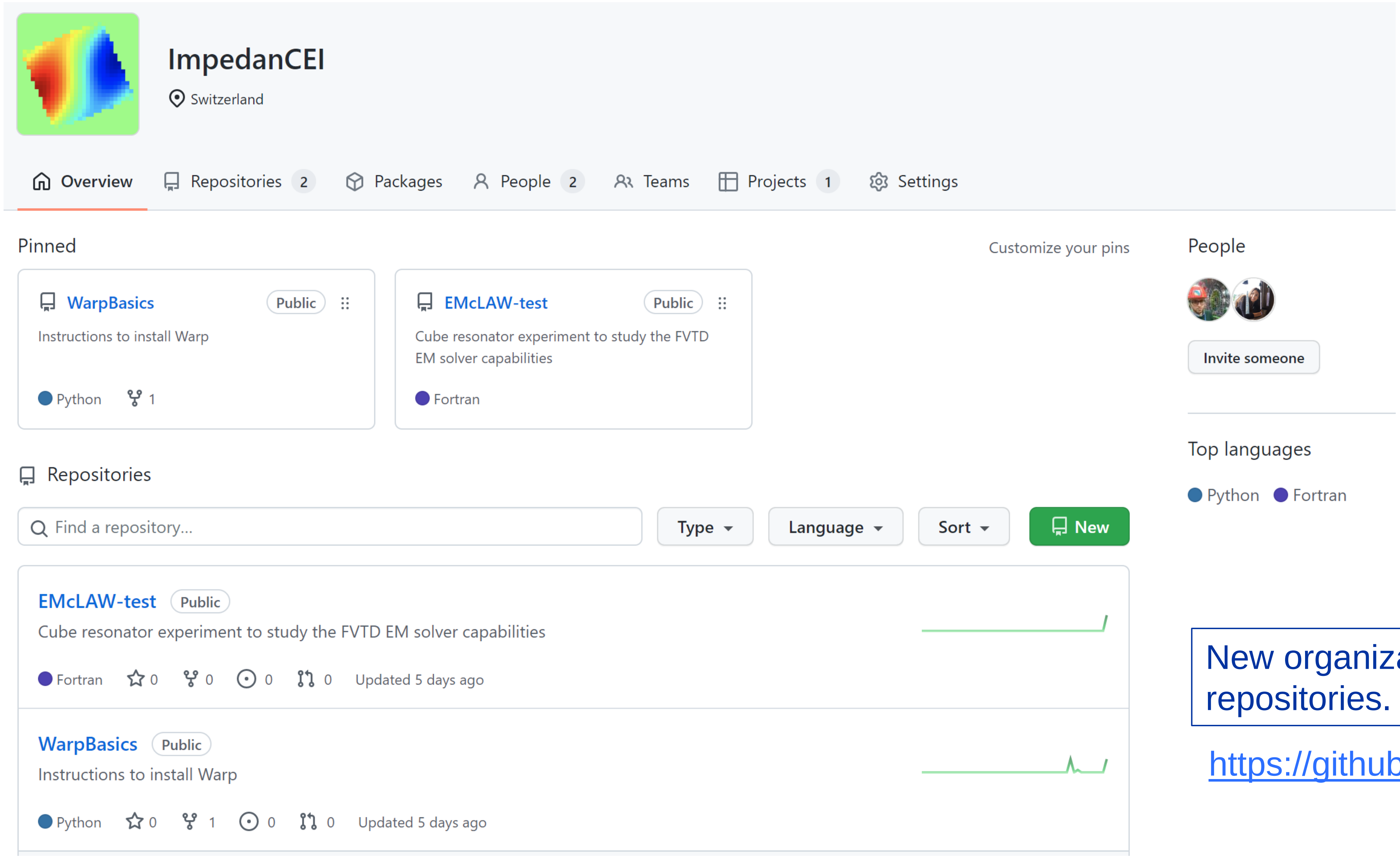
EMWSD

Electromagnetic and Wake Solver Developement

Meeting #01

Elena de la Fuente, Carlo Zannini, Lorenzo Giacomel, Giovanni Iadarola

Outline



ImpedanCEI
Switzerland

Pinned

WarpBasics Public
Instructions to install Warp
● Python 1

EMcLAW-test Public
Cube resonator experiment to study the FVTD EM solver capabilities
● Fortran

Repositories

EMcLAW-test Public
Cube resonator experiment to study the FVTD EM solver capabilities
● Fortran 0 0 0 0 Updated 5 days ago

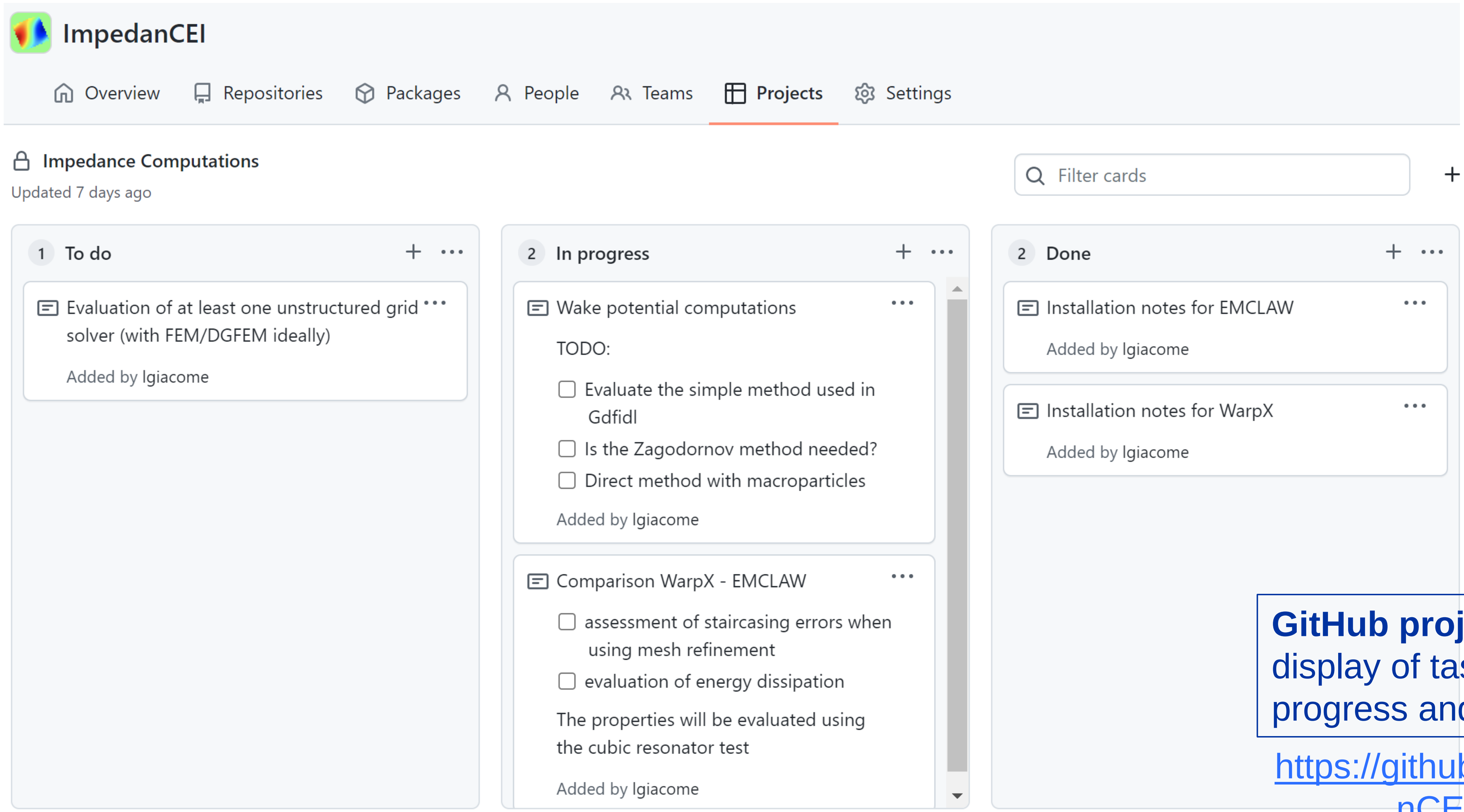
WarpBasics Public
Instructions to install Warp
● Python 0 1 0 0 Updated 5 days ago

People

Top languages
● Python ● Fortran

New organization for GitHub repositories.

https://github.com/ImpedanCEI


ImpedanCEI

Overview Repositories Packages People Teams Projects Settings

Impedance Computations

Updated 7 days ago

Filter cards

1 To do

Evaluation of at least one unstructured grid solver (with FEM/DGFEM ideally)

Added by lgiacome

2 In progress

Wake potential computations

TODO:

- [ ] Evaluate the simple method used in Gdfidl
- [ ] Is the Zagodornov method needed?
- [ ] Direct method with macroparticles

Added by lgiacome

Comparison WarpX - EMCLAW

- [ ] assessment of staircasing errors when using mesh refinement
- [ ] evaluation of energy dissipation

