Electromagnetic code ECHO

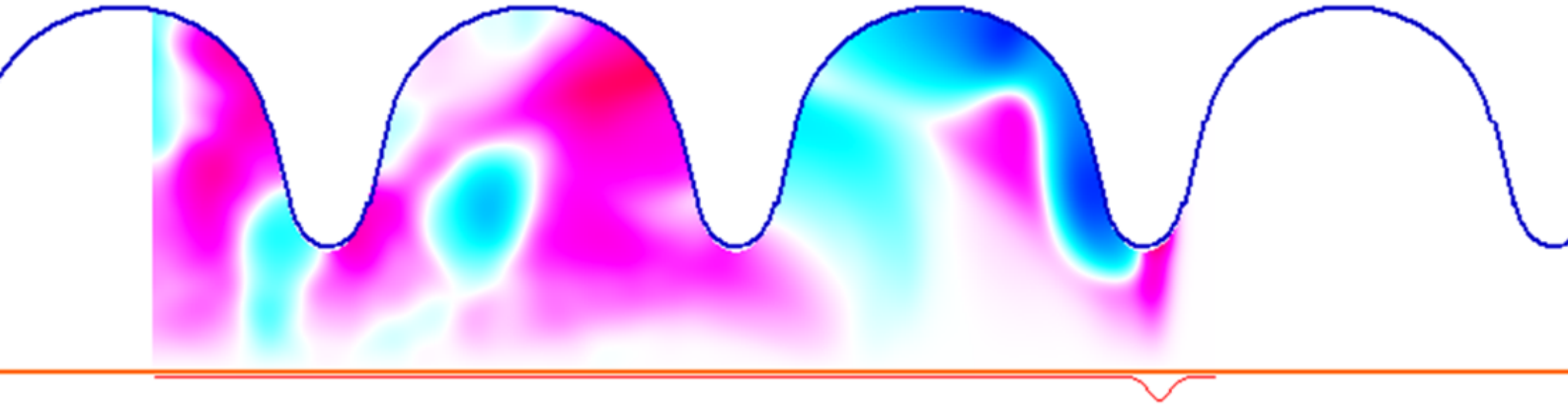
User's Manual

Igor Zagorodnov

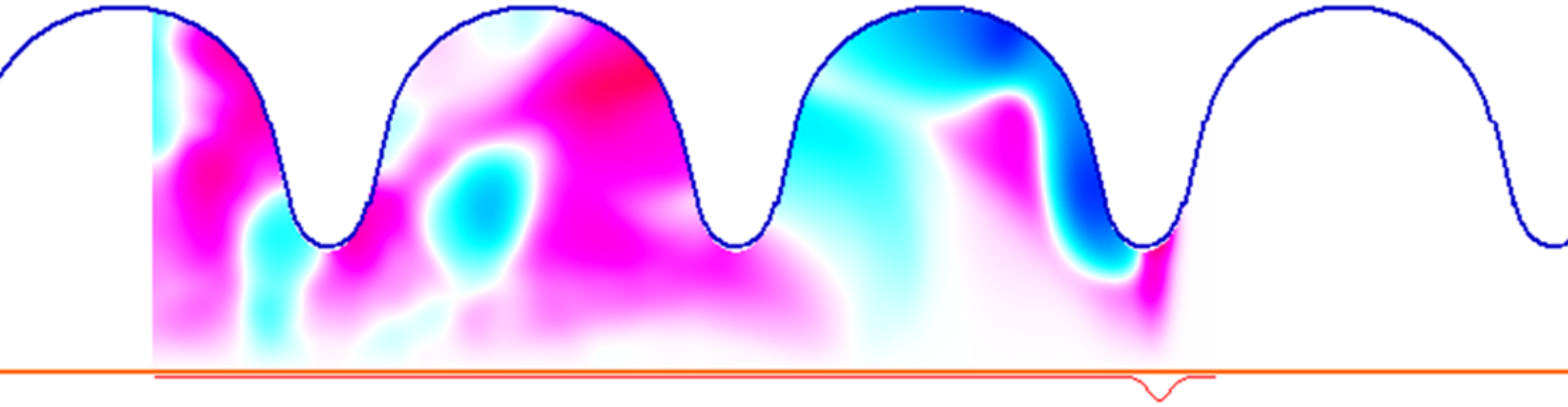
PUBLISHED BY IGOR ZAGORODNOV

WWW.ECHO4D.DE

Printing, 3 April 2020



Contents



1. Introduction

Program ECHO calculates electromagnetic fields of charged bunches in accelerators.

The package consists of three archives:

- **ECHO1D.zip**
- **ECHO2D.zip**
- **ECHO3D.zip**

Archive **ECHO1D.zip** contains program ECHO1D which calculates impedances and wakes of rotationally symmetric and rectangular waveguides.

Archive **ECHO2D.zip** includes three different programs: ECHOz1, ECHOz2 and ECHO2D for rotationally symmetric and rectangular geometries.

Archive **ECHO3D.zip** contains program ECHO3D for arbitrary three dimensional structures.

R Under rectangular geometries we mean structures having rectangular cross-section, where the height can vary as function of longitudinal coordinate but the width and side walls remain fixed.

In order to start I would advice to use code ECHOz1 or code ECHOz2 as they have GUI and only few simulation parameters.

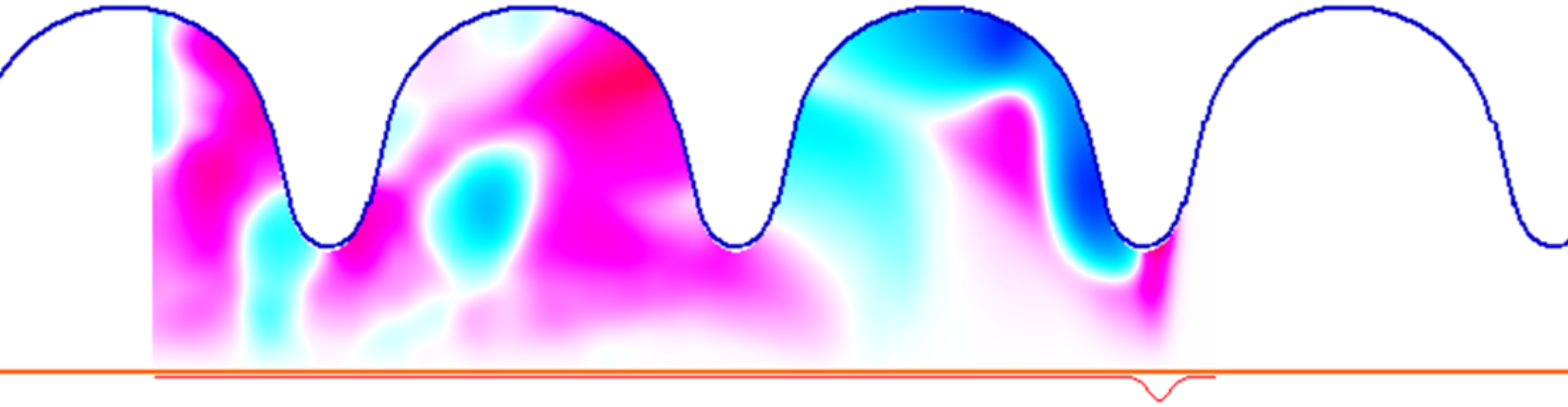
The codes use conformal finite-difference method. In the time-domain wake filed calculations the mesh resolution with 5 mesh points on the rms bunch width should be enough to obtain accurate results. To check the accuracy I would advice to do 2 tests:

- for the coarsest mesh resolution to change bunch offset (with indirect integration algorithm switched on);
- double the mesh resolution.

If the first test fails it means that there is a meshing error. In this case change a little the mesh steps to try to avoid it and contact me for fixing a possible bug in the code. If the first test is OK then check the convergence and the accuracy with the second test.

ECHOz2 and ECHO2D include a model of conductive walls. The meshes in the vacuum and in the metal should agree as follows. As a default mesh use 5 mesh lines on sigma in vacuum and “NStepsInConductive=10” in metal. It means that the skin depth in the metal will be meshed

with 10 mesh lines. If you increase the mesh density in vacuum by factor 2 (10 mesh lines on sigma) you should simultaneously increase by the same factor “NStepsInConductive” in metal: “NStepsInConductive=20”. It means that the calculation depth in the metal remains the same. For 20 mesh lines on sigma use”NStepsInConductive=40” and so on. In this case the calculation domain remains the same and the wake converges.



2. ECHOz1: Full Rotational Symmetry

2.1 Introduction

Code ECHOz1 calculates in time domain the electromagnetic fields generated by an electron bunch passing through rotationally symmetric perfectly conducting structure on axis of symmetry [8].

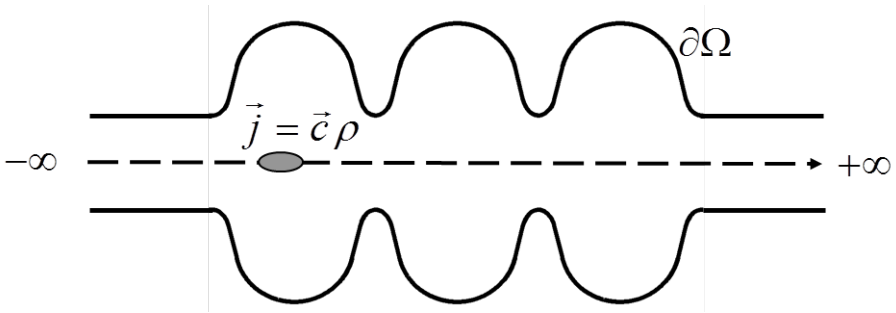


Figure 2.1: The beam moves on axis in rotationally symmetric stricture.

We consider a charged bunch moving with light velocity c through a rotationally symmetric structure as shown in Fig. 2.1. The bunch has Gaussian longitudinal charge density $\lambda(s)$ with rms length σ_z. It moves on the symmetry axis and the whole problem is rotationally symmetric.

The charge density in time domain can be written as

$$\rho(r,\varphi,z,t) = Q\frac{\delta(r-r_0)}{2\pi r_0}\lambda(z-ct), \quad \lambda(s) = \frac{1}{\sqrt{2\pi}\sigma_z}e^{\frac{s^2}{2\sigma_z^2}}, \tag{2.1}$$

where Q is the bunch charge, r_0 is the "hollow" bunch radius, c is velocity of light in vacuum, and $\delta(\cdot)$ means the Dirac delta function.

We are interested in longitudinal wake potential as defined in [1]. For fully rotationally symetric problem it can be written as

$$W_{\parallel}(r_0,r,\varphi,s) = W_0(s), \tag{2.2}$$

where s is the position of the wittness particle in the bunch.

2.2 Installation

The program ECHOz1 is compiled for Windows. It can be downloaded as archive **ECHO2D.zip** from `https://www.echo4d.de`. Extract the archive keeping the stricture of folders and files.

The archive contains the following folders.

1. **Docs**. It contains this manual.
2. **Codes**. It contains the executable **ECHOz1.exe**.
3. **Examples**. It contains several examples.
4. **MatLib4ECHO**. It contains Matlab functions for postprocessing.
5. **PostProcessor2D**. It contains Matlab scripts for postprocessing.
6. **System**. It contains two files which are required for parallel execution. If **ECHOz1** do not start or start with error, install **vc_redist.x64.exe** on your computer. It puts file **vcomp140.dll** in Windows system directory. Alternatively you can put only the file **vcomp140.dll** to the directory **ECHOz1**.

2.3 Input files

The program ECHOz1 does not require any input files. A geometry and a setup can be done in the program itself and saved in a binary file with extension "*.e2d".

Alternatively two files can be used as input:

- a file with geometry description in ASCII format; it can have an arbitrary name and it will be imported in the programm through GUI menu,
- a file with parameters of the simulation and the geometry in binary format with extension "*.e2d" created early with ECHOz1.

2.3.1 Geometry description

The geometry can be imported as a file in ASCII format with extension "*.txt".

The geometry file has the following format.

N

$z_{1,1}$ $r_{1,1}$ $z_{2,1}$ $r_{2,1}$ $z_{3,1}$ $r_{3,1}$ $z_{4,1}$ $r_{4,1}$ d_1

...

$z_{1,N}$ $r_{1,N}$ $z_{2,N}$ $r_{2,N}$ $z_{3,N}$ $r_{3,N}$ $z_{4,N}$ $r_{4,N}$ d_N

The parameters in the geometry file are:

- N - total number of segments (lines or elliptical arcs).
- $z_{1,i}$, $r_{1,i}$ - coordinates in cm of start point for segment number i.
- $z_{2,i}$, $r_{2,i}$ - coordinates in cm of end point for segment number i.
- $z_{3,i}$, $r_{3,i}$, $z_{4,i}$, $r_{4,i}$ - - coordinates in cm of square in which the ellipse is inscribed (for lines these parameters should be zeros).
- $z_{3,i}$, $r_{3,i}$ - coordinates in cm of top left corner.
- $z_{4,i}$, $r_{4,i}$ - coordinates in cm of bottom right corner.
- d_i - orientation (0-clock, 1-anticlock).

As example let us consider the geometry shown in Fig. 2.2. The corresponding file will have the following content

```
3
z0 r1 z1 r1 0 0 0 0 0
z1 r1 z2 r2 z3 r3 z4 r4 0
z2 r2 z5 r2 0 0 0 0 0.
```

In oder to export the geometry in ECHOz1 go to GUI menu "Geometry/Import". Alternatively it is possible to create a geometry in ECHOz1 GUI. Use for it menu "Geometry/Edit" and the button

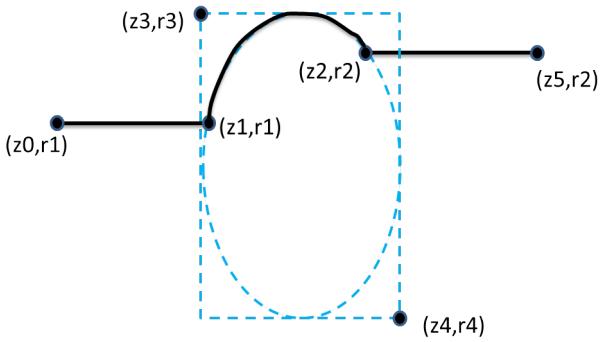


Figure 2.2: Example for geometry file format.

"Add" in the dialog box. The format is the same as described above. After the geometry creation save it with help of menu "File/Save As" in file with extension "*.e2d".

2.3.2 Parameters of simulation

The parameters of simulation can be set only through the GUI. The setup of the simulation can be done only after the geometry description is created or imported in the program.

In order to set the Gaussian bunch length σ go to menu "Bunch" and set the value in cm in the box "Sigma". It is only the parameter in the dialog "Bunch".

The mesh can be set through menu "Mesh". In order to use 5 mesh points on sigma press the button "Default". If you are going to use different mesh steps then put the new values in the boxes "Z step" and "R step" in cm and press the button "Apply".

After the setup of parameters is finished save them in "*.e2d" file: go to menu "File/Save" or use the "Save" symbol in the tool-bar.

2.4 Wakefield Calculation

After creation of the mesh, setup of the bunch length and setup of the mesh steps you can go to menu "Solver". It opens the dialog box shown in Fig. 2.3.

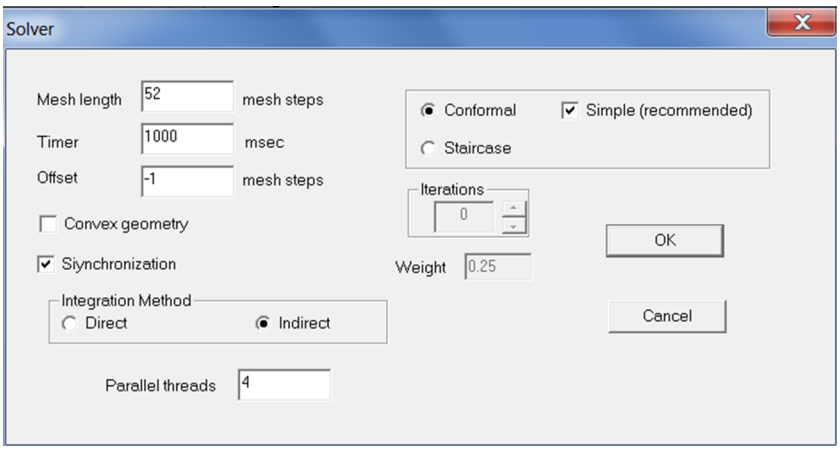


Figure 2.3: Parameters of solver.

The parameters in the dialog are:

- **Mesh length** - length og the calculation window moving with the bunch. It is given as number of steps. The length of window in cm can be found by multiplication of this number with value of "Step Z" from "Mesh" dialog.
- **Timer** - this parameter defines the time interval of update of the field picture on the display during the calculation. The program shows the scaterred field potential rA_φ.
- **Offset** - defines value of r_0 in mesh lines. It can be found as r_0 = ("Offset"+0.5)*"Step R". The value "-1" mens that we use r_0 as large as possible. The last choice provides the best accuracy.
- **Convex geometry** - check ON this check-box to accelerate the calculation for "convex" geometry. "Convex" means here a geometry that has only one connected vacuum region in each plane transverse to the symmetry axis.
- **Syncronization** - check OFF this check-box to accelerate the calculation if you are not interested in synchronization of field map with the geometry. It has impact only on the display picture during the calculation.
- **Integration Method** - use "Indirect" choice if you do not really know what "Direct" means.
- **Conformal** - use this choice together with "Simple" check-box. Other choices can be used only if you have problems with this one.
- **Parallel threads** - set up how many threads will be used. Usually it should be equal to the number of cores in your computer, but check the efficiency of parallelism experimentally.

Press "OK" button to start the calculation. After the box "Ready!" finish the calculation with menu "Stop". Press the green button "W" to see the wake and the loss factor. After the calculation is finished or interrupted with menu "Stop", save the parameters in "*.e2d" file with menu command "File/Save". It will save the parameters of the solver as well.

2.5 Output files

After execution of **ECHOz1.exe** the folder will contain two files:

- **wake.dat** - with longitudinal wake. It has two columns. In the first column is s-coordinate in cm, in the second column is function W(s) in V/pC.
- **bunch.dat** - with bunch charge profile. It has two columns. In the first column is s-coordinate in cm, in the second column is current profile in arbitrary units.

2.6 Postprocessing

Use matlab script **PP_ECHOz1** from directory **PostProcessor2D/ Wakes/ Round**. It plots the wake and calculates the loss factor and the rms spread of the wake.

2.7 Examples

In this section we consider several examples included in the archive at the directory **Examples**.

2.7.1 Example 1: Round collimator

The example of round collimator can be found in directory **Examples/ N1_RoundCollimatorLong**.

In order to make the simulation proceed as follows:

- Go to directory **Codes** and start **ECHOz1.exe**.
- Go to menu "File/Open" and navigate to directory **Examples/ N1_RoundCollimatorLong/ ECHOz1**. Open the input file **N1.e2d**. You should see the geometry shown in Fig. 2.4.
- Go to menu "Bunch" and press "OK".
- Go to menu "Mesh" and press "Close".

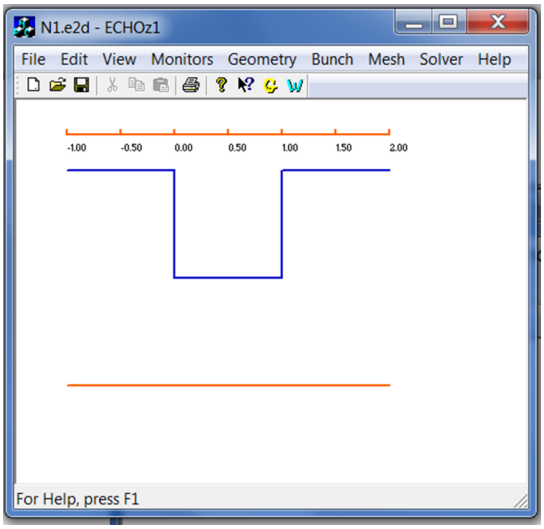


Figure 2.4: Geometry of round collimator.

- Go to menu "Sover" and press "OK". The calculation starts.
- Wait until message "Ready" appears and press "OK". The calculations is done.
- Go to menu "Stop".
- Press button with green "W" in the panel under main menu. You will see the wake and the loss factor.
- Press button with yellow "G" to return to the geometry.
- Close the program.

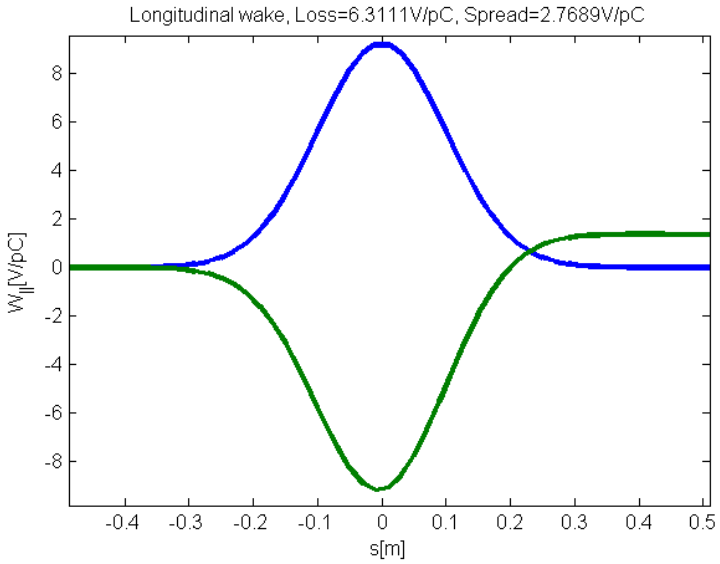


Figure 2.5: Longitudinal wake of round collimator (in green).

Now the wake is saved in file **wake.dat** in directory **Examples/ N1_RoundCollimatorLong/ ECHOz1**. You can use the matlab script **PostProcessor2D/ Round/ PP_ECHOz1.m** to see the wake shown in Fig. 2.5.

2.7.2 Example 2: TESLA cavity

The example of TESLA cavity can be found in directory **Examples/ N10_TESLACavityLong**.

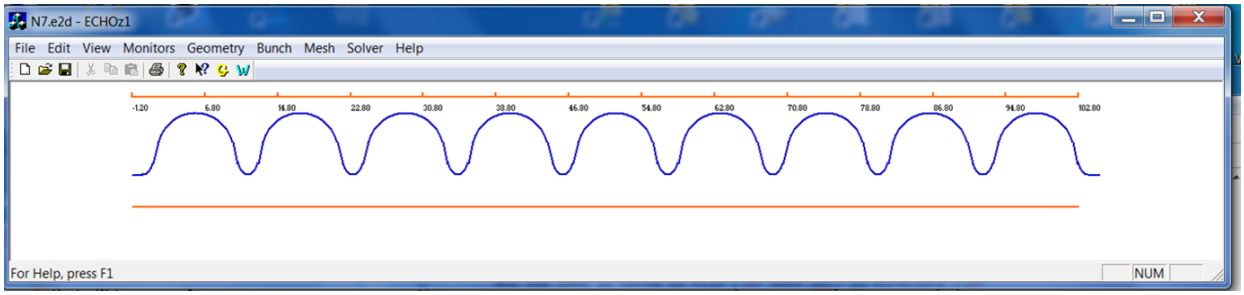


Figure 2.6: Geometry of TESLA cavity.

In order to make the simulation proceed as follows:

- Go to directory **Codes** and start **ECHOz1.exe**.
- Go to menu "File/Open" and navigate to directory **Examples/ N10_TESLACavityLong/ECHOz1**. Open the input file **N10.e2d**. You should see the geometry shown in Fig. 2.6.
- Go to menu "Bunch" and press "OK".
- Go to menu "Mesh" and press "Close".
- Go to menu "Sover" and press "OK". The calculation starts.
- Wait until message "Ready" appears and press "OK". The calculations is done.
- Go to menu "Stop".
- Press button with green "W" in the panel under main menu. You will see the wake and the loss factor.
- Press button with yellow "G" to return to the geometry.
- Close the program.

