BagelFit

Release 1.0

Neelesh Soni

Mar 06, 2025

CONTENTS:

1 BagelFit **1**
- 1.1 BagelFitter . . . 1
- 1.2 Examples . . . 5
- 1.3 version . . . 6

2 Indices and tables **7**

Python Module Index **9**

Index **11**

CHAPTER

ONE

BAGELFIT

1.1 BagelFitter

1.1.1 BagelFitter Class

class src.BagelFitter.**BagelFitter**

Bases: object

Fits a torus onto a nuclear membrane by searching for the best parameters.

best_torus

The best-fitting torus found during the fitting process.

Type
Torus

dmap

Input density map of the nuclear membrane.

Type
IMP.em.DensityMap

dmap_out

Output density map after fitting.

Type
IMP.em.DensityMap

dmap_out_binary_flag

Flag indicating if the density map is binary.

Type
bool

input_map_path

Path to the input density map file.

Type
str

voxel_size

Size of each voxel in the density map (default is 10).

Type
int

calculate_dice_coefficient(*dmap1: DensityMap*, *dmap2: DensityMap*) → float

Computes the Dice Coefficient (F1 Score) for binary overlap between two density maps.

Parameters

- **dmap1** (*IMP.em.DensityMap*) – First density map.
- **dmap2** (*IMP.em.DensityMap*) – Second density map.

Returns
Dice Coefficient representing the overlap of the two maps.

Return type
float

create_blank_density_map(*n_voxels: int*) → DensityMap

Creates an empty density map centered at (0,0,0).

Parameters
n_voxels (*int*) – Number of voxels in each dimension.

Returns
A blank density map with specified voxel size.

Return type
IMP.em.DensityMap

fill_binary_density(*torus:* Torus) → None

Generates a binary density map based on the torus parameters.

Parameters
torus (Torus) – Torus object used to define the density map.

fill_nonbinary_density(*torus:* Torus) → None

Generates a non-binary density (extracted from the input map) map based on the torus parameters.

Parameters
torus (Torus) – Torus object used to define the density map.

fit_binary_torus(*tor_R_range: tuple = (670, 680, 10)*, *tor_r_range: tuple = (160, 180, 20)*, *tor_th_range: tuple = (85.0, 95, 10)*, *extension: float = 0.0*) → *Torus*

Fits a binary torus to the density map by searching for optimal parameters.

Parameters

- **tor_R_range** (*tuple, optional*) – Range of major radius values (start, stop, step).
- **tor_r_range** (*tuple, optional*) – Range of minor radius values (start, stop, step).
- **tor_th_range** (*tuple, optional*) – Range of thickness values (start, stop, step).
- **extension** (*float, optional*) – Additional extension factor (default is 0.0).

Returns
Best-fitting torus object based on maximum cross-correlation coefficient.

Return type
Torus

fit_nonbinary_torus(*tor_R_range: tuple = (670, 680, 10)*, *tor_r_range: tuple = (160, 180, 20)*, *tor_th_range: tuple = (85.0, 95, 10)*, *extension: float = 0.0*) → *Torus*

Fits a non-binary torus to the density map by searching for optimal parameters.

Parameters

- **tor_R_range** (*tuple, optional*) – Range of major radius values (start, stop, step).
- **tor_r_range** (*tuple, optional*) – Range of minor radius values (start, stop, step).
- **tor_th_range** (*tuple, optional*) – Range of thickness values (start, stop, step).
- **extension** (*float, optional*) – Additional extension factor (default is 0.0).

Returns
Best-fitting torus object based on maximum cross-correlation coefficient.

Return type
Torus

generate_binary_torus(*tor_R: float*, *tor_r: float*, *tor_th: float*, *extension: float = 0.0*, *boundingbox_length: float = 2240*, *voxel_size: float = 10*, *outmap_fname: str = 'torus_yeast_fitted.mrc'*) → *Torus*

Generates and writes a binary torus density map based on input parameters.

Parameters

- **tor_R** (*float*) – Major radius of the torus.
- **tor_r** (*float*) – Minor radius of the torus.
- **tor_th** (*float*) – Thickness of the bilipid layer.
- **extension** (*float, optional*) – Additional extension factor (default is 0.0).
- **boundingbox_length** (*float, optional*) – Length of the bounding box for Torus centered at (0,0,0). Default value 2240 Å.
- **voxel_size** (*float, optional*) – Individual voxel size in the output map file. Default value 10 Å.
- **outmap_fname** (*str, optional*) – Output map file name of the torus. Default is "torus_yeast_fitted.mrc" in the current directory.

Returns
Torus object based on input parameters.

Return type
Torus

load_exprimental_map(*input_map_path: str*, *voxel_size: int | None = None*) → None

Loads an experimental density map for processing.

Parameters

- **input_map_path** (*str*) – Path to the input density map file.
- **voxel_size** (*int, optional*) – Size of each voxel (default is determined from the map).

plot_voxel_values() → None

Plots the histogram of voxel intensity values in the density map.

score_torus_maps(*map1: str*, *map2: str*) → float

Computes the cross-correlation coefficient between two torus maps.

Parameters

- **map1** (*str*) – Path to the first torus density map.
- **map2** (*str*) – Path to the second torus density map.

