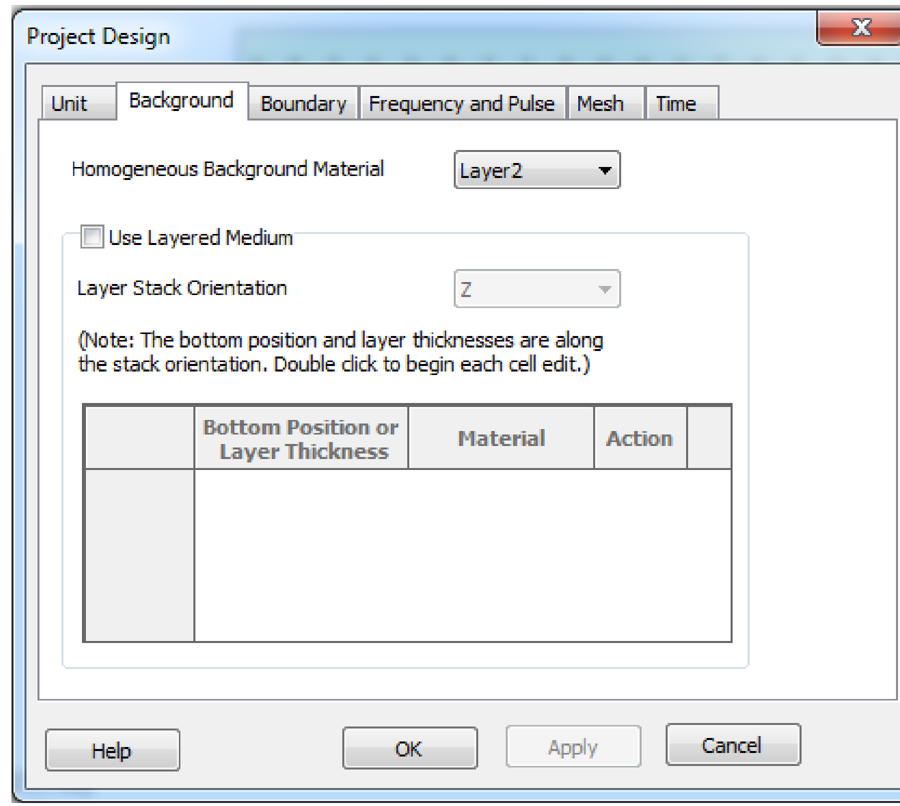
3. Project setting

Set up unit as:

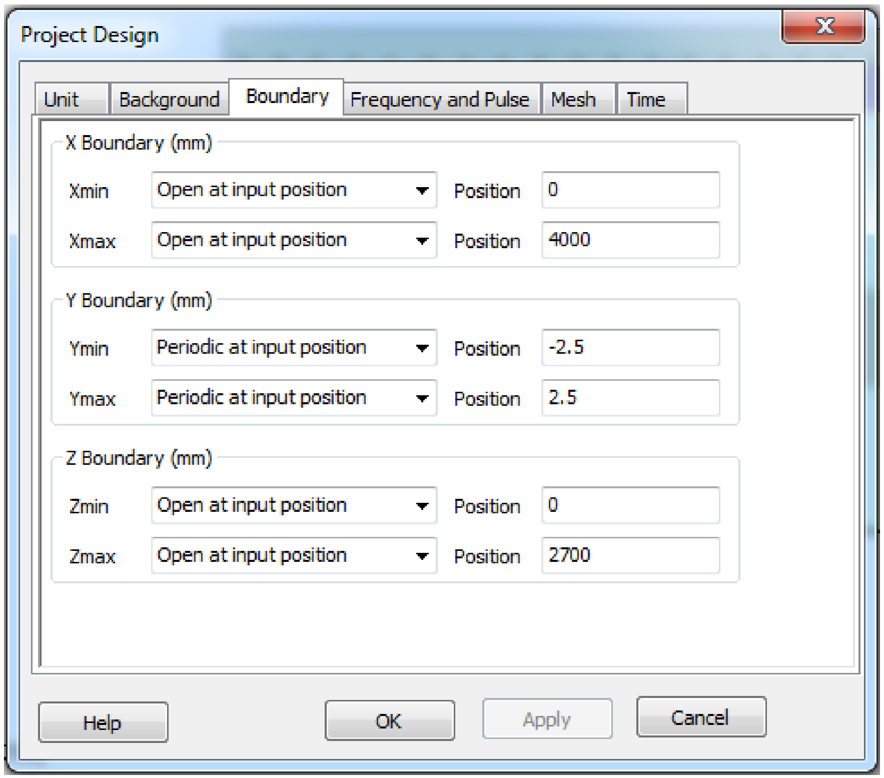
- mm in length
- ms in time,
- KHz for freq



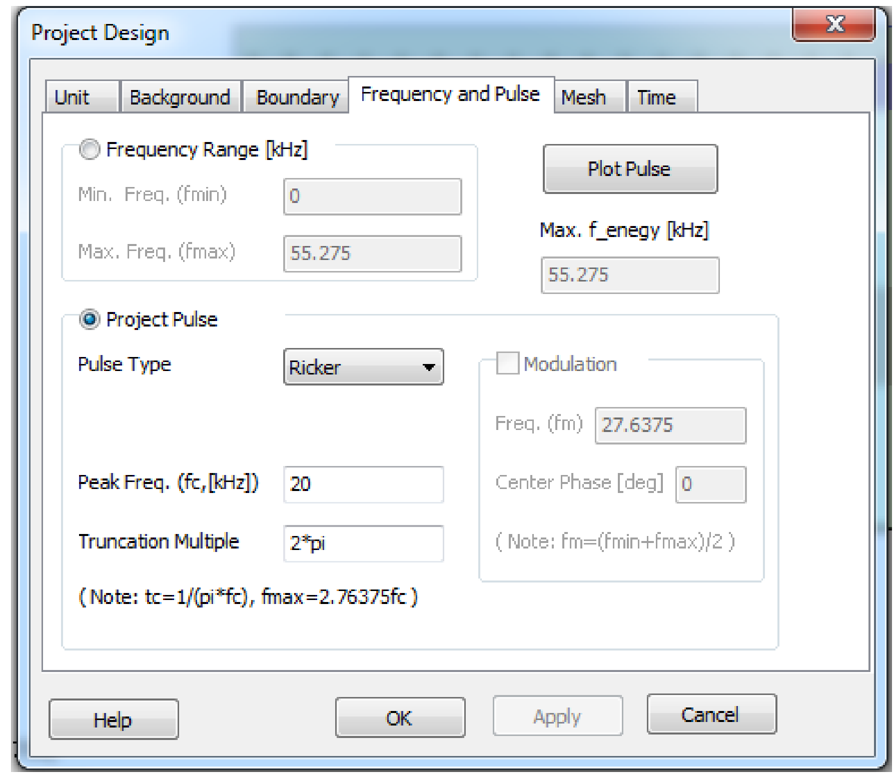
A homogenous background with material “Layer2”



