Topological Data Analysis in Python

organized by:
Michael Bleher, Maximilian Schmahl and Daniel Spitz

Heidelberg University

26^{th} - 28^{th} of October 2020

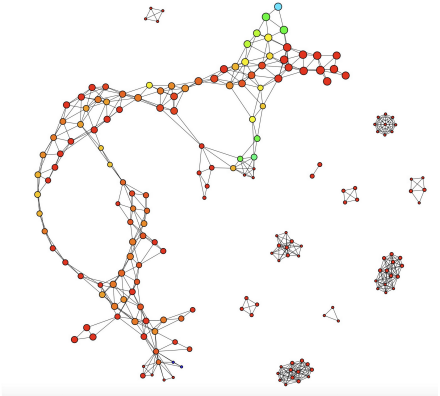
Contents

Contents

Overview sckit-tda and Programme

scikit-tda libraries

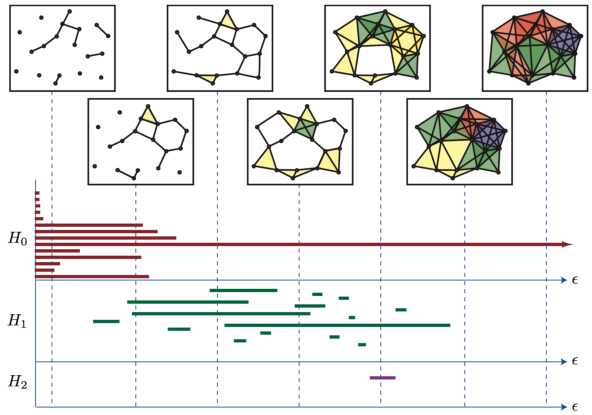
Overview sckit-tda and Programme



scikit-tda libraries

- Kepler Mapper

Overview sckit-tda and Programme

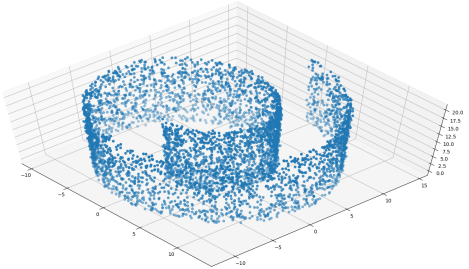


Melissa McGuirl, Lecture Notes, ICERM 2017

scikit-tda libraries

- Kepler Mapper
- Ripser.py
- Persim
- CechMate

Overview sckit-tda and Programme



scikit-tda libraries

- Kepler Mapper
- Ripser.py
- Persim
- CechMate
- TaDAsets

Overview sckit-tda and Programme

	Monday	Tuesday	Wednesday
12-13	Python Pre-Course		
14-16	Welcome **Overview scikit-tda** **Intro to Topology in Data Analysis** Mapper	Intro to Persistent Homology Ripser Guest contribution – Sebastian Damrich	Project
16-18	Tutorials	Tutorials	Project

scikit-tda libraries

- Kepler Mapper
- Ripser.py
- Persim
- CechMate
- TaDAsets

Overview sckit-tda and Programme

	Monday	Tuesday	Wednesday
12-13	Python Pre-Course		
14-16	Welcome Overview scikit-tda Intro to Topology in Data Analysis **Mapper**	Intro to Persistent Homology Ripser Guest contribution – Sebastian Damrich	Project
16-18	**Tutorials**	Tutorials	Project

scikit-tda libraries

- **Kepler Mapper**
- Ripser.py
- Persim
- CechMate
- TaDAsets

Overview sckit-tda and Programme

	Monday	Tuesday	Wednesday
12-13	Python Pre-Course		
14-16	Welcome Overview scikit-tda Intro to Topology in Data Analysis Mapper	**Intro to Persistent Homology** **Ripser** Guest contribution – Sebastian Damrich	Project
16-18	Tutorials	**Tutorials**	Project

scikit-tda libraries

- Kepler Mapper
- **Ripser.py**
- **Persim**
- CechMate
- TaDAsets

Overview sckit-tda and Programme

	Monday	**Tuesday**	**Wednesday**
12-13	Python Pre-Course		
14-16	Welcome Overview scikit-tda Intro to Topology in Data Analysis Mapper	Intro to Persistent Homology Ripser **Guest contribution – Sebastian Damrich**	**Project**
16-18	Tutorials	Tutorials	**Project**