The properties will be evaluated using the cubic resonator test

Added by lgiacome

2 Done

Installation notes for EMCLAW

Added by lgiacome

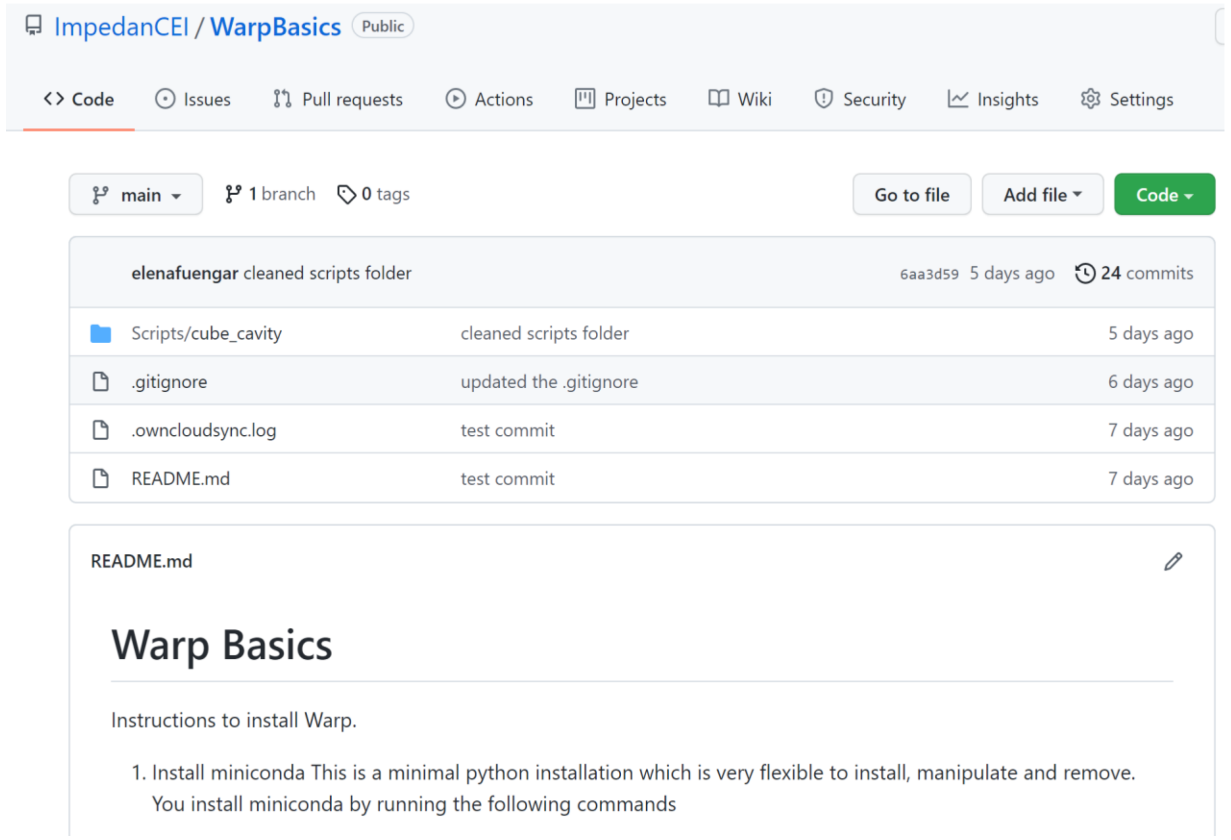
Installation notes for WarpX

Added by lgiacome

GitHub projects: Effective display of tasks to be done, in progress and done

https://github.com/orgs/ImpedanCEI/projects/1

WarpBasics repository


ImpedanCEI / WarpBasics Public

Code | Issues | Pull requests | Actions | Projects | Wiki | Security | Insights | Settings

main | 1 branch | 0 tags | Go to file | Add file | Code

elenafuengar cleaned scripts folder 6aa3d59 5 days ago 24 commits

Scripts/cube_cavity	cleaned scripts folder	5 days ago
.gitignore	updated the .gitignore	6 days ago
.owncloudsync.log	test commit	7 days ago
README.md	test commit	7 days ago

README.md

Warp Basics

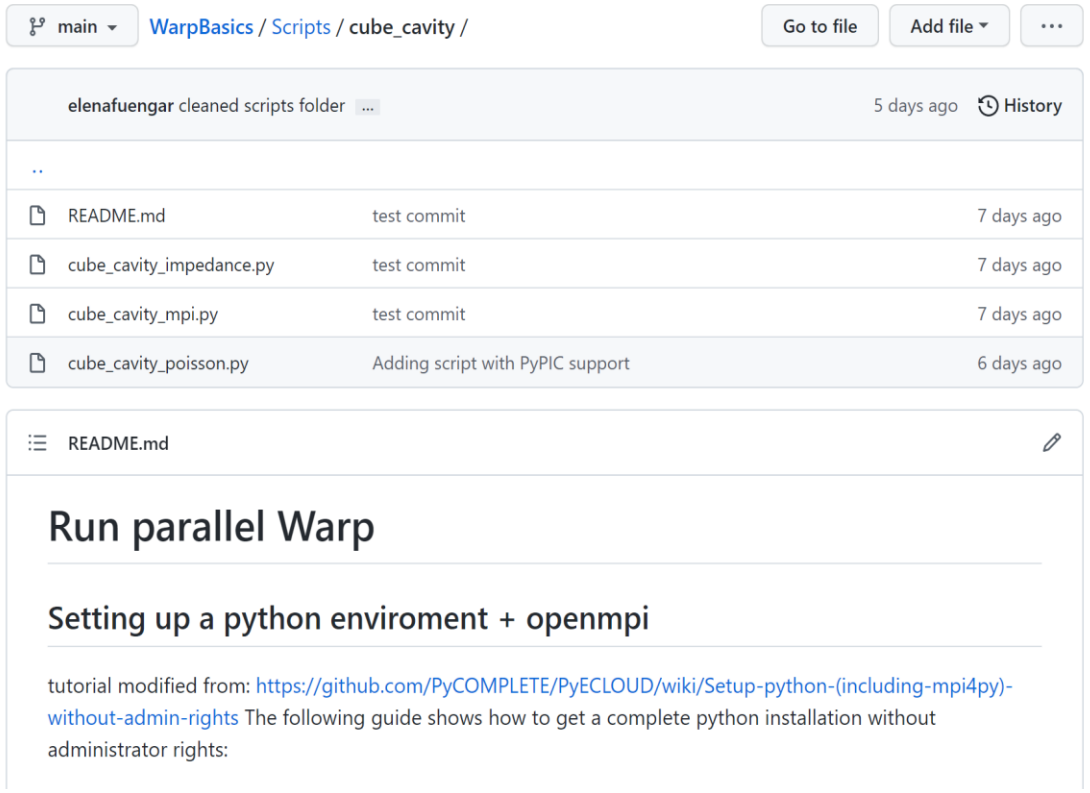
Instructions to install Warp.

1. Install miniconda This is a minimal python installation which is very flexible to install, manipulate and remove. You install miniconda by running the following commands

README.md files with the installation guide and instructions to run in parallel from the server

Scripts with the cube cavity simulations in Warp:

- **_mpi:** prepared to run on parallel in htcondor
- **_impedance:** old algorithm for WP and impedance
- **_poisson:** new algorithm [in progress]


main | WarpBasics / Scripts / cube_cavity / | Go to file | Add file | ...

elenafuengar cleaned scripts folder ... 5 days ago History

..

README.md	test commit	7 days ago
cube_cavity_impedance.py	test commit	7 days ago
cube_cavity_mpi.py	test commit	7 days ago
cube_cavity_poisson.py	Adding script with PyPIC support	6 days ago

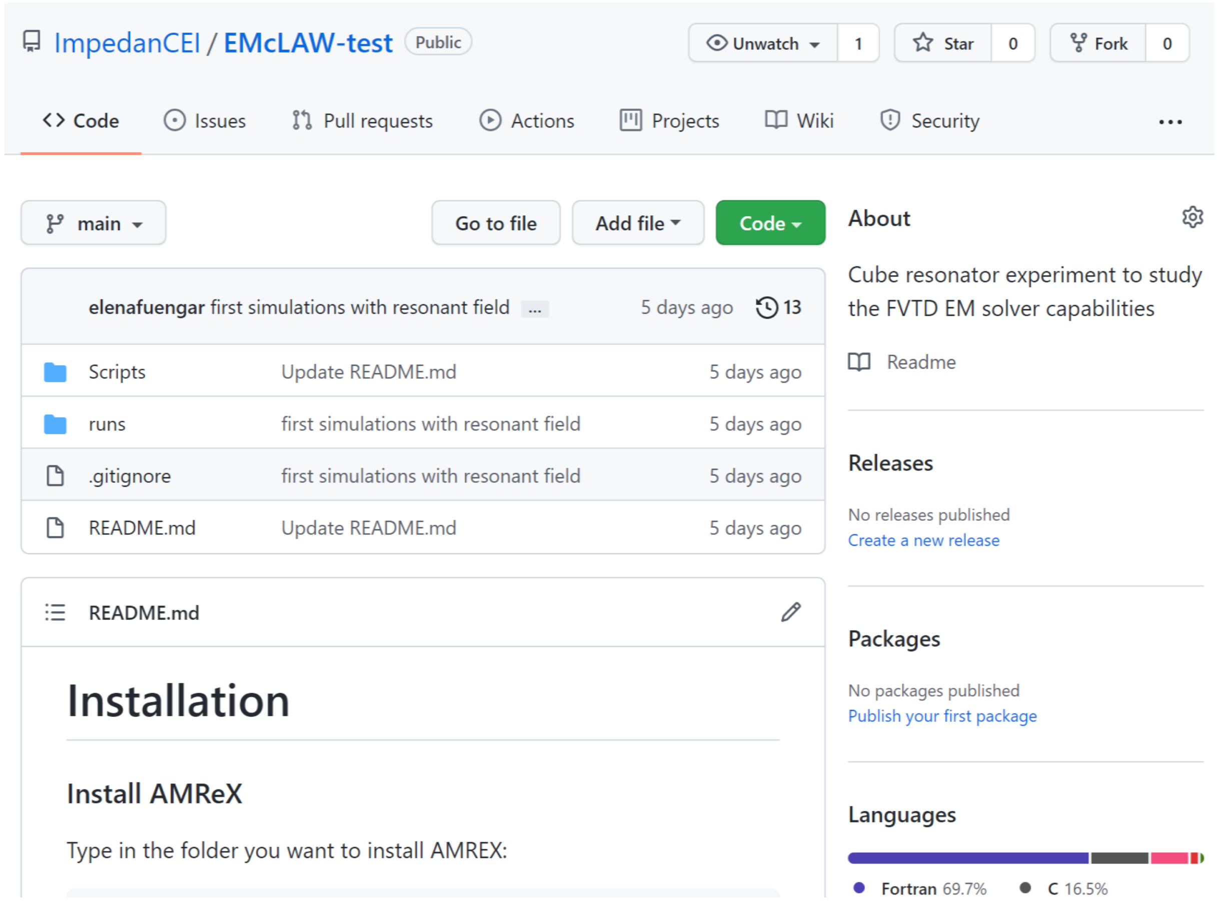
README.md

Run parallel Warp

Setting up a python enviroment + openmpi

tutorial modified from: https://github.com/PyCOMPLETE/PyECLOUD/wiki/Setup-python-(including-mpi4py)-without-admin-rights The following guide shows how to get a complete python installation without administrator rights:

EMcLAW-test repository


ImpedanCEI / EMcLAW-test Public

Unwatch 1 | Star 0 | Fork 0

Code | Issues | Pull requests | Actions | Projects | Wiki | Security

main | Go to file | Add file | Code

elenafuengar first simulations with resonant field ... 5 days ago 13

Scripts	Update README.md	5 days ago
runs	first simulations with resonant field	5 days ago
.gitignore	first simulations with resonant field	5 days ago
README.md	Update README.md	5 days ago

README.md

Installation

Install AMReX

Type in the folder you want to install AMREX:

