EPOCH1D surrogate model (PX915 Project)

Generated by Doxygen 1.8.17

Chapter 1

epoch_surra

Repo for PX915 summer project

Chapter 2

Modules Index

2.1 Modules List

Here is a list of all documented modules with brief descriptions:

Chapter 3

Data Type Index

3.1 Data Types List

Here are the data types with brief descriptions:

Chapter 4

Module Documentation

4.1 epoch_calculator Namespace Reference

Classes

- class **dist_f**

  dist_f *(*p. *15) Class.*
- class **EM_fields**

  EM_fields *(*p. *17) Class.*
- class **Laser_Plasma_Params**

  Laser_Plasma_Params *(*p. *24) Class.*

Functions

- def **winsincFIR** (omega_c, omega_s, M)

  winsincFIR
- def **bandpass** (w0, bw, omega_s, M)

  bandpass
- def **moving_av** (Q, span, period=10)

  moving_av
- def **plasmon** (ne)

  plasmon
- def **dispersion_Stokes** (k, k0, ne, omega0)

  dispersion_Stokes
- def **dispersion_EPW** (k, ne, v_th)

  dispersion_EPW
- def **dispersion_EM** (k, ne)

  dispersion_EM
- def **srs_matching** (k, k0, ne, v_th, omega0)

  srs_matching

Variables

- **c** = constants.c
- **eps0** = constants.epsilon_0
- **me** = constants.m_e
- **e** = constants.e
- **kB** = constants.k
- tuple **keV_to_K** = (e∗1e3)/kB
- **mu0** = constants.mu_0
- **pi** = np.pi
- int **pico** = 1e-12
- int **micron** = 1e-6
- int **nano** = 1e-9

4.1.1 Detailed Description

Documentation for **epoch_calculator** (p. 7) module

The epoch_calaculator module reads the EPOCH data files and calculates several parameters. There are three classes, one which handles the calculations from grid quantities (**Laser_Plasma_Params** (p. 24)), field quantities (EM_fileds) and the momentum distribution function (**dist_f** (p. 15)).

4.1.2 Function Documentation

4.1.2.1 bandpass()

```
def epoch_calculator.bandpass (
              w0,
              bw,
              omega_s,
              M )
```

bandpass

Create a band-pass filter by convolving a high-pass and a low-pass filter

Parameters

w0	: central frequency you want to filter around (fraction of omega0)
bw	: total bandwidth of your filter (fraction of omega0)
M	: half filter length (must be odd)

4.1.2.2 dispersion_EM()

```
def epoch_calculator.dispersion_EM (
              k,
              ne )
```

dispersion_EM

EM wave in plasama dipersion relation

Parameters

k	: EM wavenumber in plasma
ne	: Electron number density

4.1.2.3 dispersion_EPW()

```
def epoch_calculator.dispersion_EPW (
              k,
              ne,
              v_th )
```

dispersion_EPW

Electron Plasma wave dispersion realtion - Bohm-Gross

Parameters

k	: EPW wavenumber in plasma
ne	: Electron number density
v↩_th	: Electron thermal velocity

4.1.2.4 dispersion_Stokes()

```
def epoch_calculator.dispersion_Stokes (
              k,
              k0,
              ne,
              omega0 )
```

dispersion_Stokes

Stokes dispersion curve (Stokes branch) (maximal SRS growth where this curve intersects EPW curve)

Parameters

k	: Wavenumber in plasma
k0	: Vacuum wavenumber (laser)
ne	: Electron number density
omega0	: Laser frequency

4.1.2.5 moving_av()

```
def epoch_calculator.moving_av (
              Q,
              span,
              period = 10 )
```

moving_av

Finds movng average of an array using scipys uniform_filter1d function

Parameters

Q	: Data array
span	: length of data
period	: period to average over

4.1.2.6 plasmon()

```
def epoch_calculator.plasmon (
              ne )
```

plasmon

Calculates electron-plasma frequency

Parameters

ne	: Electron number density

4.1.2.7 srs_matching()

```
def epoch_calculator.srs_matching (
              k,
              k0,
              ne,
              v_th,
              omega0 )
```

srs_matching

SRS frequency matching condition (SRS when it returns zero)

Parameters

k	: Wavenumber in plasma
k0	: Vacuum wavenumber (laser)
ne	: Electron number density
v_th	: Electron thermal velocity
omega0	: Laser frequency

4.1.2.8 winsincFIR()

```
def epoch_calculator.winsincFIR (
              omega_c,
              omega_s,
              M )
```

winsincFIR

Windowed sinc filter function (`http://www.dspguide.com/ch16/2.htm` (Equation 16-4))

Parameters

omega↩ _c	: cutoff frequency
omega↩ _s	: sampling rate (sampling frequency)
M	: length of the filter kernel (must be odd)

4.2 utils Namespace Reference

Functions

- def **expav** (a, t)

  expav
- def **read_intensity** (dir)

  read_intensity
- def **replace_line** (line_in, line_out, fname)

  replace_line
- def **run_epoch** (intensity, data_dir='Data', output=False, np=10)

  run_epoch
- def **get_I_SRS_res** (I_array, dir, np=10)

  get_I_SRS_res
- def **append_list_as_row** (file_name, list_of_elem)

  append_list_as_row

4.2.1 Detailed Description

Documentation for utils module

The utils module houses functions which are used to either read data, run simulations or just mathematical definitions that aren't relate to plasma physics.