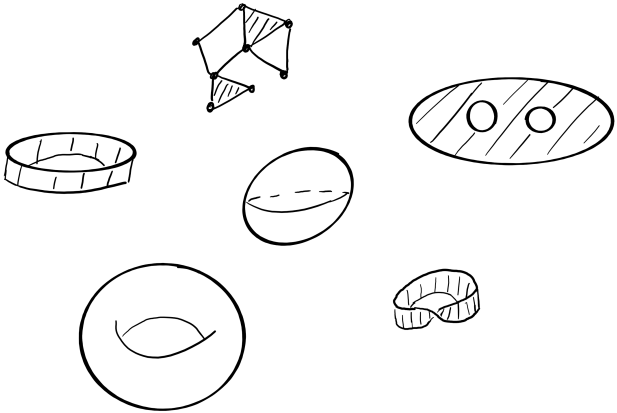
scikit-tda libraries

- Kepler Mapper
- Ripser.py
- Persim
- CechMate
- TaDAsets

Contents

Topology

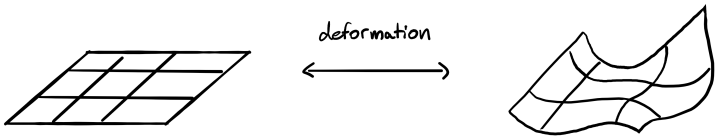
Topology is about studying shapes



Topology

Topology is about studying shapes

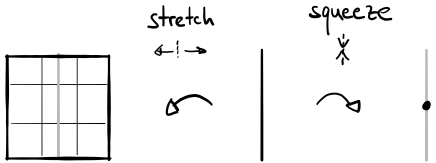
- up to deformations,



Topology

Topology is about studying shapes

- up to deformations,
- and up to stretching and squeezing,



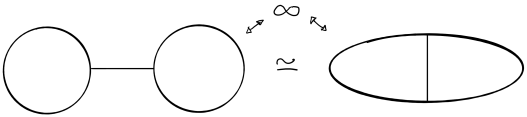
Topology

Topology is about studying shapes

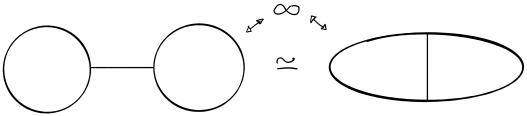
- up to deformations,
- and up to stretching and squeezing,
- **without** cutting or gluing.