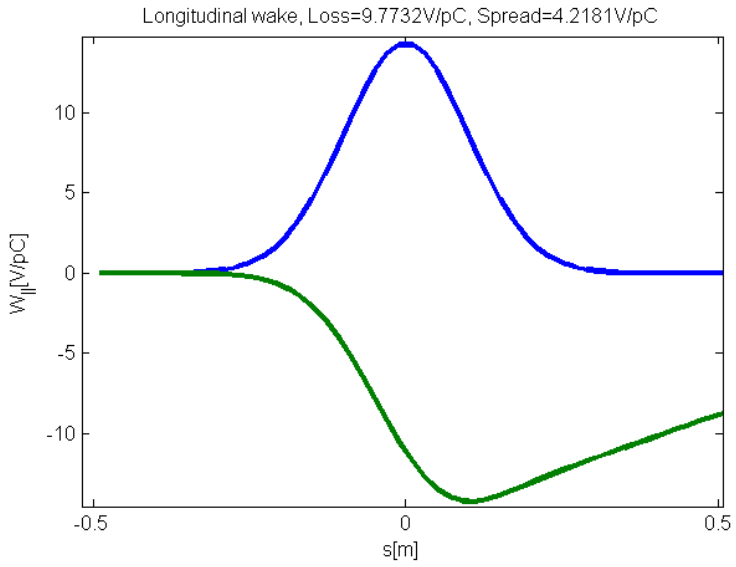
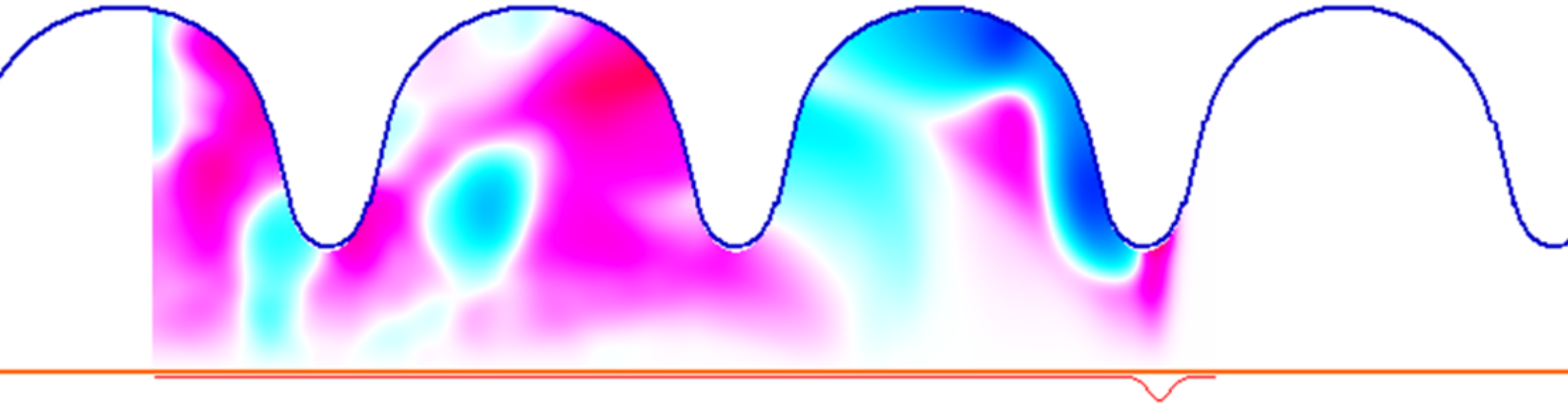


Figure 2.7: Longitudinal wake of TESLA cavity (in green).

Now the wake is saved in file **wake.dat** in directory **/Examples/ N10_TESLACavityLong/ ECHOz1**. You can use the matlab script **PostProcessor2D/ Round/ PP_ECHOz1.m** to see the wake shown in Fig. 2.5.



3. ECHOz2: Rotationally Symmetric Geometry

3.1 Introduction

Code ECHOz2 calculates in time domain the electromagnetic fields generated by an electron bunch passing through rotationally symmetric conducting structure off axis [6, 9]. The structure can have only metal conductive walls with finite or infinite conductivity.

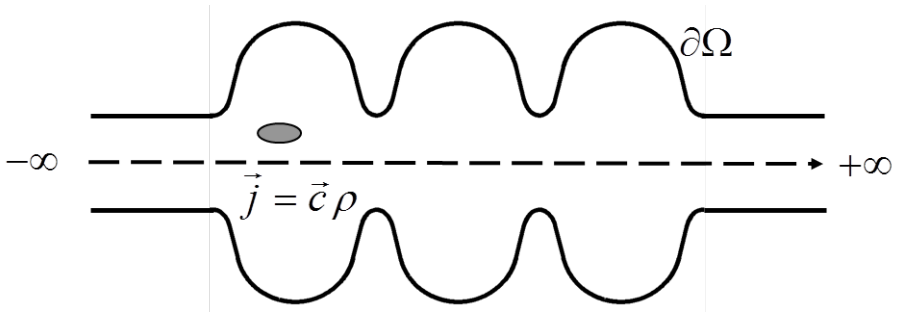


Figure 3.1: The beam moves off axis in rotationally symmetric stricture.

We consider a charged bunch moving with light velocity c through a rotationally symmetric structure as shown in Fig. 3.1. The bunch has Gaussian longitudinal charge density $\lambda(s)$ with rms width σ. It moves off axis and the whole problem is not rotationally symmetric but can be expanded in infinite number of independent problems for Fourier azimuthal harmonics.

The charge density in time domain can be written as

$$\rho(r_0,\varphi_0,r,\varphi,z,t) = \sum_{m=0}^{\infty} \rho_m(r_0,r,z,t)\cos(m(\varphi-\varphi_0)), \tag{3.1}$$

$$\rho_m(r_0,r,z,t) = Q\frac{\delta(r-r_0)}{\pi r_0(1+\delta_{m0})}\lambda(z-ct), \quad \lambda(s) = \frac{1}{\sqrt{2\pi}\sigma}e^{\frac{s^2}{2\sigma^2}}, \tag{3.2}$$

where Q is the bunch charge, r_0, φ_0 is the "pincel" bunch offset coordinates, c is velocity of light in vacuum, and $\delta(\cdot)$ means the Dirac delta function, and $\delta_{m0} = 1$ if $m = 1, 0$ otherwise.

The longitudinal wake potential can be represented through one dimensional functions, with only one function for each azimuthal mode number m,

$$W_{\parallel}(r_0,\varphi_0,r,\varphi,s)=\sum_{m=0}^{\infty}W_m(s)r_0^m r^m\cos(m(\varphi-\varphi_0)). \tag{3.3}$$

3.2 Installation

The program ECHOz2 is compiled for Windows. It can be downloaded as archive **ECHO2D.zip** from `https://www.echo4d.de`. Extract the archive keeping the stricture of folders and files.

The archive contains the following folders.

1. **Docs**. It contains this manual.
2. **Codes**. It contains the executable **ECHOz2.exe**.
3. **Examples**. It contains several examples.
4. **MatLib4ECHO**. It contains Matlab functions for postprocessing.
5. **PostProcessor2D**. It contains Matlab scripts for postprocessing.
6. **System**. It contains two files which are required for parallel execution. If **ECHOz2** do not start or start with error, install **vc_redist.x64.exe** on your computer. It puts file **vcomp140.dll** in Windows system directory. Alternatively you can put only the file **vcomp140.dll** to the directory **ECHOz2**.

3.3 Input files

The program ECHOz2 does not require any input files. A geometry and a setup can be done in the program itself and saved in a binary file with extension "*.e2dx".

Alternatively two files can be used as input:

- a file with geometry description in ASCII format; it can have an arbitrary name and it will be imported in the programm through GUI menu,
- a file with parameters of the simulation and the geometry in binary format with extension "*.e2dx" created earlier with ECHOz2.

3.3.1 Geometry description

The geometry can be imported as a file in ASCII format with extension "*.txt".

The geometry file has the following format.

N

$z_{1,1}$ $r_{1,1}$ $z_{2,1}$ $r_{2,1}$ $z_{3,1}$ $r_{3,1}$ $z_{4,1}$ $r_{4,1}$ d_1 k_1

…

$z_{1,N}$ $r_{1,N}$ $z_{2,N}$ $r_{2,N}$ $z_{3,N}$ $r_{3,N}$ $z_{4,N}$ $r_{4,N}$ d_N k_N

The parameters in the geometry file are:

- N - total number of segments (lines or elliptical arcs).
- $z_{1,i}$, $r_{1,i}$ - coordinates in cm of start point for segment number i.
- $z_{2,i}$, $r_{2,i}$ - coordinates in cm of end point for segment number i.
- $z_{3,i}$, $r_{3,i}$, $z_{4,i}$, $r_{4,i}$ - - coordinates in cm of square in which the ellipse is inscribed (for lines these parameters should be zeros).
- $z_{3,i}$, $r_{3,i}$ - coordinates in cm of top left corner.
- $z_{4,i}$, $r_{4,i}$ - coordinates in cm of bottom right corner.
- d_i - orientation (0-clock, 1-anticlock).
- k_i - conductivity in S/m.

As example let us consider the geometry shown in Fig. 2.2. The corresponding file will have the following content

```
3
z0 r1 z1 r1 0 0 0 0 0 k1
z1 r1 z2 r2 z3 r3 z4 r4 0 k2
z2 r2 z5 r2 0 0 0 0 0 k3,
```

where k1, k2, k3 are conductivities of the segments.

In oder to export the geometry in ECHOz2 go to GUI menu "Geometry/Import". Alternatively it is possible to create a geometry in ECHOz2 GUI. Use for it menu "Geometry/Edit" and the button "Add" in the dialog box. The format is the same as described above. After the geometry creation save it with help of menu "File//Save As" in file with extension "*.e2dx".

3.3.2 Parameters of simulation

The parameters of simulation can be set only through the GUI. The setup of the simulation can be done only after the geometry description is created or imported in the program.

In order to set the Gaussian bunch length σ go to menu "Bunch" and set the value in cm in the box "Sigma". It is only the parameter in the dialog "Bunch".

The mesh can be set through menu "Mesh". In order to use 5 mesh points on sigma press the button "Default". If you are going to use different mesh steps then put then in the boxes "Z step" and "R step" in cm and press the button "Apply".

The mesh dialog has box "Lossy metal 1D mesh length". It should be set to "0" if the structure is perfectly electric conductive. Otherwise it defines an one dimensional mesh length for tangential components of electromagnetic field in the conductive parts [6]. The default value is 10. Increase this value to obtain a better accuracy.

After the setup of parameters save them in "*.e2dx" file. For it go to menu "File/Save" or use the "Save" symbol in the panel under menu.

3.4 Wakefield Calculation

After creation of the mesh, setup of the bunch length and setup of the mesh steps you can go to menu "Solver". It opens the dialog box shown in Fig. 3.2.

The parameters in the dialog are:

- **Mode #** - mode number m in the azimuthal expansion.
- **Mesh length** - length og the calculation window moving with the bunch. It is given as number of steps. The length of window in cm can be found by multiplication of this number with value of "Step Z" from "Mesh" dialog.
- **Update on the screen** - this parameter defines the time interval of update of the field picture on the display during the calculation. The program shows the electric field component E_z.
- **Bunch offset** - defines value of r_0 in mesh lines. It can be found as r_0 = ("Offset"+0.5)*"Step R". The value "-1" mens that we use r_0 as large as possible. The last choice provides the best accuracy.
- **Convex geometry** - check ON this check-box to accelerate the calculation for "convex" geometry. "Convex" means here a geometry that has only one connected vacuum region in each plane transverse to the symmetry axis.
- **Syncronization** - check OFF this check-box to accelerate the calculation if you are not interested in synchronization of field map with the geometry. It has impact only on the display picture during the calculation.
- **Integration Method** - use "Indirect" choice if you do not really know what "Direct" means.

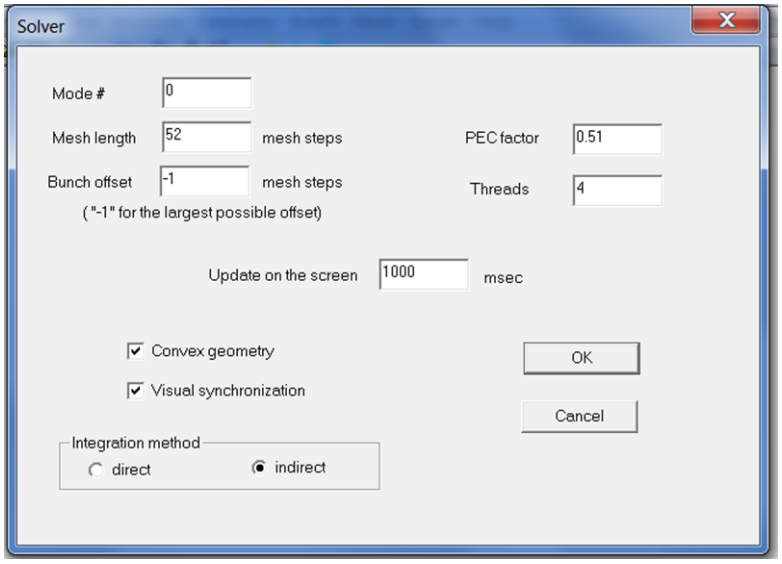


Figure 3.2: Parameters of solver ECHOz2.

- **PEC factor** - it defines which cells near the boundary will be treated with extended stencil [7]. The value shoul be between 0.5 and 1.0. The lower value gives a better accuracy, the higher value provides a better stability.
- **Parallel threads** - set up how many threads will be used. Usually it should be equal to the number of cores in your computer, but check the efficiency of parallelism experimentally.

Press "OK" button to start the calculation. After the box "Ready!" finish the calculation with menu "Stop". Press the green button "L" to see the longitudinal wake and the loss factor. Press the green button "T" to see the transverse wake and the kick factor. After the calculation is finished or interrupted with menu "Stop", save the parameters in "*.e2d" file with menu command "File/Save". It will save the parameters of the solver as well.

3.5 Output files

After execution of **ECHOz2.exe** the folder will contain three files:

- **wakeL.dat** - with longitudinal wake for mode m. It has two columns. In the first column is s-coordinate in cm, in the second column is function $W_m(s)$ in V/pC/m^{2m}.
- **wakeT.dat** - with transverse wake for mode m. It has two columns. In the first column is s-coordinate in cm, in the second column is function $\int_{-\infty}^{s} W_m(s)ds$ in V/pC/m^{2m-1}.
- **bunch.dat** - with bunch charge profile. It has two columns. In the first column is s-coordinate in cm, in the second column is current profile in arbitrary units.

3.6 Postprocessing

Use matlab script **PP_ECHOz2** from directory **PostProcessor2D/ Wakes/ Round**. It plots the wake and calculates the loss factor and the rms spread of the wake.

3.7 Examples

In this section we consider several examples included in the archive at the directory **Examples**.

3.7.1 Example 1: Round collimator

The examples of round collimator can be found in directories **Examples/ N1_RoundCollimatorLong**, **Examples/ N2_RoundCollimatorDipole**, **Examples/ N3_RoundCollimatorDipoleConductive**.

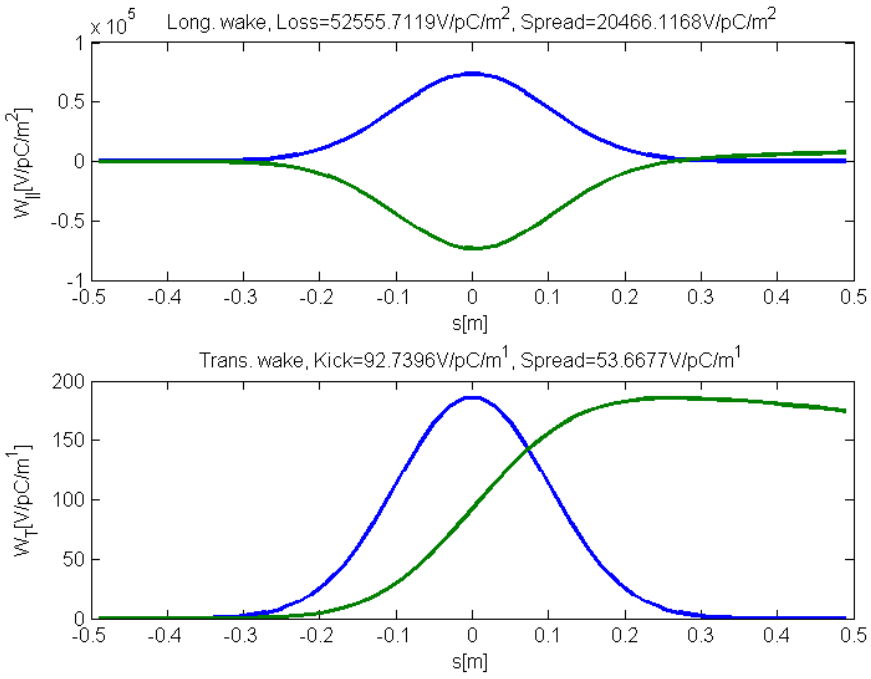


Figure 3.3: Longitudinal and transverse wakes of round collimator (dipole mode).

In order to calculate the longitudinal wake of monopole mode ($m = 0$) proceed as follows:

- Go to directory **Codes** and start **ECHOz2.exe**.
- Go to menu "File/Open" and navigate to directory **Examples/ N1_RoundCollimatorLong/ ECHOz2**. Open the input file **N1.e2dx**. You should see the geometry shown in Fig. 2.4.
- Go to menu "Bunch" and press "OK".
- Go to menu "Mesh" and press "Close".
- Go to menu "Sover" and press "OK". The calculation starts.
- Wait until message "Ready" appears and press "OK". The calculations is done.
- Go to menu "Stop".
- Press button with green "L" in the panel under main menu. You will see the wake and the loss factor.
- Press button with yellow "G" to return to the geometry.
- Close the program.

Now the wake is saved in file **wakeL.dat** in directory **Examples/ N1_RoundCollimatorLong/ ECHOz2**. You can use the matlab script **PostProcessor2D/ Round/ PP_ECHOz2.m** to see the wake shown in Fig. 2.5. The transverse wake is zero for the monopole mode.

In order to calculate the transverse wake of dipole mode ($m = 1$) proceed as follows:

- Go to directory **Codes** and start **ECHOz2.exe**.
- Go to menu "File/Open" and navigate to directory **Examples/ N2_RoundCollimatorDipole/ ECHOz2**. Open the input file **N2.e2dx**. You should see the geometry shown in Fig. 2.4.
- Go to menu "Bunch" and press "OK".
- Go to menu "Mesh" and press "Close".
- Go to menu "Sover" and press "OK". The calculation starts.

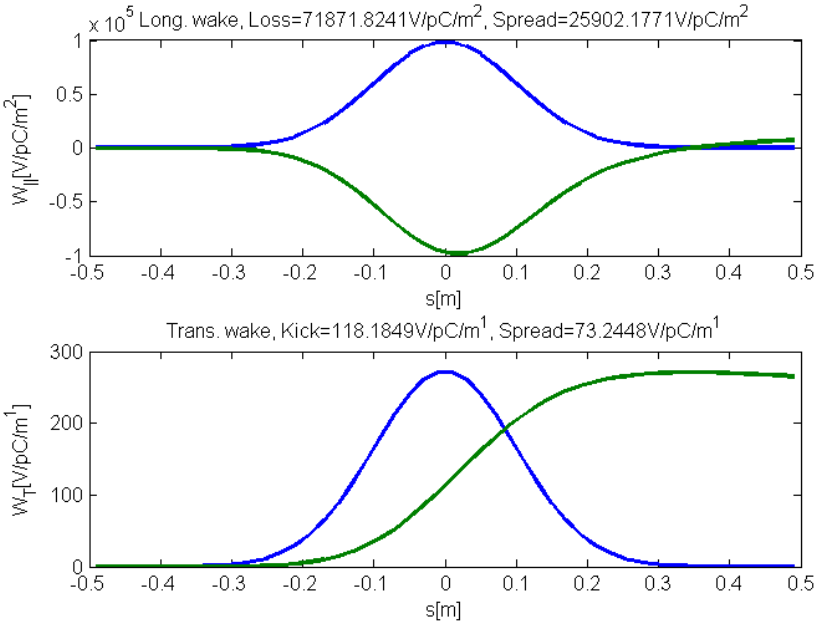


Figure 3.4: Longitudinal and transverse wakes of round conductive collimator (dipole mode).

- Wait until message "Ready" appears and press "OK". The calculations is done.
- Go to menu "Stop".
- Press button with green "T" in the panel under main menu. You will see the transverse wake and the kick factor.
- Press button with yellow "G" to return to the geometry.
- Close the program.

Now the wake is saved in file **wakeT.dat** in directory **/Examples/ N2_RoundCollimatorDipole/ ECHOz2**. You can use the matlab script **PostProcessor2D/ Round/ PP_ECHOz2.m** to see the wake shown in Fig. 3.3.

The last example is the same collimator but with conductive small pipe. The conductovity is equat ti 1 S/m. It can be seen and changed through menu "Geometry/Edit". In order to calculate the transverse wake of dipole mode ($m = 1$) for the conductive collimator proceed as follows:

- Go to directory **Codes** and start **ECHOz2.exe**.
- Go to menu "File/Open" and navigate to directory **Examples/ N3_RoundCollimatorDipoleConductive/ ECHOz2**. Open the input file **N3.e2dx**. You should see the geometry shown in Fig. 2.4.
- Go to menu "Bunch" and press "OK".
- Go to menu "Mesh" and press "Close".
- Go to menu "Sover" and press "OK". The calculation starts.
- Wait until message "Ready" appears and press "OK". The calculations is done.
- Go to menu "Stop".
- Press button with green "T" in the panel under main menu. You will see the transverse wake and the kick factor.
- Press button with yellow "G" to return to the geometry.
- Close the program.

Now the wake is saved in file **wakeT.dat** in directory **Examples/ N3_RoundCollimatorDipoleConductive/ ECHOz2**. You can use the matlab script **PostProcessor2D/ Round/ PP_ECHOz2.m** to see the wake shown in Fig. 3.4.

3.7.2 Example 2: Resistive pillbox cavity

The example of pillbox cavity can be found in directory **Examples/N9_ResistivePillbox**. The cavity walls have conductivity equal to 1000 S/m.

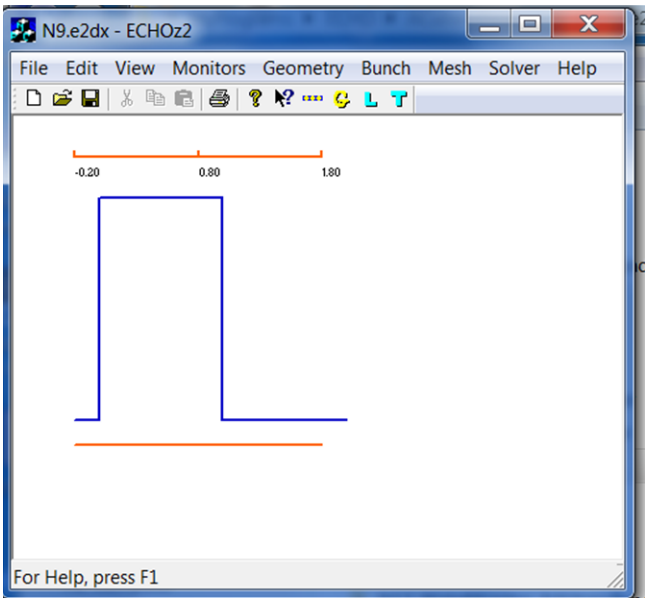


Figure 3.5: Pillbox cavity geometry.

In order to calculate the transverse wake of dipole mode ($m = 1$) proceed as follows:

- Go to directory **Codes** and start **ECHOz2.exe**.
- Go to menu "File/Open" and navigate to directory **Examples/ N9_ResistivePillbox/ ECHOz2**. Open the input file **N9.e2dx**. You should see the geometry shown in Fig. 3.5.
- Go to menu "Bunch" and press "OK".
- Go to menu "Mesh" and press "Close".
- Go to menu "Sover" and press "OK". The calculation starts.
- Wait until message "Ready" appears and press "OK". The calculations is done.
- Go to menu "Stop".
- Press button with green "L" in the panel under main menu. You will see the wake and the loss factor.
- Press button with yellow "G" to return to the geometry.
- Close the program.

Now the wake is saved in file **wakeL.dat** in directory **Examples/ N9_ResistivePillbox/ ECHOz2**. You can use the matlab script **PostProcessor2D/ Round/ PP_ECHOz2.m** to see the wake shown in Fig. 3.6. The transverse wake is zero for the monopole mode.

In order to calculate the monopole or the higher order modes change only "Mode #" value in solver box in the route described above.

3.7.3 Example 3: TESLA cavity

The example of TESLA cavity can be found in directory **Examples/ N10_TESLACavityLong**.

In order to make the simulation proceed as follows:

- Go to directory **Codes** and start **ECHOz2.exe**.
- Go to menu "File/Open" and navigate to directory **Examples/ N10_TESLACavityLong/ ECHOz2**. Open the input file **N10.e2dx**. You should see the geometry shown in Fig. 2.6.
- Go to menu "Bunch" and press "OK".

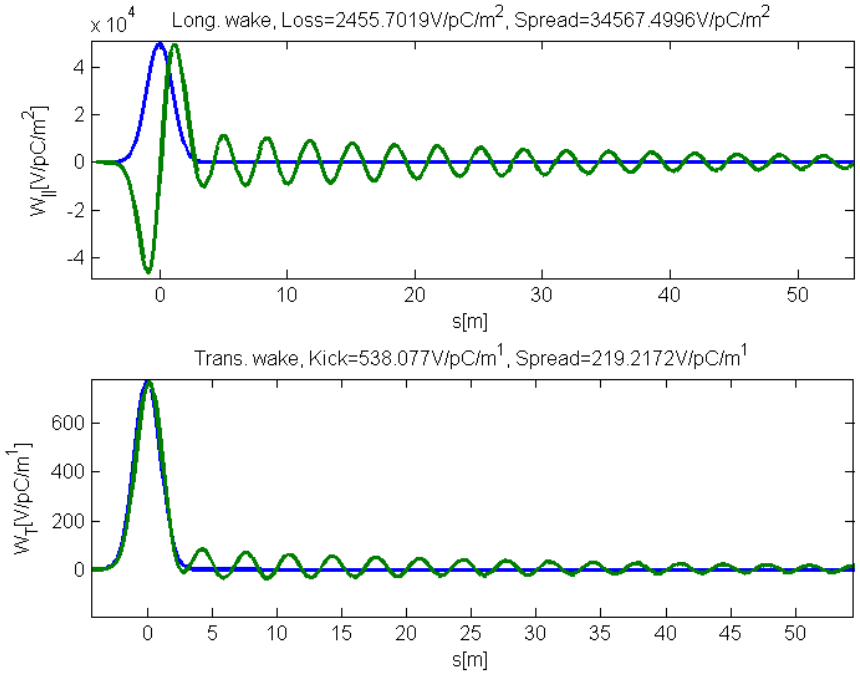
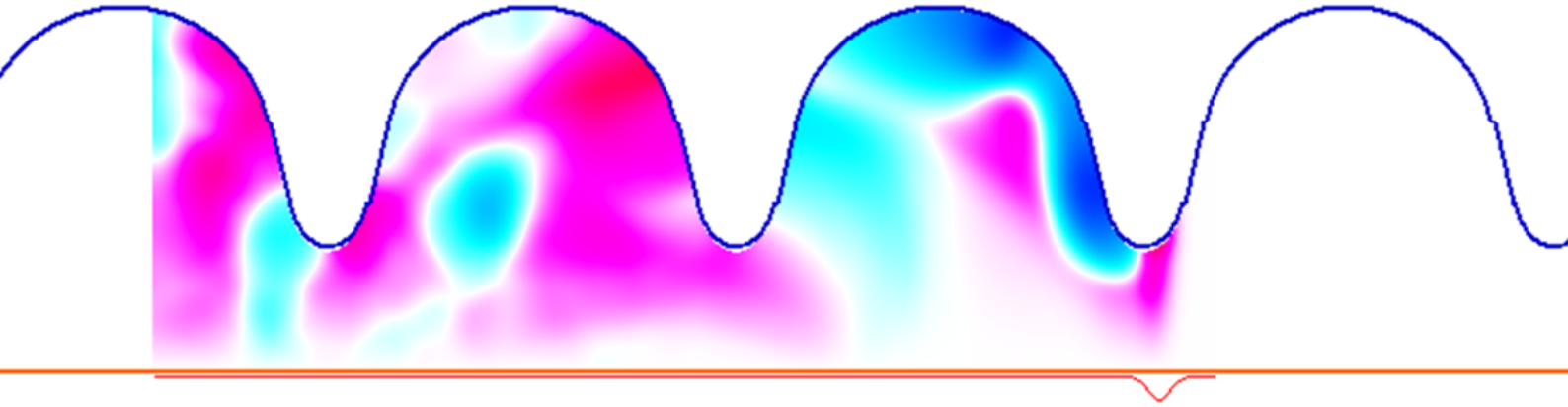


Figure 3.6: Dipole wakes of resistive pillbox.

