index
VorominkEstimation c:\users\pabst\desktop\voromink\vorominkestimation.py

Module: VorominkEstimation
Author: Dominik Pabst

This module estimates Voronoi and Minkowski tensors based on a given data set.
It includes two main functions:
  `Voromink` for general tensor estimation by solving a least squares problem
  `Vorosurf` for surface Minkowski tensor estimation
Both methods have customizable parameters for precision and algorithm control.

A tensor of rank r is a linear mapping T, which takes r vectors from R^d.
A symmetric tensor T is determined by the values T(e_i1,...,e_ir), where
1 <= i1 <= ... <= ir <= d and e_i is the i-th standard vector in R^d.
In this code a tensor is represented by a dictionary, which has a key (i1,...,ir) for each choice of i1,...,ir.
For example the value corresponding to the key (1,1,2) represents T(e1,e1,e2).

This code is based on the following paper:
  D. Hug, M.A.Klatt, D.Pabst. Minkowski tensors for voxelized data: robust asymptotically unbiased estimators.
Please cite the paper if you use the code.

Modules

Tensors	itertools	pandas
Tools	math	sys
argparse	numpy	time

Functions

Voromink(infile, r, s, n, Rmax, window=None, a=None, verbose=True, rotate=False)
    Estimate the Minkowski tensors of a set represented by a finite data set.
    This approach is consistent for sets with positive reach.

    Args:
        infile (csv or numpy.ndarray): Input data
        r, s (int): Rank parameters of the tensor, which will be estimated.
        n (int): Number of radii, where the Voronoi tensors are evaluated.
        Rmax (float or False): Maximum radius for which the Voronoi tensors are
                            evaluated. If False, the `window` parameter must be provided
                            to compute `Rmax`.
        window (list, optional): A cuboid representing the observation window where the
                            data lies. If `Rmax` is False, this window is used to
                            compute `Rmax`. Example: window = [-2, 2, 0, 1] represents
                            the cuboid [-2, 2] x [0, 1].
        a (float, optional): Average nearest neighbor distance in the input data.
                            If False, the algorithm computes this quantity automatically.
        verbose (bool, optional): If True, the algorithm prints information about the
                            current step.
        rotate (bool, optional): If True, the grid process will be rotated randomly.
                            This option is only necessary for generated test data, which
                            lies parallel to the standard axis. Note that rotation
                            increases computation time.

    Returns:
        dict: A dictionary with the following keys:
            - "Vor": Contains a list with the estimated Voronoi tensors.
            - "Min": Contains a list with the estimated Minkowski tensors Phi_d,...,Phi_0 (in this order)

    Example usage:
        result = Voromink('data.csv', 50, 0, 2, False, window=[-2, 2, 0, 1])
        print(result['Min'])   # Prints the estimated Minkowski tensors

Vorosurf(infile, r, s, epsilon, a=None, verbose=True, rotate=False)

Estimate the surface Minkowski tensors of a set represented by a finite data set.
This approach is consistent for finite unions of compact sets with positive reach.
Works for s>0 and r=s=0. In the latter the tensor is optained as the trace of the tensor for r=0, s=2 times 1/(4*pi).

Args:
infile (csv or numpy.ndarray): Input data
r, s (int): Rank parameters of the tensor to be estimated.
epsilon (float): Value of the radius, where the Voronoi tensor is evaluated.
We recommend this value to be at least 100 times a.
a (float, optional): Resolution of the grid. If None, the algorithm
computes the average nearest neighbor distance in the
input data and uses it for resolution.
verbose (bool, optional): If True, the algorithm prints information about
the current step.
rotate (bool, optional): If True, the grid process will be rotated randomly once.
This is only necessary for generated test data, which
lies parallel to the standard Cartesian axis.

Returns:
dict: Estimated Minkowski tensor.

Example usage:

```
result = Vorosurf('data.csv', 0, 2, 0.1, a=0.001)
print(result)    # Prints the estimated Minkowski surface tensor
```