About

Cube resonator experiment to study the FVTD EM solver capabilities

Readme

Releases

No releases published
Create a new release

Packages

No packages published
Publish your first package

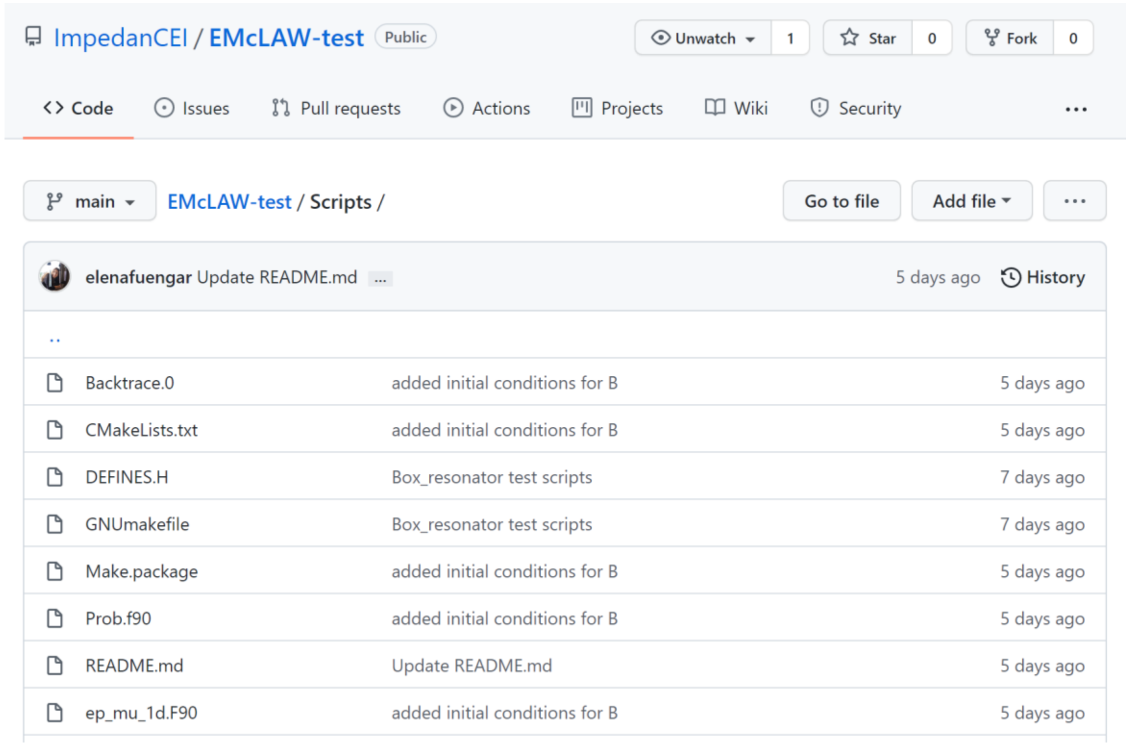
Languages

● Fortran 69.7% ● C 16.5%

README.md with installation and user guide, with modelling notes is also available

Scripts with the box resonator simulations:

- **Pron.f90:** where initial conditions are defined
- **metallic_material_3d.f90:** defines the metallic region
- **inputs:** defines the simulation domain, timesteps mesh and AMR refinement


ImpedanCEI / EMcLAW-test Public

Unwatch 1 | Star 0 | Fork 0

Code | Issues | Pull requests | Actions | Projects | Wiki | Security

main | EMcLAW-test / Scripts / | Go to file | Add file

elenafuengar Update README.md ... 5 days ago History

..

Backtrace.0	added initial conditions for B	5 days ago
CMakeLists.txt	added initial conditions for B	5 days ago
DEFINES.H	Box_resonator test scripts	7 days ago
GNUmakefile	Box_resonator test scripts	7 days ago
Make.package	added initial conditions for B	5 days ago
Prob.f90	added initial conditions for B	5 days ago
README.md	Update README.md	5 days ago
ep_mu_1d.F90	added initial conditions for B	5 days ago

Outline

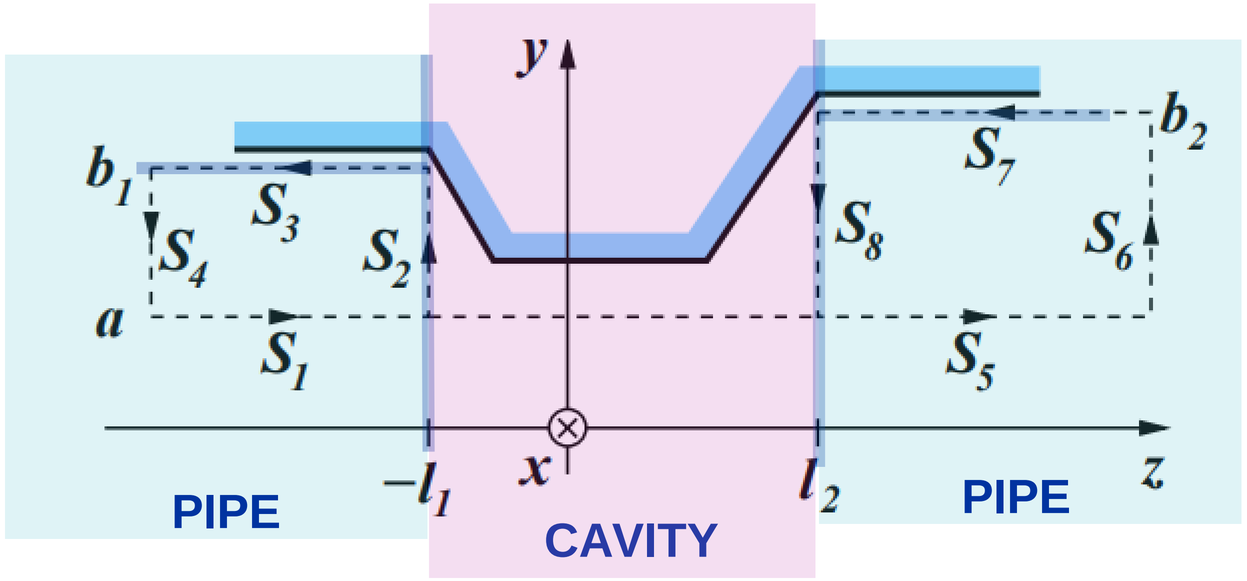
Algorithm overview

Source: https://accelconf.web.cern.ch/e06/PAPERS/WEPCH110.PDF

Deform the **path of integration** to go from the improper integral:

$$W_z(x,y,s) = -\frac{1}{q}\int_{-\infty}^{\infty} E_z(x,y,z,t=(z+s)/c)\mathrm{d}z \quad (1)$$

To a definite integral + 2 poisson problems



$$\begin{aligned} qW_z(x,y,s) &= \\ &= \left[\varphi(x,y=b_1)-\varphi(x,y)\right]_{z=-l_1} \\ &\quad -\int_{-l_1}^{l_2} E_z(x,y,z,t=(z+s)/c)dz \qquad (15) \\ &\quad +\left[\varphi(x,y)-\varphi(x,y=b_2)\right]_{z=l_2}. \end{aligned}$$

Poisson in z=-l1 surface t=(-l1+s)/c

Poisson in z=l2 surface t=(l2+s)/c

Calculation steps (I)

Source: https://accelconf.web.cern.ch/e06/PAPERS/WEPCH110.PDF

1) New field component definition:

$$\bar{u}(x,y,z,s) = u(x,y,z,t=(z+s)/c), \qquad (3)$$

such that

$$\frac{\partial \bar{u}}{\partial z} = \frac{\partial u}{\partial z} + \frac{1}{c}\frac{\partial u}{\partial t} \qquad (4)$$

2) Maxwell eqs for TM/TEM fields yield to an irrotational vector G:

$$\int_A \left(\nabla \times \vec{G}\right) \cdot d\vec{A} = 0 = \oint_S \vec{G} \cdot d\vec{s}$$
$$= \int_{l_2}^{\infty} G_z(y=a)dz + \int_a^{b_2} G_y(z=\infty)dy -$$
$$- \int_{l_2}^{\infty} G_z(y=b_2)dz - \int_a^{b_2} G_y(z=l_2)dy$$

$$0 = \int_{l_2}^{\infty} \bar{E}_z dz - \int_a^{b_2} \left(\bar{E}_y - c\bar{B}_x\right)(z=l_2)dy, \quad (10)$$

This proves that one can change an improper integral to a proper integral using the new path of integration

S8=S5 and S1=S2

3) Combining the equalities found, the wake potential is defined by:

$$qW_z(x,y,s) =$$
$$\int_a^{b_1} \left(E_y - cB_x\right)(z=-l_1, t=(-l_1+s)/c)dy$$
$$- \int_{-l_1}^{l_2} E_z(z, t=(z+s)/c)dz \qquad (11)$$
$$- \int_a^{b_2} \left(E_y - cB_x\right)(z=l_2, t=(l_2+s)/c)dy.$$

Calculation steps (II)

Source: https://accelconf.web.cern.ch/e06/PAPERS/WEPCH110.PDF

4) To extract the TM/TEM fields, we use the defined irrotational vector G since it can be derived from a scalar potential

$$\vec{G} = \vec{e}_x\left(\bar{E}_x + c\bar{B}_y\right) + \vec{e}_y\left(\bar{E}_y - c\bar{B}_x\right) + \vec{e}_z\bar{E}_z$$

$$(\nabla \times \mathbf{G})_z = 0$$

$$G_x\mathbf{e}_x + G_y\mathbf{e}_y = \nabla\varphi.$$

5) Taking the divergence this leads to a Poisson equation. The driving term has to be taken at z=-l1 and z=l2

$$\nabla\cdot(G_x\mathbf{e}_x + G_y\mathbf{e}_y) =$$
$$= \frac{\partial}{\partial x}(\bar{E}_x + c\bar{B}_y) + \frac{\partial}{\partial y}(\bar{E}_y - c\bar{B}_x)$$
$$= \underbrace{\frac{1}{c}\frac{\partial}{\partial t}E_z - \frac{\partial}{\partial z}E_z}_{\text{rho}} = \nabla^2\varphi. \qquad (13)$$

6) The wake potential becomes:

$$qW_z(x,y,s) =$$
$$= \left[\varphi(x,y=b_1) - \varphi(x,y)\right]_{z=-l_1}$$
$$- \int_{-l_1}^{l_2} E_z(x,y,z,t=(z+s)/c)dz \qquad (15)$$
$$+ \left[\varphi(x,y) - \varphi(x,y=b_2)\right]_{z=l_2}.$$