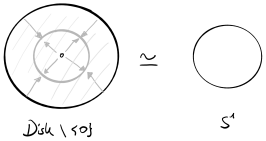
Examples

Examples

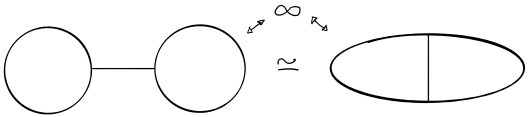


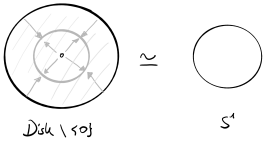
Examples

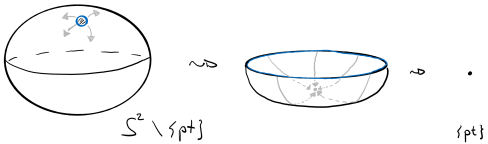




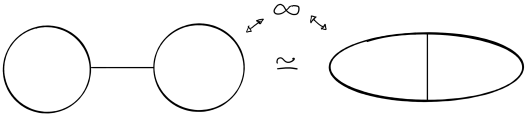
Examples

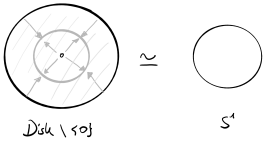


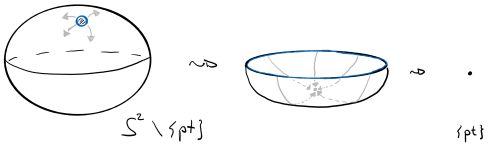


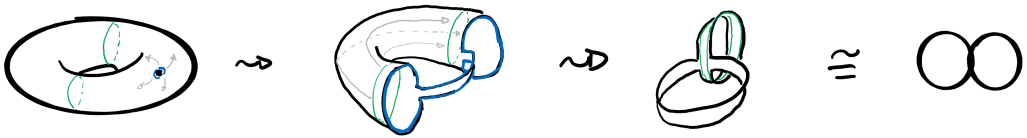


Examples



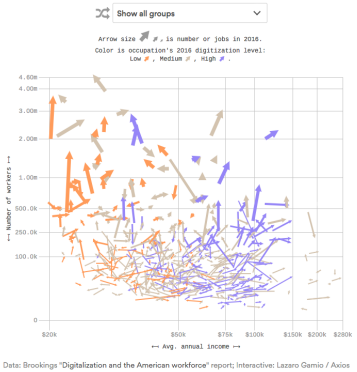






Data Analysis

Understanding data is complicated!

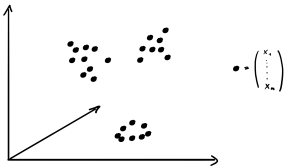


Challenges of data science: complexity, size, curse of dimensionality

Data Analysis

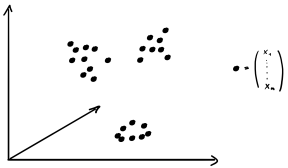
Data Analysis

Data = sample of points in $\mathbb{R}^n$

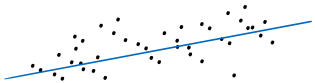


Data Analysis

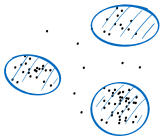
Data = sample of points in $\mathbb{R}^n$



Classical Data Analysis:

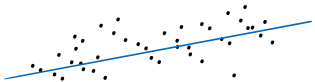


Correlation Methods

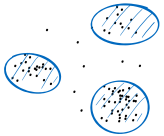


Clustering Methods

Topological Data Analysis

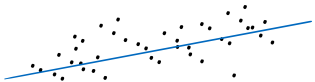


Correlation Methods

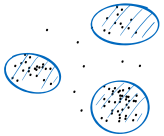


Clustering Methods

Topological Data Analysis



Correlation Methods

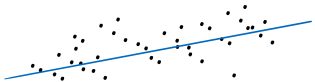


Clustering Methods

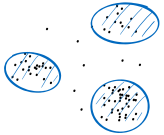
Premise of Topological Data Analysis

Data is noisy sample of some subspace in $\mathbb{R}^n$.
The "topology" of this subspace captures abstract (cor)relations in the data.

Topological Data Analysis



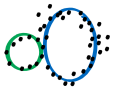
Correlation Methods



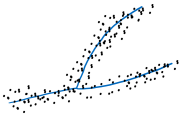
Clustering Methods

Premise of Topological Data Analysis

Data is noisy sample of some subspace in $\mathbb{R}^n$.
The "topology" of this subspace captures abstract (cor)relations in the data.



Loops
(and higher dim. generalizations)



Bifurcations

Topological Data Analysis

Promises of Topological Data Analysis:

Topological Data Analysis

Promises of Topological Data Analysis:

- model independent, non-parametric analysis.

Topological Data Analysis

Promises of Topological Data Analysis:

- model independent, non-parametric analysis.
- Compressed representation of global properties in highly complex data sets.

Topological Data Analysis

Promises of Topological Data Analysis:

- model independent, non-parametric analysis.
- Compressed representation of global properties in highly complex data sets.
- Resistance to noise and missing data. TDA retains significant features of the data.

Topological Data Analysis

Promises of Topological Data Analysis:

- model independent, non-parametric analysis.
- Compressed representation of global properties in highly complex data sets.
- Resistance to noise and missing data. TDA retains significant features of the data.
- Solid theoretical foundation, e.g. robustness with respect to change of choices.

