bnlcrl Documentation

Release 20180115.183948

mrakitin

January 15, 2018

Contents

CRL simulator

CHAPTER 1

bnlcrl package

Subpackages

bnlcrl.pkcli package

Submodules

bnlcrl.pkcli.simulate module

Utilities for X-Ray beamlines.

The module to perform the following operations:

- simulate Compound Refractive Lenses (`CRL`) in the approximation of thick lens;
- get the Index of Refraction (`Delta`) value;
- calculate ideal focal distance.

bnlcrl.pkcli.simulate.**calc_ideal_focus**(*delta*, *n*, *p0*, *radius*)
Calculate ideal focus for the CRL with specified parameters.

Parameters

- **delta** (*float*) – the index of refraction.
- **n** (*int*) – number of lenses in the CRL.
- **p0** (*float*) – distance from source to the CRL [m].
- **radius** (*float*) – radius on tip of parabola [m].

Returns dictionary with the result.

Return type dict

bnlcrl.pkcli.simulate.**find_delta**(*energy*, *calc_delta=False*, *characteristic='delta'*, *data_file=''*, *e_max=30000.0*, *e_min=30.0*, *e_step=10.0*, *formula='Be'*, *n_points=500*, *outfile=''*, *plot=False*, *precise=False*, *save=False*, *save_output=False*, *show_plot=False*, *thickness=0.1*, *use_numpy=False*, *verbose=False*)
Determine the Index of Refraction (delta).

The index of refraction can be defined by three different methods/approaches:

1. Get delta for the closest energy from the saved *.dat files (see `bnlcrl/package_data/dat/`).
2. Get delta from http://henke.lbl.gov/optical_constants/getdb2.html.
3. Calculate delta analytically (requires `periodictable` package installed).

Parameters

- **calc_delta** (*bool*) – a flag to calculate delta analytically.
- **characteristic** (*str*) – characteristic to be extracted (`atten` - attenuation length, `delta` - index of refraction, `transmission` - filter transmission).
- **data_file** (*str*) – a *.dat data file in `bnlcrl/package_data/dat/` directory with delta values for the material of the CRL (e.g., Be).
- **e_max** (*float*) – the highest available energy [eV].
- **e_min** (*float*) – the lowest available energy [eV].
- **e_step** (*float*) – energy step size used for saving data to a file [eV].
- **energy** (*float*) – photon energy [eV].
- **formula** (*str*) – material's formula of the interest.
- **n_points** (*int*) – number of points to get from the server.
- **outfile** (*str*) – optional output file.
- **plot** (*bool*) – a flag to plot the obtained data.
- **precise** (*bool*) – a flag to find delta within the energy interval +/- 1 eV from the specified energy.
- **save** (*bool*) – a flag to save the obtained data.
- **save_output** (*bool*) – a flag to save the output dictionary in JSON format.
- **show_plot** (*bool*) – a flag to show the show the plot.
- **thickness** (*float*) – thickness of the material.
- **use_numpy** (*bool*) – a flag to use NumPy.
- **verbose** (*bool*) – a flag to print output to console.

Returns dictionary with the result.

Return type dict

`bnlcrl.pkcli.simulate.`**simulate_crl**(*cart_ids, energy, beamline='smi', calc_delta=False, d_ssa_focus=8.1, data_file='Be_delta.dat', dl_cart=0.03, dl_lens=0.002, lens_array=[1, 2, 4, 8, 16], outfile='False', output_format='csv', p0=6.2, r_array=[50, 200, 500], radii_tolerance=1e-08, teta0=6e-05, use_numpy=False, verbose=False*)

Runner of the CRL simulator.

Calculate real CRL under-/over-focusing comparing with the ideal lens.

Example:

```
d = default_command(
    cart_ids=['2', '4', '6', '7', '8'],
    energy=21500,
    p0=6.52,
    verbose=True
)
```

Output:

```
"d","d_ideal","f","p0","p1","p1_ideal"
0.00120167289264,-0.0661303590822,1.0480597835,6.52,1.24879832711,1.31613035908
```