Poisson in z=-l1 surface t=(-l1+s)/c

Poisson in z=l2 surface t=(l2+s)/c

Scripting

$$qW_z(x,y,s) =$$

$$= \left[\varphi(x,y=b_1) - \varphi(x,y)\right]_{z=-l_1}$$

Poisson in z=-l1 surface t=(-l1+s)/c

$$- \int_{-l_1}^{l_2} E_z(x,y,z,t=(z+s)/c)dz \qquad (15)$$

$$+ \left[\varphi(x,y) - \varphi(x,y=b_2)\right]_{z=l_2} .$$

Poisson in z=l2 surface t=(l2+s)/c

Poisson equations are solved with **PyPIC** for each value of s_n at x=xtest, y=ytest, z=-l1 and x=xtest, y=ytest, z=l2

```
# PyPIC function to declare the implicit function for the conductors (this acts as BCs)
PyPIC_chamber = poly.polyg_cham_geom_object({'Vx' : np.array([w_rect, -w_rect, -w_rect, w_rect]),
                                             'Vy': np.array([h_rect, h_rect, -h_rect, -h_rect]),
                                             'x_sem_ellip_insc' : 0.99*w_rect, #important to add this
                                             'y_sem_ellip_insc' : 0.99*h_rect})

# solver object
picFD = PIC_FD.FiniteDifferences_Staircase_SquareGrid(chamb = PyPIC_chamber, Dh = dh, sparse_solver = 'PyKLU')

# define the left side of the laplacian (driving term rho = 1/c*dEz/dt-dEz/dz)
# rho = np.ones_like(picFD.rho)                                          #test rho to check the solver. Rho needs
rho = Ez_dt[iz_l1]-Ez_dt[it_l1]/picmi.constants.c*np.ones_like(picFD.rho)       #this rho is a constant evaluated

# solve the laplacian and obtain phi(0,0)
picFD.solve(rho = rho)      #the dimensions are selected by pyPIC solver
phi[n] = picFD.phi.copy()
```

Ez is interpolated to match the dimensions between t and z so they can be added to obtain rho

The derivatives are obtained for each value of z_k, $t_{k,n}$

```
#--- obtain the derivatives
for n in range(tot_nsteps-1):
    Ez_dt[k]= (Ez_interp[n+1, :] - Ez_interp[n, :])/dt
    Ez_dz[k]= (Ez_interp[:, n+1] - Ez_interp[:, n])/dz
```

Scripting

$$
\begin{aligned}
qW_z(x,y,s) &= \\
&= \left[\varphi(x,y=b_1)-\varphi(x,y)\right]_{z=-l_1} \\
&\quad -\int_{-l_1}^{l_2} E_z(x,y,z,t=(z+s)/c)\,dz \qquad (15) \\
&\quad +\left[\varphi(x,y)-\varphi(x,y=b_2)\right]_{z=l_2}\,.
\end{aligned}
$$

Poisson in z=-l1 surface t=(-l1+s)/c

Poisson in z=l2 surface t=(l2+s)/c

The definite integral is solved for each z_k, s_n . For each pair s, z, the index for the time dimension has to be obtained

```
#----------------------------#
# integral between -l1 and l2 #
#----------------------------#


#int(Ez(xtest, ytest, z, t=(s+z)/c))
for k in range(iz_l1, iz_l1):
    it=int(((z[k]+s[n])/c-t[0])/dt)
    integral=integral+(Ez_interp[k, it])*dz
```

```
#----------------------#
#      Obtain W(s)     #
#----------------------#
```

All is added obtaining $W(s_k)$

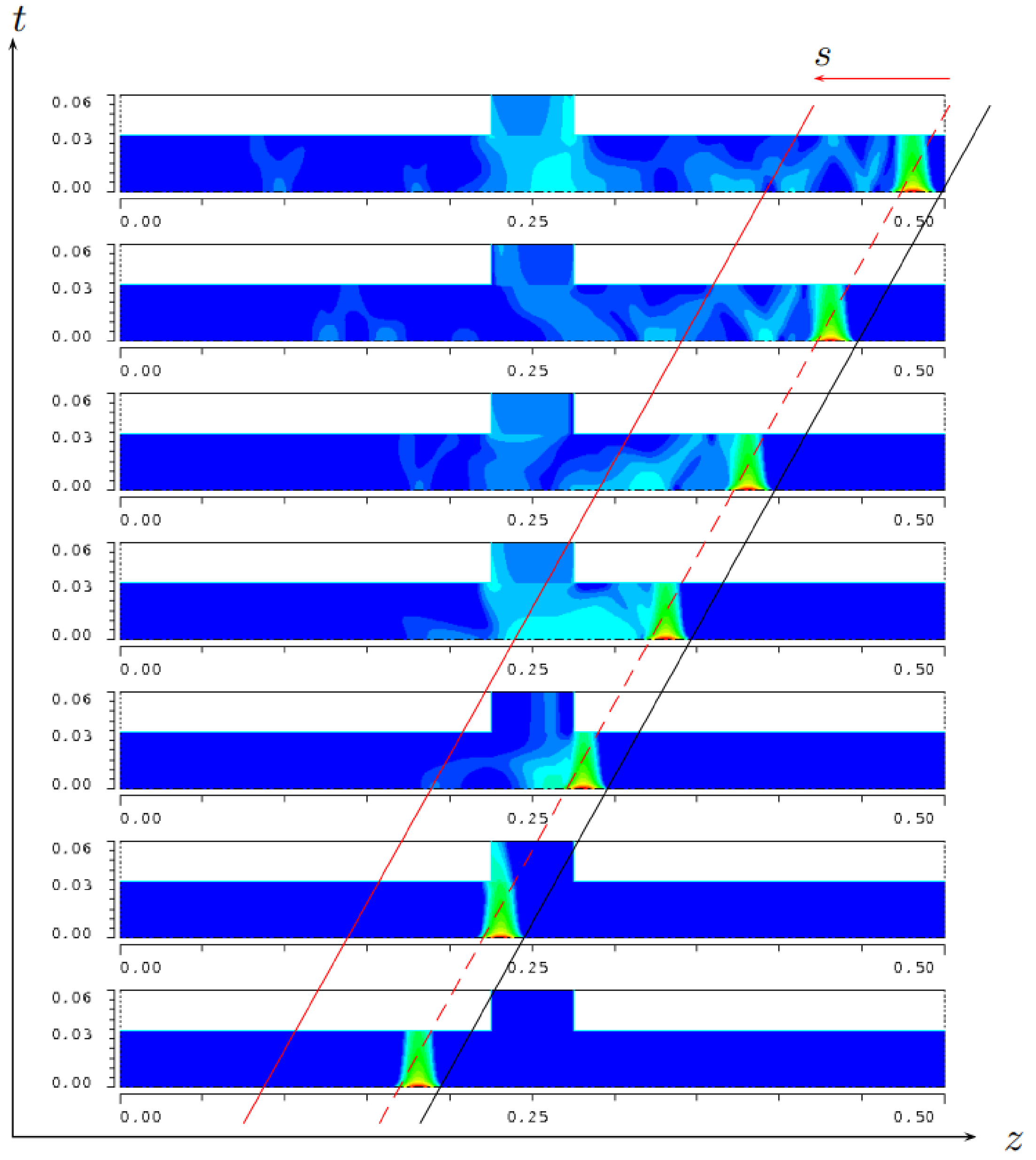
```
# Define phi(x, y=b1)
# Phis indexes go from x(-w_rect,w_rect) and y(-h_rect,h_rect)
# x direction is gridded with (w_rect*2)/dh + 10 ghost cells
# y direction is gridded with (h_rect*2)/dh + 8 ghost cells
# +info see: class PyPIC_Scatter_Gather(object):
iy_b1=int((b1-picFD.bias_y)/dh)
iy_b2=int((b2-picFD.bias_y)/dh)
ixtest_phi=int((xtest-picFD.bias_y)/dh)
ixtest_phi=int((ytest-picFD.bias_y)/dh)

phi_b1=phi(ixtest_phi, iy_b1, n) #phi(x=0, y=b1, s[n])
phi_b2=phi(ixtest_phi, iy_b1, n) #phi(x=0, y=b1, s[n])

Wake_potential[n]=(phi_b1-phi[ixtest_phi, iytest_phi])-integral-(phi[ixtest_phi, iytest_phi]-phi_b2)
```

S vector?

- What resolution and length can the s vector have?

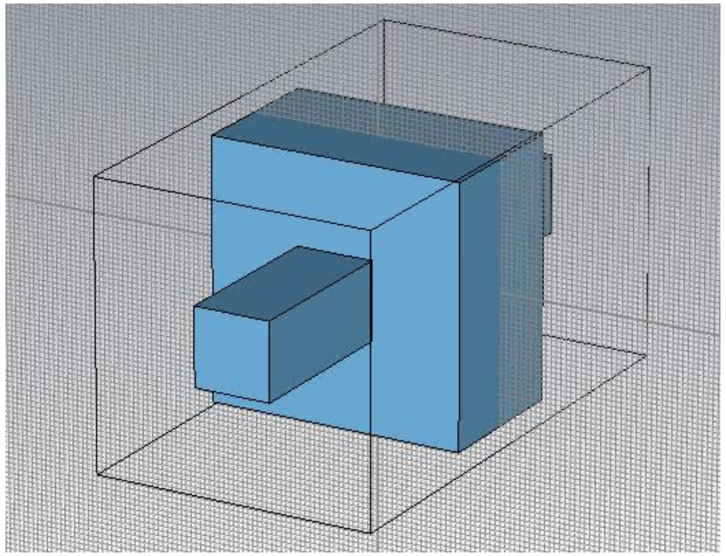
- How is the wake length related to the number of timesteps?

$$W_z(x,y,s) = -\frac{1}{q}\int_{-\infty}^{\infty} E_z(x,y,z,t=(z+s)/\mathrm{c})\mathrm{d}z$$



CST study of wakelength and nº of timesteps

The aim of these simulations was to obtain the relation between mesh cells, number of timesteps and s vector resolution



Background PEC extended 2.5 mm in x & y
Resonance at ~4.3 GHz

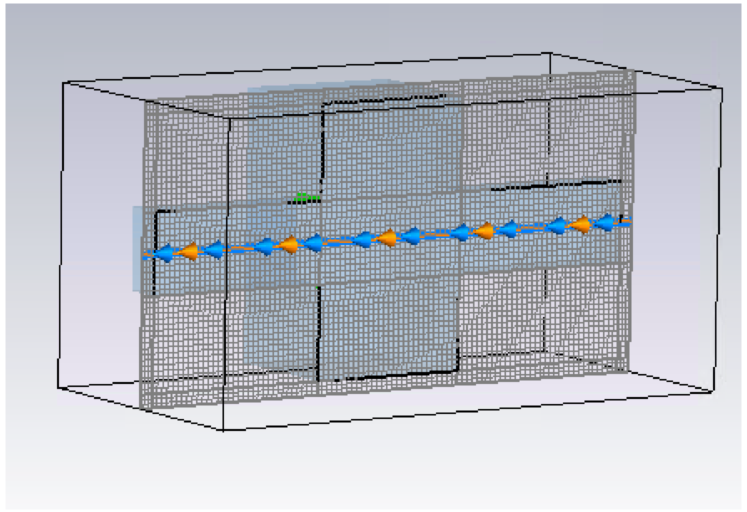
Cube cavity surrounded by PEC:

