index
Tools c:\users\pabst\desktop\voromink\finalcode\tools.py

Module: Tools
Author: Dominik Pabst

This module provides some basic tools for the module VorominkEstimation

Modules

itertools math numpy time

Functions

R_values(Rmin, Rmax, n)
Computes n values equidistant between Rmin,Rmax (those values excluded)

Args:
  Rmin (float): Lower bound
  Rmax (float): Upper bound
  n (int): Number of values

Returns:
  list: Contains the n values.

average_NN(in_data)
Calculates the average nearest neighbour distance in the data.

Args:
  in_data (numpy.ndarray): Input data

Returns:
  float: Average Nearest Neighbour Distance

create_window(in_data, dist, F=False)
Constructs a cuboid, whose boundary has in each direction distance equal to the parameter dist from the data.

Args:
  in_data (numpy.ndarray): Input data
  dist (float): Distance of the boundary of the output cuboid to the data
  F (boolean,optional): Decides in which format the cuboid is returned

Returns:
  list: Contains the coordinates of the cuboid.
    Example: [-2,2]x[-2,2]
    if F is True, the algorithm returns [[-2,2], [-2,2]]
    if F is False, the algorithm returns [-2, 2, -2, 2]

dist(x, y)
Computes Eudlidean distance of two vectors x and y.

Args:
  x,y (list): Eclidean vectors
Returns:
  float: Computed distance

distance_window(in_data, W)
Calculates the distance of the boundary of the cuboid W to the input data.

Args:

in_data (numpy.ndarray): Input data
W (list): Cuboid, where the data lies in. Should be of the form like the
function create_window creates it with the option F=True

Returns:
float: Distance of the boundary of the cuboid W and the data.

euclidean(x)
Computes Eudlidean norm of a vector x.

Args:
x (list): Eclidean vector
Returns:
float: Computed norm

grid_process(W, res)
Constructs a grid process in an observation window.
This is a grid with resolution res randomly translated (by a uniformly distributed vector).

Args:
window (list): Cuboid representing the observation window where the
data lies. Example: window = [[-2, 2],[0, 1]] represents
the cuboid [-2, 2] x [0, 1]
res (float): Resolution of the grid process.

Returns:
list: Entries are lists itself. The j-th list contains the values
of the j-th coordinates of the points of the grid. Example: [[0,1],[2,3]]
represents the grid consisting of the 4 points (0,2),(0,3),(1,2),(1,3)

grid_process_rotated(W, res)
Constructs a randomly rotated grid process in an observation window.

Args:
window (list): A cuboid representing the observation window where the
data lies. Example: window = [[-2, 2],[0, 1]] represents
the cuboid [-2, 2] x [0, 1]
res (float): The resolution of the grid process.

Returns:
list: The elements of the list are the points of the grid process.

kappa(n)
Computes the n-dimensional volume of the n-dimensional unit ball.

Args:
n (int): Dimension
Returns:
float: Computed Volume

minimal_NN(in_data)
Calculates the minimal nearest neighbour distance in the data.

Args:
in_data (numpy.ndarray): Input data

Returns:
float: Minimal Nearest Neighbour Distance

Data

gamma = <ufunc 'gamma'>