- Go to menu "Mesh" and press "Close".
- Go to menu "Sover" and press "OK". The calculation starts.
- Wait until message "Ready" appears and press "OK". The calculations is done.
- Go to menu "Stop".
- Press button with green "L" in the panel under main menu. You will see the wake and the loss factor.
- Press button with yellow "G" to return to the geometry.
- Close the program.

Now the wake is saved in file **wake.dat** in directory **Examples/ N10_TESLACavityLong/ ECHOz2**. You can use the matlab script **PostProcessor2D/ Round/ PP_ECHOz2.m** to see the wake shown in Fig. 2.5.

In order to calculate the dipole or the higher order modes change only "Mode #" value in solver box in the route described above.



4. ECHO2D: Rectangular and Round Geometries

4.1 Introduction

Code ECHO2D calculates in time domain the electromagnetic fields generated by an electron bunch passing through rotationally symmetric or rectangular structures [6, 9]. The structure can consist of several materials with different permeabilities, permitivities and conductivities. The wall conductivity model for metals is available as well. This code has all possibilities of ECHOz1 and ECHOz2. Additionally it is able to calculate wakefields in rectangular structures.The bunch form can be arbitrary and the bunch can have finite energy. At the current version there is possibility to do particle tracking for fully rotationally symmetric case.

Let us consider a line-charge beam with vanishing transverse dimensions,

$$\begin{aligned} \rho(x_0,y_0,x,y,s) &= Q\delta(x-x_0)\delta(y-y_0)\lambda(s), \\ j_z(x_0,y_0,x,y,s) &= c\rho(x_0,y_0,x,y,s), \end{aligned} \tag{4.1}$$

where x_0, y_0, define the transverse offset of the beam, $s = z - ct$ is the local longitudinal coordinate in the bunch, Q is the bunch charge and $\lambda(s)$ is the longitudinal bunch profile [for a point charge, $\lambda(s) = \delta(s)$]. The longitudinal wake potential $W_\parallel$ at point (x,y,s) is defined as [1]

$$W_\parallel(x_0,y_0,x,y,s) = Q^{-1}\int_{-\infty}^{\infty}[E_z(x,y,z,t)]_{t=(z-s)/c}dz, \tag{4.2}$$

where the electric field on the right-hand side is the solution to Maxwell's equation with the sources of Eqs. (4.1) (this field, of course, is also a function of x_0 and y_0 omitted in the arguments of E_z for brevity).

The charge distribution can be Fourier expanded as

$$\begin{aligned} \rho(x_0,y_0,x,y,s) &= \frac{1}{w}\sum_{m=1}^{\infty}\rho_m(y_0,y,s)\sin(k_{x,m}x_0)\sin(k_{x,m}x), \\ \rho_m(y_0,y,s) &= Q\delta(y-y_0)\lambda(s). \end{aligned} \tag{4.3}$$

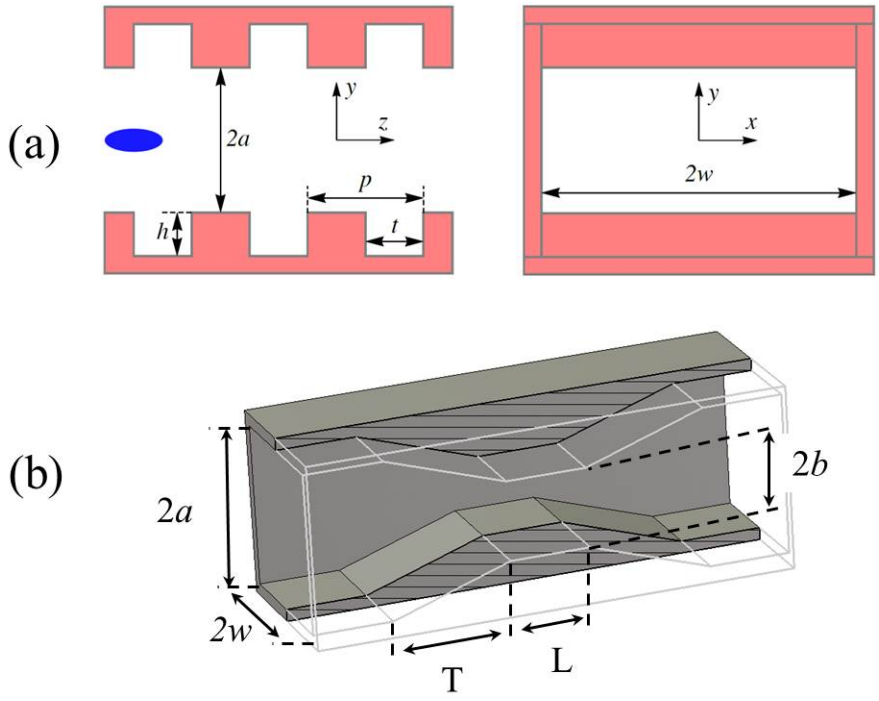


Figure 4.1: Structures of rectangular geometry: (a) dechirper, (b) tapered collimator.

The longitudinal wake potential can be written as:

$$W_{\parallel}(x_0,y_0,x,y,s) = \frac{1}{w}\sum_{m=1}^{\infty} W_m(y_0,y,s)\sin(k_{x,m}x_0)\sin(k_{x,m}x), \tag{4.4}$$

where

$$\begin{aligned} W_m(y_0,y,s) = &[W_m^{cc}(s)\cosh(k_{x,m}y_0) + W_m^{sc}(s)\sinh(k_{x,m}y_0)]\cosh(k_{x,m}y) \\ &+ [W_m^{cs}(s)\cosh(k_{x,m}y_0) + W_m^{ss}(s)\sinh(k_{x,m}y_0)]\sinh(k_{x,m}y). \end{aligned} \tag{4.5}$$

Thus in structures of constant width, for each mode number m four functions are needed to completely describe the longitudinal wake potential. These functions can be calculated as follows

$$\begin{aligned} W_m^{cc} &= W_m(0,0,s), & W_m^{sc} &= \frac{1}{k_{x,m}}\frac{\partial}{\partial y}W_m(0,0,s), \\ W_m^{cs} &= \frac{1}{k_{x,m}}\frac{\partial}{\partial y_0}W_m(0,0,s), & W_m^{ss} &= \frac{1}{(k_{x,m})^2}\frac{\partial^2}{\partial y\partial y_0}W_m(0,0,s), \end{aligned} \tag{4.6}$$

where the m^{th} modal component of the wake potential

$$W_m(y_0,y,s) = Q^{-1}\int_{-\infty}^{\infty}[E_{z,m}(y,z,t)]_{t=(z-s)/c}dz \tag{4.7}$$

is excited by a charge distribution that does not depend on x,

$$\rho_m(y_0,y,s) = Q\delta(y-y_0)\lambda(s). \tag{4.8}$$

With a knowledge of the longitudinal wake we can calculate the transverse wakes. For example, the vertical wake potential, W_y, can be easily found through the Panofsky-Wenzel theorem

$$\frac{\partial}{\partial s}W_y(x_0,y_0,x,y,s) = \frac{\partial}{\partial y}W_{\parallel}(x_0,y_0,x,y,s). \tag{4.9}$$

Let us consider a structure of constant width $2w$ that also has a vertical symmetry plane, at $y = 0$. Structures in Fig. 6.1 (a) and (b) possess this symmetry; hence, they have a symmetry axis located at $x = w$, $y = 0$. Due to the symmetry, the wake potential satisfies the equation

$$W_{\|}(x_0, y_0, x, y, s) = W_{\|}(x_0, -y_0, x, -y, s), \tag{4.10}$$

and Eq. (4.5) simplifies:

$$W_m(y_0, y, s) = W_m^{cc}(s)\cosh(k_{x,m}y_0)\cosh(k_{x,m}y) + W_m^{ss}(s)\sinh(k_{x,m}y_0)\sinh(k_{x,m}y). \tag{4.11}$$

Note that

$$W_m(y_0, y, s) = W_m(y, y_0, s). \tag{4.12}$$

Let us consider the transverse wakes in such structures. We first introduce the integrated wake functions (sometimes called the step function response)

$$S_m^{cc} = \int_{-\infty}^{s} W_m^{cc}(s')ds', \quad S_m^{ss} = \int_{-\infty}^{s} W_m^{ss}(s')ds'. \tag{4.13}$$

It then follows from (4.9) that the transverse wake function can be written as

$$W_y(x_0, y_0, x, y, s) = \frac{1}{w}\sum_{m=1}^{\infty} k_{x,m}W_{y,m}(y_0, y, s)\sin(k_{x,m}x_0)\sin(k_{x,m}x), \tag{4.14}$$

$$W_x(x_0, y_0, x, y, s) = \frac{1}{w}\sum_{m=1}^{\infty} k_{x,m}W_{x,m}(y_0, y, s)\sin(k_{x,m}x_0)\cos(k_{x,m}x), \tag{4.15}$$

where

$$W_{y,m}(y_0, y, s) = S_m^{cc}(s)\cosh(k_{x,m}y_0)\sinh(k_{x,m}y) + S_m^{ss}(s)\sinh(k_{x,m}y_0)\cosh(k_{x,m}y),$$
$$W_{x,m}(y_0, y, s) = S_m^{cc}(s)\cosh(k_{x,m}y_0)\cosh(k_{x,m}y) + S_m^{ss}(s)\sinh(k_{x,m}y_0)\sinh(k_{x,m}y).$$

Representations (4.14), (4.15), are valid for arbitrary offsets of leading and trailing particles.

For small offsets near the symmetry axis, $x = w$, $y = 0$, the transverse wake potential is usually expanded in Taylor series,

$$W_y(w, y_0, w, y, s) \approx y_0\frac{\partial}{\partial y_0}W_y(w, y_0, w, 0, s)\big|_{y_0=0} + y\frac{\partial}{\partial y}W_y(w, 0, w, y, s)\big|_{y=0}. \tag{4.16}$$

The first term in (4.16) is usually called the transverse *dipole* wake in the y-direction. It can be calculated as follows

$$W_{y,d}(s) \equiv \frac{\partial}{\partial y_0}W_y(w, y_0, w, 0, s)\big|_{y_0=0} = \frac{1}{w}\sum_{m=1,\text{odd}}^{\infty}(k_{x,m})^2 S_m^{ss}(s). \tag{4.17}$$

The second term in (4.16) is called the transverse *quadrupole* wake in y-direction; it is obtained by

$$W_{y,q}(s) \equiv \frac{\partial}{\partial y}W_x(w, 0, w, y, s)\big|_{y=0} = \frac{1}{w}\sum_{m=1,\text{odd}}^{\infty}(k_{x,m})^2 S_m^{cc}(s). \tag{4.18}$$

The transverse wakes in the x direction are obtained by equations corresponding to those of Eqs. (4.17), (4.18). Note that $W_{y,q}(s) = -W_{x,q}(s)$.

In numerical calculations of structures with symmetry we can use the approach of paper [4] that allows us to reduce the calculation domain in half. Indeed the charge distribution (4.8) can be written as a sum of symmetric and antisymmetric parts

$$\rho_m(y_0,y,s) = \rho_m^E(y_0,y,s) + \rho_m^H(y_0,y,s), \tag{4.19}$$

where

$$\rho_m^H(y_0,y,s) = \frac{1}{2} Q\left[\delta(y-y_0) + \delta(y+y_0)\right]\lambda(s), \tag{4.20}$$

$$\rho_m^E(y_0,y,s) = \frac{1}{2} Q\left[\delta(y-y_0) - \delta(y+y_0)\right]\lambda(s). \tag{4.21}$$

In problems with the symmetric driving charges (4.20), the tangential component of the magnetic field will be zero in the symmetry plane (the so called "magnetic" boundary condition). In problems with the antisymmetric driving charges (4.21) the tangential component of the electric field will be zero in the symmetry plane (the "electric" boundary condition). Thus, instead of solving the system of equations in the whole domain, one can solve two independent problems in half of the domain: one problem with the "magnetic" boundary condition at $y = 0$ and one problem with the "electric" boundary condition at $y = 0$. This is true not only for the line-charge current distribution (4.1), but for any arbitrary three dimensional charge distribution $\rho(x,y,z,t)$. From solutions $W_m^H(y_0,y,s)$ and $W_m^E(y_0,y,s)$ of the two problems we can easily find the one dimensional modal functions in Eq. (4.11):

$$W_m^{cc}(s) = W_m^H(0,0,s), \qquad W_m^{ss}(s) = (k_{x,m})^{-2} \frac{\partial^2}{\partial y_0 \partial y} W_m^E(y_0,y,s)\big|_{y,y_0=0}. \tag{4.22}$$

The current version of ECHO2D allows to treat only rectangular structures with vertical plane of symmetry.

4.2 Installation

The program ECHO2D is compiled for Windows. It can be downloaded as archive **ECHO2D.zip** from `https://www.echo4d.de`. Extract the archive keeping the stricture of folders and files.

The archive contains the following folders.

1. **Docs**. It contains this manual.
2. **Codes/ECHO2D**. It contains the executables: console application **ECHO2D.exe** and GUI application **ECHO2D_GUI.exe** .
3. **Examples**. It contains several examples.
4. **MatLib4ECHO**. It contains Matlab functions for postprocessing.
5. **PostProcessor2D**. It contains Matlab scripts for postprocessing.

In the following we will describe usage of console application only.

4.3 Input files

The program ECHO2D requires two input files:

- a file with geometry description in ASCII format; it can have an arbitrary name,
- a file with parameters of the simulation in ASCII format; it has a fixed name **input_in.txt**.

Additionally some special directories and files can be present as explained in the following Sections.

4.3.1 Geometry description

The geometry can be imported as a file in ASCII format with extension "*.txt".

The geometry file is ASCII file with extension "*.txt". It has the following format:

%Number of materials
N_m
% Number of elements in metal with conductive walls, permeability, permitivity, cond.
N^1 ε^1 μ^1 σ^1
% Segments of lines and elipses with wall conductivity
$z^1_{1,1}$ $r^1_{1,1}$ $z^1_{2,1}$ $r^1_{2,1}$ $z^1_{3,1}$ $r^1_{3,1}$ $z^1_{4,1}$ $r^1_{4,1}$ d^1_1 k^1_1
...
$z^1_{1,N}$ $r^1_{1,N}$ $z^1_{2,N}$ $r^1_{2,N}$ $z^1_{3,N}$ $r^1_{3,N}$ $z^1_{4,N}$ $r^1_{4,N}$ d^1_N k^1_N
...
% Number of elements in material N_m, permetivity, permeability, conductivity
N^{N_m} ε^{N_m} μ^{N_m} σ^{N_m}
% Segments of lines and elipses
$z^{N_m}_{1,1}$ $r^{N_m}_{1,1}$ $z^{N_m}_{2,1}$ $r^{N_m}_{2,1}$ $z^{N_m}_{3,1}$ $r^{N_m}_{3,1}$ $z^{N_m}_{4,1}$ $r^{N_m}_{4,1}$ $d^{N_m}_1$ 0
...
$z^{N_m}_{1,N}$ $r^{N_m}_{1,N}$ $z^{N_m}_{2,N}$ $r^{N_m}_{2,N}$ $z^{N_m}_{3,N}$ $r^{N_m}_{3,N}$ $z^{N_m}_{4,N}$ $r^{N_m}_{4,N}$ $d^{N_m}_N$ 0

The parameters in the geometry file are:

- N_m - number of materials.
- ε^j, μ^j, σ^j - relative permitivity, permeability and conductivity in S/m of material number j.
- N^j - total number of segments (lines or elliptical arcs) in material j.
- $z_{1,i}$, $r_{1,i}$ - coordinates in cm of start point for segment number i.
- $z_{2,i}$, $r_{2,i}$ - coordinates in cm of end point for segment number i.
- $z_{3,i}$, $r_{3,i}$, $z_{4,i}$, $r_{4,i}$ - - coordinates in cm of square in which the ellipse is inscribed (for lines these parameters should be zeros).
- $z_{3,i}$, $r_{3,i}$ - coordinates in cm of top left corner.
- $z_{4,i}$, $r_{4,i}$ - coordinates in cm of bottom right corner.
- d_i - orientation (0-clock, 1-anticlock).
- k_i - wall conductivity in S/m (only for the first material).

In this listing the strings which begin with % are not comments. They are separators and are obligatory. For rectangular geometry the format is the same with replacing $r \to y$.

As example let us consider the geometry shown in Fig. 2.2. The corresponding file will have the following content

```
%Number of materials
1
% Number of elements in metal with conductive walls, permeability, permitivity, cond.
3 1 1 0
% Segments of lines and elipses with wall conductivity
z0 r1 z1 r1 0 0 0 0 0 k1
z1 r1 z2 r2 z3 r3 z4 r4 0 k2
z2 r2 z5 r2 0 0 0 0 0 k3,
```

where k1, k2, k3 are conductivities of the segments.

4.3.2 Parameters of simulation

The parameters of simulation are listed in input command file with fixed name **input_in.txt**. This file has a following format.

%%%%%%%%%%%%%% geometry %%%%%%%%%%%%%%%%%%%%

GeometryFile=$*$.txt
Units=m/cm/mm
GeometryType=round/recta
Width=W
SymmetryCondition=magn/elec
Convex=0/1

%%%%%%%%%%%%%% beam %%%%%%%%%%%%%%%%%%%%%%%%

InPartFile=$-$/$*$.txt/$*$.bin
BunchSigma=σ_z
Offset=y_0
InjectionTimeStep=t_{inj}

%%%%%%%%%%%%%% field %%%%%%%%%%%%%%%%%%%%%%%

InFieldDir=$-$/string
PortDir=$-$/string
PortPosition=z_p

%%%%%%%%%%%%%% model %%%%%%%%%%%%%%%%%%%%%%%

WakeIntMethod=dir/ind
Modes=m_0 ...m_N
ParticleMotion=0/1
ParticleField=0/1
CurrentFilter=n_F
ParticleLoss=0/1

%%%%%%%%%%%%%% mesh %%%%%%%%%%%%%%%%%%%%%%%%

MeshLength=N_z
StartPosition=z_s
TimeSteps=n_t
StepY=h_y
StepZ=h_z
NStepsInConductive=N_c
AdjustMesh=0/1
MeshMotionFile=$-$/$*$.txt

%%%%%%%%%%%%%% monitors %%%%%%%%%%%%%%%%%%%%%

WakeMonitor= M_1 M_2 M_3
BeamMonitor= M_1 M_2 M_3 M_4
FieldMonitor= { F t_F z_0 z_1 y_0 y_1 s_0 s_1 N }
DumpField=0/1
DumpParticles=0/1

```
DumpCurrent=0/1
DumpMesh=0/1
```

The parameters in this command file are:

- **GeometryFile** [string]. Name of ASCII file with extension '*.txt'. It defines the name of file with the geometry description.
- **Units** [string]. Units of the geometry description: 'm'/'cm'/'mm'.
- **GeometryType** [string]. It defines type of geometry: 'round'/'recta'.
- **Width** [float/m]. Width of rectangular geometry W in x direction in m. The parameter is obsolete for round geometry.
- **SymmetryCondition** [string] It defines the boundary condition on axis for rectangular geometry: elec/magn.
- **Convex** [boolean]. Use '1' to accelerate the calculation for "convex" geometry. "Convex" means here a geometry that has only one connected vacuum region in each plane transverse to the symmetry axis.
- **InPartFile** [string]. Input bunch as a particle file ('*.bin') or as a pencil beam profile ('*.txt'). If you would like to use the default Gaussian pencil bunch with rms length σ_z use here option '-'.
- **BunchSigma** [float/m] The Gaussian pencil bunch rms length σ_z in m.
- **Offset** [integer]. It defines value of y_0 for pencil beam in mesh lines. In metric units it can be found as $(y_0+0.5)\cdot h_y$ for round geometry or as $y_0\cdot h_y$ for rectangular one. The value "-1" mens that we use y_0 as large as possible. The last choice provides the best accuracy.
- **InjectionTimeStep** [integer]. Time of particle distribution injection in time steps. In metric units it can be found as $t_{inj}\cdot h_z/c$, where c is the light velocity.
- **InFieldDir** [string]. It defines the name of directory with files of initial filed. Use '-' if initial field should be calculated in the program itself.
- **PortDir** [string]. It defines the name of directory with file of transverse mode in waveguide port. Use '-' if it is absent.
- **PortPosition** [integer]. It defines the position z_p of the waveguide port in mesh lines. Use '-1' if the port is absent.
- **WakeIntMethod** [string]. Direct or indirect wake potential integration: 'dir'/'ind'.
- **Modes** [integer list]. It defines Fourier modes m_0 ...m_N to be calculated.
- **ParticleMotion** [boolean]. It defines whether equations of motion are used ('1') or the particle distributian is frozen ('0').
- **ParticleField** [boolean]. It defines whether fields are calculated ('1') or not ('0').
- **CurrentFilter** [integer]. It defines how many times n_F a simple 2-points low-pass filter will be applied longitudinally to the current profile.
- **ParticleLoss** [boolean]. It defines whether particles are lost in materials ('1') or not ('0').
- **MeshLength** [integer]. It defines length of the moving mesh N_z in the mesh lines. In metric units the length is $N_z\cdot h_z$.
- **StartPosition** [integer]. It defines the longitudinal start position of moving mesh in mesh lines.
- **TimeSteps** [integer]. It defines the number of time steps in the calculation. Use '-1' to fly through the whole structure.
- **StepY** [float/m]. It defines the transverse mesh step h_y in m.
- **StepZ** [float/m]. It defines the longitudinal mesh step h_z in m.
- **NStepsInConductive** [integer]. It should be set to '0' if the structure is perfectly electric conductive. Otherwise it defines an one dimensional mesh length for tangential components of electromagnetic field in the conductive parts [6]. The default value is 10. Increase this value to obtain a better accuracy.

- **AdjustMesh** [boolean]. It defines whether the transverse mesh step is adjusted to the outgoing waveguide size ('1') or not ('0').
- **MeshMotionFile** [string]. Name of ASCII file with extension '*.txt'. It defines mesh motion. Use '-' to fly in positive direction with the light velocity.
- **WakeMonitor** [integer list: M_1 M_2 M_3]. Defines save points of the wake potential: from time step M_1 to time step $M2$ with step M_3.
- **BeamMonitor** [integer list: M_1 M_2 M_3 M_4]. It defines beam monitor. The parameters are explained in Section...
- **FieldMonitor** [string: F string: t_F integer list: F t_F z_0 z_1 y_0 y_1 s_0 s_1 N]. It defines field monitor for the field component F: 'Ex'/'Ey'/'Ez'/'Bx'/'By'/'Bz'. Parameter t_F defines type of the monitor: 'z'/'s'. Other parameters are explained in Section 4.3.6.
- **DumpField** [boolean]. It defines whether the filed is dumped ('1') or not ('0').
- **DumpParticles** [boolean]. It defines whether the particles are dumped ('1') or not ('0').
- **DumpCurrent** [boolean]. It defines whether the current is dumped ('1') or not ('0').
- **DumpMesh** [boolean]. It defines whether the mesh is dumped ('1') or not ('0').

4.3.3 Beam setup

The beam setup is done by parameter **InPartFile** [string] in the command file **input_in.txt**. The beam can be defined in three ways: (1) the default Gaussian pencil bunch; (2) a pencil beam with arbitrary longitudinal profile; (3) a three dimensional particle distribution.

Option '-' defines the default Gaussian pencil bunch with rms length defined by parameter **BunchSigma** in the command file **input_in.txt**.

A pencil beam with arbitrary longitudinal profile should be described in file with extension "*.txt". This file has the following format:

% s[m] charge [normalized]
s_0 $\rho(s_0)$
s_1 $\rho(s_1)$
...
s_N $\rho(s_N)$

The first line is a comment. The first column describes the bunch coordinate with uniform step. The second column defines the bunch shape in arbitrary units. The s-coordinate should be positive and it increases from the head to the tail of the bunch. This shape will be projected on the moving mesh with the longitudinal coordinates in interval [0, StepZ*MeshLength].

