

EPOCH1D surrogate model (PX915 Project)

Generated by Doxygen 1.8.17

1 epoch_surra	1
2 Modules Index	3
2.1 Modules List	3
3 Data Type Index	5
3.1 Data Types List	5
4 Module Documentation	7
4.1 epoch_calculator Namespace Reference	7
4.1.1 Detailed Description	8
4.1.2 Function Documentation	8
4.1.2.1 bandpass()	8
4.1.2.2 dispersion_EM()	8
4.1.2.3 dispersion_EPW()	9
4.1.2.4 dispersion_Stokes()	9
4.1.2.5 moving_av()	10
4.1.2.6 plasmon()	10
4.1.2.7 srs_matching()	10
4.1.2.8 winsincFIR()	11
4.2 utils Namespace Reference	11
4.2.1 Detailed Description	12
4.2.2 Function Documentation	12
4.2.2.1 append_list_as_row()	12
4.2.2.2 expav()	12
4.2.2.3 get_I_SRS_res()	12
4.2.2.4 read_intensity()	13
4.2.2.5 replace_line()	13
4.2.2.6 run_epoch()	14
4.3 visualiser Namespace Reference	14
4.3.1 Detailed Description	14
5 Data Type Documentation	15
5.1 epoch_calculator.dist_f Class Reference	15
5.1.1 Detailed Description	15
5.1.2 Constructor & Destructor Documentation	16
5.1.2.1 __init__()	16
5.1.3 Member Function Documentation	16
5.1.3.1 plot_p_dist_func()	16
5.1.3.2 read_dist_data()	16
5.2 epoch_calculator.EM_fields Class Reference	17
5.2.1 Detailed Description	17
5.2.2 Constructor & Destructor Documentation	18
5.2.2.1 __init__()	18

5.2.3 Member Function Documentation	18
5.2.3.1 get_2D_Electric_Field_x()	18
5.2.3.2 get_2D_Electric_Field_y()	18
5.2.3.3 get_2D_FFT()	19
5.2.3.4 get_2D_Magnetic_Field_z()	19
5.2.3.5 get_filtered_signals()	19
5.2.3.6 get_flux()	21
5.2.3.7 get_flux_grid_av()	21
5.2.3.8 get_space_FFT()	22
5.2.3.9 get_time_FFT()	22
5.3 visualiser.epoch_plotter Class Reference	22
5.3.1 Detailed Description	23
5.3.2 Constructor & Destructor Documentation	23
5.3.2.1 __init__()	23
5.3.3 Member Function Documentation	23
5.3.3.1 density_plot()	24
5.3.3.2 dispersion_2D_plot()	24
5.4 epoch_calculator.Laser_Plasma_Params Class Reference	24
5.4.1 Detailed Description	26
5.4.2 Constructor & Destructor Documentation	26
5.4.2.1 __init__()	26
5.4.3 Member Function Documentation	26
5.4.3.1 get_matching_conds()	26
5.4.3.2 get_plasma_param()	27
5.4.3.3 get_spatio_temporal()	27
5.4.3.4 read_data()	27
Index	29

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:

epoch_calculator	7
utils	11
visualiser	14

Chapter 3

Data Type Index

3.1 Data Types List

Here are the data types with brief descriptions:

epoch_calculator.dist_f	
Dist_f Class	15
epoch_calculator.EM_fields	
EM_fields (p. 17) Class	17
visualiser.epoch_plotter	
Epoch_plotter Class	22
epoch_calculator.Laser_Plasma_Params	
Laser_Plasma_Params (p. 24) Class	24

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_calculator 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_files) 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

<i>w0</i>	: central frequency you want to filter around (fraction of omega0)
<i>bw</i>	: total bandwidth of your filter (fraction of omega0)
<i>M</i>	: 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

<i>k</i>	: EM wavenumber in plasma
<i>ne</i>	: 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

<i>k</i>	: EPW wavenumber in plasma
<i>ne</i>	: Electron number density
$v_{\leftrightarrow th}$: 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