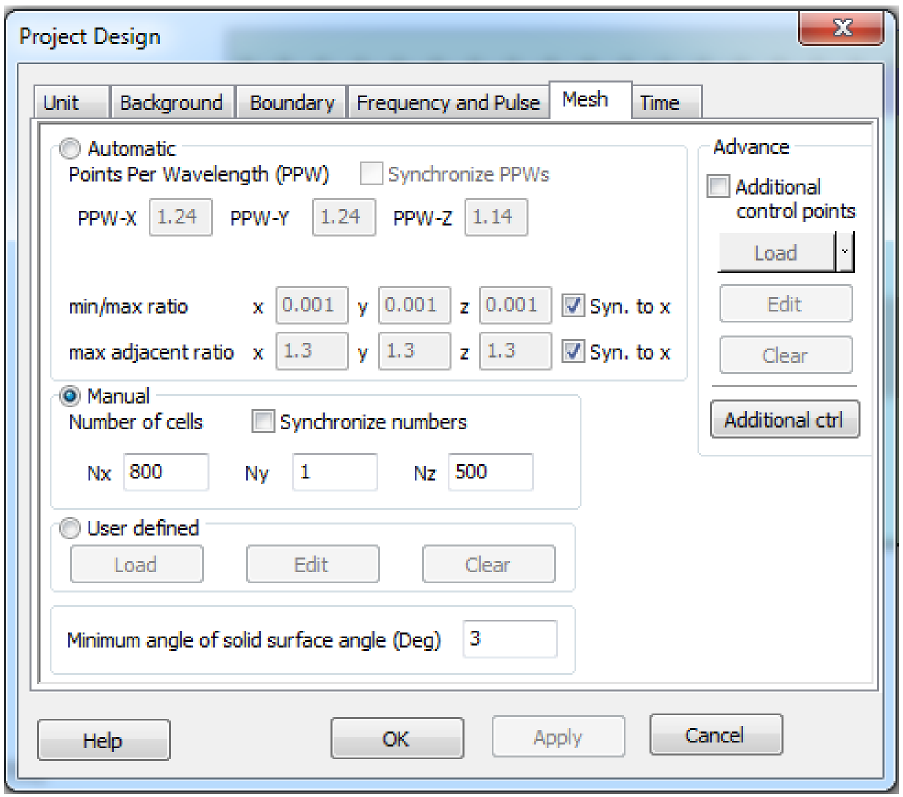
Boundary position & type



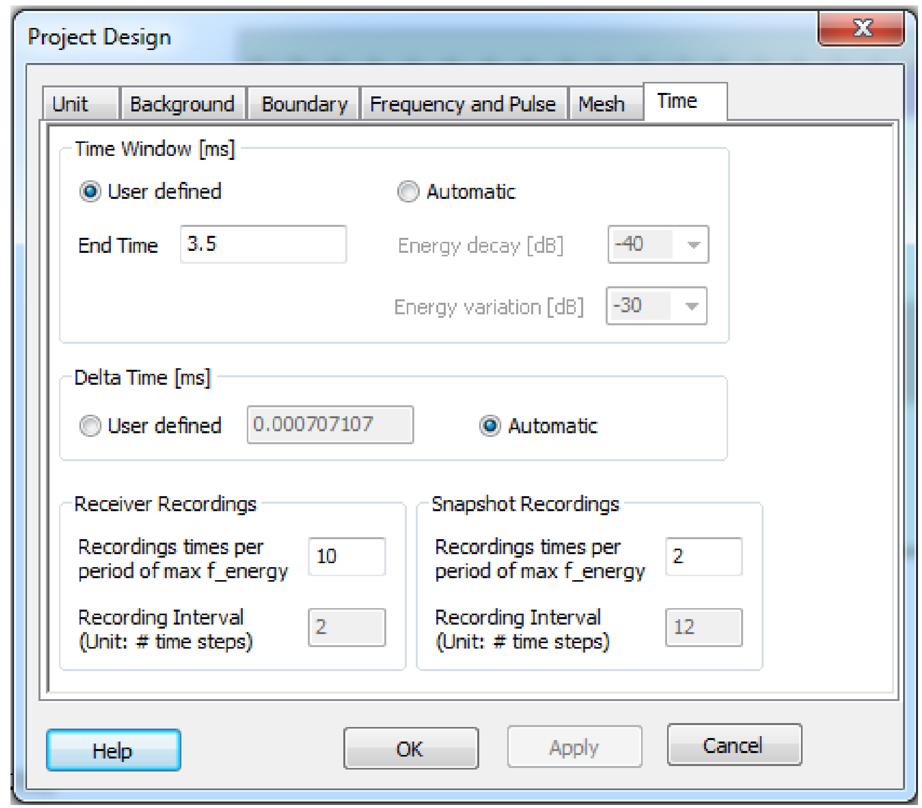
Excitation pulse is 1st order BHW waveform with f_{max}≈55 KHz