- Total length: 100 mm
- Cavity length: 30 mm
- Pipe width: 15 mm
- Cavity width: 50 mm

Nx=55,
Ny=55,
Nz=96
(ncells = 277020) --> Δh=1 mm

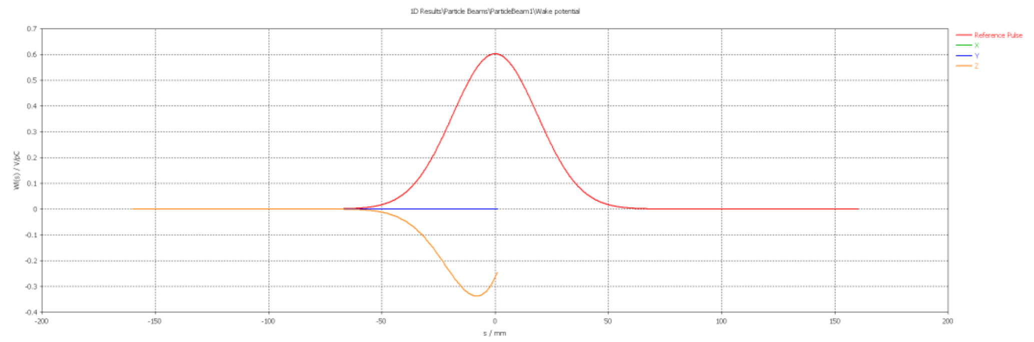
Beam excitation:
Sigmat=¼ ns –-> sigmaz=18.73 mm
Excitation duration: 6.50460411e-001 ns

```
LOGFILE
  Number of mesh cells:                277020
  Excitation duration:               6.50460411e-001 ns
  Calculation time for excitation:     0     s
  Number of calculated pulse widths:    1.3386
  Simulated number of time steps:       450
  Maximum number of time steps:         450
  Time step width:
     without subcycles:               1.93490105e-003 ns
```

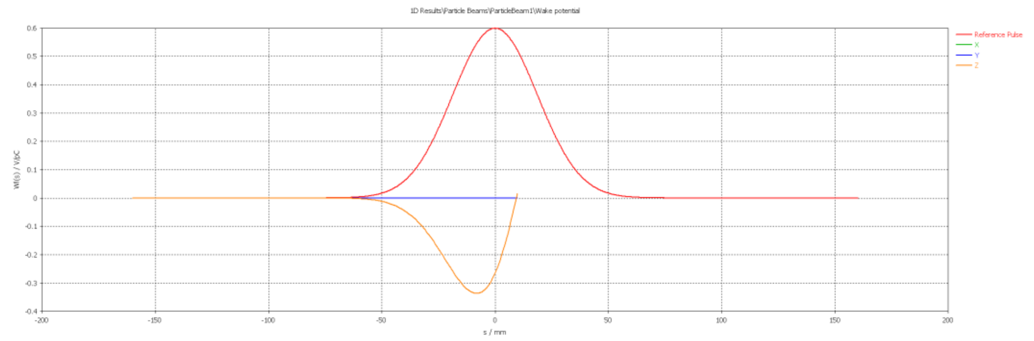


CST study of wavelength and nº of timesteps

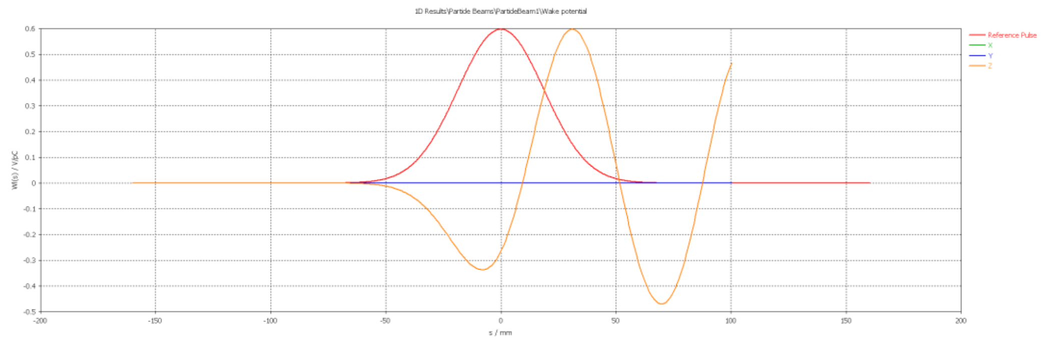
Wakelength=1 --> nsteps=450,
dt=1.93490105e-003 ns,
Npulse=1.3386 (t=sigmat*npulse)
Length s=279 (277 negative values)



Wakelength = 10 --> nsteps=465,
dt=1.93490105e-003 ns,
Npulse=1.38322
Length s=294 (277 negative values)

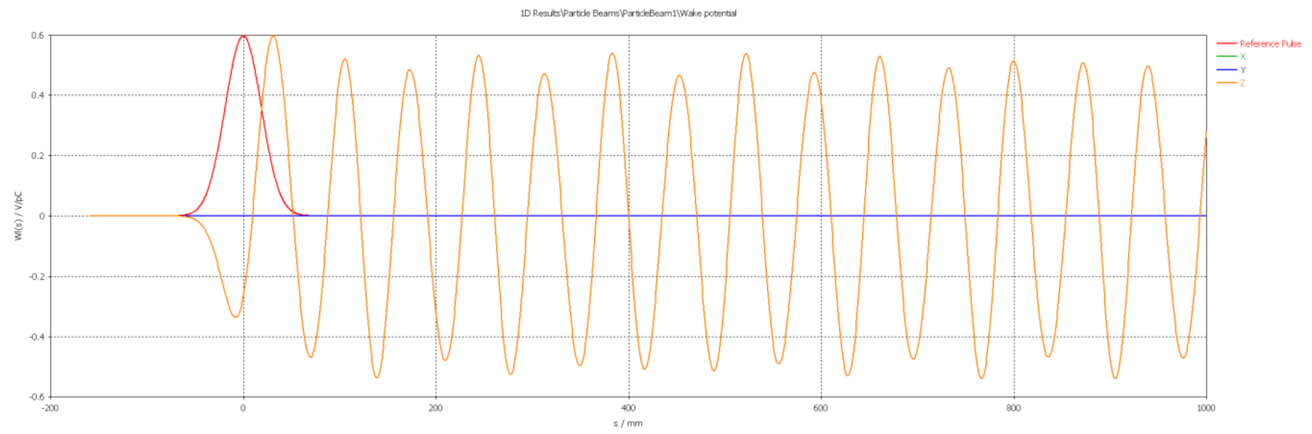


Wakelength = 100 --> nsteps=620,
dt=1.93490105e-003 ns,
Npulse=1.84429
Length s=450 (277 negative values)

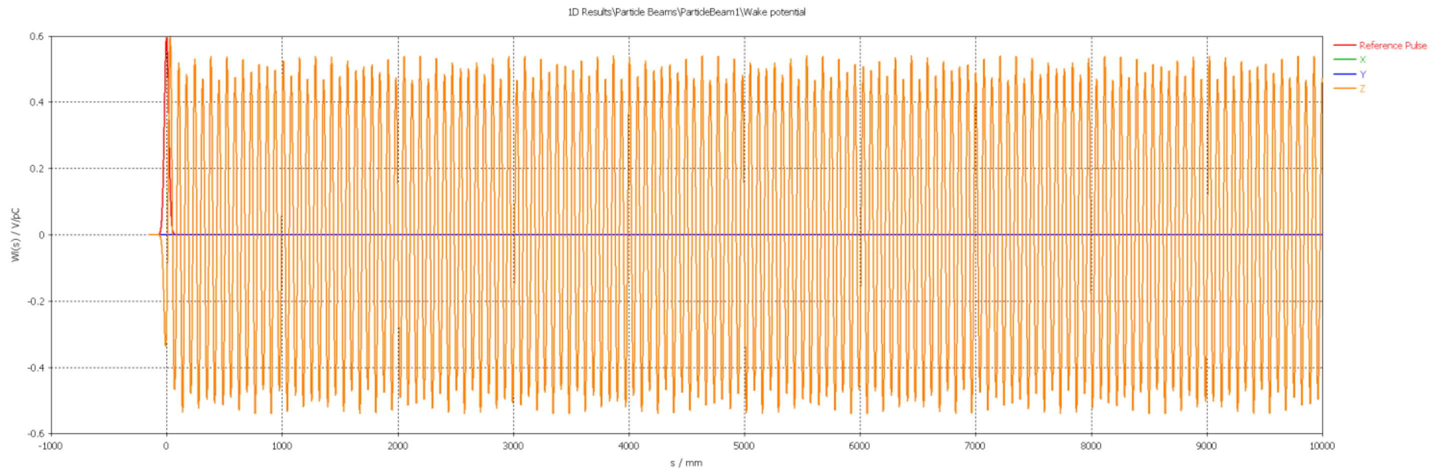


CST study of wavelength and nº of timesteps

Wakelength = 1000 --> nsteps=2172, dt=1.93490105e-003 ns, Npulse=6.46097
Length s=2001 (277 negative values)



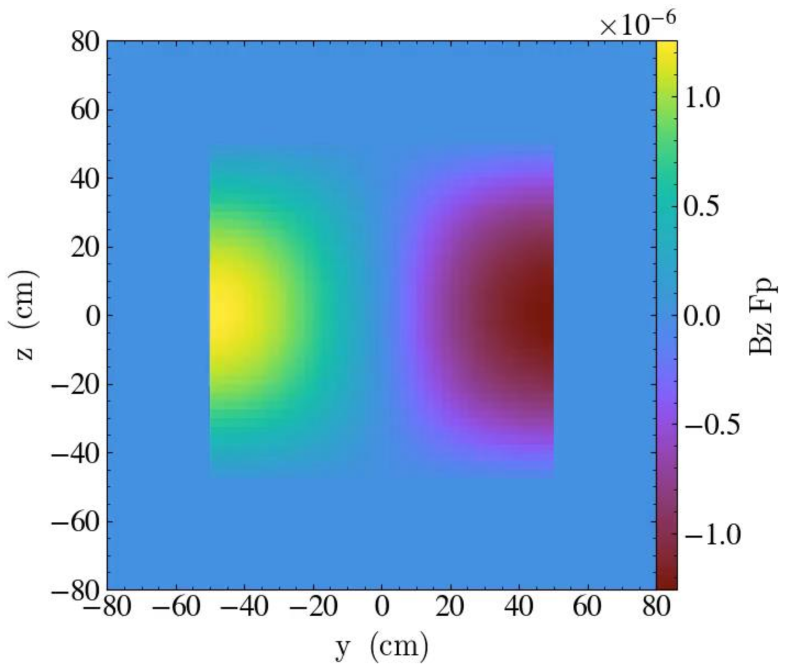
Wakelength = 10000 --> nsteps=17687, dt=1.93490105e-003 ns, Npulse=52.6129
Length s=17518 (277 negative values)