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

4.1.2.5 moving_av()

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

moving_av

Finds moving average of an array using scipys uniform_filter1d function

Parameters

<i>Q</i>	: Data array
<i>span</i>	: length of data
<i>period</i>	: period to average over

4.1.2.6 plasmon()

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

plasmon

Calculates electron-plasma frequency

Parameters

<i>ne</i>	: 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

<i>k</i>	: Wavenumber in plasma
<i>k0</i>	: Vacuum wavenumber (laser)
<i>ne</i>	: Electron number density
<i>v_th</i>	: Electron thermal velocity
<i>omega0</i>	: 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

<i>omega_c</i>	: cutoff frequency
<i>omega_s</i>	: sampling rate (sampling frequency)
<i>M</i>	: 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

<i>file_name</i>	: file name of csv file
<i>list_of_elem</i>	: 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

<i>a</i>	: Array
<i>t</i>	: 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 outputs backsatter SRS intensity

Parameters

<i>I_array</i>	: Intensity array (list of data to sim)
<i>dir</i>	: Directory to store epoch data to and where the input.deck file is
<i>np</i>	: Number of processors to run 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

<i>dir</i>	: 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

<i>line_in</i>	: original line in input.deck
<i>line_out</i>	: replacement 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

<i>intensity</i>	: Intensity to write in input.deck
<i>data_dir</i>	: Directory to store epoch data to and where the input.deck file is
<i>output</i>	: Ouput to command line (True) or to run.log file (False)
<i>np</i>	: 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 ouputs 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

<i>self</i>	: The object pointer
<i>dir</i>	: 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

<i>self</i>	: The object pointer
<i>scaled_x</i>	: (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

<i>self</i>	: 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

<i>self</i>	: The object pointer
<i>dir</i>	: 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

<i>self</i>	: 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

<i>self</i>	: 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

<i>self</i>	: The object pointer
<i>field</i>	: EM Field to FFT (inputs are either 'Ex', 'Ey', 'Bz')
<i>square_mod</i>	: (Logical) outputs the squared 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

<i>self</i>	: 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

<i>self</i>	: The object pointer
<i>laser</i>	: (Logical) Whether to output laser signal (true) or SRS signal (false)
<i>plot↔ _E</i>	: (Logical) Whether to plot the filter result at set grid point to test if it works (Ey field)
<i>plot↔ _B</i>	: (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 $S_x = (E_y B_z - B_y E_z) / \mu_0$ (SRS produces scattered light with same polarisation as the laser (i.e E_z and B_y are negligible) thus $S_x = E_y B_z / \mu_0$)

Parameters

<i>self</i>	: The object pointer
<i>laser</i>	: (Logical) Whether to use laser signal (true) or SRS signal (false)
<i>plot↔ _E</i>	: (Logical) Whether to output the S_x 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

<i>self</i>	: The object pointer
<i>ncells</i>	: Number of cells to average over (default 50)
<i>laser</i>	: (Logical) Whether to use laser signal (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

<i>self</i>	: The object pointer
<i>field</i>	: EM Field to FFT (inputs are either 'Ex', 'Ey', 'Bz')
<i>square_mod</i>	: (Logical) outputs the squared 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

<i>self</i>	: The object pointer
<i>field</i>	: EM Field to FFT (inputs are either 'Ex', 'Ey', 'Bz')
<i>square_mod</i>	: (Logical) outputs the squared 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 often used.