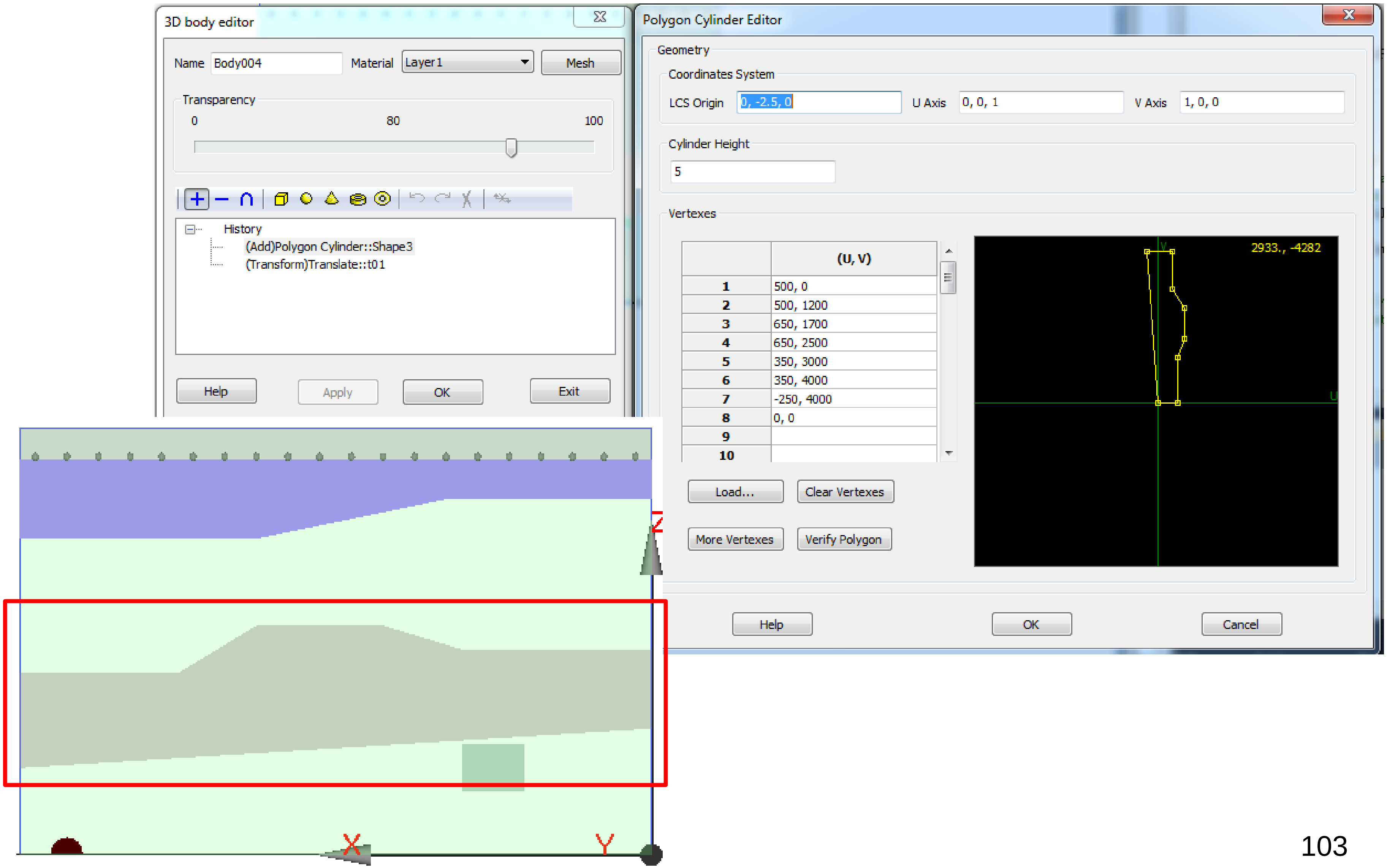
Uniform mesh

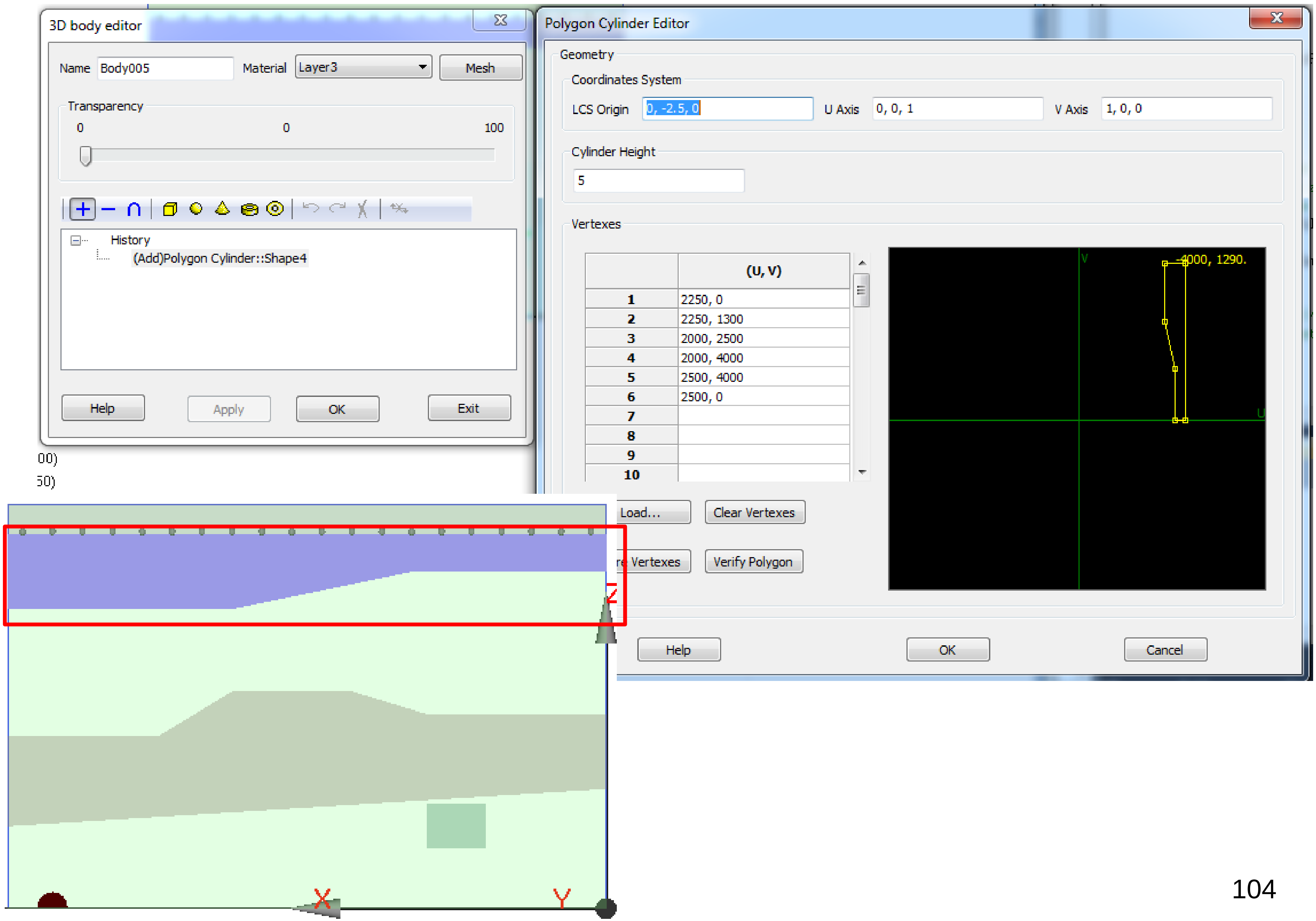


