# *k*-means considered harmful

## …as clustering for Mapper…

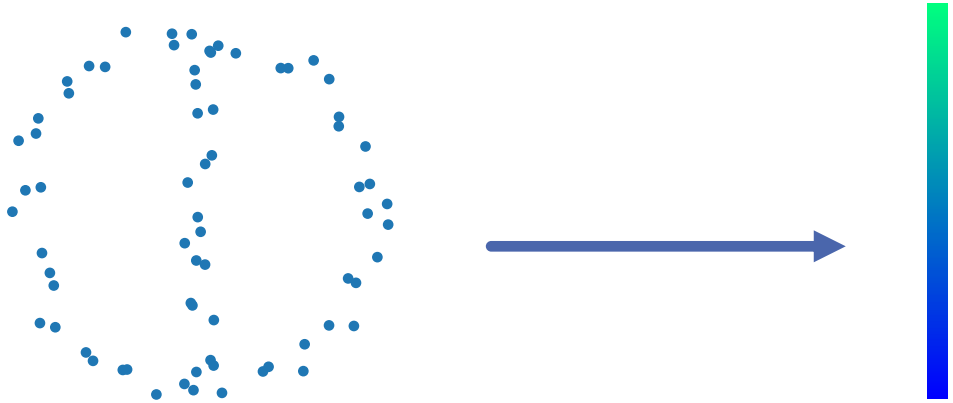
Mikael Vejdemo-Johansson

CUNY College of Staten Island: Mathematics

CUNY Graduate Center: Computer Science, Data Science, Mathematics

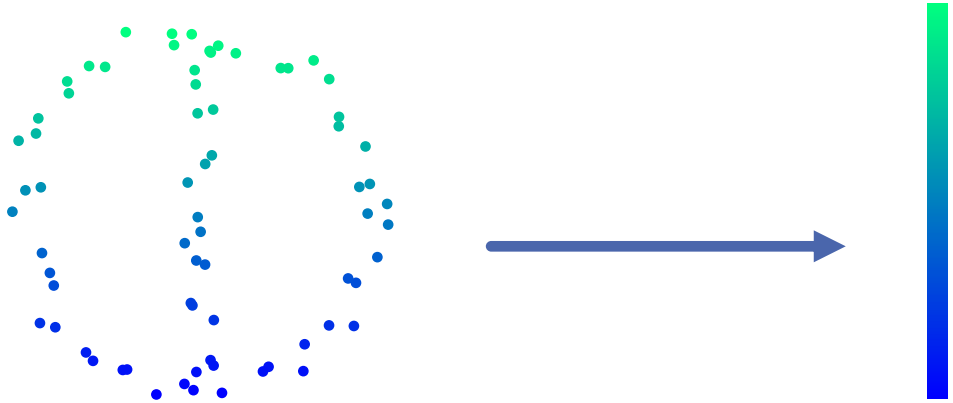
# Recall the Mapper Algorithm structure

Point Cloud



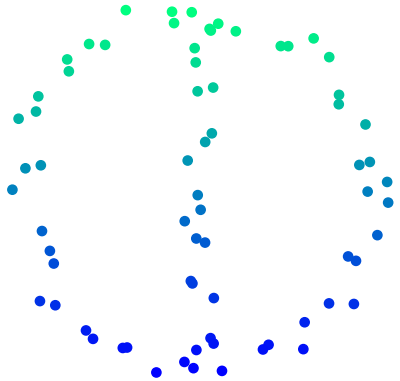