5.3.2 Constructor & Destructor Documentation

5.3.2.1 __init__()

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

init

The constructor

Parameters

<i>self</i>	: The object pointer
<i>dir</i>	: 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

<i>self</i>	: 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

<i>self</i>	: The object pointer
<i>field</i>	: 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

<i>self</i>	: The object pointer
<i>dir</i>	: 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

<i>self</i>	: 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

<i>self</i>	: 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

<i>self</i>	: The object pointer
<i>mic</i>	: (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

<i>self</i>	: The object pointer
-------------	----------------------

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

- epoch_calculator.py

Index

- `__init__`
 - `epoch_calculator.dist_f`, 16
 - `epoch_calculator.EM_fields`, 18
 - `epoch_calculator.Laser_Plasma_Params`, 26
 - `visualiser.epoch_plotter`, 23
- `append_list_as_row`
 - `utils`, 12
- `bandpass`
 - `epoch_calculator`, 8
- `density_plot`
 - `visualiser.epoch_plotter`, 23
- `dispersion_2D_plot`
 - `visualiser.epoch_plotter`, 24
- `dispersion_EM`
 - `epoch_calculator`, 8
- `dispersion_EPW`
 - `epoch_calculator`, 9
- `dispersion_Stokes`
 - `epoch_calculator`, 9
- `epoch_calculator`, 7
 - `bandpass`, 8
 - `dispersion_EM`, 8
 - `dispersion_EPW`, 9
 - `dispersion_Stokes`, 9
 - `moving_av`, 10
 - `plasmon`, 10
 - `srs_matching`, 10
 - `winsincFIR`, 11
- `epoch_calculator.dist_f`, 15
 - `__init__`, 16
 - `plot_p_dist_func`, 16
 - `read_dist_data`, 16
- `epoch_calculator.EM_fields`, 17
 - `__init__`, 18
 - `get_2D_Electric_Field_x`, 18
 - `get_2D_Electric_Field_y`, 18
 - `get_2D_FFT`, 19
 - `get_2D_Magnetic_Field_z`, 19
 - `get_filtered_signals`, 19
 - `get_flux`, 21
 - `get_flux_grid_av`, 21
 - `get_space_FFT`, 22
 - `get_time_FFT`, 22
- `epoch_calculator.Laser_Plasma_Params`, 24
 - `__init__`, 26
 - `get_matching_conds`, 26
- `get_plasma_param`, 26
- `get_spatio_temporal`, 27
- `read_data`, 27
- `expav`
 - `utils`, 12
- `get_2D_Electric_Field_x`
 - `epoch_calculator.EM_fields`, 18
- `get_2D_Electric_Field_y`
 - `epoch_calculator.EM_fields`, 18
- `get_2D_FFT`
 - `epoch_calculator.EM_fields`, 19
- `get_2D_Magnetic_Field_z`
 - `epoch_calculator.EM_fields`, 19
- `get_filtered_signals`
 - `epoch_calculator.EM_fields`, 19
- `get_flux`
 - `epoch_calculator.EM_fields`, 21
- `get_flux_grid_av`
 - `epoch_calculator.EM_fields`, 21
- `get_I_SRS_res`
 - `utils`, 12
- `get_matching_conds`
 - `epoch_calculator.Laser_Plasma_Params`, 26
- `get_plasma_param`
 - `epoch_calculator.Laser_Plasma_Params`, 26
- `get_space_FFT`
 - `epoch_calculator.EM_fields`, 22
- `get_spatio_temporal`
 - `epoch_calculator.Laser_Plasma_Params`, 27
- `get_time_FFT`
 - `epoch_calculator.EM_fields`, 22
- `moving_av`
 - `epoch_calculator`, 10
- `plasmon`
 - `epoch_calculator`, 10
- `plot_p_dist_func`
 - `epoch_calculator.dist_f`, 16
- `read_data`
 - `epoch_calculator.Laser_Plasma_Params`, 27
- `read_dist_data`
 - `epoch_calculator.dist_f`, 16
- `read_intensity`
 - `utils`, 13
- `replace_line`
 - `utils`, 13
- `run_epoch`

- utils, 13
- srs_matching
 - epoch_calculator, 10
- utils, 11
 - append_list_as_row, 12
 - expav, 12
 - get_I_SRS_res, 12
 - read_intensity, 13
 - replace_line, 13
 - run_epoch, 13
- visualiser, 14
- visualiser.epoch_plotter, 22
 - __init__, 23
 - density_plot, 23
 - dispersion_2D_plot, 24
- winsincFIR
 - epoch_calculator, 11