Time setting



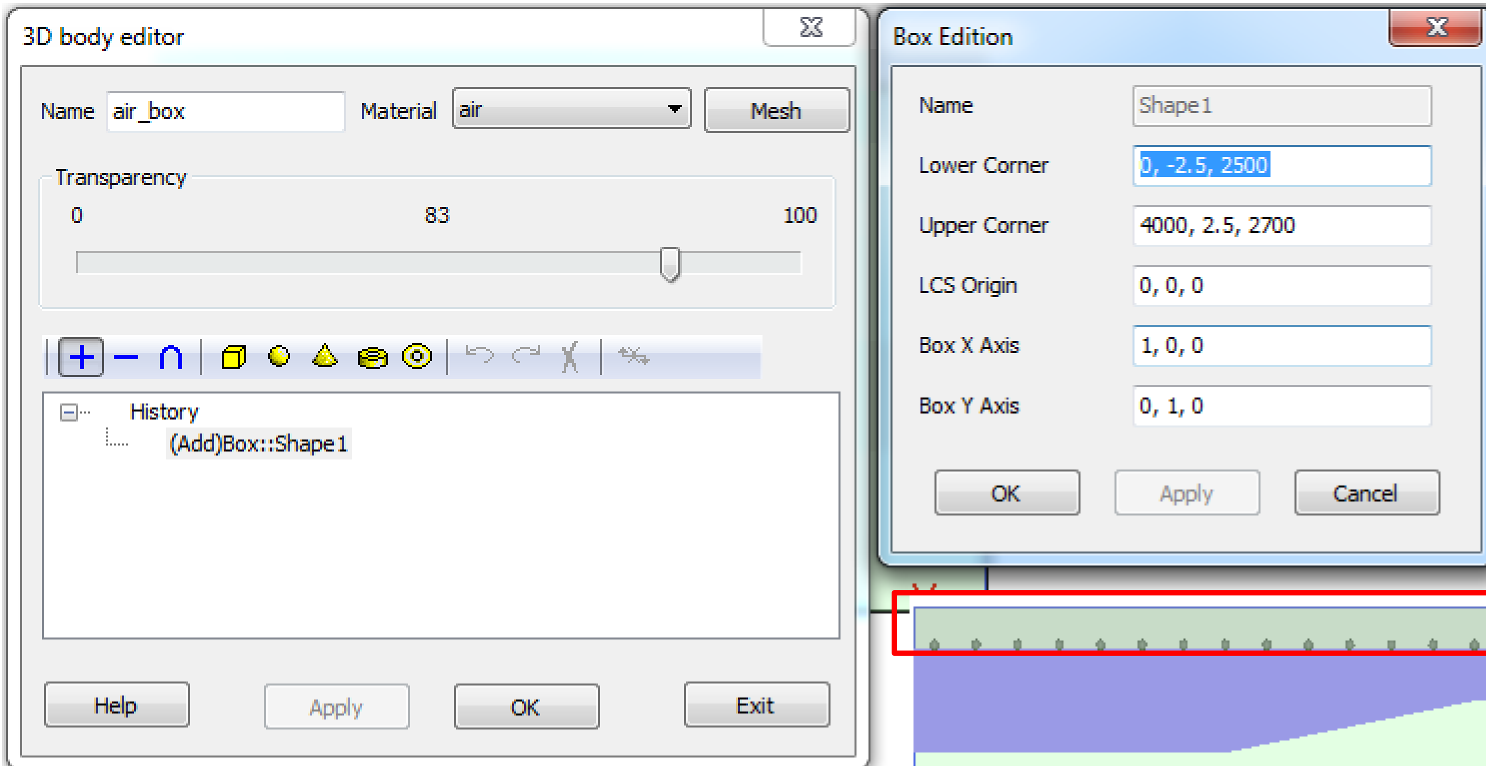
4. Build layers in the homogenous background