Directory **Examples/ N14_WakeMonitor_ArbitraryBunchShape** contains an example of a special bunch profile.

Finally the last option to define the bunch shape is to create a file with extension "*.bin" which contains a particle distribution. This files has binary format: ...

4.3.4 Initial field setup

4.3.5 Waveguide port setup

4.3.6 Field monitors setups

In code ECHO2D two types of fields monitors exists: s-time and z-time.

The field monitor is described by line **FieldMonitor =** F t_F z_0 z_1 y_0 y_1 s_0 s_1 N. Here F defines the filed component: 'Ex'/'Ey'/'Ez'/'Hx'/'Hy'/'Hz'. The second parameter t_f defines the type of the field monitor: 's'/'z'. The parameters y_0 and y_1 define the transverse interval in meters in which field is saved (see Figs). The last parameter N defines sampling interval in timesteps $h_t = h_z/c$, where c is the light velocity.

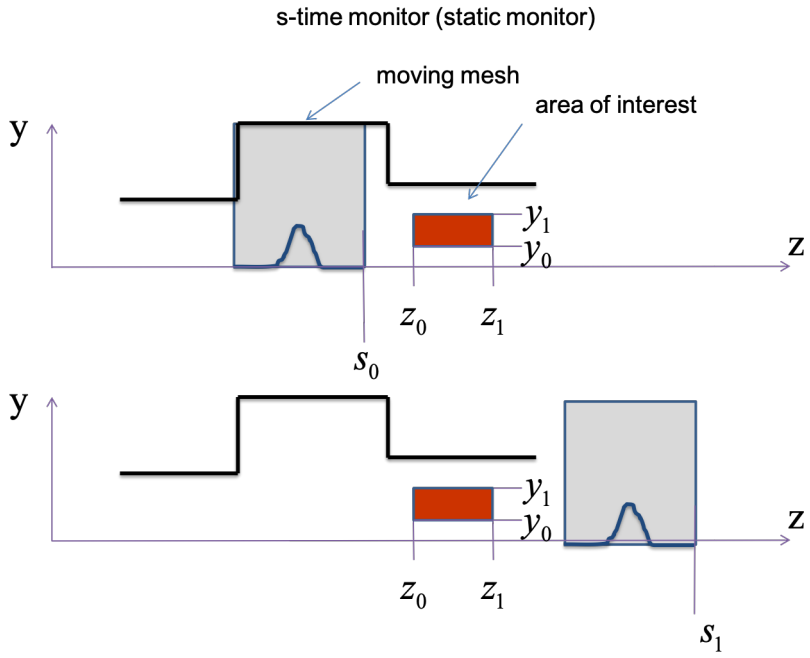


Figure 4.2: Field monitor of type s-time.

The s-time monitor is a static monitor. The window is defined as static rectangle in the calculation domain with longitudinal coordinates z_0, z_1 in meters. The filed is saved from time $t_0 = s_0/c$ to time $t_1 = s_1/c$ with interval $h_z/c/N$ .The principle of s-time monitor is explained in Fig. 4.2.

The z-time monitor is a moving monitor. The window is defined as moving rectangle in the moving mesh with longitudinal coordinates s_0, s_1 in meters. The filed is saved from time $t_0 = z_0/c$ to time $t_1 = z_1/c$ with interval $h_z/c/N$ .The principle of z-time monitor is explained in Fig. 4.3.

The output formats and postprocessing are described below. An example can be found in the directory **Examples/ N8_FlatTaperWithFieldMonitor**.

4.4 Wakefield Calculation

The local folder should contain three files:

- geometry file,
- command file **input_in.txt**,
- command file **run.bat**, which starts **ECHO2D.exe**.

The calculations starts by execution of **run.bat**. During the simulation the progress in percents is shown. All modes are calculated in parallel.

4.5 Output files

After execution of **ECHO2D.exe** the folder "round"/"magn"/"elec" will be created. It contains N_m files with modal wakes. They have name pattern WakeL_XX.txt, where XX is the mode number m_i. Each file is text file with two columns and contains a longitudinal modal wake.

```
% vertical mesh step h[m] offset[mesh lines]
1.990050e−04 48
% rectangular width [m]  bunch rms [m]
0.000000e+00  1.000000e−03
```

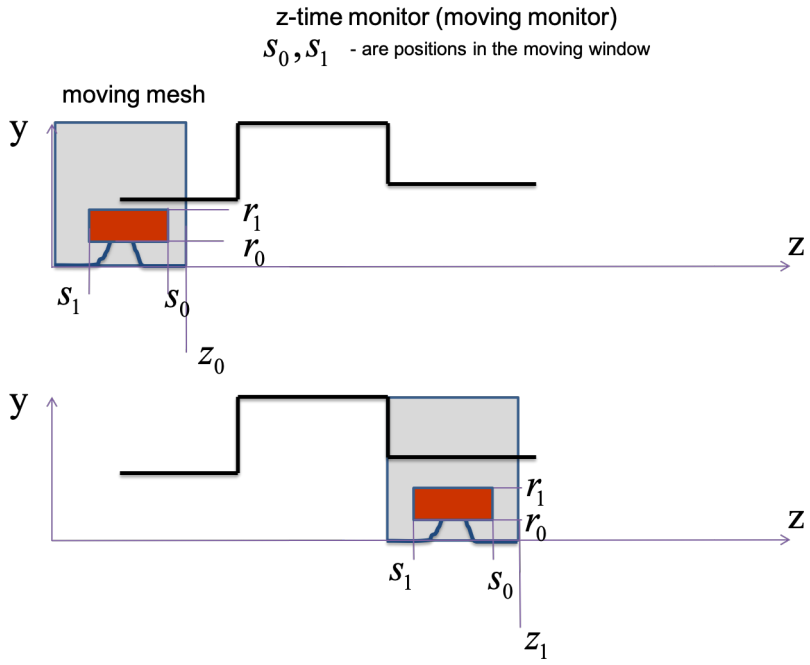


Figure 4.3: Field monitor of type z-time.

```
% modal wake
% s[m]  W(s)[m*V/nC]
1.000000e-04 -2.752929e-03
```

For round geometry the units are V/nC.

R The modal wakes are not normalized on the beam offset. It means, for example, that in order to obtain the same transverse dipole wake (mode $m = 1$, file **wakeT.dat**) as in ECHOz2 you need to integrate the wake from file **wakeL_01.txt** divided by the beam offset squared. See script **PP_WakeDipole.m** from the post-processor directory.

If field monitors had been setup in file **input_in.txt** then they are saved in the same directory in ASCII files with name pattern **Monitor_mXX_NYY.txt**, where XX is the mode number and YY the ordinal number of the monitor.

The s-type monitor file has the following format:

% Field=F time=s width= W
% k_ct=k_{ct} h_ct=h_{ct} ct0=s_0
% k_r=k_r h_r=h_r r0=r_0
% k_z=k_z h_z=h_z z0=z_0
s_0
$F(r_0,z_0)$ ... $F(r_0,z_1)$
...
$F(r_1,z_0)$ ... $F(r_1,z_1)$
s_0+h_{ct}
$F(r_0,z_0)$ ... $F(r_0,z_1)$
...
$F(r_1,z_0)$ ... $F(r_1,z_1)$
...

s_1
$F(r_0,z_0)$ ...$F(r_0,z_1)$
...
$F(r_1,z_0)$ ... $F(r_1,z_1)$

The z-type monitor file has the following format:

% Field=F time=s width= W
% k_ct=k_{ct} h_ct=h_{ct} ct0=z_0
% k_r=k_r h_r=h_r r0=r_0
% k_s=k_s h_s=h_z s0=s_0
z_0
$F(r_0,s_0)$ ... $F(r_0,s_1)$
...
$F(r_1,s_0)$ ... $F(r_1,s_1)$
z_0+h_{ct}
$F(r_0,s_0)$ ... $F(s_0,z_1)$
...
$F(r_1,s_0)$ ... $F(r_1,s_1)$
...
z_1
$F(r_0,s_0)$ ...$F(r_0,s_1)$
...
$F(r_1,s_0)$ ... $F(r_1,s_1)$

For rectangular geometry the harmonic fileds E_x, E_y, E_z are saved in V/m^2. For round geometry the electric field components E_r, E_z are saved in V/m^{2k+1}, where k is the mode number. The azimuthal component E_φ is in the same units but multiplied yet by radial coordinate in meters. Magnetic field components are saved multiplied by velocity of light c as cB and hence the units are the same as for the electric field components .

In order to obtain the total field use the malab scripts described in Section "Postprocessing".

The folder "round"/"magn"/"elec" contains several files with initial beam currents: **Iz0.tx**, **Ir0.txt**. File **Iz0.tx** contains z-component of the current on mesh and has the following format:

s_0 $I_z(s_0,r_0)/c$ $I_z(s_0,r_1)/c$ ... $I_z(s_0,r_{N_r})/c$
...
s_{N_z} $I_z(s_{N_z},r_0)/c$ $I_z(s_{N_z},r_1)/c$ ... $I_z(s_{N_z},r_{N_r})/c$

Here the first column is a longitudinal bunch coordinate in meters. The current component I_z/c is given in Coulombs. File **Ir0.tx** contains r-component of the initial current in the same format.

4.6 Postprocessing

The folder PostProcessor2D contains two subfolders:

- Fields,
- Wakes.

to be continued...

4.6.1 Wakes

The matlab scripts ...

4.7 Examples

In this section we consider several examples included in the archive at the directory **Examples**.

4.7.1 Example 1: Round collimator

The examples of round collimator can be found in directories **Examples/ N1_RoundCollimatorLong**, **Examples/ N2_RoundCollimatorDipole**, **Examples/ N3_RoundCollimatorDipoleConductive**.

In order to calculate the longitudinal wake of monopole mode ($m = 0$) proceed as follows:

- Go to directory **Examples/ N1_RoundCollimatorLong/ ECHO2D** and run **run.but** It calls the console executable from directory **Codes/ECHO2D**.
- Alternatively you can use GUI application and file **N1.echo2d**.

After the code execution a directory **round** is created. The monopole wake is saved in file **wakeL_00.txt**. Use the matlab script **PostProcessor2D/ Round/ PP_Wake_Monopole.m** to see the wake shown in Fig. 2.5. The transverse wake is zero for the monopole mode.

In order to calculate the transverse wake of dipole mode ($m = 1$) proceed as follows:

- Go to directory **Examples/ N2_RoundCollimatorDipole/ ECHO2D** and run **run.but** It calls the console executable from directory **Codes/ ECHO2D**.
- Alternatively you can use GUI application and file **N2.echo2d**.

After the code execution a directory **round** is created. The dipole wake is saved in file **wakeL_01.txt**. Use the matlab script **PostProcessor2D/Round/ PP_Wake_Dipole.m** to see the wake shown in Fig. 3.3.

The last example is the same collimator but with conductive small pipe. The conductivity is equal to 1 S/m. In order to calculate the transverse wake of dipole mode ($m = 1$) for the conductive collimator proceed as follows:

- Go to directory **Examples/ N3_RoundCollimatorDipoleConductive/ ECHO2D** and run **run.but** It calls the console executable from directory **Codes/ ECHO2D**.
- Alternatively you can use GUI application and file **N3.echo2d**.

After the code execution a directory **round** is created. The dipole wake is saved in file **wakeL_01.txt**. Use the matlab script **PostProcessor2D/ Round/ PP_Wake_Dipole.m** to see the wake shown in Fig. 3.4.

4.7.2 Example 2: Resistive pillbox cavity

The example of pillbox cavity can be found in directory **Examples/ N9_ResistivePillbox**. The cavity walls have conductivity equal to 1000 S/m.

- Go to directory **Examples/ N9_ResistivePillbox/ ECHO2D** and run **run.but** It calls the console executable from directory **Codes/ ECHO2D**.
- Alternatively you can use GUI application and file **N9.echo2d**.

After the code execution a directory **round** is created. The dipole wake is saved in file **wakeL_01.txt**. Use the matlab script **PostProcessor2D/ Round/ PP_Wake_Dipole.m** to see the wake shown in Fig. 3.6. The transverse wake is zero for the monopole mode.

In order to calculate the monopole or the higher order modes change only "Modes" value in the input file.

4.7.3 Example 3: TESLA cavity

The example of TESLA cavity can be found in directory **Examples/ N10_TESLACavityLong**.

In order to calculate the longitudinal wake of monopole mode ($m = 0$) proceed as follows:

- Go to directory **Examples/ N10_TESLACavityLong/ ECHO2D** and run **run.but** It calls the console executable from directory **Codes/ECHO2D**.
- Alternatively you can use GUI application and file **N10.echo2d**.

After the code execution a directory **round** is created. The monopole wake is saved in file **wakeL_00.txt**. Use the matlab script **PostProcessor2D/ Round/ PP_Wake_Monopole.m** to see the wake shown in Fig. 2.5.

In order to calculate the dipole or the higher order modes change only "Modes" value in in the input file.

4.7.4 Example 4: Flat absorber

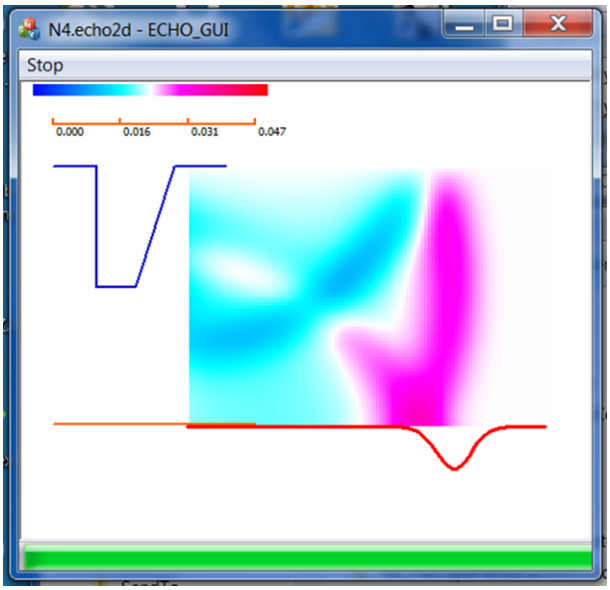


Figure 4.4: Geometry of flat absorber.

The example of a flat abosrber can be found in directories **Examples/ N4_FlatAbsorberLongQuad** and **Examples/ N4_FlatAbsorberDipole**.

The absorber has geometry shown in Fig. 4.4 with *Width = 0.07* defined in file **input_in.txt**. The bunch flies in ZY symmetry plane ($x_0 = x = 0$) and we calculate only odd modes *Modes = 1 3 5 7 9 11 13 15*. In order to calculate the longitudinal and quadrupole wakes we use *SymmetryCondition = magn* and proceed as follows:

- Go to directory **Codes/ ECHO2D** and start **ECHO2D_GUI.exe**.
- Go to menu "File/Open" and navigate to directory **Examples/ N4_FlatAbsorberLongQuad/ ECHO2D**. Open the input file **N4.echo2d**. You should see the geometry shown in Fig. 4.4.
- Go to menu "Sover/ Start". The calculation starts.
- Wait until message "Ready" appears and press "OK" . The calculations is done.
- Go to menu "Stop".
- Alternatively you can run the console application with command file **run.bat**.

The results of calculation are placed in directory **magn**. It contains 8 modal wakes. Run matlab script **PostProcessor2D/ Flat/ PP_Wcc.m** to calculate coefficients $W_{cc}(k_x, s)$ shown in Fig. 4.5. Finally run matlab script **PostProcessor2D/ Flat/ PP_WakeLQ.m** to calculate longitudinal and quadrupole transverse wakes shown in Fig. 4.6.

In order to calculate the dipole wake we use *SymmetryCondition = elec* and proceed as follows:

- Go to directory **Codes/ECHO2D** and start **ECHO2D_GUI.exe**.
- Go to menu "File/Open" and navigate to directory **Examples/ N5_FlatAbsorberLongQuad/ ECHO2D**. Open the input file **N5.echo2d**. You should see the geometry shown in Fig. 4.4.
- Go to menu "Sover/ Start". The calculation starts.
- Wait until message "Ready" appears and press "OK" . The calculations is done.

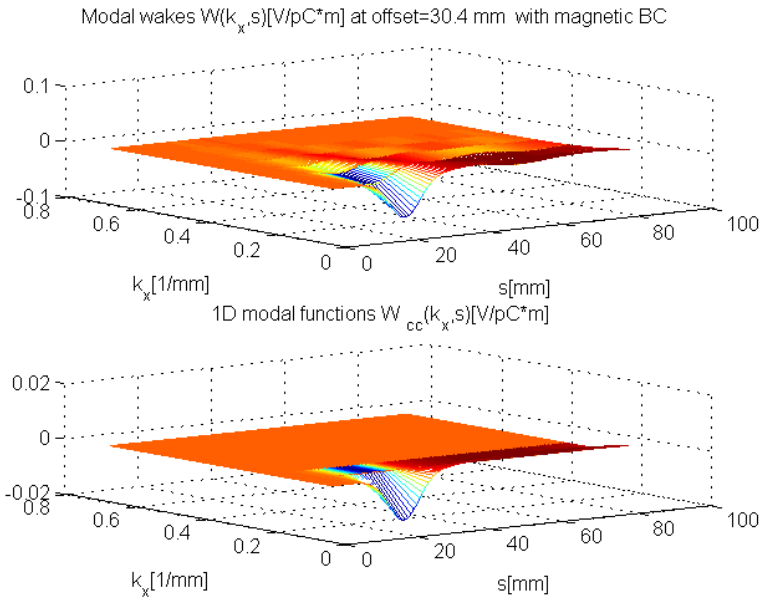


Figure 4.5: Flat absorber. Coefficients.

- Go to menu "Stop".
- Alternatively you can run the console application with command file **run.bat**.

The results of calculation are placed in directory **elec**. It contains 8 modal wakes. Run matlab script **PostProcessor2D/ Flat/ PP_Wss.m** to calculate coefficients $W_{ss}(k_x, s)$ shown in Fig. 4.7.

Before to proceed copy there directory **magn** from the previous case (or calculate it by changing *SymmetryCondition=magn*). Run matlab script **PostProcessor2D/ Flat/ PP_Wcc.m** to calculate coefficients $W_{cc}(k_x, s)$. Finally run matlab script **PostProcessor2D/ Flat/ PP_WakeLQD.m** to calculate longitudinal, quadrupole transverse and dipole transcerse wakes shown in Fig. 4.8.

Start Matlab and open **PostProcessor/ Flat/ PP_WakeZY**. Adjust “y”, “y0”. Run this matlab file to create file **WakeZY.txt** with longitudinal and transverse wakes for the offsets y, y0. Matlab script bf PP_WakeZY.m creates 4 plots shown in Fig. 4.9. The longitudinal and transverse wakes are on the right side. 3D plot at the left side are for estimation of the number of modes. It can be seen that we need more than 8 modes in this example (near the boundary!).

4.7.5 Example 5: Pohang Dechirper

The example of the dechirper can be found in directoriy **Examples/ N6_PohangDechirper**.

The absorber has geometry shown in Fig. 4.10 with *Width = 0.05* m defined in file **input_in.txt**. The bunch flies in ZY symmetry plane ($x_0 = x = 0$) and we calculate only odd modes *Modes = 1 3 5 7 9 11 13 15 17 19 21 23 25 27 29*. In order to calculate the longitudinal and quadrupole wakes we use *SymmetryCondition = magn* and proceed as follows:

- Check that parameter *SymmetryCondition* has value "magn" in file **input_in.txt**.
- Go to directory **Codes/ ECHO2D** and start **ECHO2D_GUI.exe**.
- Go to menu "File/Open" and navigate to directory **Examples/ N6_PohangDechirper/ ECHO2D**. Open the input file **N6.echo2d**. You should see the geometry shown in Fig. 4.10.
- Go to menu "Sover/ Start". The calculation starts.
- Wait until message "Ready" appears and press "OK" . The calculations is done.
- Go to menu "Stop" and **close the GUI program**.
- Alternatively you can run the console application with command file **run.bat**.

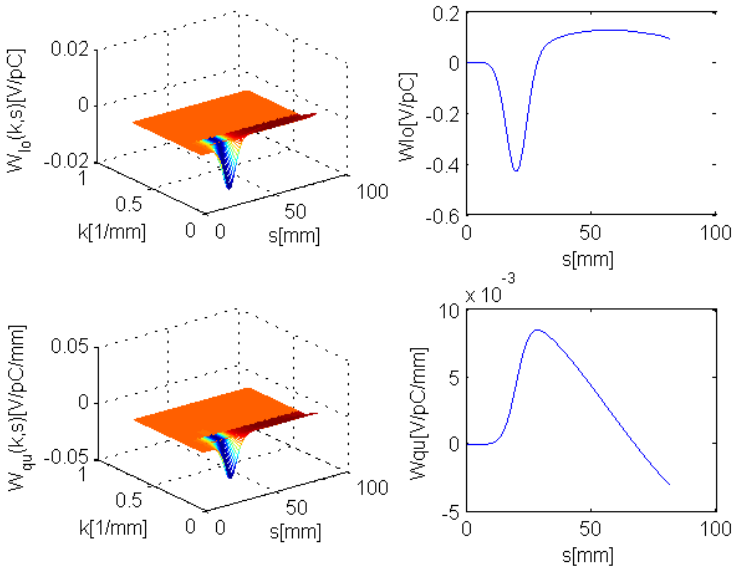


Figure 4.6: Flat absorber. Longitudinal and quadrupole transverse wakes.

The results of calculation are placed in directory **magn**. It contains 15 modal wakes. Run matlab script **PostProcessor2D/ Flat/ PP_Wcc.m** to calculate coefficients $W_{cc}(k_x, s)$ shown in Fig. 4.11. Finally run matlab script **PostProcessor2D/ Flat/ PP_WakeLQ.m** to calculate longitudinal and quadrupole transverse wakes.

In order to calculate the dipole wake we use *SymmetryCondition* = *elec* and proceed as follows:

- Change the parameter *SymmetryCondition* to value "elec" in file **input_in.txt**.
- Go to directory **Codes/ECHO2D** and start **ECHO2D_GUI.exe**.
- Go to menu "File/Open" and navigate to directory **Examples/ N6_PohangDechirper/ ECHO2D**. Open the input file **N6.echo2d**. You should see the geometry shown in Fig. 4.10.
- Go to menu "Sover/ Start". The calculation starts.
- Wait until message "Ready" appears and press "OK" . The calculations is done.
- Go to menu "Stop".
- Alternatively you can run the console application with command file **run.bat**.

The results of calculation are placed in directory **elec**. It contains 15 modal wakes. Run matlab script **PostProcessor2D/ Flat/ PP_Wss.m** to calculate coefficients $W_{ss}(k_x, s)$ shown in Fig. 4.12.

Finally run matlab script **PostProcessor2D/ Flat/ PP_WakeLQD.m** to calculate longitudinal, quadrupole transverse and dipole transcerse wakes shown in Fig. 4.13.

4.7.6 Example 7: Flat tapered collimator with resistivity

The example of the flat tapered collimator with resistivity can be found in directory **Examples/ N7_TaperedResistiveCollimator**.

The collimator has geometry shown in Fig. 4.14 with *Width* = *0.01*m defined in file **input_in.txt**. The bunch flies in ZY symmetry plane ($x_0 = x = 0$) and we calculate only odd modes *Modes* = *1 3 5 7 9 11 13 15*. In order to calculate the longitudinal and quadrupole wakes we use *SymmetryCondition* = *magn* and proceed as follows:

- Go to directory **Codes/ ECHO2D** and start **ECHO2D_GUI.exe**.
- Go to menu "File/Open" and navigate to directory **Examples/ N7_TaperedResistiveCollimator/ ECHO2D**. Open the input file **N7.echo2d**. You should see the geometry shown in Fig. 4.14.

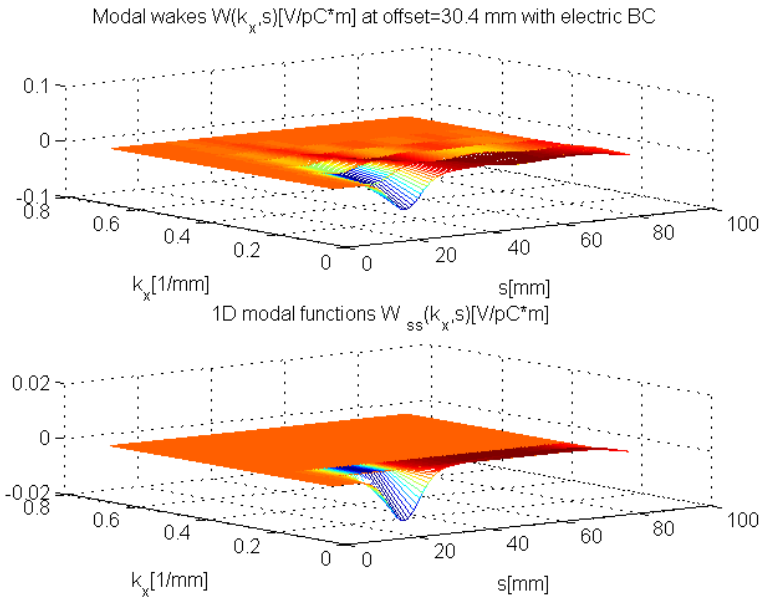


Figure 4.7: Flat absorber. Coefficients.

- Go to menu "Sover/ Start". The calculation starts.
- Wait until message "Ready" appears and press "OK" . The calculations is done.
- Go to menu "Stop" and close the GUI program.
- Alternatively you can run the console application with command file **run.bat**.