Outline

Box resonator (I)

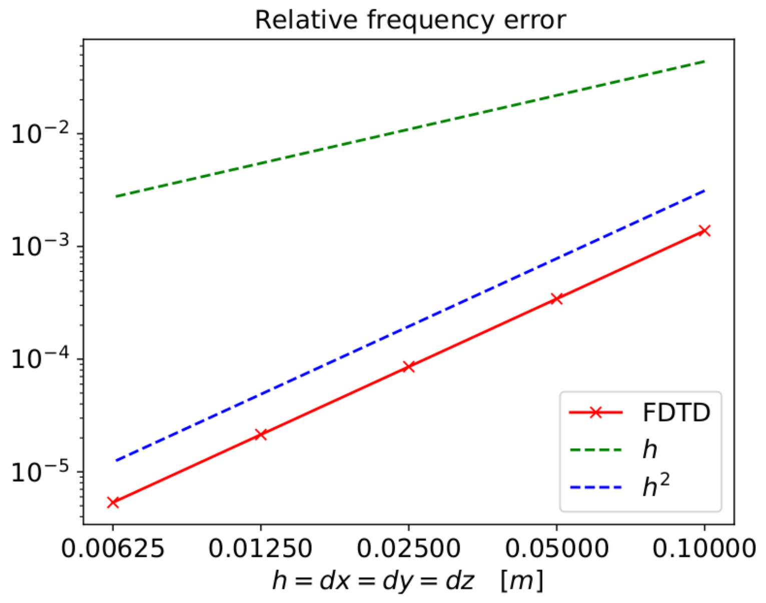
Objective 1: assess the order of the numerical scheme with a convergence analysis



Relative frequency error

$$Err(h) = \frac{Freq(h) - Freq^*}{Freq^*}.$$

$Freq(h)$ is the estimated frequency with $h = \Delta x = \Delta y = \Delta z =$ and $Freq^*$ is the analytic frequency.



Source: Lorenzo's presentation

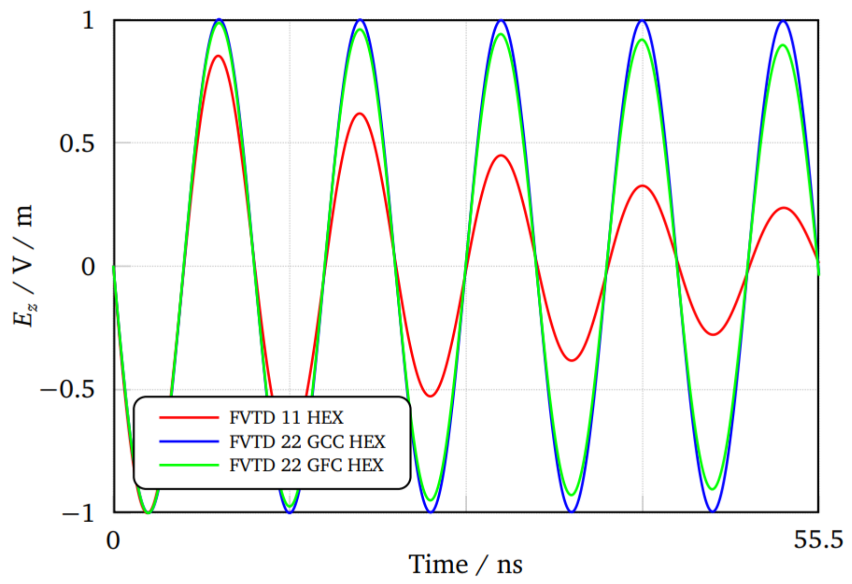
Box resonator (II)

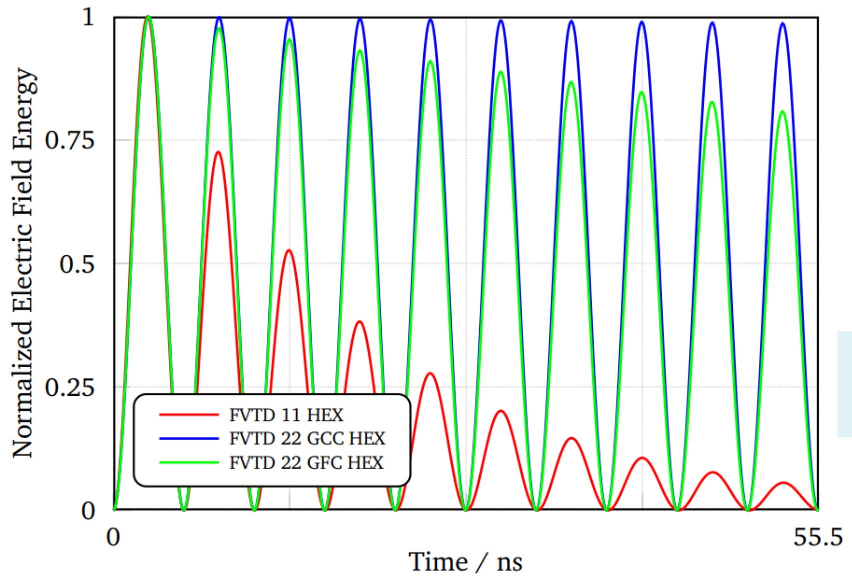
Source: Darmstadt thesis *"Investigating Finite Volume Time Domain Methods in Computational Electromagnetics"*

Objective 2: study the energy dissipation of FVTD

$$E_x(x,y,z) = \frac{-1}{\gamma^2+k^2}\frac{m\pi}{a}\frac{p\pi}{c}E_0\cos\left(\frac{m\pi}{a}x\right)\sin\left(\frac{n\pi}{b}y\right)\sin\left(\frac{p\pi}{c}z\right)$$
$$E_y(x,y,z) = \frac{-1}{\gamma^2+k^2}\frac{n\pi}{b}\frac{p\pi}{c}E_0\sin\left(\frac{m\pi}{a}x\right)\cos\left(\frac{n\pi}{b}y\right)\sin\left(\frac{p\pi}{c}z\right)$$
$$E_z(x,y,z) = E_0\sin\left(\frac{m\pi}{a}x\right)\sin\left(\frac{n\pi}{b}y\right)\cos\left(\frac{p\pi}{c}z\right)$$

$$H_x(x,y,z) = \frac{j\omega\varepsilon}{\gamma^2+k^2}\frac{n\pi}{b}E_0\sin\left(\frac{m\pi}{a}x\right)\cos\left(\frac{n\pi}{b}y\right)\cos\left(\frac{p\pi}{c}z\right)$$
$$H_y(x,y,z) = \frac{-j\omega\varepsilon}{\gamma^2+k^2}\frac{m\pi}{a}E_0\cos\left(\frac{m\pi}{a}x\right)\sin\left(\frac{n\pi}{b}y\right)\cos\left(\frac{p\pi}{c}z\right)$$
$$H_z(x,y,z) = 0$$





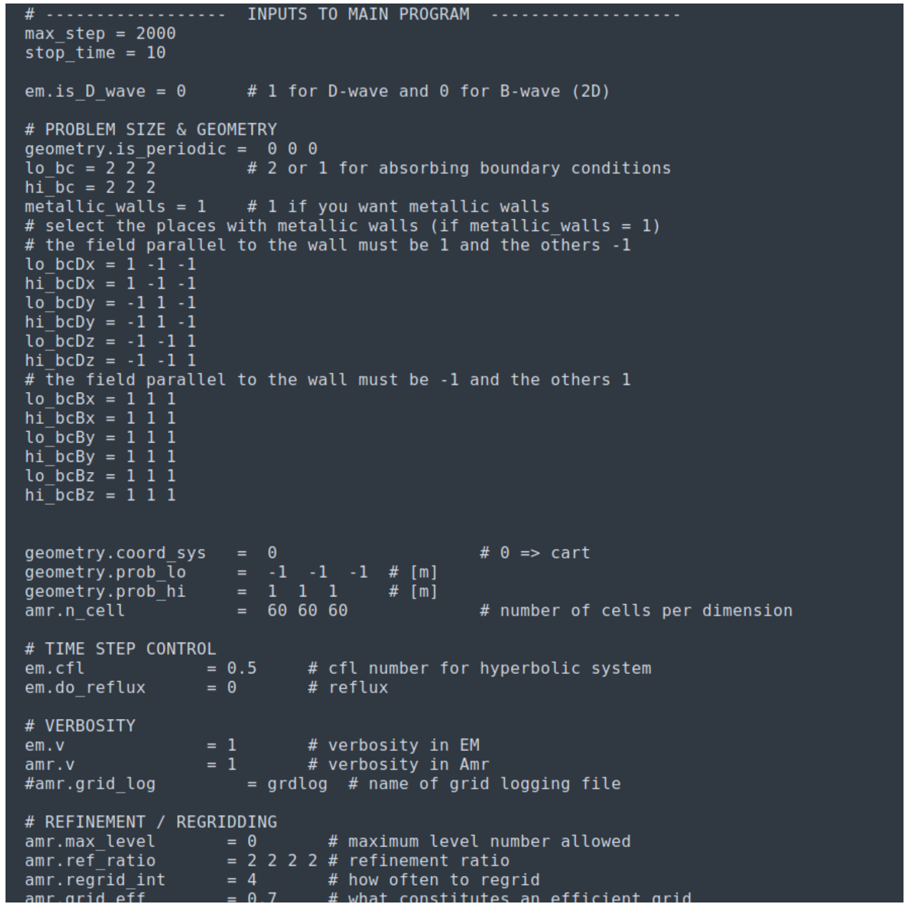
Energy decay at cavity center:

