“Table-Top Tunable Chiral Photonic Emitter” https://journals.aps.org/prl/abstract/10.1103/PhysRevLett.133.113804)

–code package for C++ (heavy code with FDTD)–

Lu Wang

September 11, 2024

1 Copyright (MIT License)

2 Introduction

This code serves as the numerical package for the publication: “Table-Top Tunable Chiral Photonic Emitter” `https://journals.aps.org/prl/abstract/10.1103/PhysRevLett.133.113804`, where details related to the underlying physics can be found in the supplementary material. The path of the files is indicated by “.../...”, and the variables are referred to by variable. All the numbers are in SI units.

This code is written in C++. It is paralleled using OpenMP on an HPC cluster. It is developed to describe the emission of the high-harmonic electric fields driven by a laser pulse after propagating through a Weyl semimetal (WSM) heterostructure. It is valid for:

1. analysing currents along y, z dimensions
2. geometry of a single WSM slab or multiple WSM slabs separated by the air
3. input driving fields at both IR and THz frequency ranges
4. checking the electric field at any spatial location and time

3 Configure the code

The code requires some configuration of the computing system to get working.

1. in addition to the standard C++ library, two external libraries OpenMP and EIGEN are required
2. the path of the EIGEN should be linked inside the Makefile (see Fig.1)
3. the OpenMP should be linked by a flag (see Fig.1)

```
.SUFFIXES: .c .cpp .o .ex


  CC =mpic++  -fopenmp  -ggdb  -Wall  -std=gnu++17  -Wfatal-errors

INCLUDE= -I /public1/soft/eigen/3.4.0/include/eigen3

.cpp.o:
  $(CC) $(INCLUDE) -c $<

.c.o:
  gcc $(INCLUDE) -c $<

.o.ex:
  @echo g++ ... -o $@ $< ... $(OBJS) ... $(LIBS)
  @$(CC) -o $@ $< $(OBJS) $(LIBS)

#####################################################################
# targets
#####################################################################
OBJS  =

objs: $(OBJS)

clean:
  rm -f *.o *.ex *~

编码: ascii
```

Link openmp flag

Change to your own path of Eigen3

Figure 1: Two things to modify in the Makefile. DO NOT TOUCH ANYTHING ELSE IN THE MAKE FILE!

4 Run the code

In the terminal, use the command `cd` into the target path (the path contains the `main.cpp`). Create the executable from the code using

```
make main.ex
```

Execute the code via the batch file (SLURM system)

```
sbatch best_lu.sh
```

Alternatively one can also execute it directly via

```
./main.ex 1
```

where the integer “1” is associated with the air gap distance between the two WSM slabs via the following line in `main.cpp` line 71,

```
double L_air=(150e-9)*atof(argv[1]);
```

It means the air gap distance is set to be $1 \times 150\text{e}^{-9} = 150$ nm.
Similar as the package 1, to customize the code, please follow the steps below.

Define constants: physical constants can be defined by class C under the path `my_func/constants_file.hpp`.

Define input electric field: input driving electric field can be defined by class E under the path `my_mesh/input_field.hpp`.

Numerical method: everything related to the nonlinear current calculation and the finite difference time domain method (FDTD) is defined by the class WSM under the path `my_num_method/`

Change the input wavelength: switching from the IR driving pulse to THz driving pulse is achieved by simply enable/disable the following paragraph of codes in `main.cpp` starting from line 45

```
//------------------
//for IR
//double lambda   = 1000e-9;
//double tau = 10e-15;
```

```
//double E0=1e8;
//int t_check_num=6000;
//for THz
double lambda    = 10*1000e-9;
double tau = 40e-15;
double E0=1e7;
int t_check_num=20000;
```

5 Print out messages while running

If executed correctly, you should see the following print messages in the terminal (or in the .o file if you submit via the batch file). The printed-out numbers are the iterations in time.

```
a single node
SLURM_JOB_ID           11747120
SLURM_ARRAY_JOB_ID     11747120
SLURM_ARRAY_TASK_ID    1
SLURM_ARRAY_TASK_COUNT 1
SLURM_ARRAY_TASK_MAX   1
SLURM_ARRAY_TASK_MIN   1
SLURM_JOB_NODELIST     f0509
total threads=128
0
save check point iteration0 to my_output/mesh60_b0p06_Ef100_ConHalf_xyApart_E2000.00E5V_per_mtau10fs_1.00um_Lwsm100nm_Lair450nm_dx6nm/
500
1000
1500
2000
2500
3000
save check point iteration3000 to my_output/mesh60_b0p06_Ef100_ConHalf_xyApart_E2000.00E5V_per_mtau10fs_1.00um_Lwsm100nm_Lair450nm_dx6nm/
3500
4000
4500
****************************************************************
4725
****************************************************************
5000
5500
6000
save check point iteration6000 to my_output/mesh60_b0p06_Ef100_ConHalf_xyApart_E2000.00E5V_per_mtau10fs_1.00um_Lwsm100nm_Lair450nm_dx6nm/
6500

码: ascii
```

Figure 2: The iteration of time is printed out every 500 iterations. The checkpoint data is saved every 3000 iterations.

6 Output data

All the output files including the data saved at the checkpoint are stored under the path `my_output`. Under this path you should see the saving files as the following

Eyz_3location_t0.txt	2024-02-27 18:03:19	4.05MB	.txt
Eyz_3location_t12000.txt	2024-02-27 21:18:24	4.08MB	.txt
Eyz_3location_t3000.txt	2024-02-27 18:03:53	4.05MB	.txt
Eyz_3location_t6000.txt	2024-02-27 18:36:06	4.06MB	.txt
Eyz_3location_t9000.txt	2024-02-27 19:55:15	4.07MB	.txt
Eyz_t.txt	2024-02-27 18:03:19	2.06MB	.txt
Eyz_x_dis0.txt	2024-02-27 18:03:20	87.66KB	.txt
Eyz_x_dis12000.txt	2024-02-27 21:18:24	89.44KB	.txt
Eyz_x_dis3000.txt	2024-02-27 18:03:53	88.31KB	.txt
Eyz_x_dis6000.txt	2024-02-27 18:36:06	89.47KB	.txt
Eyz_x_dis9000.txt	2024-02-27 19:55:15	90.01KB	.txt
save_note.txt	2024-02-27 18:04:11	126B	.txt
saveNx.txt	2024-02-27 21:18:25	17B	.txt
t.txt	2024-02-27 18:03:19	738.92KB	.txt
x.txt	2024-02-27 18:03:19	45.82KB	.txt

Figure 3: The saved data.

The ones with file names ending with numbers are saved at checkpoints with that time iteration. The ones without a number ending are the final output files after the entire calculation. Inside the `saveNx.txt`, one can find 3 numbers indicating the location of the saved electric field along the propagation direction

```
399
1865
1948
```

This means, combined with the `x.txt` file, the first position of the electric field is saved at `x[399]`. These three numbers correspond to before the WSM slabs, in between the WSM slabs, and after the WSM slabs. The transmitted and reflected electric fields correspond to the first and last numbers in the `saveNx.txt`. The saved electric field data is stored in the file `Eyz_3location_t.txt`. Ploting this file with `t.txt`, you will see the electric field distribution over time.

Enjoy life and happy coding ♡. Any questions please address to `lu.wangTHz@outlook.com`