The results of calculation are placed in directory **magn**. It contains 8 modal wakes. Run matlab script **PostProcessor2D/ Flat/ PP_Wcc.m** to calculate coefficients $W_{cc}(k_x, s)$. Finally run matlab script **PostProcessor2D/ Flat/ PP_WakeLQ.m** to calculate longitudinal and quadrupole transverse wakes shown in Fig. 4.15.

4.7.7 Example 8: Field monitor for flat taper

The example of the flat taper can be found in directory **Examples/ N8_FlatTaperWithFieldMonitor**.

The taper has geometry shown in Fig. 4.16 with *Width = 0.05*m defined in file **input_in.txt**. The bunch flies in ZY symmetry plane ($x_0 = x = 0$) and we calculate only odd modes *Modes = 1 3 5 7*. The bunch flies on axis and we use *SymmetryCondition = magn* and proceed as follows:

- Go to directory **Codes/ ECHO2D** and start **ECHO2D_GUI.exe**.
- Go to menu "File/Open" and navigate to directory **Examples/ N8_FlatTaperWithFieldMonitor/ ECHO2D**. Open the input file **N8.echo2d**. You should see the geometry shown in Fig. 4.16.
- Go to menu "Sover/ Start". The calculation starts.
- Wait until message "Ready" appears and press "OK" . The calculations is done.
- Go to menu "Stop" and close the GUI program.
- Alternatively you can run the console application with command file **run.bat**.

The results of calculation are placed in directory **magn**. It contains 4 modal wakes. Run matlab script **PostProcessor2D/ Fields/ Flat/ PP_CreateTotalField_EzEyHx.m** to create the full field from monitor defined by string **MonitorNumber=2** in the script. Then run script **PP_FieldMonitor** to see the total field shown in Fig. 4.17.

4.7.8 Example 9: Round dielectric pipe

The example of the round dielectric pipe can be found in directory **Examples/ N11_Round_Dielectric**.

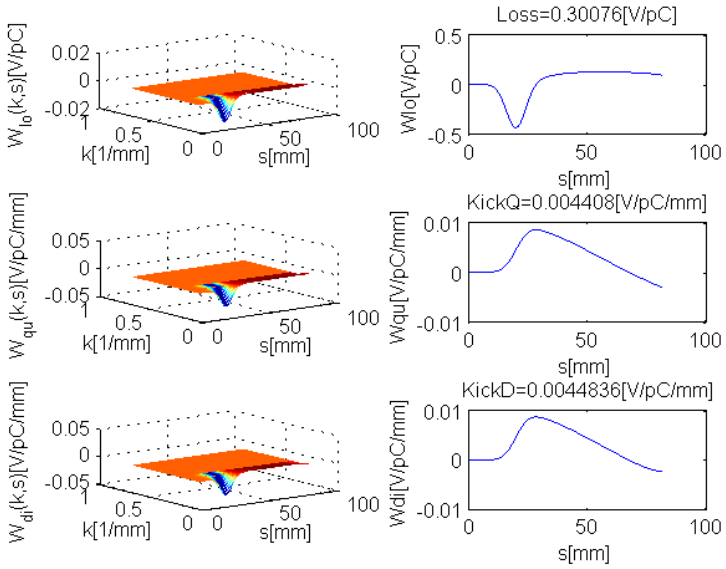


Figure 4.8: Flat absorber. Longitudinal, quadrupole and dipole transverse wakes.

The pipe has geometry shown in Fig. 6.3. In order to estimate the steady state solution we will calculate wakes for pipe of length 1.1m and for pipe of length 1m. Then we subtract the second wake from the first one. We proceed as follows:

- Open file **input_in.txt** and set *GeometryFile = PipeCondLayer_110cm.txt*.
- Go to directory **Codes/ ECHO2D** and start **ECHO2D_GUI.exe**.
- Go to menu "File/Open" and navigate to directory **Examples/ N8_FlatTaperWithFieldMonitor/ ECHO2D**. Open the input file **N11.echo2d**. You should see the geometry shown in Fig. 6.6.
- Go to menu "Sover/ Start". The calculation starts.
- Wait until message "Ready" appears and press "OK" . The calculations is done.
- Go to menu "Stop" and close the GUI program.
- Alternatively you can run the console application with command file **run.bat**.

The results of calculation are placed in directory **round**. Run matlab scripts **PostProcessor2D/ round/ PP_Wake_Monopole.m** and **PP_Wake_Dipole.m** . Rename the directory **round** in **round_1m10**. In file **input_in.txt** set *GeometryFile = PipeCondLayer_100cm.txt* and repeat the calculations together with execution of the matlab scripts. Rename the directory **round** in **round_1m**. The comparison of the results from ECHO2D with ECHO1D can be seen by running the script **Compare_2D_vs_1D.m** in Matlab. The result is shown in Fig. 6.7.

4.7.9 Example 10: Flat dielectric pipe

The example of the flat dielectric pipe can be found in directory **Examples/ N11_Flat_Dielectric**. Proceed as in the previous example and see the corresponding section in ECHO1D.

4.7.10 Example 11: TESLA cavity with restart procedure, wake monitors and arbitrary bunch shape

The examples can be found in directory **Examples/ N14_WakeMonitor_ArbitraryBunchShape** and **Examples/ N13_Restart**. As manual use the PPTX file in directory **Docs** of these examples.

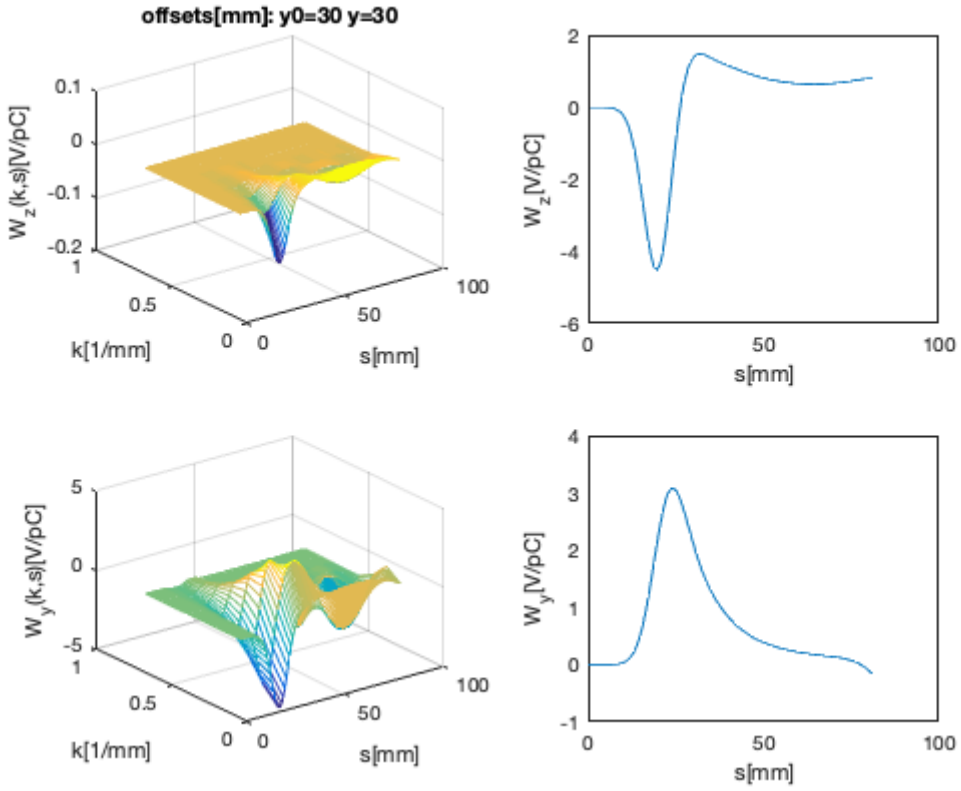


Figure 4.9: Flat absorber. Longitudinal and transverse wakes for offsets $y_0 = y = 30$ mm.

4.7.11 Example 12: Particle tracking in dielectric pipe

The examples can be found in directory **Examples/ N15_ParticleTracking** and **Examples/ N13_Restart**. As manual use the PPTX file in directory **Docs** of these examples.

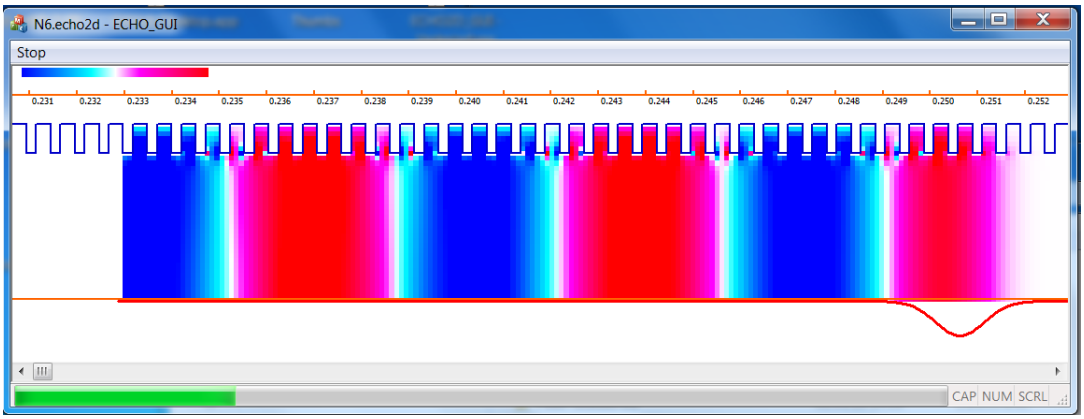


Figure 4.10: Geometry of the dechirper.

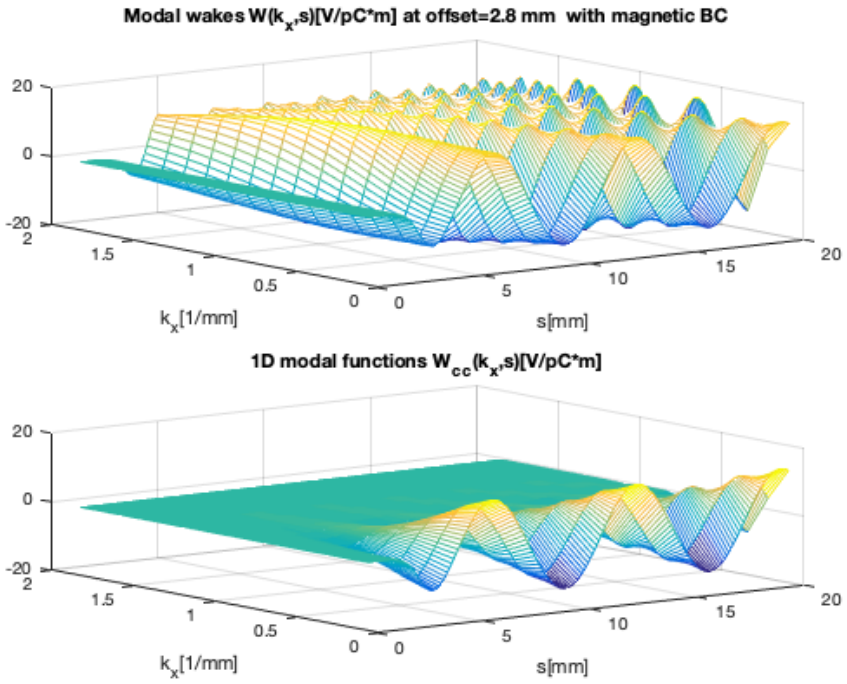


Figure 4.11: Dechirper. Coefficients.

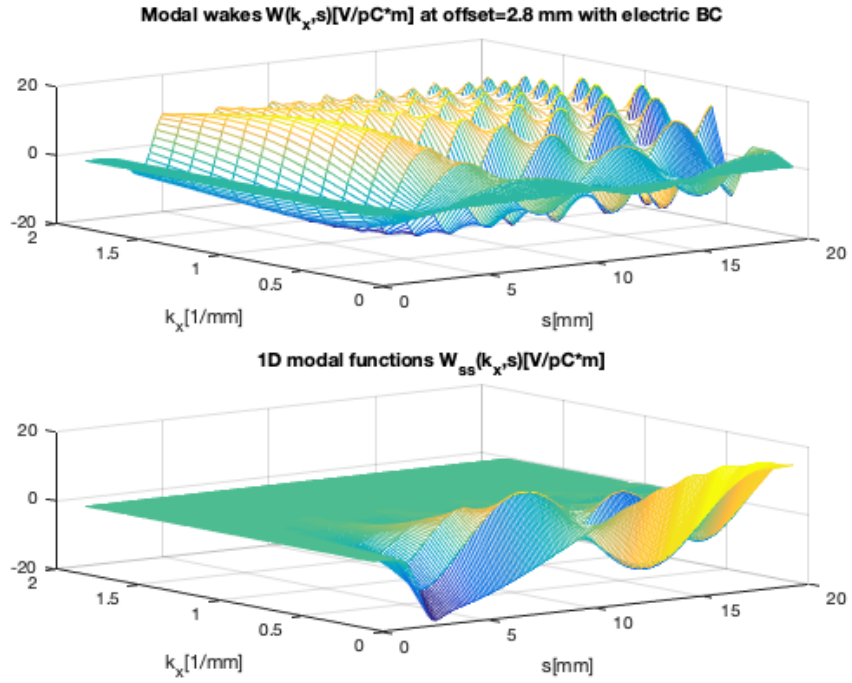


Figure 4.12: Dechirper Coefficients.

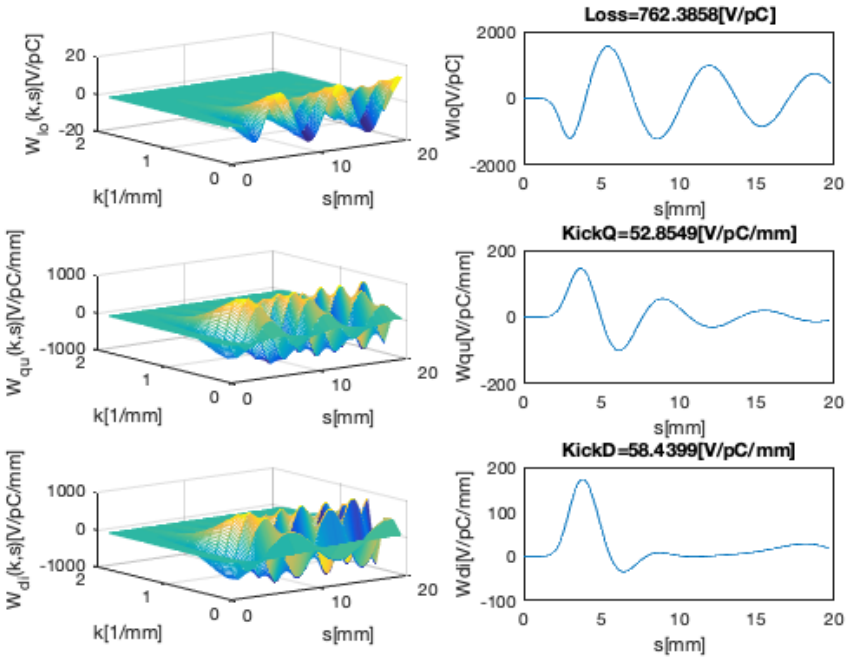


Figure 4.13: Dechirper. Wakes.

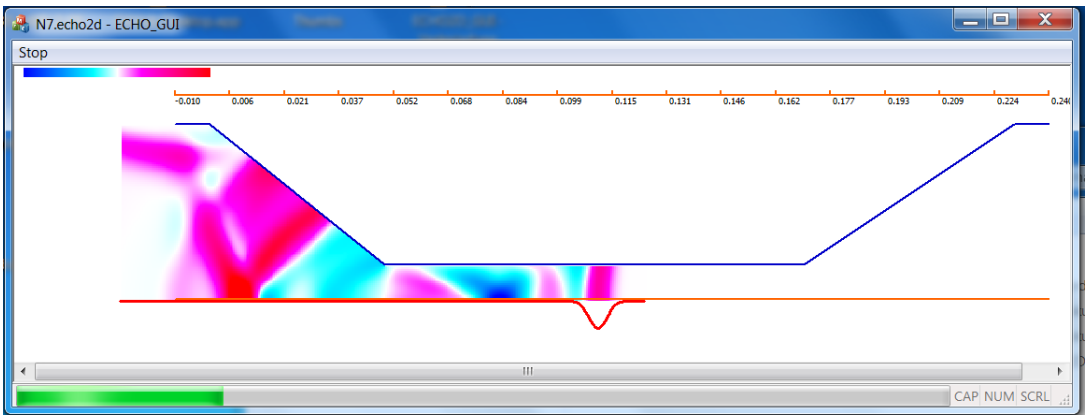


Figure 4.14: Geometry of flat tapered collimator with resistivity.

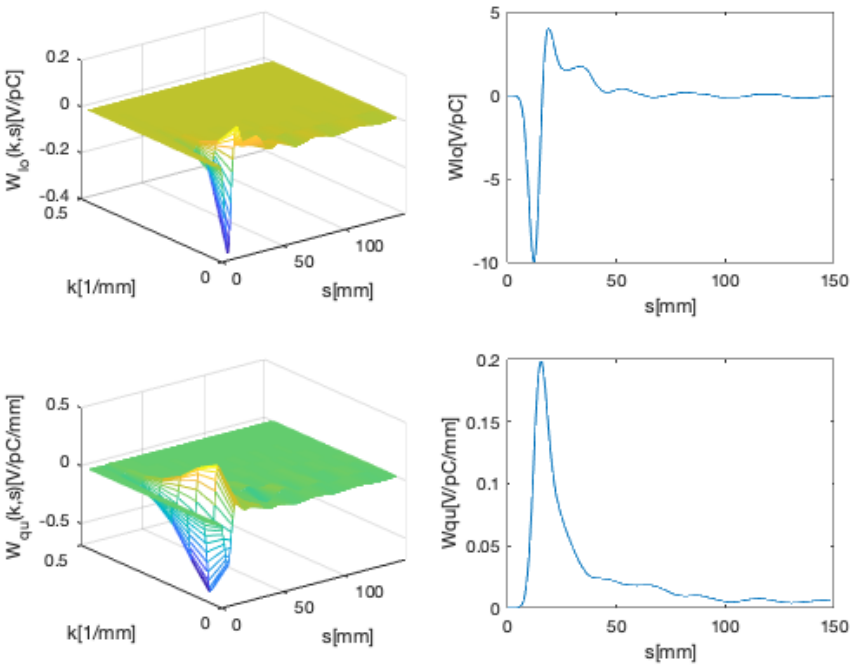


Figure 4.15: Tapered collimator. Wakes.

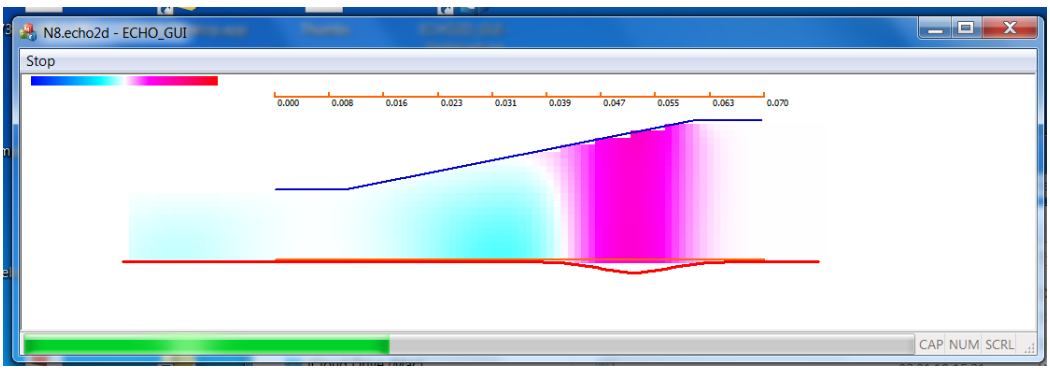


Figure 4.16: Geometry of flat taper.

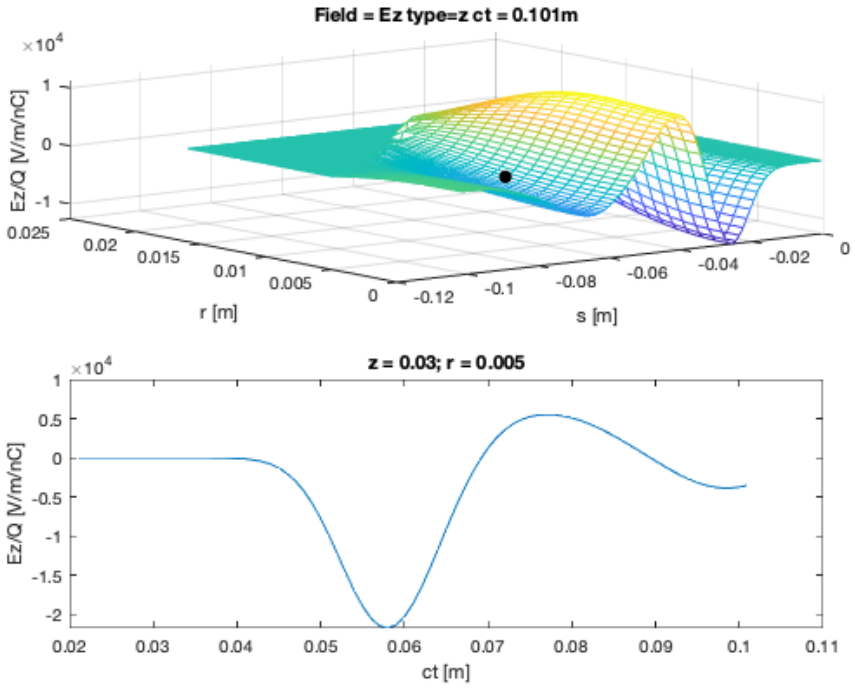
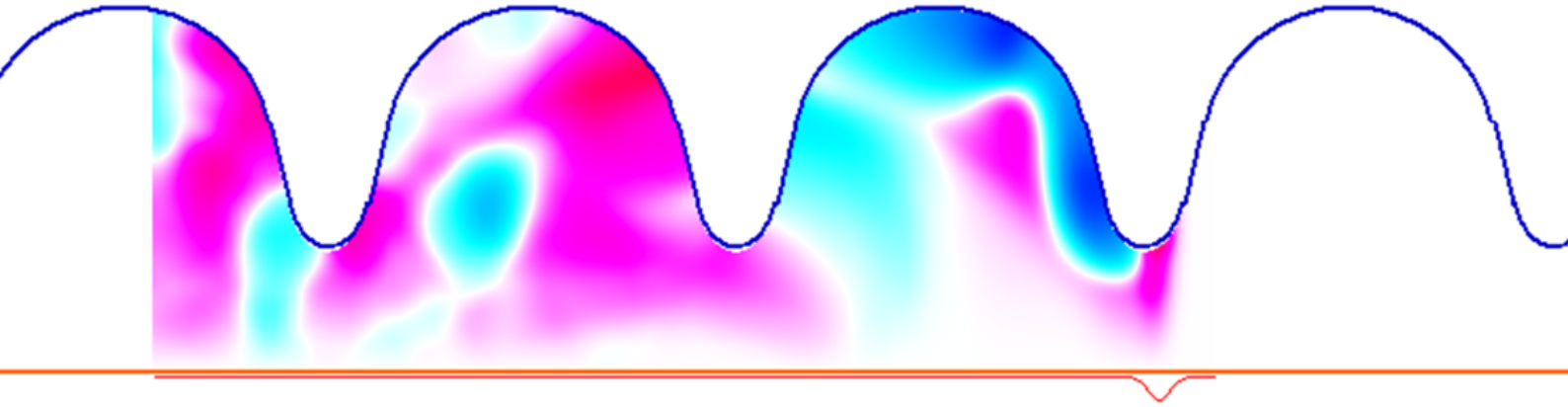


Figure 4.17: Flat taper. Field monitor.



5. ECHO3D: Three Dimensional Geometry

ECHO3D allows to calculate wakefields in three dimensional structures. The version 3.2 of the code is thread parallelized and allows to use different materials. The volume and wall conductivities are not implemented and will be available in the next releases.

5.1 Introduction

Code ECHO3D calculates in time domain the electromagnetic fields generated by an electron bunch passing through arbitrary three dimensional chamber. The structure can consist of several materials with different permeabilities and permitivities. The volume and wall conductivity model for metals are not available. The bunch form is a Gaussian pencil bunch. The arbitrary bunch form is possible but this option is not described here.

For the time being the beam can fly only along x-axis. Hence the notation below is different from the previous sections.

Let us consider a line-charge beam with vanishing transverse dimensions,

$$\begin{aligned} \rho(y_0,z_0,y,z,s) &= Q\delta(z-z_0)\delta(y-y_0)\lambda(s), \\ j_z(y_0,z_0,y,z,s) &= c\rho(y_0,z_0,y,z,s), \end{aligned} \tag{5.1}$$

where y_0, z_0, define the transverse offset of the beam, $s = x - ct$ is the local longitudinal coordinate in the bunch, Q is the bunch charge and $\lambda(s)$ is the longitudinal bunch profile [for a point charge, $\lambda(s) = \delta(s)$]. The longitudinal wake potential $W_\parallel$ at point (x,y,s) is defined as [1]

$$W_\parallel(x_0,y_0,x,y,s) = Q^{-1}\int_{-\infty}^{\infty}[E_x(x,y,z,t)]_{t=(x-s)/c}dx, \tag{5.2}$$

where the electric field on the right-hand side is the solution to Maxwell's equation with the sources of Eqs. 5.1 (this field, of course, is also a function of y_0 and z_0 omitted in the arguments of E_x for brevity).