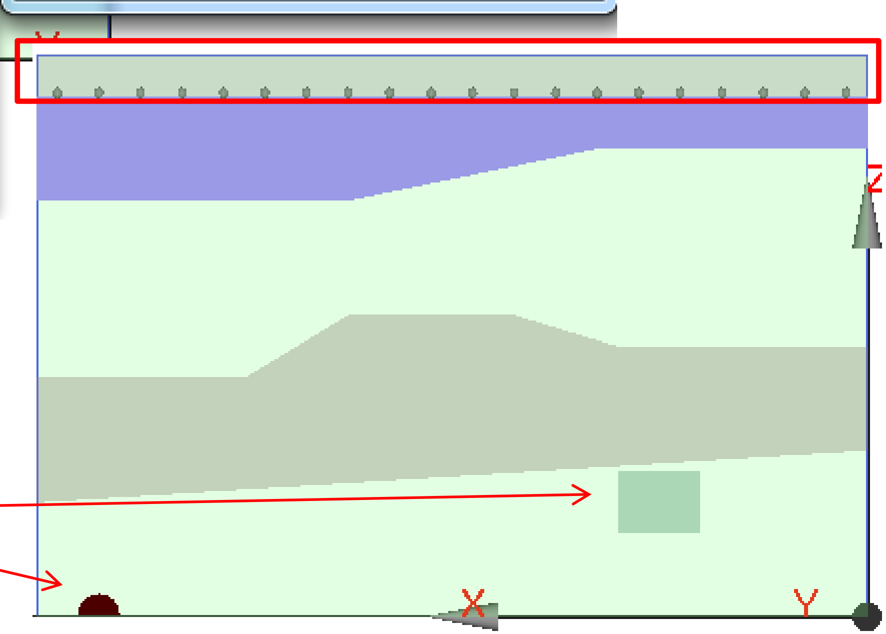




3D body editor
Name Body005
Material Layer3
Mesh
Transparency
0
0
100
History
(Add)Polygon Cylinder::Shape4
Help
Apply
OK
Exit
Polygon Cylinder Editor
Geometry
Coordinates System
LCS Origin 0, -2.5, 0
U Axis 0, 0, 1
V Axis 1, 0, 0
Cylinder Height
5
Vertexes
(U, V)
1 2250, 0
2 2250, 1300
3 2000, 2500
4 2000, 4000
5 2500, 4000
6 2500, 0
7
8
9
10
Load...
Clear Vertexes
Vertexes
Verify Polygon
Help
OK
Cancel

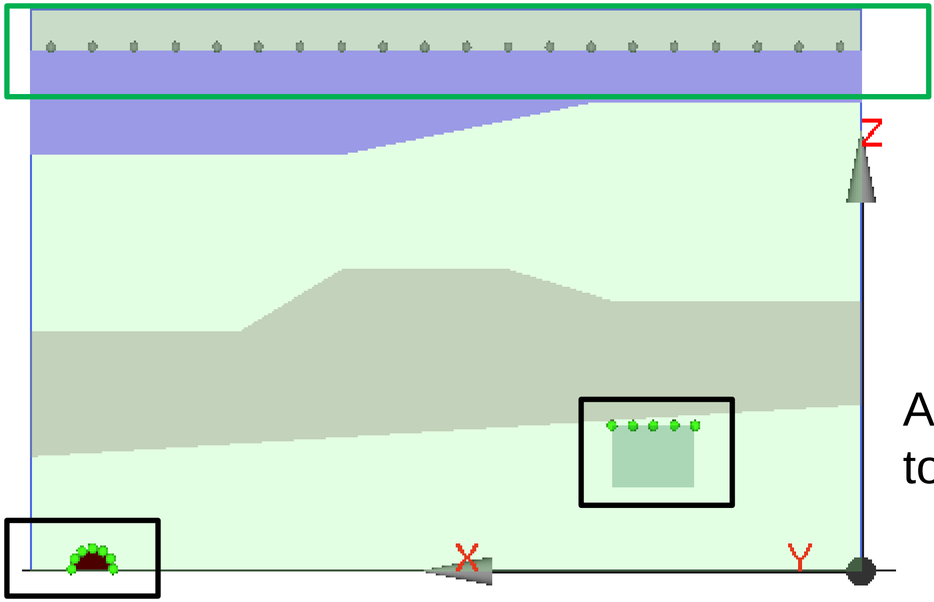
Top layer is an air box





Then two objects

5. Add array of elastic monopole source at the top edge of object, and create an array of observers to record T_{au_xx} and T_{au_zz}, as following



Array of source on the top edge of target 1

Array of source on the top edge of target 2

6. Simulate the project, we get the transient T_{au_xx} field on the sensor. The data files for it are:

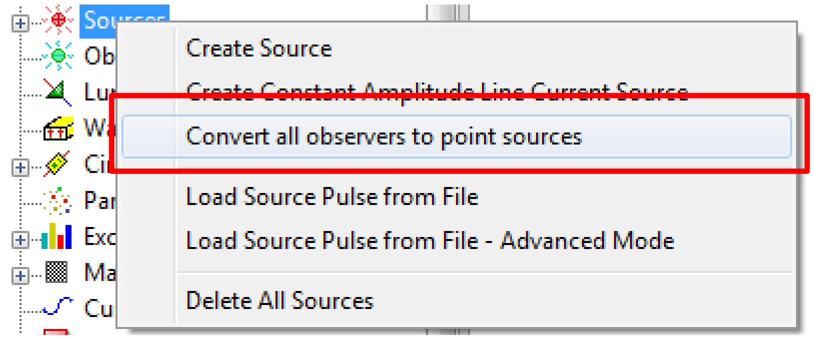
Layered_2D_01_res\observers \ Layered_2D_01_obv_txx_time.txt

Layered_2D_01_res\observers \ Layered_2D_01_obv_tzz_time.txt

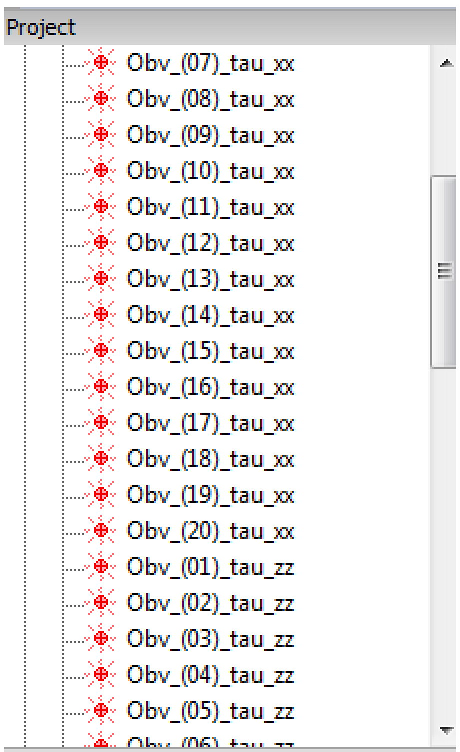
These data will be considered as the signal for the time reversal imaging case.