Topological Data Analysis

Promises of Topological Data Analysis:

- model independent, non-parametric analysis.
- Compressed representation of global properties in highly complex data sets.
- Resistance to noise and missing data. TDA retains significant features of the data.
- Solid theoretical foundation, e.g. robustness with respect to change of choices.
- Invariance. Only 'relations' matter. The skew, size, or orientation of data does not fundamentally change that data.

Topological Data Analysis

Promises of Topological Data Analysis:

- model independent, non-parametric analysis.
- Compressed representation of global properties in highly complex data sets.
- Resistance to noise and missing data. TDA retains significant features of the data.
- Solid theoretical foundation, e.g. robustness with respect to change of choices.
- Invariance. Only 'relations' matter. The skew, size, or orientation of data does not fundamentally change that data.
- A data exploration tool. Get answers to questions you haven't even asked yet.

Overview

	Monday	**Tuesday**	**Wednesday**
12-13	Python Pre-Course		
14-16	Welcome **Overview scikit-tda** **Intro to Topology in Data Analysis** Mapper	Intro to Persistent Homology Ripser Guest contribution – Sebastian Damrich	Project
16-18	Tutorials	Tutorials	Project

Overview

	Monday	**Tuesday**	**Wednesday**
12-13	Python Pre-Course		
14-16	Welcome Overview scikit-tda Intro to Topology in Data Analysis Mapper	Intro to Persistent Homology Ripser Guest contribution – Sebastian Damrich	Project
16-18	Tutorials	Tutorials	Project

Break

Continue at:

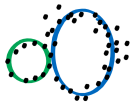
Overview

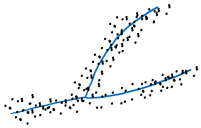
	Monday	**Tuesday**	**Wednesday**
12-13	Python Pre-Course		
14-16	Welcome Overview scikit-tda Intro to Topology in Data Analysis **Mapper**	Intro to Persistent Homology Ripser Guest contribution – Sebastian Damrich	Project
16-18	Tutorials	Tutorials	Project

Contents

The Mapper Algorithm - Intro

The Mapper algorithm provides a way to extract non-trivial properties from high-dimensional data.

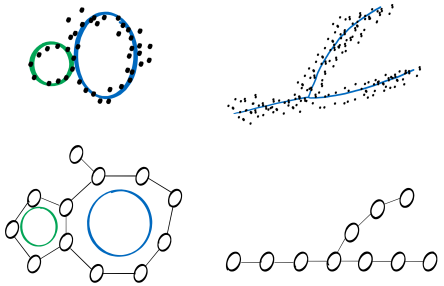




The Mapper Algorithm - Intro

The Mapper algorithm provides a way to extract non-trivial properties from high-dimensional data.

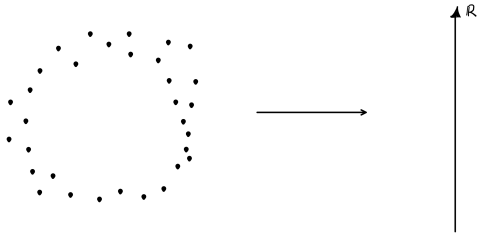
Mapper produces a graph that captures connectedness and topological properties of the data.



The Mapper Algorithm

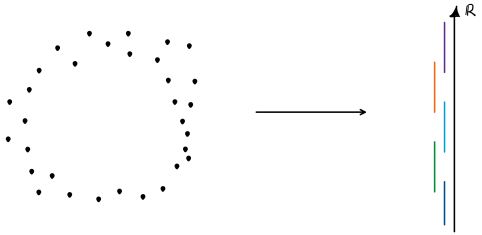
The Mapper Algorithm

1. **Project** (filter dependency)



