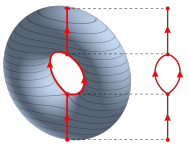
Exercise 1 *Kepler Mapper - Getting Started*

Revisit the example from the lecture and play around with different filter functions, noise-levels and covers.

```
# Initialize Mapper
import kmapper as km
mapper = km.KeplerMapper()

# Import sample data (2 disjoint circles)
from sklearn import datasets
data, labels = datasets.make_circles(n_samples=5000, noise=0.03, factor=0.3)

# Fit to and transform the data
projected_data = mapper.fit_transform(data, projection=[0,1])

# Create dictionary called 'graph' with nodes, edges and meta information
graph = mapper.map(projected_data, data)

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

Exercise 2 *The Reeb Graph*



Let X be a topological space and $f : X \to \mathbb{R}$ a continuous function. The Reeb Graph $R_f(X)$ is the space obtained by identifying $x, y \in f^{-1}(c)$, whenever they lie in the same connected component of the level set $f^{-1}(c)$.

(a) Determine the Reeb Graph of the standard embedding of a torus $T = S^1 \times S^1$ in $\mathbb{R}^3$ for different filter functions f. (E.g. rotate the torus before projecting to the z-axis, or come up with your own filter).

(b) Use `tadasets.torus()` to construct the Mapper graph of a noisy torus for several predefined and/or custom filters.

(c) Add a puncture to the torus by removing (enough) points in a certain area. How does this affect the Reeb and Mapper graphs?

Exercise 3 *Kepler Mapper - Digits Datasets*

based on https://kepler-mapper.scikit-tda.org/generated/gallery/plot_digits.html

full code at https://github.com/micbl/TDAworkshop/tree/main/Exercises/exercise3.py

In this exercise we will apply Mapper to a dataset of noisy images of handwritten digits. We will also see how one can adjust the information presented in the html-vizualization.

The dataset we will use is part of `sklearn.datasets` and can be accessed via:

```
from sklearn import datasets
import matplotlib.pyplot as plt

#Load the digits dataset
digits = datasets.load_digits()

#Display the first digit
plt.figure(1, figsize=(3, 3))
plt.imshow(digits.images[0], cmap=plt.cm.gray_r, interpolation='nearest')
plt.show()
```

Make yourself familiar with the data: Look at the data structure, plot the first few digits.

We will now create custom tooltips. This will help interpreting the html-output, since datapoints will not be listed by their index, but by the actual image instead. Note that you will need to run `pip install pillow` and `pip install imageio` first.

```
import io
import base64
import numpy as np
import imageio
from PIL import Image

# Configure the tooltips such that the html-visualization shows helpful information for members of
    a node
tooltip_s = []
for image_data in digits.data:
    output = io.BytesIO()
    img = Image.fromarray(image_data.reshape((8, 8))) # Data was a flat row of 64 ''pixels''.
    imageio.imwrite(output, img, format="PNG")
    contents = output.getvalue()
    img_encoded = base64.b64encode(contents)
    img_tag = """<img src="{}">""".format(img_encoded.decode('utf-
        8'))
    tooltip_s.append(img_tag)
    output.close()

tooltip_s = np.array(tooltip_s) # need to make sure to feed it as a NumPy array, not a list
```

Now it's time to apply Kepler Mapper to the dataset. We use a t-SNE filter function with 2 components (reduces data to 2 dimensions) that is provided by `sklearn`.

```
import sklearn
import kmapper as km
mapper = km.KeplerMapper(verbose=2)

# Fit and transform data
projected_data = mapper.fit_transform(digits.data, projection=sklearn.manifold.TSNE())

# Create the graph (we cluster on the projected data and suffer projection loss)
graph = mapper.map(projected_data,
                   clusterer=sklearn.cluster.DBSCAN(eps=0.3, min_samples=15),
                   cover=km.Cover(35, 0.4))
```

To create a visualization that will be written to html we finally run:

```
mapper.visualize(graph,
                 title="Handwritten_digits_Mapper",
                 path_html="digits_custom_tooltips.html",
                 color_function=digits.target,
                 custom_tooltips=tooltip_s)
# Tooltips with the target y-labels (i.e. 1,2,3,...,9) for every cluster member
mapper.visualize(graph,
                 title="Handwritten_digits_Mapper",
                 path_html="digits_ylabel_tooltips.html",
                 custom_tooltips=digits.target)
```

Note that in the first of these files, the member summary in the panel "Cluster Details" on the left now shows the images that the datapoints correspond to, instead of their index in the dataset.

Exercise 4 *Kepler Mapper - The Breast Cancer Dataset*
The dataset used by Nicolau et al (2011)[1] is accessible at
https://www.kaggle.com/uciml/breast-cancer-wisconsin-data
Have a look at the data and experiment with it.

A few instructions, in particular two possible filter functions, can be found in the Kepler Mapper Gallery at
https://kepler-mapper.scikit-tda.org/generated/gallery/plot_breast_cancer.html
note: The Kepler Mapper Gallery has been found to be outdated and often uses deprecated functions.

[1] https://www.pnas.org/content/108/17/7265