4.2.2 Function Documentation

4.2.2.1 append_list_as_row()

```
def utils.append_list_as_row (
              file_name,
              list_of_elem )
```

append_list_as_row

Append data to csv file (for appending I and I_SRS result)

Parameters

file_name	: file name of csv file
list_of_elem	: list to write to csv file

4.2.2.2 expav()

```
def utils.expav (
              a,
              t )
```

expav

Calculates exponential moving average (EMA) mu_i = exp(-1/T)mu_i-1 + (1-exp(-1/T))∗x_i

Parameters

a	: Array
t	: Period/Lengthscale of EMA

4.2.2.3 get_I_SRS_res()

```
def utils.get_I_SRS_res (
              I_array,
              dir,
              np = 10 )
```

get_I_SRS_res

Runs epoch1d simulations for changing intensity and ouptputs backsactter SRS intensity

Parameters

I_array	: Intensity array (list of data to sim)
dir	: Directory to store epoch data to and where the input.deck file is
np	: Number of processors to eun epoch1d on (MPI)

4.2.2.4 read_intensity()

```
def utils.read_intensity (
              dir )
```

read_intensity

Read intensity in W/cm2 from input.deck

Parameters

dir	: Directory which holds input.deck file you want to read (str)

4.2.2.5 replace_line()

```
def utils.replace_line (
              line_in,
              line_out,
              fname )
```

replace_line

Function rewrite line in input.deck via python

Parameters

line_in	: original line in input.deck
line_out	: repacement of line_in in input.deck

4.2.2.6 run_epoch()

```
def utils.run_epoch (
              intensity,
              data_dir = 'Data',
              output = False,
              np = 10 )
```

run_epoch

Runs epoch1d simulations for set intensity

Parameters

intensity	: Intensity to write in input.deck
data_dir	: Directory to store epoch data to and where the input.deck file is
output	: Ouput to command line (True) or to run.log file (False)
np	: Number of processors to eun epoch1d on (MPI)

4.3 visualiser Namespace Reference

Classes

- class **epoch_plotter**
  epoch_plotter *(p. 22) Class.*

4.3.1 Detailed Description

Documentation for visualiser module

The visualiser module houses functions which are used to perform plotting routines on epoch data.

Chapter 5

Data Type Documentation

5.1 epoch_calculator.dist_f Class Reference

dist_f (p. 15) Class.

Public Member Functions

- def **__init__** (self, dir)

  init
- def **read_dist_data** (self)

  read_dist_data
- def **plot_p_dist_func** (self, scaled_x=False)

  plot_p_dist_func

Public Attributes

- **directory**
- **files**
- **nfiles**
- **p_max**
- **p_min**
- **res**
- **epoch_data**
- **v_th**
- **p_norm**
- **times**
- **dist_funcs**
- **momenta_bins**

5.1.1 Detailed Description

dist_f (p. 15) Class.

Class that reads and ouptputs the electron momentum distrubution function from output dist_ .. .sdf files.

5.1.2 Constructor & Destructor Documentation

5.1.2.1 __init__()

```
def epoch_calculator.dist_f.__init__ (
              self,
              dir )
```

init

The constructor

Parameters

self	: The object pointer
dir	: Directory where data is stored (str)

5.1.3 Member Function Documentation

5.1.3.1 plot_p_dist_func()

```
def epoch_calculator.dist_f.plot_p_dist_func (
              self,
              scaled_x = False )
```

plot_p_dist_func

Plots all distribution functions

Parameters

self	: The object pointer
scaled↩ _x	: (Logical) Scales momentum using p_norm

5.1.3.2 read_dist_data()

```
def epoch_calculator.dist_f.read_dist_data (
              self )
```

read_dist_data

Read and store distrinution functions at output times

Parameters

self	: The object pointer

The documentation for this class was generated from the following file:

- epoch_calculator.py

5.2 epoch_calculator.EM_fields Class Reference

EM_fields (p. 17) Class.