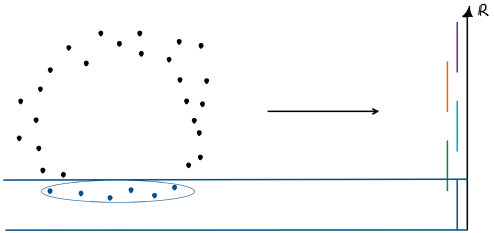
The Mapper Algorithm

1. Project (filter dependency)
2. **Cover** (cover dependency)



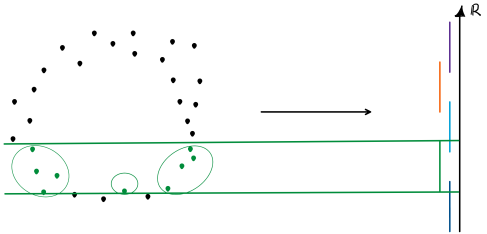
The Mapper Algorithm

1. Project (filter dependency)
2. Cover (cover dependency)
3. **Cluster** (metric & cluster algorithm)



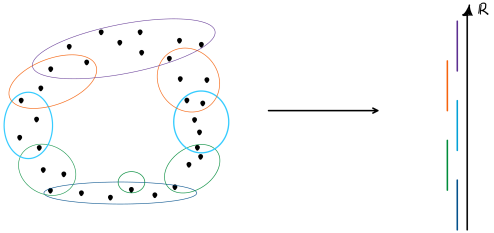
The Mapper Algorithm

1. Project (filter dependency)
2. Cover (cover dependency)
3. **Cluster** (metric & cluster algorithm)



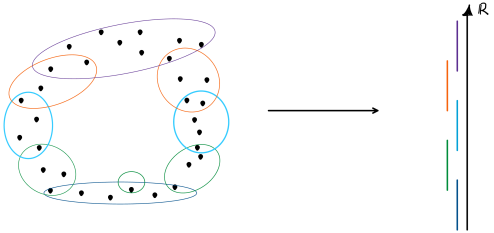
The Mapper Algorithm

1. Project (filter dependency)
2. Cover (cover dependency)
3. **Cluster** (metric & cluster algorithm)



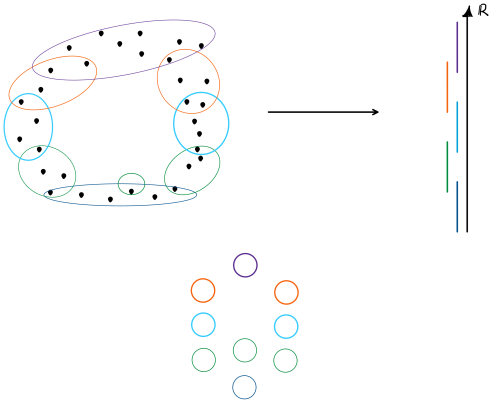
The Mapper Algorithm

1. Project (filter dependency)
2. Cover (cover dependency)
3. Cluster (metric & cluster algorithm)
4. **Graph**



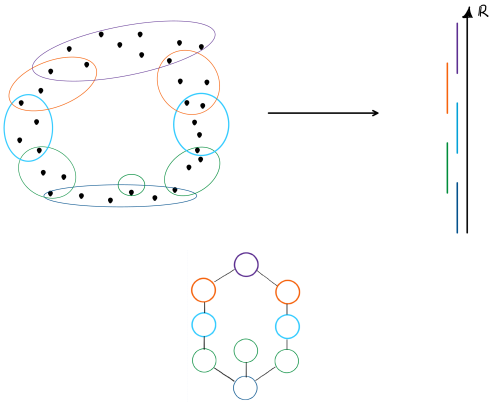
The Mapper Algorithm

1. Project (filter dependency)
2. Cover (cover dependency)
3. Cluster (metric & cluster algorithm)
4. **Graph**
   - Draw a **node** for each **cluster**



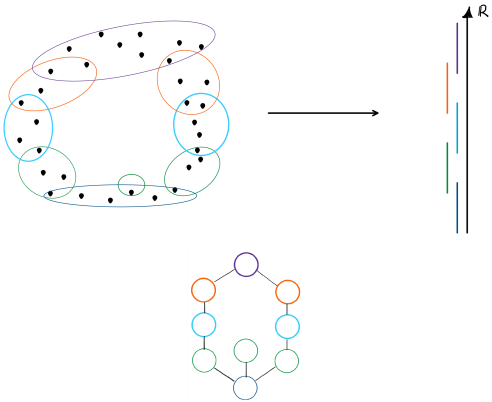
The Mapper Algorithm

1. Project (filter dependency)
2. Cover (cover dependency)
3. Cluster (metric & cluster algorithm)
4. **Graph**
   - Draw a **node** for each **cluster**
   - Draw an **edge** when clusters **interesect**