Part 2: Project for the time-reversal imaging

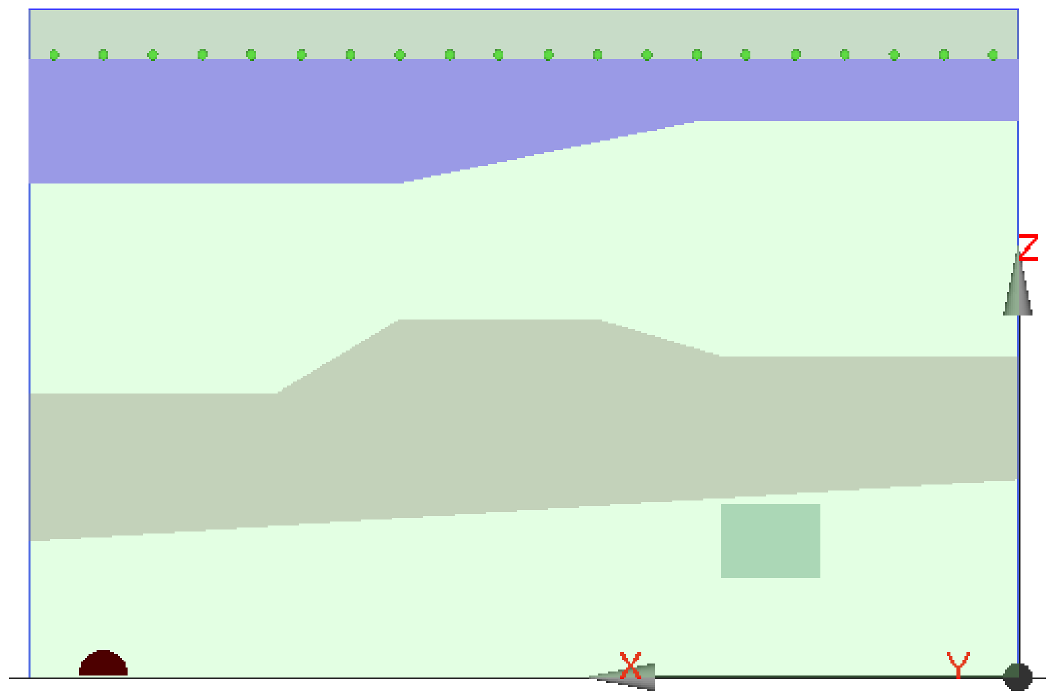
- we can “**SaveAs**” project “**Layered_2D_01.wnt**” as “**Layered_2D_01_tri.wnt**”
- delete the original sources
- convert all observers to monopole sources



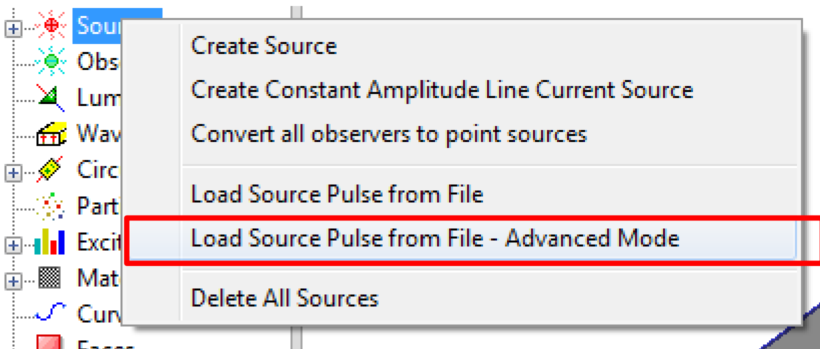
Then we get the project as



Converted sources array



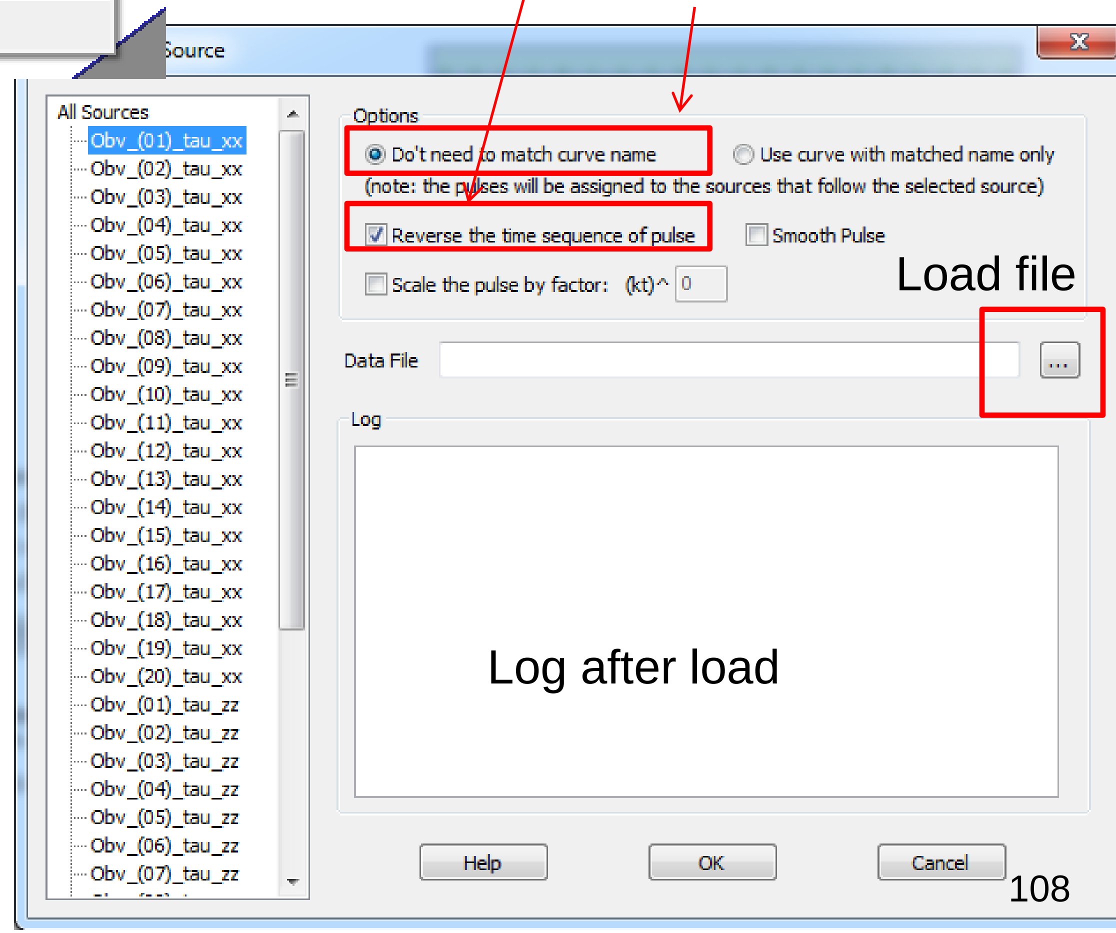
- Load signal to sources as the excitation pulse