$$u_e = \frac{1}{2}\varepsilon_0|\mathbf{E}|^2.$$

$$f(t) = a_0 e^{-\alpha t}\sin(\omega t+\delta)$$

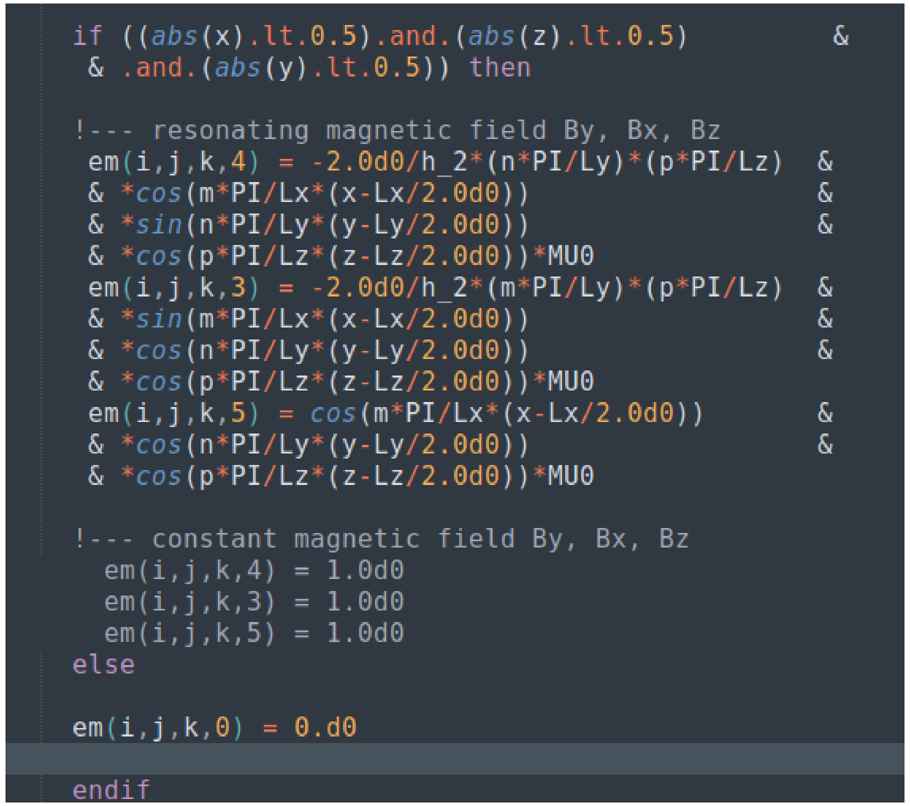
EMcLaw simulations set up

inputs: defining domain, nº cells, simulation time



```
# ------------------  INPUTS TO MAIN PROGRAM  -------------------
max_step = 2000
stop_time = 10

em.is_D_wave = 0      # 1 for D-wave and 0 for B-wave (2D)

# PROBLEM SIZE & GEOMETRY
geometry.is_periodic =  0 0 0
lo_bc = 2 2 2         # 2 or 1 for absorbing boundary conditions
hi_bc = 2 2 2
metallic_walls = 1    # 1 if you want metallic walls
# select the places with metallic walls (if metallic_walls = 1)
# the field parallel to the wall must be 1 and the others -1
lo_bcDx = 1 -1 -1
hi_bcDx = 1 -1 -1
lo_bcDy = -1 1 -1
hi_bcDy = -1 1 -1
lo_bcDz = -1 -1 1
hi_bcDz = -1 -1 1
# the field parallel to the wall must be -1 and the others 1
lo_bcBx = 1 1 1
hi_bcBx = 1 1 1
lo_bcBy = 1 1 1
hi_bcBy = 1 1 1
lo_bcBz = 1 1 1
hi_bcBz = 1 1 1


geometry.coord_sys   =  0                    # 0 => cart
geometry.prob_lo     =  -1  -1  -1  # [m]
geometry.prob_hi     =  1  1  1     # [m]
amr.n_cell           =  60 60 60             # number of cells per dimension

# TIME STEP CONTROL
em.cfl             = 0.5     # cfl number for hyperbolic system
em.do_reflux       = 0       # reflux

# VERBOSITY
em.v               = 1       # verbosity in EM
amr.v              = 1       # verbosity in Amr
#amr.grid_log         = grdlog  # name of grid logging file

# REFINEMENT / REGRIDDING
amr.max_level       = 0       # maximum level number allowed
amr.ref_ratio       = 2 2 2 2 # refinement ratio
amr.regrid_int      = 4       # how often to regrid
amr.grid_eff        = 0.7     # what constitutes an efficient grid
```

Prob.f90: initializing Bx, By, Bz field

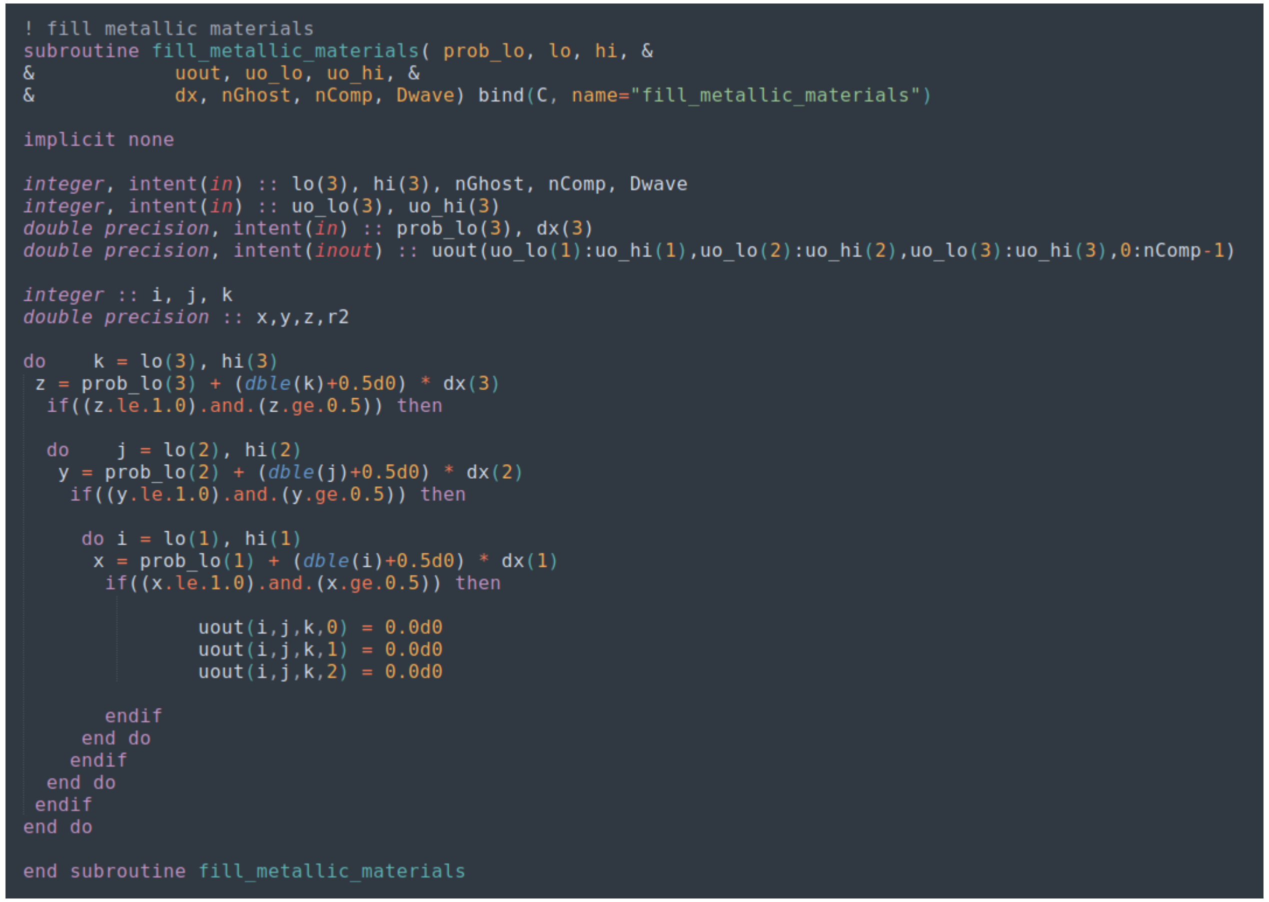


```
if ((abs(x).lt.0.5).and.(abs(z).lt.0.5)        &
 & .and.(abs(y).lt.0.5)) then

!--- resonating magnetic field By, Bx, Bz
 em(i,j,k,4) = -2.0d0/h_2*(n*PI/Ly)*(p*PI/Lz)  &
 & *cos(m*PI/Lx*(x-Lx/2.0d0))                  &
 & *sin(n*PI/Ly*(y-Ly/2.0d0))                  &
 & *cos(p*PI/Lz*(z-Lz/2.0d0))*MU0
 em(i,j,k,3) = -2.0d0/h_2*(m*PI/Ly)*(p*PI/Lz)  &
 & *sin(m*PI/Lx*(x-Lx/2.0d0))                  &
 & *cos(n*PI/Ly*(y-Ly/2.0d0))                  &
 & *cos(p*PI/Lz*(z-Lz/2.0d0))*MU0
 em(i,j,k,5) = cos(m*PI/Lx*(x-Lx/2.0d0))       &
 & *cos(n*PI/Ly*(y-Ly/2.0d0))                  &
 & *cos(p*PI/Lz*(z-Lz/2.0d0))*MU0

!--- constant magnetic field By, Bx, Bz
  em(i,j,k,4) = 1.0d0
  em(i,j,k,3) = 1.0d0
  em(i,j,k,5) = 1.0d0
else

em(i,j,k,0) = 0.d0

endif
```

Same equations used in the WarpX test

EMcLaw simulations set up

Metallic_materials_rd.f90: changing the if sentences to impose material relations (in this case, PEC)