With a knowledge of the longitudinal wake we can calculate the transverse wakes as usually.

5.2 Installation and work-flow

The program ECHO3D is compiled for Windows. It can be downloaded as archive **ECHO3D.zip** from `https://www.echo4d.de`. Extract the archive keeping the stricture of folders and files.

The archive contains the following folders.

1. **Docs**. It contains this manual.
2. **Codes**. It contains the executables: console application **ECHO3D.exe**, GUI application **ECHO3D_GUI.exe**. Additionally the folder contains three executables: for meshing (**Mesher.exe**), initial field/current creation (**InitField.exe**) and indirect wake potential calculation (**IndirectIntegration.exe**).
3. **Examples**. It contains several examples.
4. **Matlab4ECHO**. It contains Matlab functions for postprocessing.
5. **PostProcessor3D**. It contains Matlab scripts for postprocessing.

As can be seen from Fig. 5.1 we will use not one but several programs.

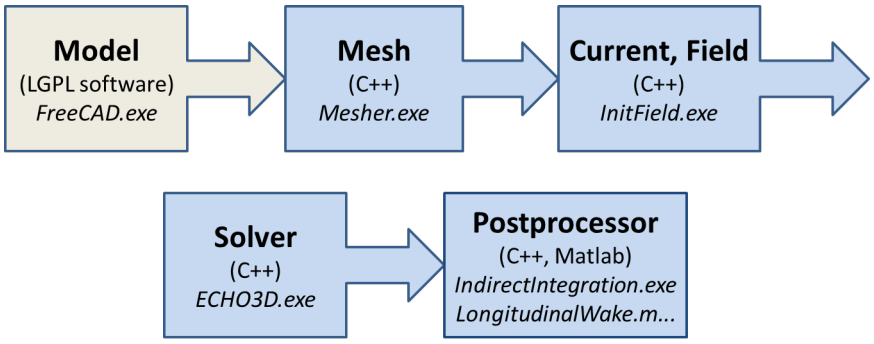


Figure 5.1: Work-flow diagram for 3D calculations.

The program package does not provide any tool for modeling of geometry. It is suggested that each material is described by file in format STL. The reader can use an arbitrary CAD program for it. In our examples we use non-commercial code **FreeCAD**. It can be installed from web site `https://www.freecadweb.org`.

5.3 Input files

The program ECHO3D requires two input files:

- a file with geometry description in ASCII format; it can have an arbitrary name,
- a file with parameters of the simulation in ASCII format; it has a fixed name **input.txt**.

The geometry itself is a collection of STL files placed in folder **Geometry**.The geometry should be created in millimeters. Additionally some special directories and files could be present as explained in the following Sections.

5.3.1 Geometry description

The geometry file is ASCII file with extension "*.txt". It has the following format:

%%%%%%%%%% Materials %%%%%%%%%%%%%%%%%%

MaterialsNumber = N_m
$Material_1$.stl 1 1 0
$Material_2$.stl ε_2 μ_2 0
...
$Material_{N_m}$.stl ε_{N_m} μ_{N_m} 0

%%%%%%%%%%% Meshing parts %%%%%%%%%%%%%%%

MeshParts = N_p
$part_1$ x_1^{min} x_1^{max}
$part_2$ x_2^{min} x_2^{max}
...
$part_{N_p}$ $x_{N_p}^{min}$ $x_{N_p}^{max}$

%%%%%%%%%%% Geometry list %%%%%%%%%%%%%%%

GeometryParts = N_g
$meshpart_1$ $iter_1$
$meshpart_2$ $iter_2$
...
$meshpart_{N_g}$ $iter_{N_g}$

The geometry file contains three sections. The first section is a list of materials from folder **Geometry**. The parameters here are:

- N_m [integer] - number of materials.
- $Material_i$.stl [string]- name of the STL file from directory **Geometry** which describes the geometry of the material number i .
- ε^i, μ^i [float]- relative permitivity, permeability of material number i.

The second section is a list of the geometry parts which will be meshed by the mesher **Mesher.exe** and placed in a new folder **Mesh**. The parameters here are:

- N_p [integer]- number of geometry parts for the meshing.
- $part_i$.stl [string]- an arbitrary but unique name of meshing part number i.
- x_i^{min}, x_i^{max} [float/mm] - the minimal and the maximal longitudinal coordinates in mm of the geometry part number i.

Finally the third section is a list of the geometry parts from the second section which compose the structure. The parameters here are:

- N_g [integer] - number of geometry parts in the list.
- $meshpart_i$.stl [string] - an arbitrary geometry part from the second section.
- $iter_i$ [integer] - the number of copies of geometry part number i in the list.

As example let us consider the geometry of dielectric pipe shown in Fig. **??**. The corresponding file will have the following content

```
%%%%%%%%%%% Materials %%%%%%%%%%%%%%%%%%%

MaterialsNumber = 2
PEC.stl 1 1 0
Dielectric.stl 11 1 0

%%%%%%%%%%% Meshing parts %%%%%%%%%%%%%%%

MeshParts = 2
pipe 1 3
diel 7 9

%%%%%%%%%%% Geometry list %%%%%%%%%%%%%%%
```

GeometryParts = 3
pipe 1
diel 50
pipe 1.

The two STL files **PEC.stl** and **Dielectric.stl** are created with program **FreeCAD** and can be found in directory **Examples/ N9_Round_Dielectric/ ECHO3D/ Geometry**.

5.3.2 Parameters of simulation

The parameters of simulation are listed in input command file with fixed name **input.txt**. This file has a following format.

%%%%%%%%%%%%%% geometry %%%%%%%%%%%%%%%%%%%%

GeometryFile = '$*$.txt'
Units = 'mm'
BoundaryConditionsX = $[BCx_0\ BCx_1]$
BoundaryConditionsY = $[BCy_0\ BCy_1]$
BoundaryConditionsZ = $[BCz_0\ BCz_1]$

%%%%%%%%%%%%%% beam %%%%%%%%%%%%%%%%%%%%

BunchSigma = σ_x
BunchPosition = $[y_0\ z_0]$

%%%%%%%%%%%%%% mesh %%%%%%%%%%%%%%%%%%%%

TimeSteps=n_t
MeshLength = N_x
dY = $[y_{min}\ y_{max}]$
dZ = $[z_{min}\ z_{max}]$
Steps = $[h_x\ h_y\ h_z]$
Tolerance = tol

%%%%%%%%%%%%%% solver %%%%%%%%%%%%%%%%%%%%

SolverType = 'adi'/'ati'
Conformal = 0/1/2
Iterations = $iter$
InitialIterations = $iter_0$
Damping = k_{damp}
ThreadsNumber = $N_{threads}$.

%%%%%%%%%%%%%% monitors %%%%%%%%%%%%%%%%%%%%

FieldMonitor= { $F\ t_F\ x_0\ x_1\ y_0\ y_1\ z_0\ z_1\ s_0\ s_1\ N$ }

The parameters in this command file are:

- **GeometryFile** [string]. Name of ASCII file with extension '*.txt'. It defines the name of file with the geometry description.
- **Units** [string]. Units of the geometry description. In current version only 'mm' can be used. Hence the geometry in the STL files should be in millimeters.
- **BoundaryConditionsX** [boolean list: BCx_0 BCx_1]. It defines boundary conditions of the global mesh domain in x-direction: '0' defines the magnetic boundary condition (tangential component of the magnetic filed is zero),'1' defines the electric boundary condition (tangential component of the electric filed is zero).
- **BoundaryConditionsY** [boolean list: BCy_0 BCy_1]. It defines boundary conditions of the global mesh domain in y-direction.
- **BoundaryConditionsZ** [boolean list: BCz_0 BCz_1]. It defines boundary conditions of the global mesh domain in z-direction.
- **BunchSigma** [float/mm] The Gaussian pencil bunch rms length σ_x in mm.
- **BunchPosition** [integer list: y_0 z_0]. It defines values of y_0, z_0 for pencil beam in mesh lines. In metric units they can be found as $y_{min} + y_0 \cdot h_y$, $z_{min} + z_0 \cdot h_y$.
- **TimeSteps** [integer]. It defines the number of time steps in the calculation. Use '-1' to fly through the whole structure.
- **MeshLength** [integer]. It defines length of the moving mesh N_x in the mesh lines. In metric units the length is $N_x \cdot h_x$.
- **dY** [float list:y_{min} y_{max} /mm]. It defines the transverse mesh dimension in y-direction in mm.
- **dZ** [float list:z_{min} z_{max} /mm]. It defines the transverse mesh dimension in z direction in mm.
- **Steps** [float list: h_x h_y h_z /mm]. It defines the mesh steps in mm.
- **Tolerance** [double]. It should be a positive value smaller than 1. The mesh facets whose material fraction is less than tol are considered as fully metallic ones. The default value is 0.01. Increase this value if any instability appears.
- **SolverType** [string]. This parameter can have two values: 'adi' - alternative-direction solver,'ati' - alternative-triangular solver. In most cases 'adi' should be used. Solver 'ati' can be used if the transverse mesh steps are smaller trhan the longitudinal one.
- **Conformal** [integer]. This parameter can have three values: '0' - staircase mesh,'1' - simple conformal method, '2' - uniformly stable conformal method. In most cases '1' should be used.
- **Iterations** [integer]. This parameter defines number of additional iterations to improve accuracy and stability of the solvers. Usually no iterations are required.
- **InitialIterations** [integer]. This parameter defines number of additional iterations to improve accuracy of the initial field. Usually no iterations are required.
- **Damping** [float]. This value could be between 0 and 0.5. Usually we do not need any damping and it should be '0'.
- **ThreadsNumber** [integer]. The parameter defines how many threads will be used. Usually it should be equal to the number of cores in your computer, but check the efficiency of parallelism experimentally.
- **FieldMonitor** [string: F string: t_F integer list: F t_F x_0 x_1 y_0 y_1 z_0 z_1 s_0 s_1 N]. It defines field monitor for the field component F: 'Ex'/'Ey'/'Ez'/'Bx'/'By'/'Bz'. Parameter t_F defines type of the monitor: 'x'/'s'. Other parameters are explained in Section 5.3.3.

5.3.3 Field monitors setups

In code ECHO3D two types of fields monitors exists: s-time and x-time.

The field monitor is described by line **FieldMonitor =**{ F t_F x_0 x_1 y_0 y_1 z_0 z_1 s_0 s_1 N}. Here F defines the filed component: 'Ex'/'Ey'/'Ez'/'Bx'/'By'/'Bz'. The second parameter t_f defines the type of the field monitor: 's'/'x'. The parameters y_0, y_1, z_0, z_1 define the transverse interval in

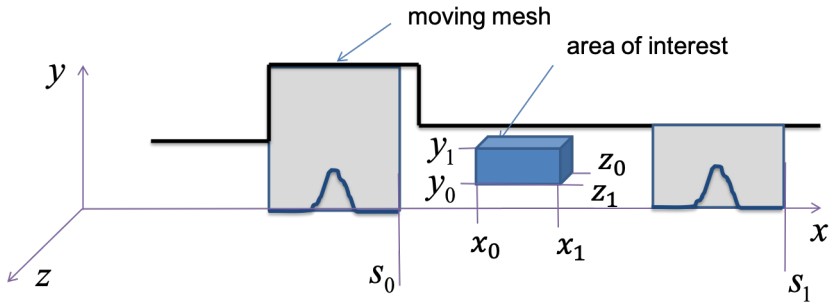


Figure 5.2: Field monitor of type s-time.

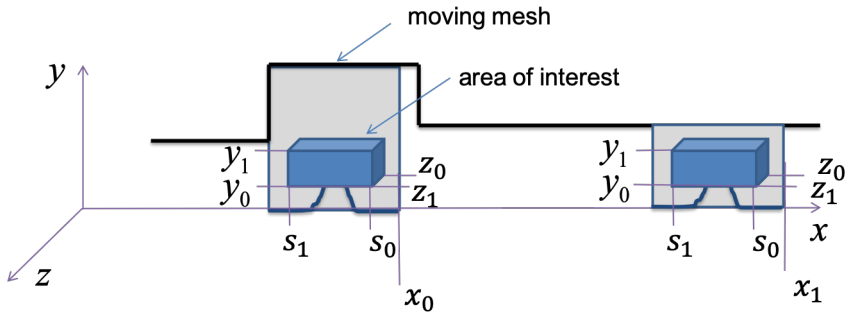


Figure 5.3: Field monitor of type x-time.

meters in which field is saved (see Figs. 5.2, 5.3). The last parameter N defines sampling interval in timesteps $h_t = h_z/c$, where c is the light velocity.

The s-time monitor is a static monitor. The window is defined as a static cuboid in the calculation domain with longitudinal coordinates x_0, x_1 in meters. The filed is saved from time $t_0 = s_0/c$ to time $t_1 = s_1/c$ with interval $h_z/c/N$ .The principle of s-time monitor is explained in Fig. 5.2.

The z-time monitor is a moving monitor. The window is defined as moving cuboid in the moving mesh with longitudinal coordinates s_0, s_1 in meters. The filed is saved from time $t_0 = x_0/c$ to time $t_1 = x_1/c$ with interval $h_z/c/N$ .The principle of z-time monitor is explained in Fig. 5.3.

The output formats and postprocessing are described below. An example can be found in the directory **Examples/ N8_FlatTaperWithFieldMonitor**.

5.4 Wakefield Calculation

The local folder should contain three files:

- geometry file,
- command file **input.txt**,
- command file **run.bat**, which uses **ECHO3D.exe** or command file **run_GUI.bat**, which uses **ECHO3D_GUI.exe**.

The geometry itself as a collection of STL files should be placed in folder **Geometry**.

The calculations starts by execution of **run.bat**. During the simulation the progress is shown. The script starts the programms in the following order:

- **Mesher.exe** creates directory **Mesh** and subdirectories with parts of the geometry meshed,
- **InitField.exe** creates directories **Bunch** and **Fields** and places their the corresponding files,
- **ECHO3D.exe** creates directory **Results**, makes the wakefield calculation and saves the results, **IndirectIntegration.exe** calculates wake potential taking into account field propaga-

tion in the outgoing waveguide.

5.5 Output files

After execution of **run.bat** the folder **Results** is created. It contains file **Wake3Dindirect.bin** in binary format which should be post-processed with the matlab scripts from directory **PostProcessor3D**.

If field monitors had been setup in file **input.txt** then they are saved in the same directory in ASCII files with name pattern **Monitor_NXX.txt**, where XX is the ordinal number of the monitor.

The s-type monitor file has the following format:

% Field=F time=s
% k_ct=k_{ct} h_ct=h_{ct} ct0=s_0
% k_y=k_y h_y=h_y y0=y_0
% k_z=k_z h_z=h_z z0=z_0
% k_x=k_x h_x=h_x x0=x_0
s_0
$F(x_0,y_0,z_0)$ $F(x_0+h_x,y_0,z_0)$ ... $F(x_0+k_x*h_x,y_0,z_0)$
$F(x_0,y_0+h_y,z_0)$ $F(x_0+h_x,y_0+h_y,z_0)$ ... $F(x_0+k_x*h_x,y_0+h_y,z_0)$
...
$F(x_0,y_0+k_y*h_y,z_0)$ $F(x_0+h_x,y_0+k_y*h_y,z_0)$ ... $F(x_0+k_x*h_x,y_0+k_y*h_y,z_0)$
$F(x_0,y_0,z_0+h_z)$ $F(x_0+h_x,y_0,z_0+h_z)$ ... $F(x_0+k_x*h_x,y_0,z_0+h_z)$
$F(x_0,y_0+h_y,z_0+h_z)$ $F(x_0+h_x,y_0+h_y,z_0+h_z)$ ... $F(x_0+k_x*h_x,y_0+h_y,z_0+h_z)$
...
$F(x_0,y_0+k_y*h_y,z_0+k_z*h_z)$ $F(x_0+h_x,y_0+k_y*h_y,z_0+k_z*h_z)$ ... $F(x_0+k_x*h_x,y_0+k_y*h_y,z_0+k_z*h_z)$
s_0+h_{ct}
...

The x-type monitor file has the following format:

% Field=F time=s
% k_ct=k_{ct} h_ct=h_{ct} ct0=z_0
% k_y=k_y h_y=h_y y0=y_0
% k_z=k_z h_z=h_y z0=z_0
% k_s=k_s h_s=h_z s0=s_0
x_0
$F(s_0,y_0,z_0)$ $F(s_0+h_s,y_0,z_0)$ ... $F(s_0+k_s*h_s,y_0,z_0)$
$F(s_0,y_0+h_y,z_0)$ $F(s_0+h_s,y_0+h_y,z_0)$ ... $F(s_0+k_s*h_s,y_0+h_y,z_0)$
...
$F(s_0,y_0+k_y*h_y,z_0)$ $F(s_0+h_s,y_0+k_y*h_y,z_0)$ ... $F(s_0+k_s*h_s,y_0+k_y*h_y,z_0)$
$F(s_0,y_0,z_0+h_z)$ $F(s_0+h_s,y_0,z_0+h_z)$ ... $F(s_0+k_s*h_s,y_0,z_0+h_z)$
$F(s_0,y_0+h_y,z_0+h_z)$ $F(s_0+h_s,y_0+h_y,z_0+h_z)$ ... $F(s_0+k_s*h_s,y_0+h_y,z_0+h_z)$
...
$F(s_0,y_0+k_y*h_y,z_0+k_z*h_z)$ $F(s_0+s_x,y_0+k_y*h_y,z_0+k_z*h_z)$ ... $F(s_0+k_s*h_s,y_0+k_y*h_y,z_0+k_z*h_z)$
x_0+h_{ct}
...

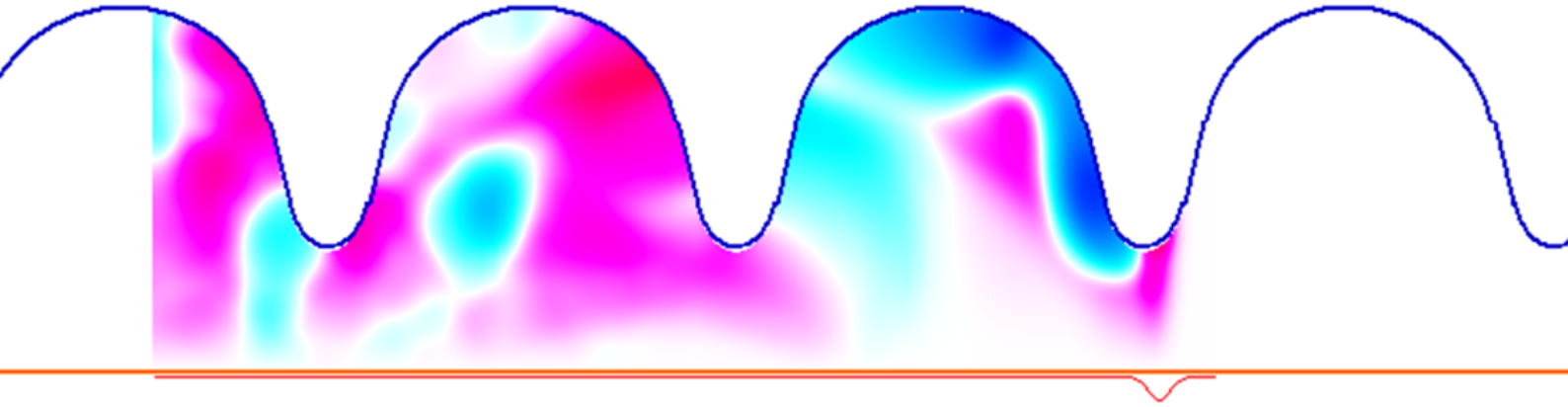
The electric field components E_x, E_y, E_z are saved in V/m/nC. Magnetic field components are saved multiplied by velocity of light c as cB and hence the units are the same as for the electric field components.

5.6 Postprocessing

The folder PostProcessor3D contains several scripts. Their usage is clarified in the examples.

5.7 Examples

The example manual can be found in directory **Docs**.



6. ECHO1D: Anisotropic Waveguides

6.1 Introduction

Code ECHO1D calculates in frequency domain the electromagnetic fields generated by an electron bunch passing through an anisotropic transversally non-homogeneous vacuum chamber of round or rectangular cross-section with translational symmetry in the beam direction [5].

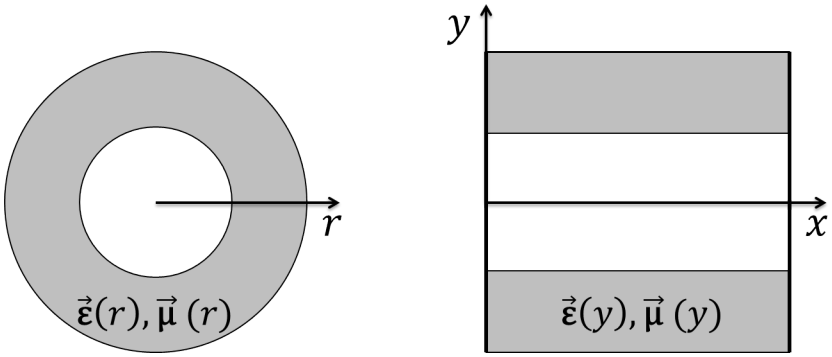


Figure 6.1: Examples of "round" and "rectangular" geometry.

We consider a point-charge q moving with constant velocity v through a structure with round or rectangular cross-section. In the following we call the structure "round" if it is axially symmetric. If the structure has a constant width between two perfectly conducting planes and has rectangular cross-sections then we call such structure "rectangular". Fig. 6.1 shows examples of round and rectangular structures. In the following we consider only an anisotropic materials with diagonal material permittivity and permeability tensors, where the optical axes coincide with coordinate ones. Hence their diagonals are given by complex vectors $\vec{\varepsilon}, \vec{\mu}$.

We assume that the charge is moving along a straight line parallel to the longitudinal axis of the system, and we neglect the influence of the wakefields on the charge motion. For round structures we will use cylindrical coordinates r, φ, z. The charge density in the frequency domain can be

expanded in Fourier series

$$\rho(r,\varphi,z,k) = e^{-ikz/\beta}\sum_{m=0}^{\infty}\rho_m(r)\cos(m(\varphi-\varphi_0)), \quad \rho_m(r) = \frac{q\delta(r-r_0)}{\pi v r_0(1+\delta_{m0})}, \tag{6.1}$$

where r_0, φ_0 are coordinates of the point charge q, $\beta = v/c$, c is velocity of light in vacuum, and $\delta_{m0} = 1$ if $m = 1$, 0 otherwise.

From the linearity of Maxwell's equations the components of the electromagnetic field can be represented by infinite sums:

$$\begin{pmatrix} H_\varphi(r,\varphi,z,k) \\ E_r(r,\varphi,z,k) \\ E_z(r,\varphi,z,k) \end{pmatrix} = e^{-ikz/\beta}\sum_{m=0}^{\infty}\begin{pmatrix} H_{\varphi,m}(r,k) \\ E_{r,m}(r,k) \\ E_{z,m}(r,k) \end{pmatrix}\sin(m\varphi),$$
$$\begin{pmatrix} E_\varphi(r,\varphi,z,k) \\ H_r(r,\varphi,z,k) \\ H_z(r,\varphi,z,k) \end{pmatrix} = e^{-ikz/\beta}\sum_{m=0}^{\infty}\begin{pmatrix} E_{\varphi,m}(r,k) \\ H_{r,m}(r,k) \\ H_{z,m}(r,k) \end{pmatrix}\cos(m\varphi). \tag{6.2}$$

The electric displacement $\vec{D}$ and the magnetic induction $\vec{B}$ are defined using complex permittivity and permeability diagonal tensors

$$\vec{D} = \begin{pmatrix} \varepsilon_r(r,k) & 0 & 0 \\ 0 & \varepsilon_\varphi(r,k) & 0 \\ 0 & 0 & \varepsilon_z(r,k) \end{pmatrix}\vec{E}, \quad \vec{B} = \begin{pmatrix} \mu_r(r,k) & 0 & 0 \\ 0 & \mu_\varphi(r,k) & 0 \\ 0 & 0 & \mu_z(r,k) \end{pmatrix}\vec{H}.$$

We do not have to assume any particular frequency dependence. In order to include conductivity and other losses in our code ECHO1D we use the following expressions (here we consider as example r-component):

$$\varepsilon_r(r,k) = \varepsilon_0\hat{\varepsilon}_r(r,k) + i\frac{\kappa_r(r)}{\omega(1+i\omega\tau_r(r))}, \quad \mu_r(r,k) = \mu_0\hat{\mu}_r(r,k), \quad \omega = kc,$$

where ε_0, μ_0 are permettivity and permeability of vacuum, and the loss can be introduced with the help of dielectric loss tangent $\delta_r^\varepsilon = \frac{\Im\hat{\varepsilon}_r}{\Re\hat{\varepsilon}_r}$, magnetic loss tangent $\delta_r^\mu = \frac{\Im\hat{\mu}_r}{\Re\hat{\mu}_r}$ or/and with AC conductivity following the Drude model [2], where κ_r is the DC conductivity of the material and τ_r its relaxation time. We use similar expressions for φ- and z- components of the permittivity and the permeability tensors.