Reverse trace time in loading

Do not need name matching

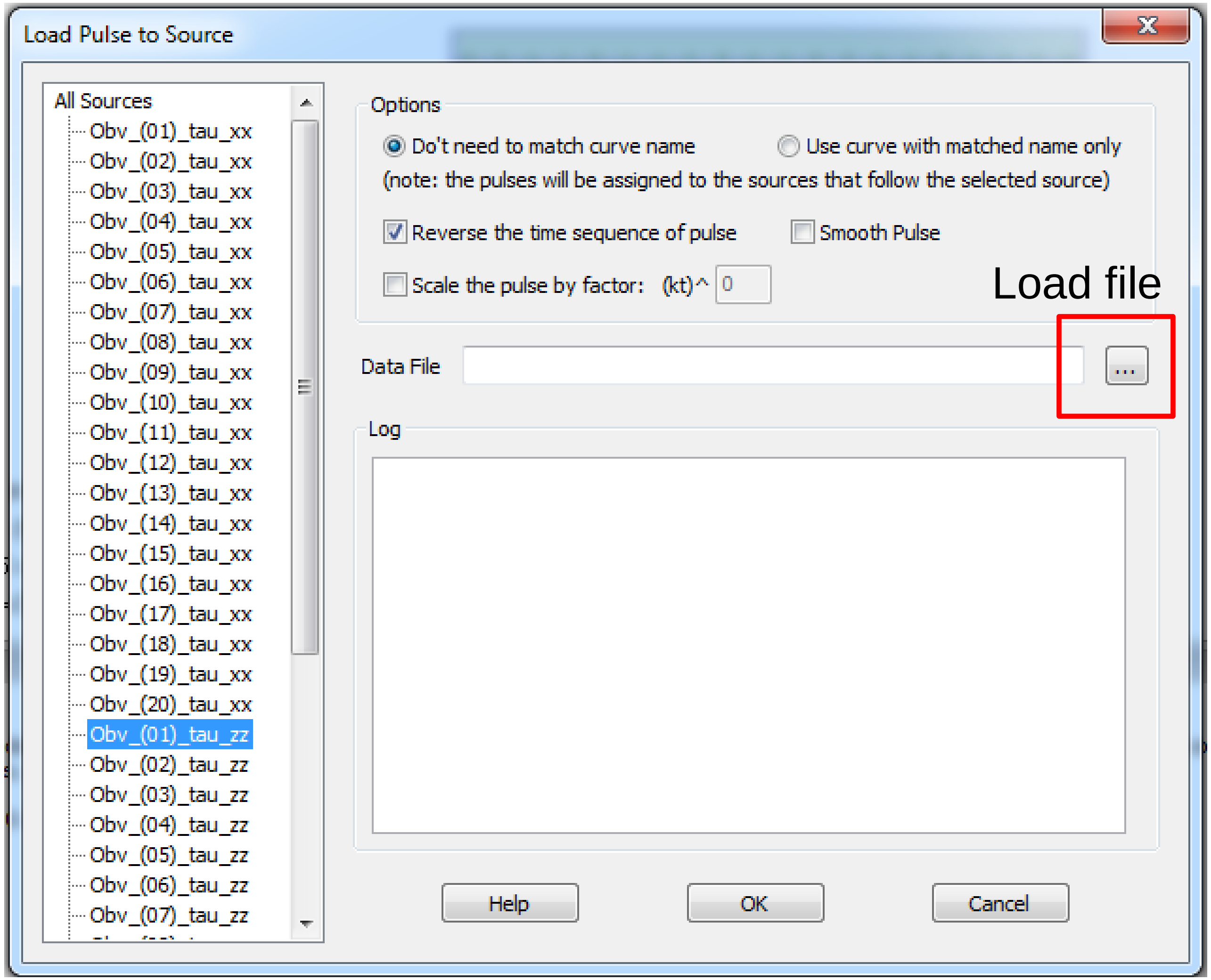
Select the 1st named with *Tau_xx* source. Load pulse from data file: **Layered_2D_01_obv_txx_time.txt**



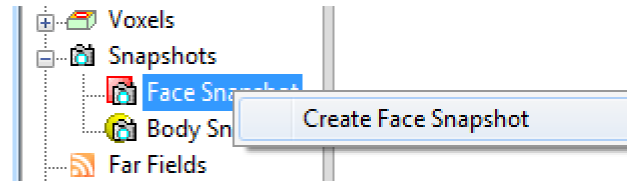
Load file

Log after load

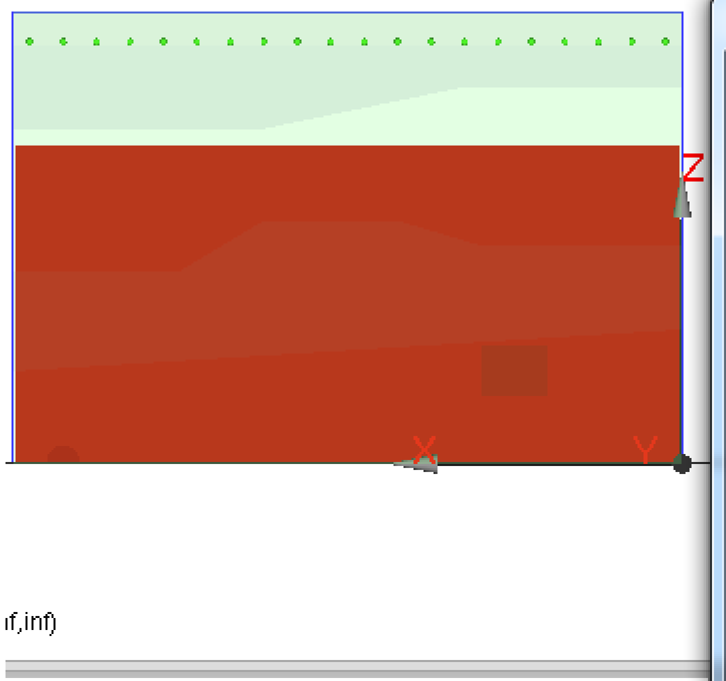
Select the 1st named with *Tau_zz* source. Load pulse from data file: **Layered_2D_01_obv_tzz_time.txt**