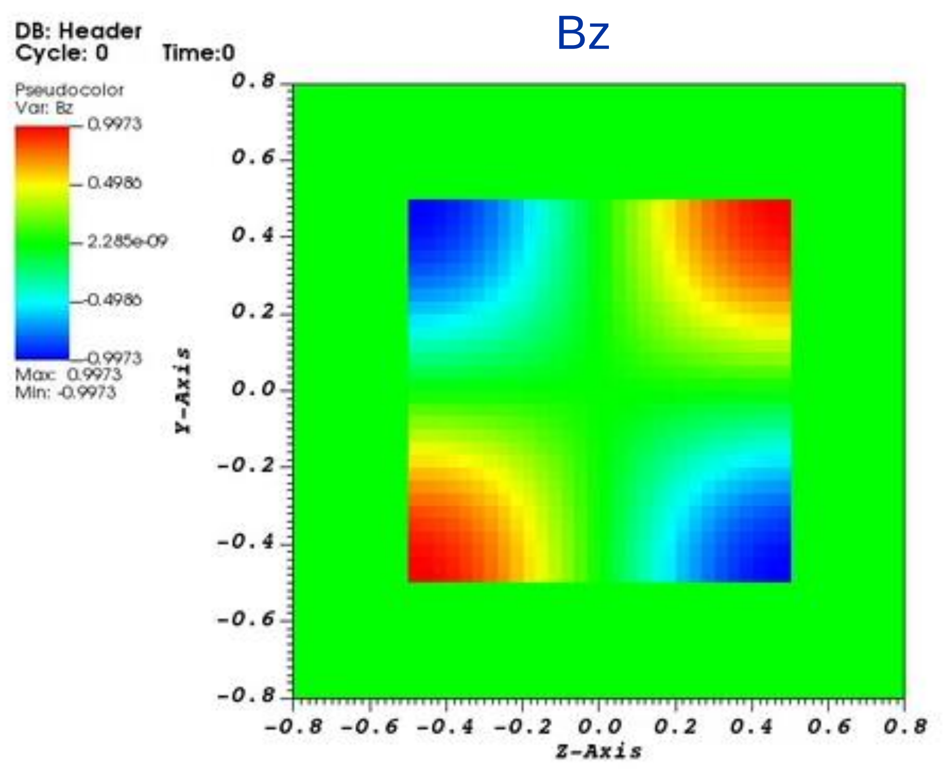


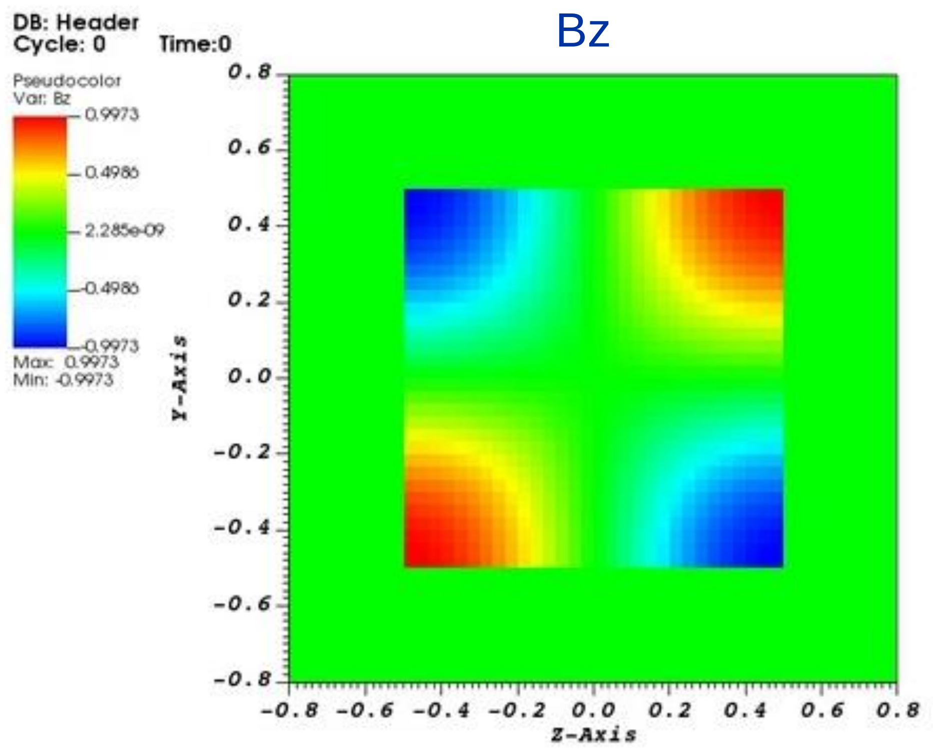
```fortran
! fill metallic materials
subroutine fill_metallic_materials( prob_lo, lo, hi, &
&             uout, uo_lo, uo_hi, &
&             dx, nGhost, nComp, Dwave) bind(C, name="fill_metallic_materials")

implicit none

integer, intent(in) :: lo(3), hi(3), nGhost, nComp, Dwave
integer, intent(in) :: uo_lo(3), uo_hi(3)
double precision, intent(in) :: prob_lo(3), dx(3)
double precision, intent(inout) :: uout(uo_lo(1):uo_hi(1),uo_lo(2):uo_hi(2),uo_lo(3):uo_hi(3),0:nComp-1)

integer :: i, j, k
double precision :: x,y,z,r2

do    k = lo(3), hi(3)
 z = prob_lo(3) + (dble(k)+0.5d0) * dx(3)
  if((z.le.1.0).and.(z.ge.0.5)) then

  do    j = lo(2), hi(2)
   y = prob_lo(2) + (dble(j)+0.5d0) * dx(2)
    if((y.le.1.0).and.(y.ge.0.5)) then

     do i = lo(1), hi(1)
      x = prob_lo(1) + (dble(i)+0.5d0) * dx(1)
       if((x.le.1.0).and.(x.ge.0.5)) then

               uout(i,j,k,0) = 0.0d0
               uout(i,j,k,1) = 0.0d0
               uout(i,j,k,2) = 0.0d0

       endif
     end do
    endif
  end do
 endif
end do

end subroutine fill_metallic_materials
```

Seems to be some issue with the material properties defined in the DEFINES.H, and the routines in ep_mu_3d.f90

Already contacted Eduardo to look at this

First simulations





Material definition is not working correctly.
Also, initial conditions need to be checked

Outline

1. GitHub organization

2. Wake potential algorithm

3. Box resonator

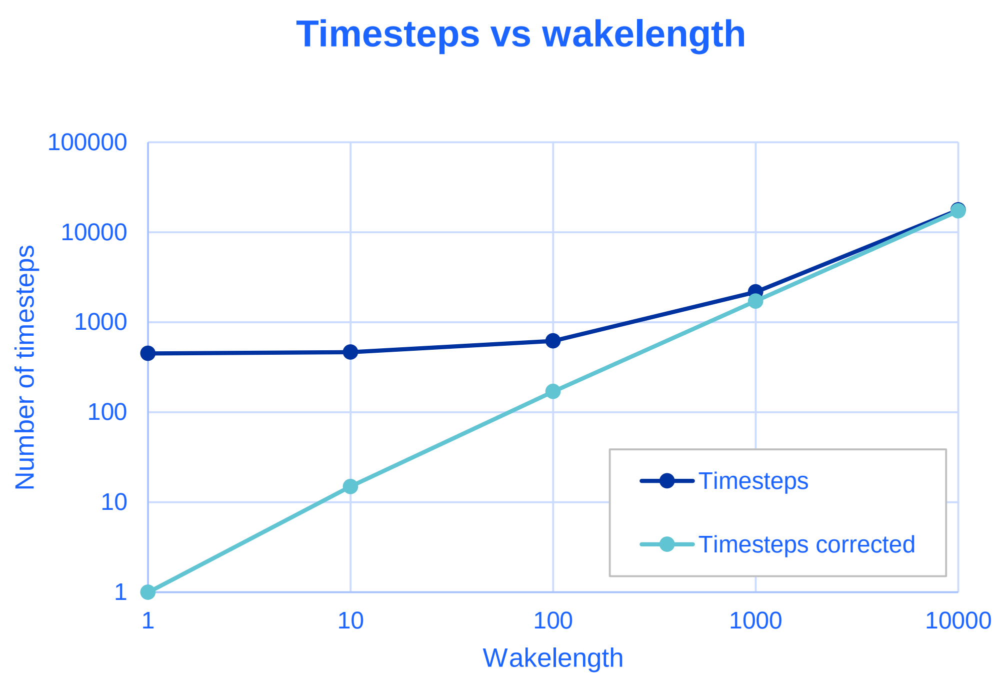
Conclusions and next steps

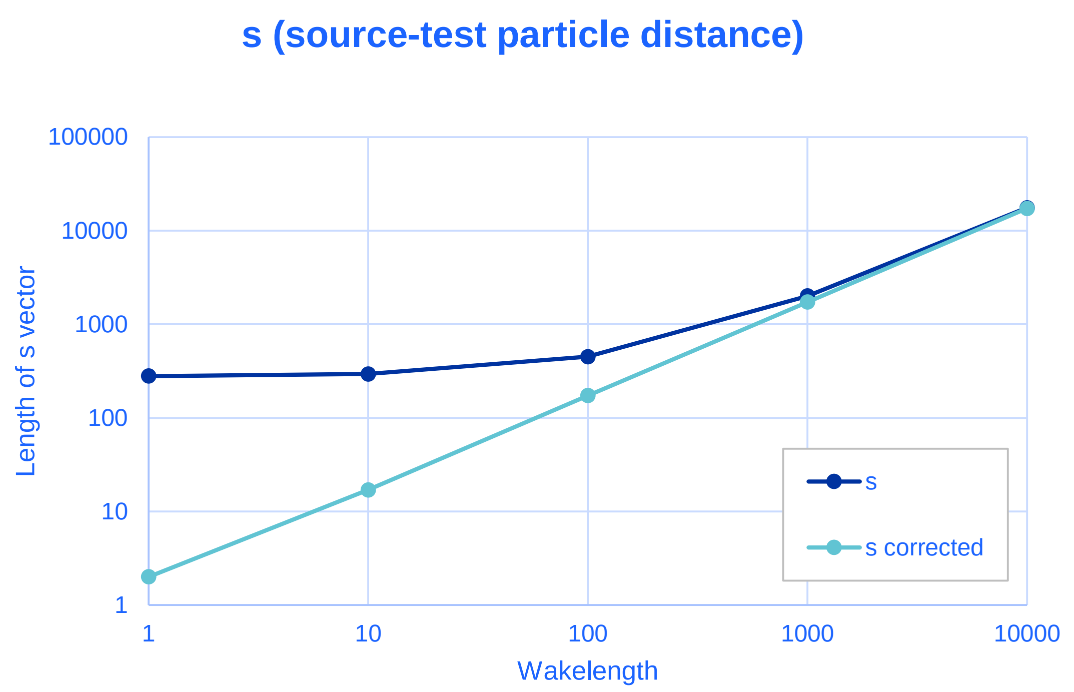
Next steps

- Debug and try the implemented algorithm
- Define the relation between time and the s vector
- Fix the problems with the Box resonator test and continue with the convergence analysis and the energy dissipation test
- Is the Napoly alogorithm needed?

CERN

Plots of the wakelength vs timestep study


Timesteps vs wakelength
Number of timesteps
Wakelength
Timesteps
Timesteps corrected


s (source-test particle distance)
Length of s vector
Wakelength
s
s corrected