For each mode number m we can write an independent system of equations

$$\begin{aligned}
&\frac{m}{r}H_{z,m} + i\frac{k}{\beta}H_{\varphi,m} = i\omega\varepsilon_r E_{r,m},\\
&-i\frac{k}{\beta}H_{r,m} - \frac{\partial}{\partial r}H_{z,m} = i\omega\varepsilon_\varphi E_{\varphi,m},\\
&\frac{1}{r}\frac{\partial}{\partial r}(rH_{\varphi,m}) - \frac{m}{r}H_{r,m} = i\omega\varepsilon_z E_{z,m} + v\rho_m,\\
&-\frac{m}{r}E_{z,m} + i\frac{k}{\beta}E_{\varphi,m} = -i\omega\mu_r H_{r,m},\\
&-i\frac{k}{\beta}E_{r,m} - \frac{\partial}{\partial r}E_{z,m} = -i\omega\mu_\varphi H_{\varphi,m},\\
&\frac{1}{r}\frac{\partial}{\partial r}(rE_{\varphi,m}) + \frac{m}{r}E_{r,m} = -i\omega\mu_z H_{z,m},\\
&\frac{1}{r}\frac{\partial}{\partial r}(rH_{r,m}\mu_r) - \frac{m}{r}H_{\varphi,m}\mu_\varphi - ikH_{z,m}\mu_z = 0,\\
&\frac{1}{r}\frac{\partial}{\partial r}(rE_{r,m}\varepsilon_r) + \frac{m}{r}E_{\varphi,m}\varepsilon_\varphi - ikE_{z,m}\varepsilon_z = \rho_m.
\end{aligned} \tag{6.3}$$

We have reduced the initial three-dimensional problem to an infinite set of independent dimensional problems, Eqs. (6.3), for the Fourier componets of the field.

In rectangular case we choose a coordinate system with y in the vertical and x in the horizontal directions; the z coordinate is directed along the beam direction. The structures considered in this paper have constant width $2w$ in x-direction between two perfectly conducting side walls.

The charge density can be expanded in Fourier series

$$\rho(x,y,z,k) = \frac{e^{-ikz/\beta}}{w}\sum_{m=1}^{\infty}\rho_m(y)\sin(k_{x,m}x_0)\sin(k_{x,m}x), \quad k_{x,m} = \frac{\pi m}{2w}, \quad \rho_m(y) = \frac{q\delta(y-y_0)}{v},$$

where x_0, y_0 are coordinates of the point charge. Again it follows from the linearity of Maxwell's equations that the components of electromagnetic field can be represented by infinite sums:

$$\begin{pmatrix} H_x(x,y,z,k) \\ E_y(x,y,z,k) \\ E_z(x,y,z,k) \end{pmatrix} = \frac{e^{-ikz/\beta}}{w}\sum_{m=1}^{\infty}\begin{pmatrix} H_{x,m}(y,k) \\ E_{y,m}(y,k) \\ E_{z,m}(y,k) \end{pmatrix}\sin(k_{x,m}x),$$
$$\begin{pmatrix} E_x(x,y,z,k) \\ H_y(x,y,z,k) \\ H_z(x,y,z,k) \end{pmatrix} = \frac{e^{-ikz/\beta}}{w}\sum_{m=1}^{\infty}\begin{pmatrix} E_{x,m}(y,k) \\ H_{y,m}(y,k) \\ H_{z,m}(y,k) \end{pmatrix}\cos(k_{x,m}x).$$

For each mode number m we can write an independent system of equations

$$\begin{aligned}
&-k_{x,m}H_{z,m} + i\frac{k}{\beta}H_{x,m} = i\omega\varepsilon_y E_{y,m},\\
&-i\frac{k}{\beta}H_{y,m} - \frac{\partial}{\partial y}H_{z,m} = i\omega\varepsilon_x E_{x,m},\\
&\frac{\partial}{\partial y}H_{x,m} + k_{x,m}H_{y,m} = i\omega\varepsilon_z E_{z,m} + v\rho_m,\\
&k_{x,m}E_{z,m} + i\frac{k}{\beta}E_{x,m} = -i\omega\mu_y H_{y,m},\\
&-i\frac{k}{\beta}E_{y,m} - \frac{\partial}{\partial y}E_{z,m} = -i\omega\mu_x H_{x,m},\\
&\frac{\partial}{\partial y}(E_{x,m}) - k_{x,m}E_{y,m} = -i\omega\mu_z H_{z,m},\\
&\frac{\partial}{\partial y}(H_{y,m}\mu_y) + k_{x,m}H_{x,m}\mu_x - ikH_{z,m}\mu_z = 0,\\
&\frac{\partial}{\partial y}(E_{y,m}\varepsilon_y) - k_{x,m}E_{x,m}\varepsilon_x - ikE_{z,m}\varepsilon_z = \rho_m.
\end{aligned} \tag{6.4}$$

We are interested in coupling impedances as defined in [1, 3]. For round pipe the coupling impedance can be written as

$$Z_{\parallel}(r_0,\varphi_0,r,\varphi,k,\gamma) = \sum_{m=0}^{\infty} Z_m(k,\gamma)I_m\left(\frac{kr_0}{\gamma\beta}\right)I_m\left(\frac{kr}{\gamma\beta}\right)\cos(m(\varphi-\varphi_0)) + Z_{sc}(r_0,\varphi_0,r,\varphi,k,\gamma),$$
$$Z_{sc}(r_0,\varphi_0,r,\varphi,k,\gamma) = -\frac{kZ_0}{2\pi(\gamma^2-1)}K_0\left(\frac{k\sqrt{r_0^2+r^2-2r_0 r cos(\varphi-\varphi_0)}}{\gamma\beta}\right), \tag{6.5}$$

where γ is the relative relativistic energy and we have written explicitly the space charge contribution Z_{sc}.

For a rectangular pipe the impedance reads

$$Z_{\|}(x_0,y_0,x,y,k) = \frac{1}{w}\sum_{m=1}^{\infty} Z_m(y_0,y,k,\gamma)\sin(k_{x,m}x_0)\sin(k_{x,m}x) + Z_{sc}(x_0,y_0,x,y,k,\gamma),$$

$$Z_{sc}(x_0,y_0,x,y,k,\gamma) = -\frac{kZ_0}{2\pi(\gamma^2-1)}K_0\left(\frac{k\sqrt{(x-x_0)^2+(y-y_0)^2}}{\gamma\beta}\right), \tag{6.6}$$

where

$$Z_m(y_0,y,k,\gamma) = [Z_m^{cc}(k,\gamma)\cosh(k_{y,m}y_0) + Z_m^{sc}(k,\gamma)\sinh(k_{y,m}y_0)]\cosh(k_{y,m}y)$$
$$+ [Z_m^{cs}(k,\gamma)\cosh(k_{y,m}y_0) + Z_m^{ss}(k,\gamma)\sinh(k_{y,m}y_0)]\sinh(k_{y,m}y),$$
$$k_{y,m} = \sqrt{k_{x,m}^2 + \frac{k^2}{\gamma^2\beta^2}}.$$

In Eqs.(6.5, 6.6) the infinite sum defines a so-called wall impedance. The longitudinal and the transverse impedances are connected by Panofsky-Wentzel theorem (see [3] for a detailed discussion):

$$\vec{Z}_{\perp} = \frac{\beta}{k}\nabla Z_{\|}, \tag{6.7}$$

where the gradient is taken on coordinates of the witness particle.

The wake field effect in time domain is described by a longitudinal wake function which can be obtained by the Fourier transform of the longitudinal impedance

$$w_{\|}(s) = \frac{c}{2\pi}\int_{-\infty}^{\infty} Z_{\|}(k)e^{iks/\beta}dk,$$

where s is the distance between the source and the test particles [1].

6.2 Installation

The program ECHO1D can be downloaded as archive **ECHO1D.zip** from `https://www.echo4d.de`. Extract the archive keeping the stricture of folders and files.

The archive contains the following folders.

1. **Docs**. It contains this manual.
2. **Codes**. It contains executables **ECHO1D.exe** and **ECHO2D_GUI.exe**.
3. **Examples**. It contains several examples.
4. **MatLib4ECHO**. It contains Matlab functions for postprocessing.
5. **PostProcessor1D**. It contains Matlab scripts for postprocessing.

6.3 Input files

The program ECHO1D requires two input files:

- a file with geometry description in ASCII format; it can have an arbitrary name,
- a file with parameters of the simulation in ASCII format; it has a fixed name **input_in.txt**.

6.3.1 Geometry description

The geometry file describes the layered structure shown in Fig. 6.2. It is a text file with arbitrary name. In the examples considered below the geometry files have names with pattern **ExampleXX.txt**, where XX is the example number.

The geometry file has the following format.

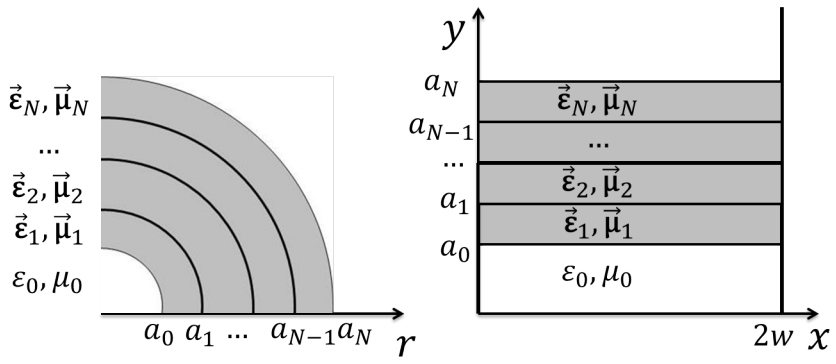


Figure 6.2: Examples of "round" and "rectangular" layered geometry.

% N – Number of layers
N
% boundaries
a_0 a_1 ... a_{N-1} a_N
% Re(EpsR[i]) Im(EpsR[i]) Re(EpsFi[i]) Im(EpsFI[i]) Re(EpsZ[i]) Im(EpsZ[i]), i=1,N
$\Re\hat{\varepsilon}_{r,1}$ $\Im\hat{\varepsilon}_{r,1}$ $\Re\hat{\varepsilon}_{\varphi,1}$ $\Im\hat{\varepsilon}_{\varphi,1}$ $\Re\hat{\varepsilon}_{z,1}$ $\Im\hat{\varepsilon}_{z,1}$
$\Re\hat{\varepsilon}_{r,2}$ $\Im\hat{\varepsilon}_{r,2}$ $\Re\hat{\varepsilon}_{\varphi,2}$ $\Im\hat{\varepsilon}_{\varphi,2}$ $\Re\hat{\varepsilon}_{z,2}$ $\Im\hat{\varepsilon}_{z,2}$
...
$\Re\hat{\varepsilon}_{r,N}$ $\Im\hat{\varepsilon}_{r,N}$ $\Re\hat{\varepsilon}_{\varphi,N}$ $\Im\hat{\varepsilon}_{\varphi,N}$ $\Re\hat{\varepsilon}_{z,N}$ $\Im\hat{\varepsilon}_{z,N}$
% Re(MueR[i]) Im(MueR[i]) Re(MueFi[i]) Im(MueFI[i]) Re(MueZ[i]) Im(MueZ[i]), i=1,N
$\Re\hat{\mu}_{r,1}$ $\Im\hat{\mu}_{r,1}$ $\Re\hat{\mu}_{\varphi,1}$ $\Im\hat{\mu}_{\varphi,1}$ $\Re\hat{\mu}_{z,1}$ $\Im\hat{\mu}_{z,1}$
$\Re\hat{\mu}_{r,2}$ $\Im\hat{\mu}_{r,2}$ $\Re\hat{\mu}_{\varphi,2}$ $\Im\hat{\mu}_{\varphi,2}$ $\Re\hat{\mu}_{z,2}$ $\Im\hat{\mu}_{z,2}$
...
$\Re\hat{\mu}_{r,N}$ $\Im\hat{\mu}_{r,N}$ $\Re\hat{\mu}_{\varphi,N}$ $\Im\hat{\mu}_{\varphi,N}$ $\Re\hat{\mu}_{z,N}$ $\Im\hat{\mu}_{z,N}$
% Conductivity[i], Relaxation Time[i], i=1,N
κ_1 τ_1
κ_2 τ_2
...
κ_N τ_N

In this listing the strings which begin with % are comments. For rectangular geometry the format is the same with replacing $r \to y, \varphi \to x$.

6.3.2 Parameters of simulation

The parameters of simulation are listed in input command file with fixed name **input_in.txt**. This file has a following format.

%%%%%%%%%%%%%% geometry %%%%%%%%%%%%%%%%%%%%
Geometry_File = ExampleXX.txt
Boundary_Condition = Open/PEC
Geometry_Width = W

%%%%%%%%%%%%%% beam %%%%%%%%%%%%%%%%%%%%
Gamma = γ

%%%%%%%%%%%%%% Model %%%%%%%%%%%%%%%%%%%%
Method = FM/FD/Mix

Steps_on_Wavelength = N_λ

%%%%%%%%%%%%%% output %%%%%%%%%%%%%%%%%%%%
Modes = m_0 m_1 ... m_{N_m}
Wavenumbers = k_{min} k_{max} Δk

The parameters in this command file are:

- **Geometry_File** [string]. It defines the name of file with the geometry description.
- **Boundary_Condition** [string]. It defines the boundary condition at a_N (see Fig. 6.2). The boundary condition could be 'PEC' or 'Open'. 'PEC' means perfectly electrically conducting material. 'Open' can be used if the last material with parameters $\vec{\varepsilon}_N, \vec{\mu}_N$ is infinite and has uniaxial anisotropy: $\varepsilon_{r,N} = \varepsilon_{\varphi,N}, \mu_{r,N} = \mu_{\varphi,N}$.
- **Geometry_Width** [float/m]. It could be '0' or a positive number $W > 0$. '0' defines rotationally symmetric geometry. A positive W defines in meters the width of the rectangular structure in x direction.
- **Gamma** [float/m]. It defines the relative energy $\gamma = \frac{E}{mc^2}$ of the charged particle.
- **Method** [string]. It could be 'FM', 'FD' or 'Mix'. 'FM' defines the field matching method and can be used if the all materials have the uniaxial anisotropy: $\varepsilon_{r,i} = \varepsilon_{\varphi,i}, \mu_{r,i} = \mu_{\varphi,i}, i = 1,..,N$. 'FD' defines finite-difference method and can be used for full anisotropy. 'Mix' defines a mixed method which should be used if the anisotropic layers are thin.
- **Steps_on_Wavelength** [integer]. It defines the number of mesh lines N_λ on wavelength in vacuum. This parameter has no impact on field matching method (Method=FM).
- **Modes =** m_0 m_1 **...** m_{N_m} [integer list]. It defines the modes which are calculated.
- **Wavenumbers =** k_{min} k_{max} Δk [float list]. The impedance is calculated from k_{min} to k_{max} with step Δk. The units are 1/meter.

6.4 Impedance Calculation

The local folder should contain three files:

- geometry file,
- command file **input_in.txt**,
- command file **run.bat**, which starts **ECHO1D.exe**.

The calculations starts by execution of **run.bat**. During the simulation the progress in percents is shown. All modes are calculated in parallel.

6.5 Output files

After execution of **ECHO1D.exe** the folder will contain N_m files with modal impedances. They have name pattern Impedance_MXXX.txt, where XXX is the mode number m_i. Each file is text file with four columns. The contents of the files is different for round and rectangular geometry

For round geometry each file contains "longitudinal" and "transverse wakes" for each mode.

```
% ECHO1D output
%k[m^-1] Re(Zlong)[Omm/m] Im(Zlong)[Omm/m]
%Re(Ztrans)[Omm/m] Im(Ztrans)[Omm/m]
1.0000000e+00 2.6112007e+11 5.9538041e+12 1.3056010e+05 2.9769036e+06
```

Here $Zlong = Z_m(k,\gamma)$ from Eq.(6.5), and $Ztrans = \frac{Z_m k}{2\gamma^2\beta}$ is an auxiliary scaled function for non-relativistic case.

For rectangular geometry each file contains Z_{cc} and Z_{ss} modal impedances from Eq.(**??**).

```
% ECHO1D output
% k[m^-1] Re(Zcc)[Omm/m] Im(Zcc)[Omm/m]
% Re(Zss)[Omm/m] Im(Zss)[Omm/m]
1.0000000e+00 3.5376930e-03 7.7756244e-02 2.1169888e-02 3.1981001e-01
```

6.6 Postprocessing

The folder PostProcessor1D contains two subfolders:

- round,
- flat.

6.6.1 Impedances

For round structure the longitudinal wall impedance of beam near the axis can be approximated as

$$Z_{\parallel}^{wall}(r_0,\varphi_0,r,\varphi,k,\gamma) \approx Z_{long}(k,\gamma) = Z_0(k,\gamma). \tag{6.8}$$

The transverse wall impedance near the axis can be approximated as

$$Z_r^{wall}(r_0,\varphi_0,r,\varphi,k,\gamma) \approx Z_{dip}(k,\gamma) r_0 cos(\varphi_0-\varphi) + Z_{quad}(k,\gamma) r, \tag{6.9}$$

$$Z_{\varphi}^{wall}(r_0,\varphi_0,r,\varphi,k,\gamma) \approx Z_{dip}(k,\gamma) r_0 sin(\varphi_0-\varphi), \tag{6.10}$$

where

$$Z_{dip}(k,\gamma) = Z_1(k,\gamma)\frac{k}{4\beta\gamma^2}, \qquad Z_{quad}(k,\gamma) = Z_0(k,\gamma)\frac{k}{2\beta\gamma^2}. \tag{6.11}$$

The matlab script Impedance_round.m plots graphically the terms $Z_{long}(k,\gamma)$, $Z_{dip}(k,\gamma)$ and $Z_{quad}(k,\gamma)$. Additionally it saves these terms in file **ImpedanceLQD.txt**.

For rectangular structure the longitudinal wall impedance of beam near the axis can be approximated as

$$Z_{\parallel}^{wall}(x_0,y_0,x,y,k,\gamma) \approx Z_{long}(k,\gamma) = \frac{1}{w}\sum_{m=1}^{\infty} Z_{2m-1}^{cc}(k,\gamma). \tag{6.12}$$

The transverse wall impedance near the axis can be approximated as

$$Z_y^{wall}(x_0,y_0,x,y,k,\gamma) \approx Z_{dip}^{y}(k,\gamma) y_0 + Z_{quad}^{y}(k,\gamma) y, \tag{6.13}$$

$$Z_x^{wall}(x_0,y_0,x,y,k,\gamma) \approx Z_{dip}^{x}(k,\gamma) x_0 - Z_{quad}^{x}(k,\gamma) x, \tag{6.14}$$

where

$$Z_{dip}^{y}(k,\gamma) = \frac{\beta}{kw}\sum_{m=1}^{\infty} k_{y,2m-1}^2 Z_{2m-1}^{cc}(k,\gamma), \tag{6.15}$$

$$Z_{quad}^{y}(k,\gamma) = \frac{\beta}{kw}\sum_{m=1}^{\infty} k_{y,2m-1}^2 Z_{2m-1}^{ss}(k,\gamma), \tag{6.16}$$

$$Z_{dip}^{x}(k,\gamma) = \frac{\beta}{kw}\sum_{m=1}^{\infty} k_{x,2m}^2 Z_{2m}^{cc}(k,\gamma), \tag{6.17}$$

$$Z_{quad}^{x}(k,\gamma) = \frac{\beta}{kw}\sum_{m=1}^{\infty} k_{x,2m-1}^2 Z_{2m-1}^{ss}(k,\gamma). \tag{6.18}$$

The matlab script Impedance_flat.m plots graphically the terms $Z_{long}(k,\gamma)$, $Z_{dip}^{y}(k,\gamma)$ and $Z_{quad}^{y}(k,\gamma)$. Additionally it saves these terms in file **ImpedanceLQD.txt**.

6.6.2 Wakes

The matlab scripts **Wake_round.m** and **Wake_flat.m** plot graphically the corresponding wake potentials for a Gaussian bunch with rms width defined at the beginning of the scripts by line *sigma*=.... The wakes are saved in file **wakeLQD.txt**.

6.7 Examples

In this section we consider several examples included in the archive at the directory **Examples**.

6.7.1 Example 1: Round dielectric pipe

The first example can be found in directory **Examples/ N1_Round_Dielectric**. We consider a dielectric pipe with interrior radius a_0=5mm, exterior radius a_1=10mm with relative permeability $\hat{\varepsilon} = 11$. The pipe is closed by perfectly conducting metal. The geometry is shown in Fig. 6.3.

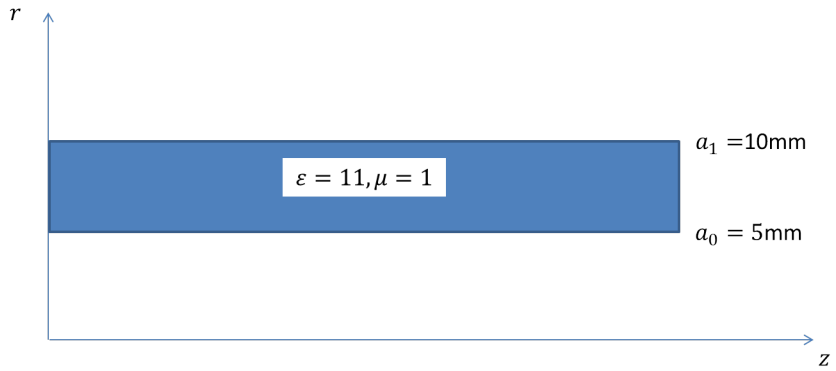


Figure 6.3: The geometry of round dielectric pipe inside of perfectly conducting pipe.

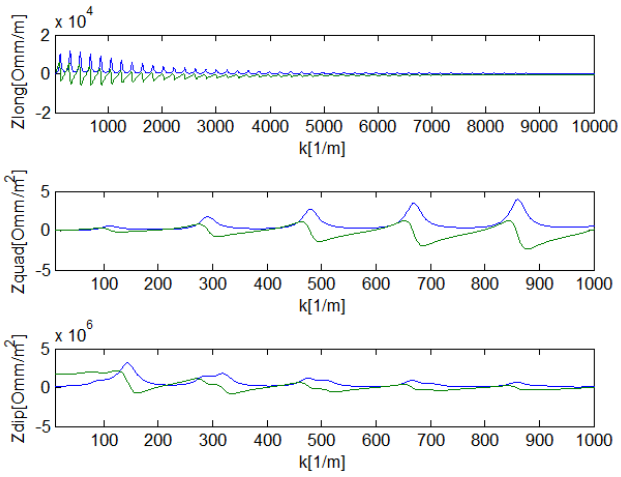


Figure 6.4: Impedances of the round dielectric pipe. The real part is shown in blue. The imaginary part is presented by the green curve.

In order to model the perfectly conducting material we set *Boundary_Condition* = *PEC* in the command file **input_in.txt**. The geometry is isotropic and we set *Method* = *FM* to choose the fastest method: field matching. For beam near to the axis we calculate only two lowest modes of

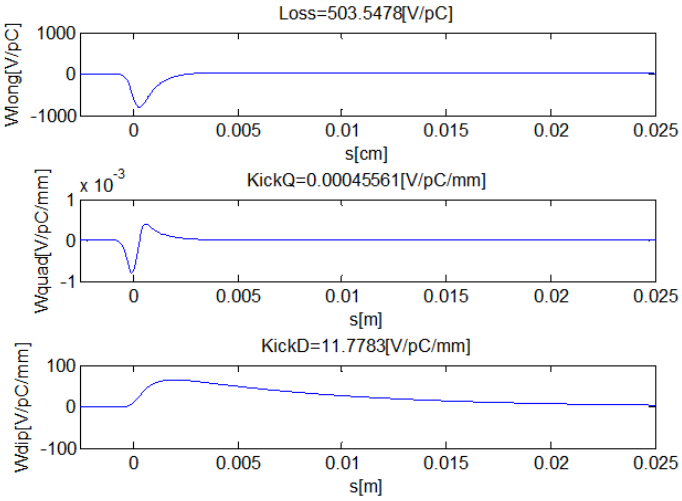


Figure 6.5: Wake potentials of the Gaussian bunch with rms width 0.25 mm in the round dielectric pipe.

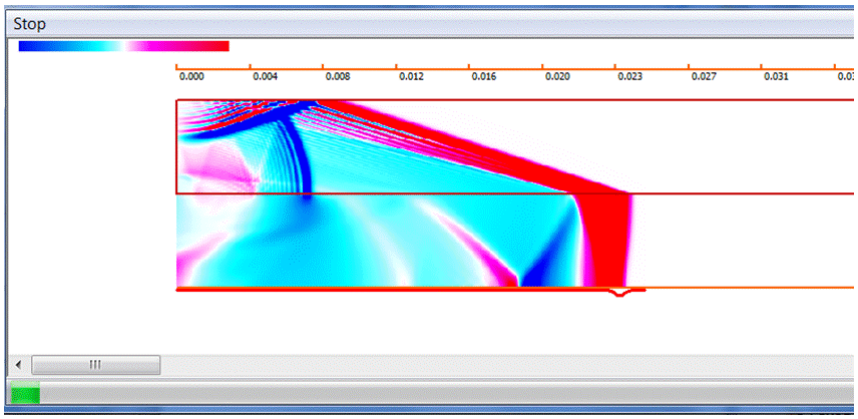


Figure 6.6: Windows GUI interface of ECHO2D code shows the E_z component of the field in time-domain in the round dielectric pipe.

the azimuthal expansion: *Modes = 0 1*. Without losses the real part of impedance is a sum of delta-functions. It can be reconstructed from imaginary part of the impedance. However we are interested here only in short range wakes and use a simpler approach: we introduce a small conductivity 1 S/m in the last row of geometry file **Example01.txt**. The obtained impedances can be seen with the matlab script **N1_Round_Dielectric/ PostProcessor1D/ round/ Impedance_round.m**. They are shown in Fig. 6.4.

We are looking for short range longitudinal and transverse wake potentials for a Gaussian bunch with rms width 0.25mm. They can be obtained with matlab script **N1_Round_Dielectric/ Postprocessor1D/ round/ Wake_round.m** and are shown in Fig. 6.5.