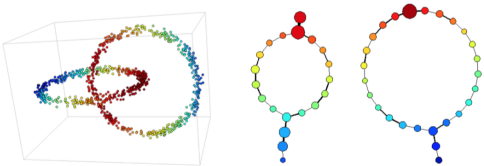


The Mapper Algorithm

1. Project (filter dependency)
2. Cover (cover dependency)
3. Cluster (metric & cluster algorithm)
4. **Graph**
   - Draw a **node** for each **cluster**
   - Draw an **edge** when clusters **interesect**
5. Prettify (e.g. node size, edge length, node colour, graph shape) and Analyze



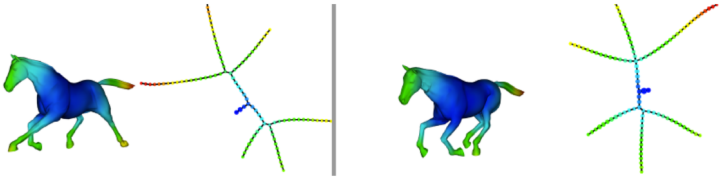
The Mapper Algorithm - Examples



The Mapper graph of two linked circles recognizes two distinct connected components and their shapes.
(filter = SVD)

The Mapper Algorithm - Examples

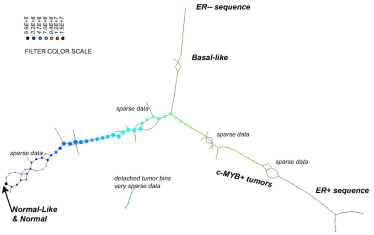
The Mapper graph is preserved throughout the animated movement of a 3d model of a horse.
(filter = eccentricity)



The Mapper Algorithm - Examples

Application to breast cancer data

data = gene expressions of tumor cells
filter $\sim$ deviation from normal cells

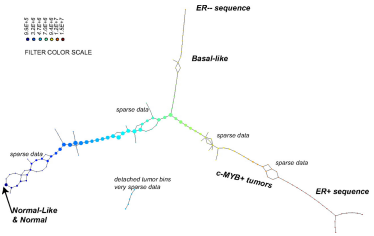


M. Nicolau et al, PNAS 108:17 (2011)

The Mapper Algorithm - Examples

Application to breast cancer data

data = gene expressions of tumor cells
filter $\sim$ deviation from normal cells



M. Nicolau et al, PNAS 108:17 (2011)

- most filters reproduce known classification of clustering methods
- special filters: graph suggests existence of previously unknown cluster c-MYB+
- further analysis shows 100% overall survival rate in this cluster

Contents

Kepler Mapper - Install

Install

```
$ pip install kmapper
$ python3
Python 3.8
> import kmapper
```

Kepler Mapper - The Main Function

kmapper.KeplerMapper.map()

Applies Mapper algorithm on a given projection and returns a graph.

Kepler Mapper - The Main Function

kmapper.KeplerMapper.map()

Applies Mapper algorithm on a given projection and returns a graph.

```
import kmapper as km
km.KeplerMapper.map(projected_data, data [,cover])
```

```
projected_data:  Numpy Array
data:  Numpy Array
cover = kmapper.Cover(n_cubes=10, perc_overlap=0.1, limits=None, verbose=0)
```

Kepler Mapper - The Main Function

Output

```
import kmapper as km
graph = km.KeplerMapper.map(projected_data, data [,cover])

graph = {
    'nodes': {'cube0_cluster0': [points], ...},
    'links': {'cube0_cluster0': [linked clusters], ...},
    'simplices': {[nodes], ..., [edges], ...},
    'meta_data': {summary of choices}
    'meta_nodes': {?}
    }
```

Kepler Mapper - kmapper projections

kmapper.KeplerMapper.project() and fit_transform()

Create a projection from a dataset.
Input the data set. Specify a projection. Output the projected data.