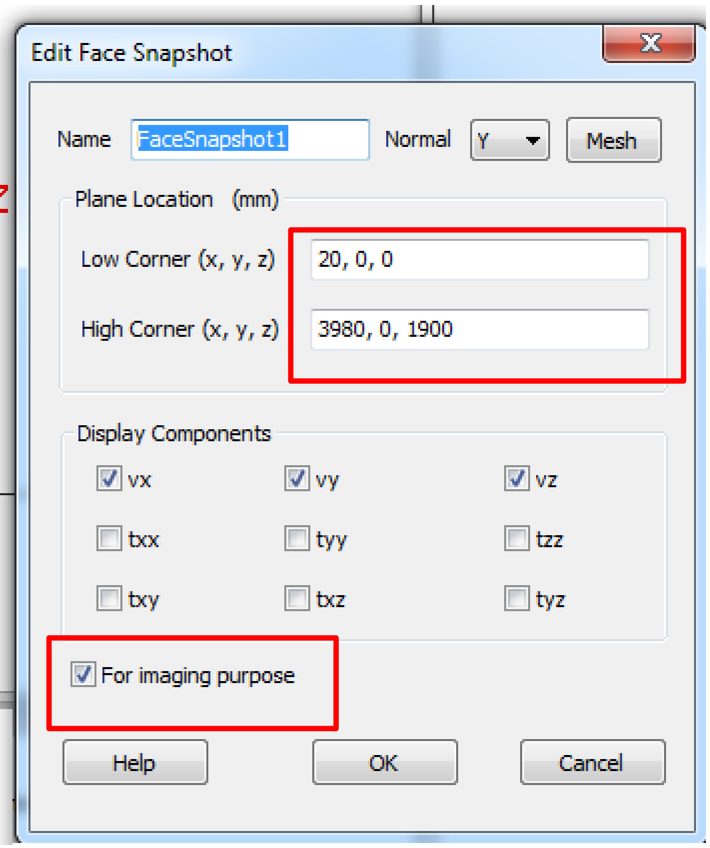


- define a 2D snapshot for imaging

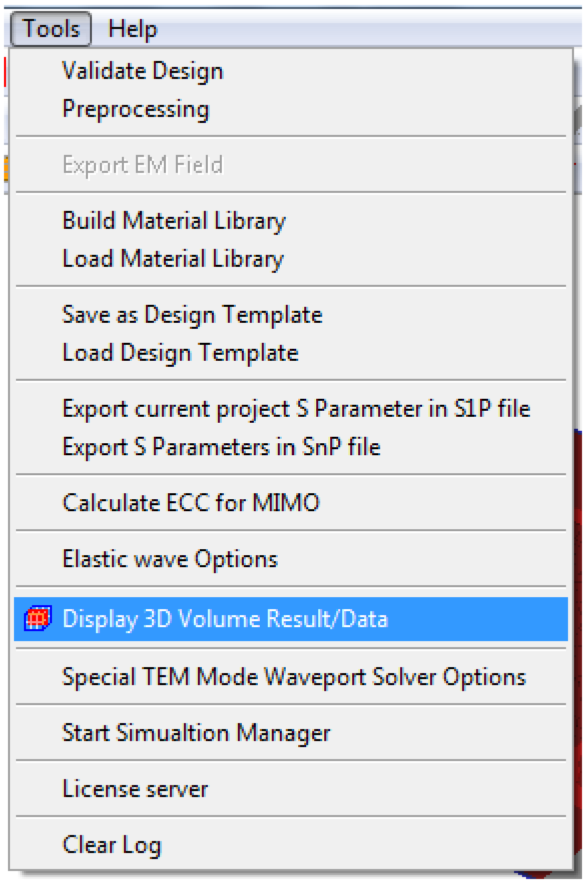


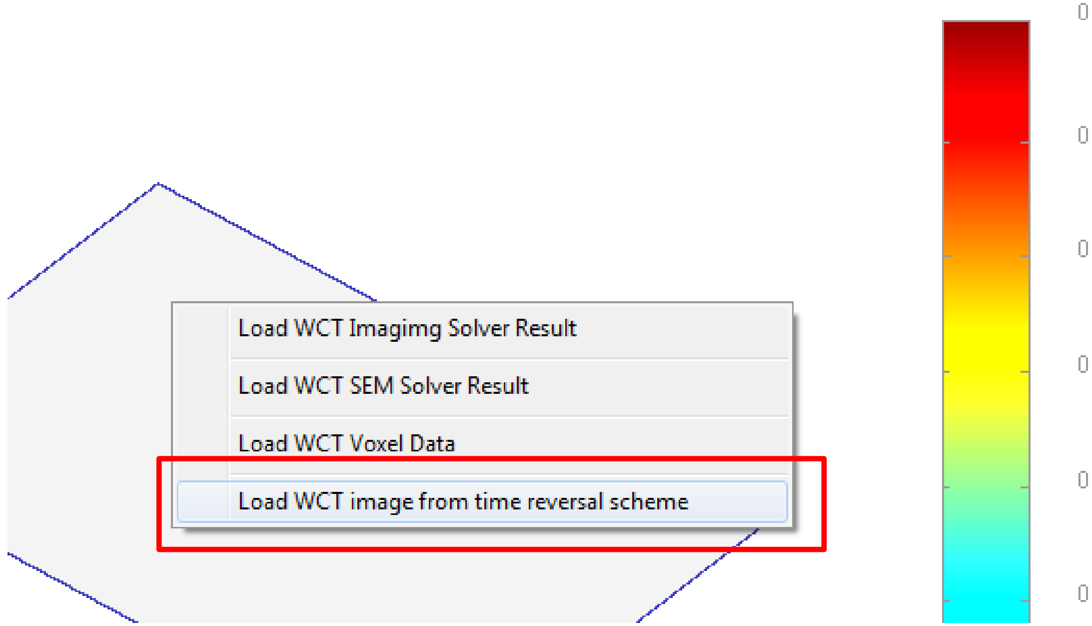
Snapshot area





Simulate the project.   After finish, we load the image with the maximum energy.





We can see the maximum energy position match the source array position.