Returns
Cross-correlation coefficient indicating similarity between the two maps.

Return type
float

`write_torusmap_to_file`(*outmap_fname: str*) → None

Saves the torus density map to a file.

Parameters
outmap_fname (*str*) – Filename to save the torus density map.

1.1.2 Torus Class

Summary

`class src.Torus.Torus`(*R: float*, *r: float*, *thickness: float*, *extension: float = 0.0*)

Bases: `object`

Represents a toroidal shape characterized by its major radius (R), minor radius (r), and thickness.

R

Major radius of the torus.

Type
float

r

Minor radius of the torus.

Type
float

thickness

Thickness of the torus.

Type
float

extension

Optional extension parameter (default is 0.0).

Type
float

eps

Small constant to avoid division by zero errors.

Type
float

dmap

Data map for storing computed values (if applicable).

Type
Any

contains_point(*x: float*, *y: float*, *z: float*) → bool

Determines whether a point (x, y, z) lies within the toroidal volume.

Parameters

- **x** (*float*) – X-coordinate of the point.
- **y** (*float*) – Y-coordinate of the point.
- **z** (*float*) – Z-coordinate of the point.

Returns
True if the point is inside the torus, False otherwise.

Return type
bool

contains_point2(*x: float*, *y: float*, *z: float*) → bool

Alternative method to check if a point (x, y, z) lies within the toroidal volume.

Parameters

- **x** (*float*) – X-coordinate of the point.
- **y** (*float*) – Y-coordinate of the point.
- **z** (*float*) – Z-coordinate of the point.

Returns
True if the point is inside the torus, False otherwise.

Return type
bool

distance(*x: float*, *y: float*, *z: float*, *d2_xy: float*) → float

Computes the shortest distance from a point (x, y, z) to the torus.

Parameters

- **x** (*float*) – X-coordinate of the point.
- **y** (*float*) – Y-coordinate of the point.
- **z** (*float*) – Z-coordinate of the point.
- **d2_xy** (*float*) – Squared distance from the torus center in the xy-plane.

Returns
Shortest distance from the given point to the torus.

Return type
float

1.2 Examples

1.2.1 examples

Exanple script for generating and scoring torus maps in nuclear membrane fitting.

examples.examples.**generate_bestfit_torus_map**()

Fits several torus models onto the nuclear membrane and saves the best fit.

Returns
None

examples.examples.**generate_binary_torus_map**()

Generates a binary torus map using predefined torus parameters and saves it to a file.

Returns
None

examples.examples.**score_torus_map_with_experimental_map**()

Compares a generated torus map with an experimental map using a scoring function.

Returns
None

1.3 version

This function provides the BagelFit version number

CHAPTER

TWO

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

e

examples.examples, 5

s

src.BagelFitter, 1
src.Torus, 4

v

version, 6

INDEX

B

BagelFitter (*class in src.BagelFitter*), 1
best_torus (*src.BagelFitter.BagelFitter attribute*), 1

C

calculate_dice_coefficient() (*src.BagelFitter.BagelFitter method*), 1
contains_point() (*src.Torus.Torus method*), 4
contains_point2() (*src.Torus.Torus method*), 5
create_blank_density_map() (*src.BagelFitter.BagelFitter method*), 2

D

distance() (*src.Torus.Torus method*), 5
dmap (*src.BagelFitter.BagelFitter attribute*), 1
dmap (*src.Torus.Torus attribute*), 4
dmap_out (*src.BagelFitter.BagelFitter attribute*), 1
dmap_out_binary_flag (*src.BagelFitter.BagelFitter attribute*), 1

E

eps (*src.Torus.Torus attribute*), 4
examples.examples
    module, 5
extension (*src.Torus.Torus attribute*), 4

F

fill_binary_density() (*src.BagelFitter.BagelFitter method*), 2
fill_nonbinary_density() (*src.BagelFitter.BagelFitter method*), 2
fit_binary_torus() (*src.BagelFitter.BagelFitter method*), 2
fit_nonbinary_torus() (*src.BagelFitter.BagelFitter method*), 2

G

generate_bestfit_torus_map() (*in module examples.examples*), 5
generate_binary_torus() (*src.BagelFitter.BagelFitter method*), 3
generate_binary_torus_map() (*in module examples.examples*), 6

I

input_map_path (*src.BagelFitter.BagelFitter attribute*), 1

L

load_exprimental_map() (*src.BagelFitter.BagelFitter method*), 3

M

module
    examples.examples, 5
    src.BagelFitter, 1
    src.Torus, 4
    version, 6

P

plot_voxel_values() (*src.BagelFitter.BagelFitter method*), 3

R

R (*src.Torus.Torus attribute*), 4
r (*src.Torus.Torus attribute*), 4

S

score_torus_map_with_experimental_map() (*in module examples.examples*), 6
score_torus_maps() (*src.BagelFitter.BagelFitter method*), 3
src.BagelFitter
    module, 1
src.Torus
    module, 4

T

thickness (*src.Torus.Torus attribute*), 4
Torus (*class in src.Torus*), 4

V

version

module, 6
voxel_size (*src.BagelFitter.BagelFitter attribute*), 1

W

write_torusmap_to_file()
(*src.BagelFitter.BagelFitter method*), 4