Public Member Functions

- def **__init__** (self, dir)
  init
- def **get_2D_Electric_Field_x** (self)
  get_2D_Electric_Field_x
- def **get_2D_Electric_Field_y** (self)
  get_2D_Electric_Field_y
- def **get_2D_Magnetic_Field_z** (self)
  get_2D_Magnetic_Field_z
- def **get_2D_FFT** (self, field, square_mod=True)
  get_2D_FFT
- def **get_time_FFT** (self, field, square_mod=True)
  get_time_FFT
- def **get_space_FFT** (self, field, square_mod=True)
  get_space_FFT
- def **get_filtered_signals** (self, laser=False, plot_E=False, plot_B=False)
  get_filtered_signals
- def **get_flux** (self, laser=False, time_series=False)
  get_flux
- def **get_flux_grid_av** (self, ncells=50, laser=False)
  get_flux_grid_av

Public Attributes

- **directory**
- **epoch_data**
- **timesteps**
- **nx**

5.2.1 Detailed Description

EM_fields (p. 17) Class.

Class that reads and calculates field quantities from fields_ output files.

5.2.2 Constructor & Destructor Documentation

5.2.2.1 __init__()

```
def epoch_calculator.EM_fields.__init__ (
              self,
              dir )
```

init

The constructor

Parameters

self	: The object pointer
dir	: Directory where data is stored (str)

5.2.3 Member Function Documentation

5.2.3.1 get_2D_Electric_Field_x()

```
def epoch_calculator.EM_fields.get_2D_Electric_Field_x (
              self )
```

get_2D_Electric_Field_x

Get time and space data of Ex field i.e Ex(x,t)

Parameters

self	: The object pointer

5.2.3.2 get_2D_Electric_Field_y()

```
def epoch_calculator.EM_fields.get_2D_Electric_Field_y (
              self )
```

get_2D_Electric_Field_y

Get time and space data of Ey field i.e Ey(x,t)

Parameters

self	: The object pointer

5.2.3.3 get_2D_FFT()

```
def epoch_calculator.EM_fields.get_2D_FFT (
              self,
              field,
              square_mod = True )
```

get_2D_FFT

Get 2D FFT (i.e space and time) for specific field

Parameters

self	: The object pointer
field	: EM Field to FFT (inputs are either 'Ex', 'Ey', 'Bz')
square_mod	: (Logical) outputs the sqaured modulus of the FFT

5.2.3.4 get_2D_Magnetic_Field_z()

```
def epoch_calculator.EM_fields.get_2D_Magnetic_Field_z (
              self )
```

get_2D_Magnetic_Field_z

Get time and space data of Bz field i.e Bz(x,t)

Parameters

self	: The object pointer

5.2.3.5 get_filtered_signals()

```
def epoch_calculator.EM_fields.get_filtered_signals (
              self,
              laser = False,
              plot_E = False,
              plot_B = False )
```

get_filtered_signals

Finds filtered signals of Ey and Bz fields (either laser signal or SRS signal)

Parameters

self	: The object pointer
laser	: (Logical) Whether to output laser sginal (true) or SRS signal (false)
plot↩ _E	: (Logical) Whether to plot the filter result at set grid point to test if it works (Ey field)
plot↩ _B	: (Logical) Whether to plot the filter result at set grid point to test if it works (Bz field)

5.2.3.6 get_flux()

```
def epoch_calculator.EM_fields.get_flux (
              self,
              laser = False,
              time_series = False )
```

get_flux

Finds Poynting flux in x direction Sx = (EyBz-ByEz)/mu0 (SRS produces scattered light with same polarisation as the laser (i.e Ez and By are negliable) thus Sx = EyBz/mu0)

Parameters

self	: The object pointer
laser	: (Logical) Whether to use laser sginal (true) or SRS signal (false)
plot↩ _E	: (Logical) Whether to output the Sx time series (true) or the time average (false)

5.2.3.7 get_flux_grid_av()

```
def epoch_calculator.EM_fields.get_flux_grid_av (
              self,
              ncells = 50,
              laser = False )
```

get_flux_grid_av

Averages Poynting flux over ncells (near LH boundary for backscatter SRS and RH boundary for laser)

Parameters

self	: The object pointer
ncells	: Number of cells to average over (default 50)
laser	: (Logical) Whether to use laser sginal (true) or SRS signal (false)

5.2.3.8 get_space_FFT()

```
def epoch_calculator.EM_fields.get_space_FFT (
              self,
              field,
              square_mod = True )
```

get_space_FFT

Produces 1D space FFT for specific field

Parameters

self	: The object pointer
field	: EM Field to FFT (inputs are either 'Ex', 'Ey', 'Bz')
square_mod	: (Logical) outputs the sqaured modulus of the FFT

5.2.3.9 get_time_FFT()

```
def epoch_calculator.EM_fields.get_time_FFT (
              self,
              field,
              square_mod = True )
```

get_time_FFT

Produces 1D time FFT for specific field

Parameters

self	: The object pointer
field	: EM Field to FFT (inputs are either 'Ex', 'Ey', 'Bz')
square_mod	: (Logical) outputs the sqaured modulus of the FFT

The documentation for this class was generated from the following file:

- epoch_calculator.py

5.3 visualiser.epoch_plotter Class Reference

epoch_plotter (p. 22) Class.