The results are cross-checked with ECHO2D. The simulations are done for 1 meters and 1.1 meters and subtracted to obtain the "steady-state" wake. The setup to run code ECHO2D can be found in folders **ECHO2D** and **PostProcessor2D**. The instructions how to use code ECHO2D are given in corresponding section of this manual. Fig. 6.6 presents the GUI interface during time-domain calculations with ECHO2D.

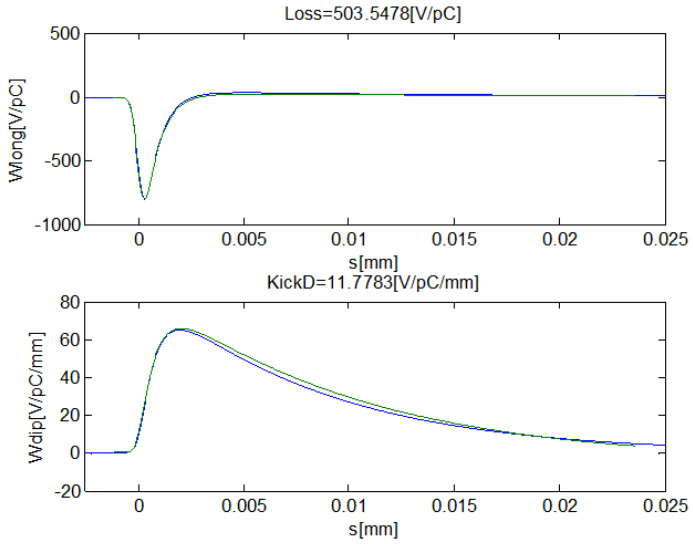


Figure 6.7: Comparison the wake potentials of the round dielectric pipe obtained by ECHO1D (blue curves) with the ones obtained by ECHO2D (green curves)

.

The comparison of the results from ECHO1D with ECHO2D can be seen by running the script **Compare_2D_vs_1D.m** in Matlab. The result is shown in Fig. 6.7.

6.7.2 Example 2: Flat dielectric pipe

The second example can be found in directory **Examples/ N2_Flat_Dielectric**. We consider a rectangular perfectly conducting pipe of width 160 mm and half hight a_1=10mm. The pipe has a dielectric layer vertically from a_0=5mm to a_1=10mm with relative permeability $\hat{\varepsilon} = 11$. The geometry is shown in Fig. 6.8.

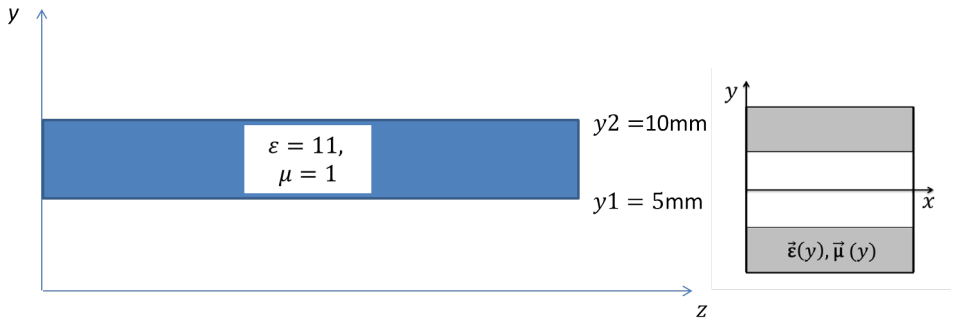


Figure 6.8: The geometry of flat dielectric pipe inside of perfectly conducting pipe.

In order to model the perfectly conducting material we set *Boundary_Condition = PEC* in the command file **input_in.txt**. The geometry is isotropic and we set *Method = FM* to choose the fastest method: field matching. For beam near to the axis we calculate only odd modes: *Modes = 1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39 41 43 45 47 49 51 53 55 57 59*. Again we introduce a small conductivity 1 S/m in the last row of geometry file **Example02.txt**. The obtained impedances can be seen with the matlab script **PostProcessing1D/ flat/ Impedance_flat.m**. They are shown in Fig. 6.9.

We are looking for short range longitudinal and transverse wake potentials for a Gaussian

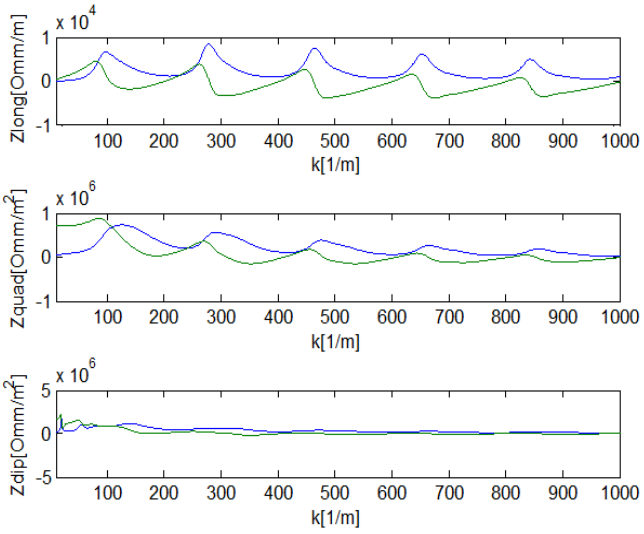


Figure 6.9: Impedances of the flat dielectric pipe. The real part is shown in blue. The imaginary part is presented by the green curve.

bunch with rms width 0.25mm. They can be obtained with matlab script **PostProcessing1D/ flat/ Wake_flat.m** and are shown in Fig. 6.10.

The results are cross-checked with ECHO2D. The simulations are done for 1 meters and 1.1 meters and subtracted to obtain "steady-state" wake. The setup to run ECHO2D code can be found in folders **ECHO2D** and **PostProcessor2D**. The instructions how to use ECHO2D code are given in corresponding section of this manual.

The comparison of the results from ECHO1D with ECHO2D can be seen by running the script **Compare_2D_vs_1D.m** in Matlab. The result is shown in Fig. 6.11.

6.7.3 Example 3: Flat anisotropic pipe

The third example can be found in directory **Examples/ N3_Flat_Anisotropic_Argonne**. We consider a rectangular perfectly conducting pipe of width 11 mm and half hight a_1=2.39mm. The pipe has an anisotropic dielectric layer vertically from a_0=1.5mm to a_1=2.39mm with relative permeabilities $\hat{\varepsilon}_y = 11.5, \hat{\varepsilon}_x = \hat{\varepsilon}_z = 9.4$. The geometry is shown in Fig. 6.12.

In order to model the perfectly conducting material we set *Boundary_Condition = PEC* in the command file **input_in.txt**. The geometry is anisotropic and we set *Method = Mix* to choose the finite difference method only in the anisotropic layer. For beam near to the axis we calculate only odd modes: *Modes = 1 3 5 7 9*. Again we introduce a small conductivity 0.05 S/m in the last row of geometry file **Example03.txt**. The obtained impedances can be seen with the matlab script **PostProcessing1D/ flat/ Impedance_flat.m**. They are shown in Fig. 6.13.

We are looking for short range longitudinal and transverse wake potentials for a Gaussian bunch with rms width 1.5mm. They can be obtained with matlab script **PostProcessing1D/ flat/ Wake_flat.m** and are shown in Fig. 6.14.

6.7.4 Example 4: Round pipe with two layers

The last example can be found in directory **Examples/N4_Round_kicker_SLAC**. We consider a round pipe. The pipe has two layers: a dielectric layer from a_0=5 mm to a_1=9 mm with relative permeability $\hat{\varepsilon} = 11$ and a ferromagnetic layer from a_1=9 mm to a_2=∞ with relative permittivity

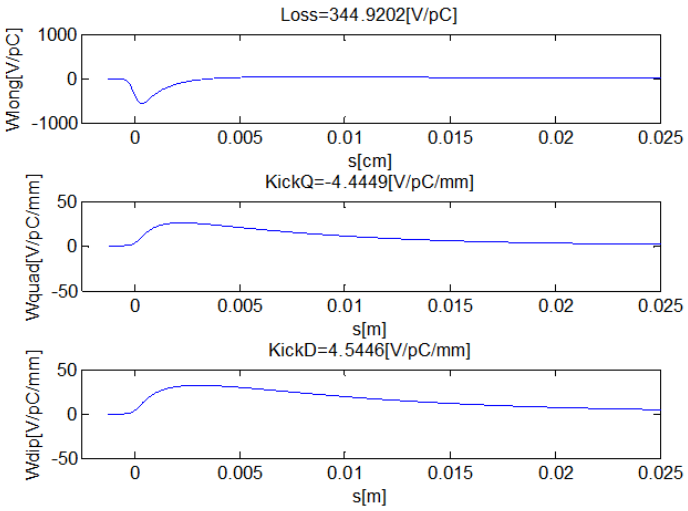


Figure 6.10: Wake potentials of the Gaussian bunch with rms width 0.25 mm in the flat dielectric pipe.

$\hat{\mu} = 10$. The geometry is shown in Fig. 6.15.

In order to model the infinite layer we set *Boundary_Condition = Open* in the command file **input_in.txt**. The geometry is isotropic and we set *Method = FM* to choose the fastest method: field matching. For beam near to the axis we calculate only two modes: *Modes = 0 1*. Again we introduce a small conductivity 0.1 S/m in the last rows of geometry file **Example04.txt**. The obtained impedances can be seen with the matlab script **PostProcessing1D/ round/ Impedance_round.m**. They are shown in Fig. 6.16.

We are looking for short range longitudinal and transverse wake potentials for a Gaussian bunch with rms width 0.25 mm. They can be obtained with matlab script **PostProcessing1D/ round/ Wake_round.m** and are shown in Fig. 6.17.

The results are cross-checked with ECHO2D. The simulations are done for 1 meters and 1.1 meters and subtracted to obtain the "steady-state" wake. We place perfectly conducting pipe at $r = 15$ mm. The setup to run code ECHO2D can be found in folders **ECHO2D** and **PostProcessor2D**. The instructions how to use ECHO2D code are given in corresponding section of this manual.

The comparison of the results from ECHO1D with ECHO2D can be seen by running the script **Compare_2D_vs_1D.m** in Matlab. The result is shown in Fig. 6.18.

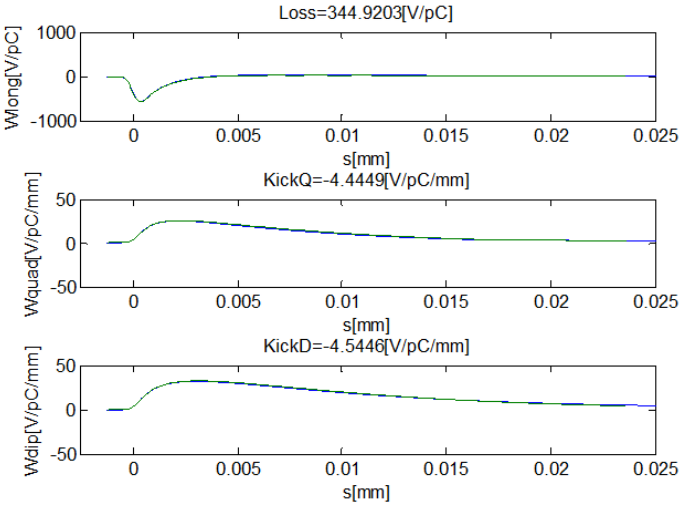


Figure 6.11: Comparison the wake potentials of the flat dielectric pipe obtained by ECHO1D (blue curves) with the ones obtained by ECHO2D (green curves)

.

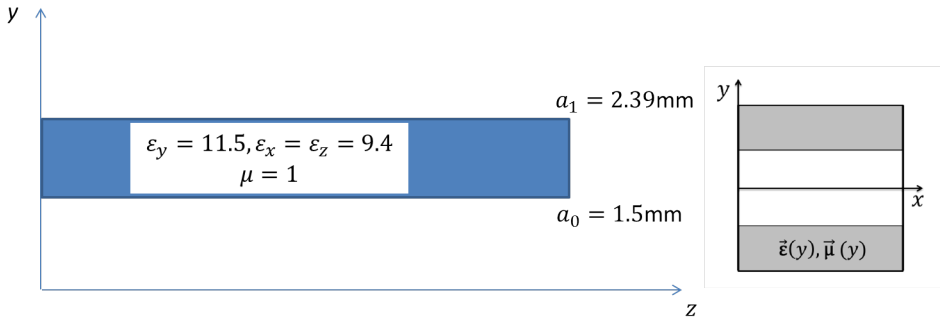


Figure 6.12: The geometry of flat anisotropic dielectric pipe inside of perfectly conducting pipe.

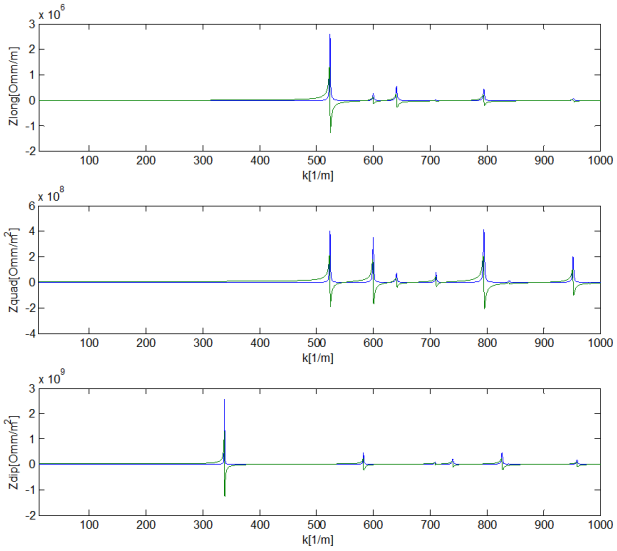


Figure 6.13: Impedances of the flat anisotropic dielectric pipe. The real part is shown in blue. The imaginary part is presented by the green curve.

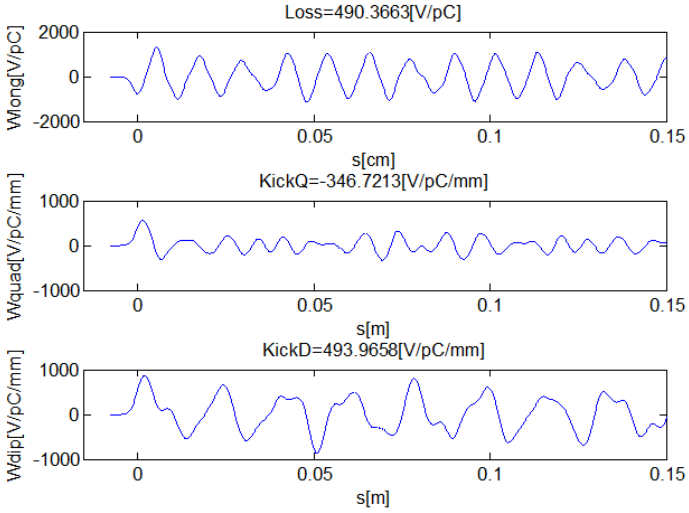


Figure 6.14: Wake potentials of the Gaussian bunch with rms width 1.5 mm in the flat anisotropic dielectric pipe.

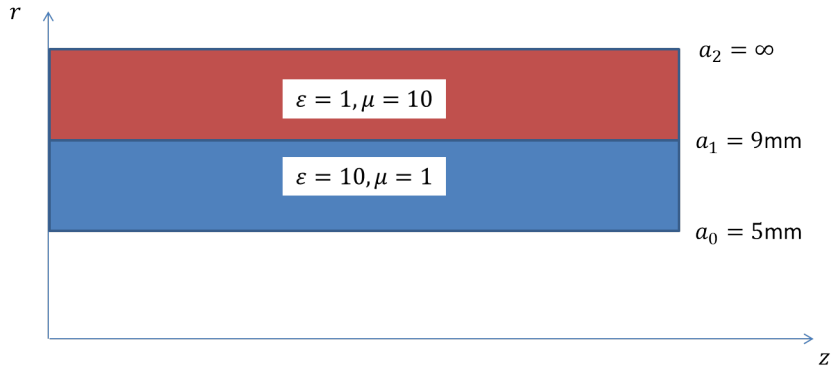


Figure 6.15: The geometry of round pipe with two layers.

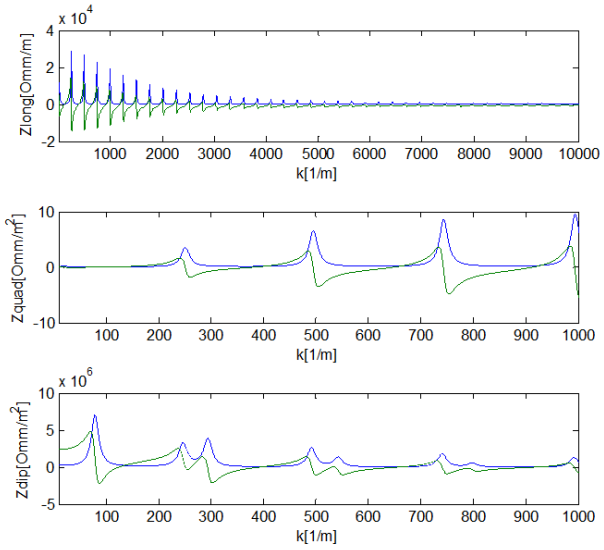


Figure 6.16: Impedances of the round two-layered pipe. The real part is shown in blue. The imaginary part is presented by the green curve.

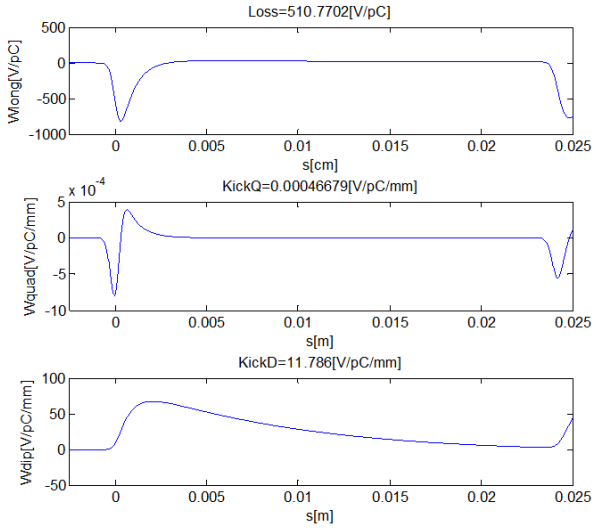


Figure 6.17: Wake potentials of the Gaussian bunch with rms width 0.25 mm in the two-layered pipe.

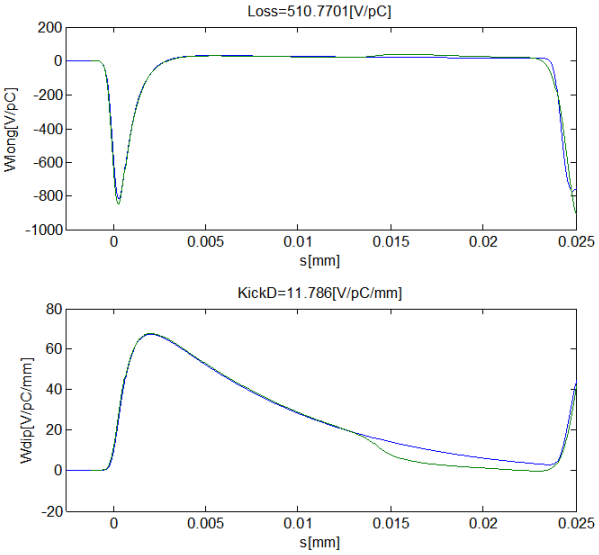
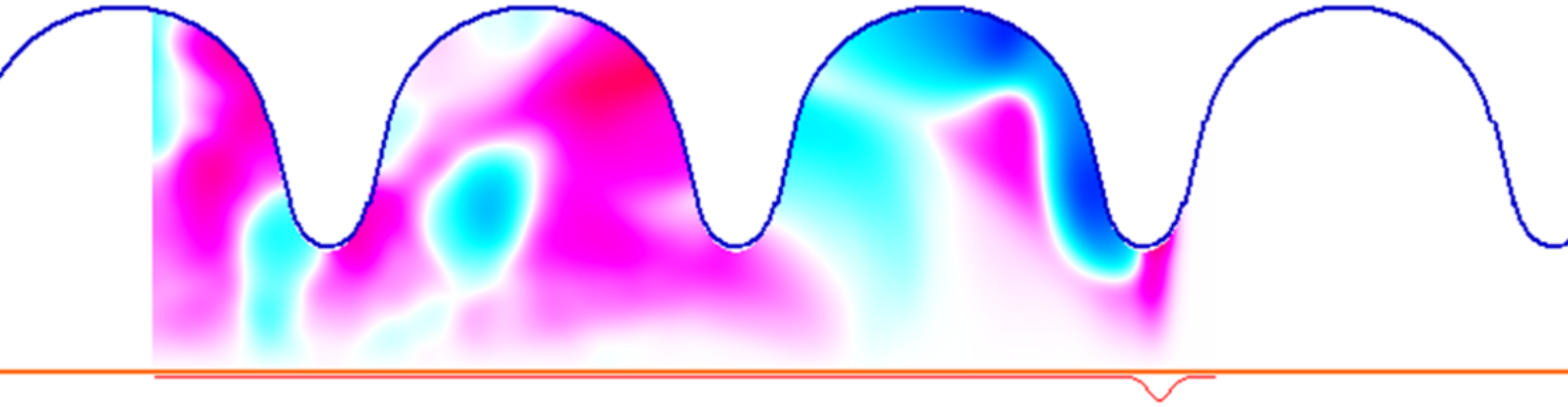


Figure 6.18: Comparison the wake potentials of the two-layered pipe obtained by ECHO1D (blue curves) with the ones obtained by ECHO2D (green curves)

.



Bibliography

Books

[1] A.W. Chao. *Physics of Collective Beam Instabilities in High Energy Accelerators*. New York: John Wiley and Sons, 1993 (cited on pages 9, 23, 45, 55, 56).

[2] J.D. Jackson. *Classical Electrodynamics*. 3rd edition. John Wiley and Sons, 1998 (cited on page 54).

[3] N. Mounet. *The LHC Transverse Coupled-Bunch Instability*. PhD Thesis. EPFL, Lausanne: CERN, 2012 (cited on pages 55, 56).

Articles

[4] T. Weiland and I. Zagorodnov. "Maxwell equations in structures with symmetries". In: *Journal of Computational Physics* 180 (2002), page 297 (cited on page 26).

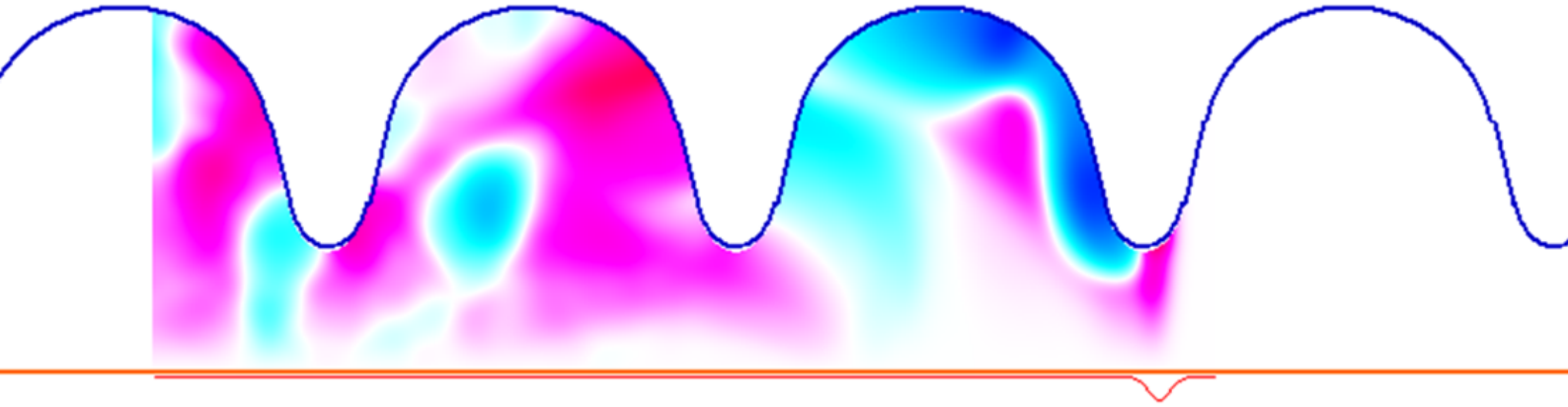
[5] I. Zagorodnov. "Impedances of anisotropic round and rectangular chambers". In: *Physical Review Accelerators and Beams* 21 (2018), page 064601 (cited on page 53).

[6] I. Zagorodnov, K.L.F. Bane, and G. Stupakov. "Calculation of wakefields in 2D rectangular structures". In: *Physical Review Accelerators and Beams* 18 (2015), page 104401 (cited on pages 15, 17, 23, 29).

[7] I. Zagorodnov, R. Schuhmann, and T. Weiland. "A uniformly stable conformal FDTD-method in Cartesian grids". In: *International Journal of Numerical Modeling* 16.2 (2003), page 127 (cited on page 18).

[8] I. Zagorodnov, R. Schuhmann, and T. Weiland. "Long-time numerical computation of electromagnetic fields in the vicinity of a relativistic source". In: *Journal of Computational Physics* 191 (2003), pages 525–141 (cited on page 9).

[9] I. Zagorodnov and T Weiland. "TE/TM field solver for particle beam simulations without numerical Cherenkov radiation". In: *Physical Review Accelerators and Beams* 8 (2005), page 042001 (cited on pages 15, 23).



Index