Parameters

- **beamline** (*str*) – beamline name.
- **calc_delta** (*bool*) – a flag to calculate delta analytically.
- **cart_ids** (*list*) – cartridges ids.
- **d_ssa_focus** (*float*) – Distance from SSA [m].
- **data_file** (*str*) – data file with delta values for the material of the CRL (e.g., Be).
- **dl_cart** (*float*) – distance between centers of two neighbouring cartridges [m].
- **dl_lens** (*float*) – distance between two lenses within a cartridge [m].
- **energy** (*float*) – photon energy [eV].
- **lens_array** (*list*) – possible number of lenses in cartridges.
- **outfile** (*str*) – output file.
- **output_format** (*str*) – output file format (CSV, JSON, plain text).
- **p0** (*float*) – distance from z=50.9 m to the first lens in the most upstream cartridge at the most upstream position of the transfocator [m].
- **r_array** (*list*) – radii of available lenses in different cartridges [um].
- **radii_tolerance** (*float*) – tolerance to compare radii [m].
- **teta0** (*float*) – divergence of the beam before CRL [rad].
- **use_numpy** (*bool*) – a flag to use NumPy for operations with matrices.
- **verbose** (*bool*) – a flag to print output to console.

Returns dictionary with the result.

Return type dict

Module contents

Submodules

bnlcrl.base_pkconfig module

Default config

bnlcrl.base_pkconfig.**alpha**()

bnlcrl.base_pkconfig.**beta**()

bnlcrl.base_pkconfig.**dev**()

bnlcrl.base_pkconfig.**prod**()

bnlcrl.bnlcrl_console module

Front-end command line for *bnlcrl*.

See pykern.pkcli for how this module is used.

copyright Copyright (c) 2016 mrakitin. All Rights Reserved.

license http://www.apache.org/licenses/LICENSE-2.0.html

bnlcrl.bnlcrl_console.**main**()

bnlcrl.crl_simulator module

class bnlcrl.crl_simulator.**CRLSimulator**(***kwargs*)

calc_T_total()

calc_delta_focus(*p*)

static calc_ideal_focus(***kwargs*)

calc_ideal_lens()

calc_lens_array(*radius*, *n*)

Calculate accumulated T_fs for one cartridge with fixed radius.

Parameters

- **radius** – radius.
- **n** – number of lenses in one cartridge.

Return T_fs_accum accumulated T_fs.

calc_real_lens()

calc_y_teta()

get_inserted_lenses()

print_result(*output_format=None*)

read_config_file()

bnlcrl.delta_finder module

A library to get index of refraction (delta) or attenuation length.

Author: Maksim Rakitin (BNL) 2016

class bnlcrl.delta_finder.**DeltaFinder**(***kwargs*)

calculate_delta()

print_info()

save_to_file()

bnlcrl.plot_delta module

bnlcrl.utils module

bnlcrl.utils.**console**(*class_name*, *parameters_file*)

bnlcrl.utils.**convert_types**(*input_dict*)
Convert types of values from specified JSON file.

bnlcrl.utils.**create_cli_function**(*function_name*, *parameters*, *config*)
The function creates the content of the CLI functions with the input from JSON config.

Parameters

- **function_name** (*str*) – name of the function.
- **parameters** (*dict*) – dictionary with the parameters of the arguments (default value, help info, type).
- **config** (*dict*) – dictionary with the parameters (descriptions, used class name, returns, parameters (optional)).

Returns resulted function represented as a string.

Return type str

bnlcrl.utils.**defaults_file**(*suffix=None*, *defaults_file_path=None*)

bnlcrl.utils.**get_cli_functions**(*config*)
Get list of CLI functions' content with the input from JSON config.

Parameters **config** (*dict*) – dictionary with the configuration in JSON format.

Returns list of functions' contents.

Return type list

bnlcrl.utils.**read_json**(*file_name*)

bnlcrl.visualize module

bnlcrl.visualize.**plot_data**(*df*, *elements*, *property*, *thickness*, *e_min*, *e_max*, *n_points*, *file_name='data'*, *x_label=None*, *figsize=(10, 6)*, *show_plot=False*)

bnlcrl.visualize.**save_to_csv**(*df*, *file_name='data'*, *index=False*)

bnlcrl.visualize.**to_dataframe**(*d*, *elements*)
Convert a list of strings, each representing the read data, to a Pandas DataFrame object.

Parameters

- **d** – a list of strings, each representing the read data.
- **elements** – Chemical elements of interest.

Returns a tuple of DataFrame and the parsed columns.

Module contents

CHAPTER 2

bnlcrl

CHAPTER 3

Indices and tables

- genindex
- modindex
- search

b