Public Member Functions

- def **__init__** (self, dir)
  init
- def **density_plot** (self)
  density_plot
- def **dispersion_2D_plot** (self, field)
  dispersion_2D_plot

Public Attributes

- **directory**
- **epoch_data**
- **epoch_fields**

5.3.1 Detailed Description

epoch_plotter (p. 22) Class.

Class that contains plotting routines that are oten used.

5.3.2 Constructor & Destructor Documentation

5.3.2.1 __init__()

```
def visualiser.epoch_plotter.__init__ (
              self,
              dir )
```

init

The constructor

Parameters

self	: The object pointer
dir	: Directory where data is stored (str)

5.3.3 Member Function Documentation

5.3.3.1 density_plot()

```
def visualiser.epoch_plotter.density_plot (
              self )
```

density_plot

Plots the number desnity over space

Parameters

self	: The object pointer

5.3.3.2 dispersion_2D_plot()

```
def visualiser.epoch_plotter.dispersion_2D_plot (
              self,
              field )
```

dispersion_2D_plot

Plots 2D FFT of the fields

Parameters

self	: The object pointer
field	: EM Field to FFT (inputs are either 'Ex', 'Ey', 'Bz')

The documentation for this class was generated from the following file:

- visualiser.py

5.4 epoch_calculator.Laser_Plasma_Params Class Reference

Laser_Plasma_Params (p. 24) Class.

Public Member Functions

- def **__init__** (self, dir)

  init
- def **read_data** (self)

  read_data
- def **get_spatio_temporal** (self, mic=True)

  get_spatio_temporal
- def **get_plasma_param** (self)

  get_plasma_param
- def **get_matching_conds** (self)

  get_matching_conds

Public Attributes

- **directory**
- **intensity**
- **wavelength**
- **timesteps**
- **omega0**
- **critical_density**
- **k0_vac**
- **grid_data**
- **field_data_0**
- **field_data_1**
- **data_final**
- **grid**
- **nodes**
- **dx**
- **Lx**
- **nx**
- **dt**
- **t_end**
- **time**
- **k_space**
- **omega_space**
- **ne_data**
- **ne**
- **ne_min**
- **ne_max**
- **Ln**
- **omega_pe_data**
- **omega_pe**
- **omega_pe_min**
- **omega_pe_max**
- **k0**
- **Te_data**
- **Te**
- **Te_kelvin**
- **v_th**
- **deb_len**
- **k_epw_bs**
- **omega_epw_bs**
- **k_bs**
- **omega_bs**
- **omega_bs_norm**
- **k_bs_norm**
- **k_epw_bs_norm**
- **k_epw_fs**
- **omega_epw_fs**
- **k_fs**
- **omega_fs**
- **omega_fs_norm**
- **k_fs_norm**
- **k_epw_fs_norm**

5.4.1 Detailed Description

Laser_Plasma_Params (p. 24) Class.

Class that reads and calculates plasama and grid quantities from grid_ output files.

5.4.2 Constructor & Destructor Documentation

5.4.2.1 __init__()

```
def epoch_calculator.Laser_Plasma_Params.__init__ (
              self,
              dir )
```

init

The constructor

Parameters

self	: The object pointer
dir	: Directory where data is stored (str)

5.4.3 Member Function Documentation

5.4.3.1 get_matching_conds()

```
def epoch_calculator.Laser_Plasma_Params.get_matching_conds (
              self )
```

get_matching_conds

Calculates SRS scattered wavenumber and frequency

Parameters

self	: The object pointer

5.4.3.2 get_plasma_param()

```
def epoch_calculator.Laser_Plasma_Params.get_plasma_param (
              self )
```

get_plasma_param

Calculates plasma parameters/variables

Parameters

self	: The object pointer

5.4.3.3 get_spatio_temporal()

```
def epoch_calculator.Laser_Plasma_Params.get_spatio_temporal (
              self,
              mic = True )
```

get_spatio_temporal

Reads the initial grid data and other files required to find sim data

Parameters

self	: The object pointer
mic	: (Logical) Output grid in microns

5.4.3.4 read_data()

```
def epoch_calculator.Laser_Plasma_Params.read_data (
              self )
```

read_data

Reads the initial grid data and other files required to find sim data

Parameters

self	: The object pointer

The documentation for this class was generated from the following file:

- epoch_calculator.py

Index