Kepler Mapper - kmapper projections

kmapper.KeplerMapper.project() and fit_transform()

Create a projection from a dataset.
Input the data set. Specify a projection. Output the projected data.

```
km.KeplerMapper.project(data,
    projection='sum',
    scaler=MinMaxScaler(copy=True, feature_range=(0, 1)),
    distance_matrix=None)
```

`projection`: `str`**, or a Scikit-learn class with fit_transform, or a list of dimension indices.

`scaler`: Scikit-Learn API compatible scaler.

`distance_matrix`: `str`** or None. If None do nothing, else compute distance matrix with chosen metric, before applying the projection.

**see `help(kmapper.KeplerMapper.project)` for more details.

Kepler Mapper - kmapper Cover

kmapper.Cover()

Calculates a cover based on number of cubes and the percentage of their overlap.

Kepler Mapper - kmapper Cover

kmapper.Cover()

Calculates a cover based on number of cubes and the percentage of their overlap.

```
km.Cover(n_cubes=10, perc_overlap=0.5, limits=None)
```

`n_cubes`: int. Number of hypercubes along each dimension.
`perc_overlap`: float. Amount of overlap between adjacent cubes.
`limits`: Numpy Array $(n_{\text{dim}}, 2)$.

- the value `np.float('inf')` corresponds to min/max value of the projection in this dimension.
- `limits` == None corresponds to min/max value of the projection for all dimensions.

Kepler Mapper - kmapper visualize

kmapper.KeplerMapper.visualize()

Generate a visualization of the simplicial complex mapper output. Turns the complex dictionary into a HTML/D3.js visualization.

Kepler Mapper - kmapper visualize

kmapper.KeplerMapper.visualize()

Generate a visualization of the simplicial complex mapper output. Turns the complex dictionary into a HTML/D3.js visualization.

km.KeplerMapper.visualize(graph, args)

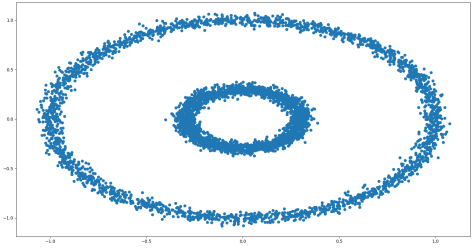
**see `help(km.KeplerMapper.visualize)` for details.

Kepler Mapper - Example

Kepler Mapper - Example

Import data

```
from sklearn import datasets
data = datasets.make_circles(n_samples=5000, noise=0.03, factor=0.3)
```



Kepler Mapper - Example

```
import kmapper as km
mapper = km.KeplerMapper

# Project data to x and y axis
projected_data = mapper.fit_transform(data, projection=[0,1])

# Create graph (using default nr_cubes=10)
graph = mapper.map(projected_data, data)

# Visualize it
mapper.visualize(
    graph,
    path_html='make_circles_keplermapper_output.html',
    title='make_circles(n_samples=5000,noise=0.03,factor=0.3)'
    )
```

Kepler Mapper - Example

```
import kmapper as km
mapper = km.KeplerMapper

# Project data to x and y axis
projected_data = mapper.fit_transform(data, projection=[0,1])

# Create graph (using default nr_cubes=10)
graph = mapper.map(projected_data, data)

# Visualize it
mapper.visualize(
    graph,
    path_html='make_circles_keplermapper_output.html',
    title='make_circles(n_samples=5000,noise=0.03,factor=0.3)'
    )
```

projection=[0,1]: `example_xy.html`
projection='sum': `example_sum.html`

Breakout Rooms

Type a room ID in chat to get assigned to it.

Room IDs

- **0 + 'group id'** create/join custom room
- **1** Exercise 1
- **2** Exercise 2
- **3** Exercise 3
- **4** Exercise 4
- **9** Break Room

- If you run into problems while in a room, hit the *contact moderator* button. This should summon a moderator into your room.
- If you want to change rooms, simply get back to the main session and let us know.