# Recall the Mapper Algorithm structure

Point Cloud with lens function



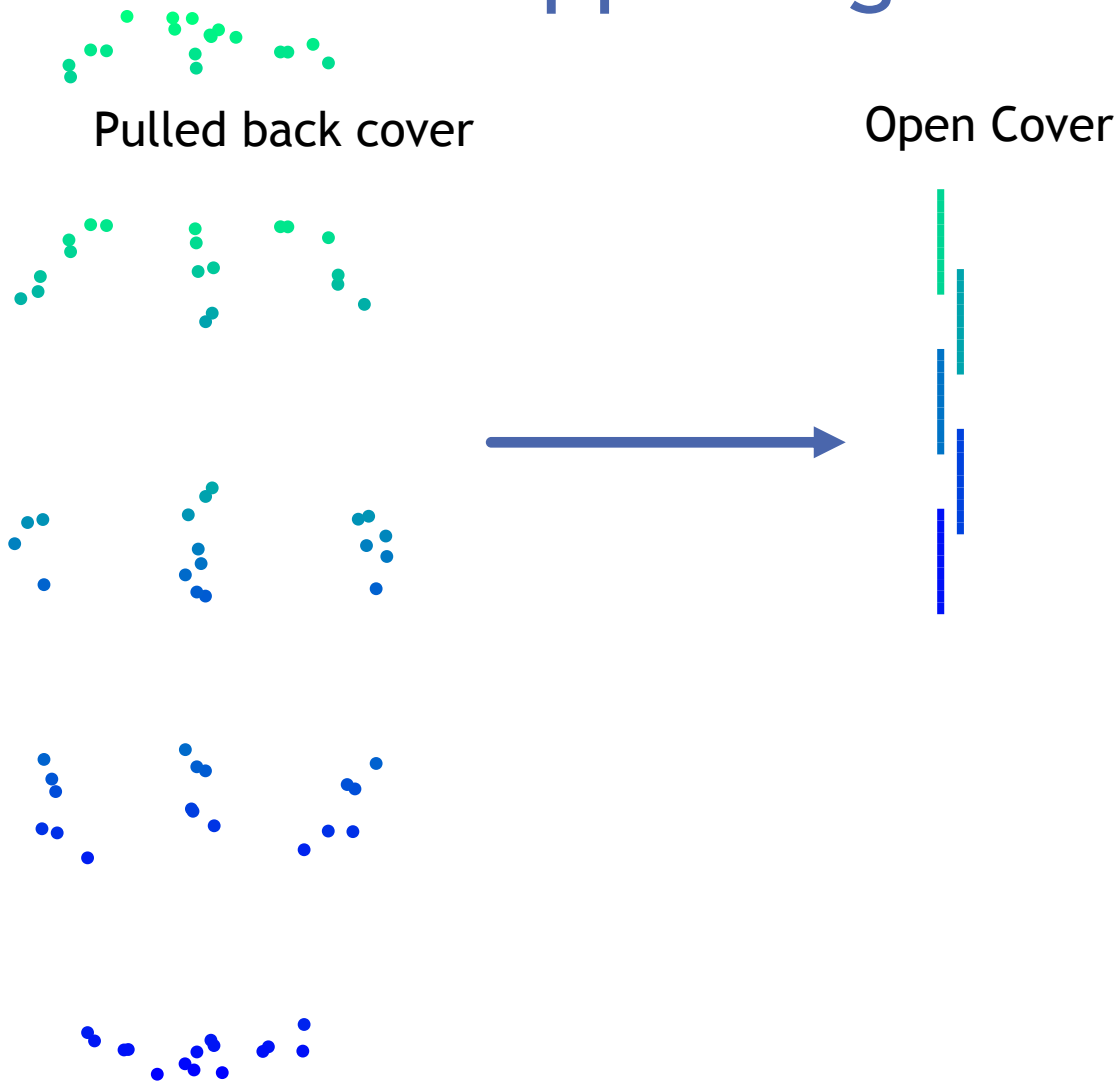
# Recall the Mapper Algorithm structure

Point Cloud with lens function



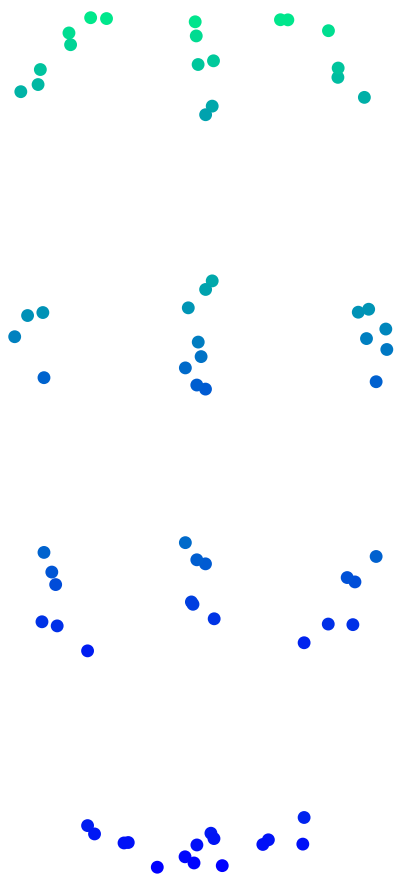
Open Cover

# Recall the Mapper Algorithm structure



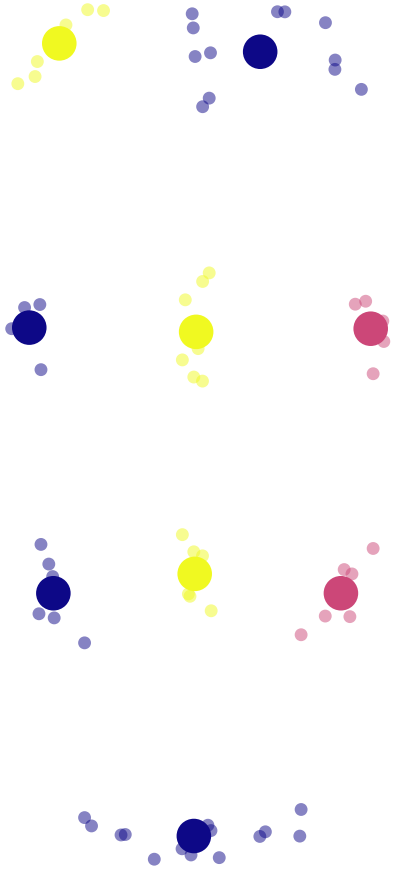
# Recall the Mapper Algorithm structure

Pulled back cover

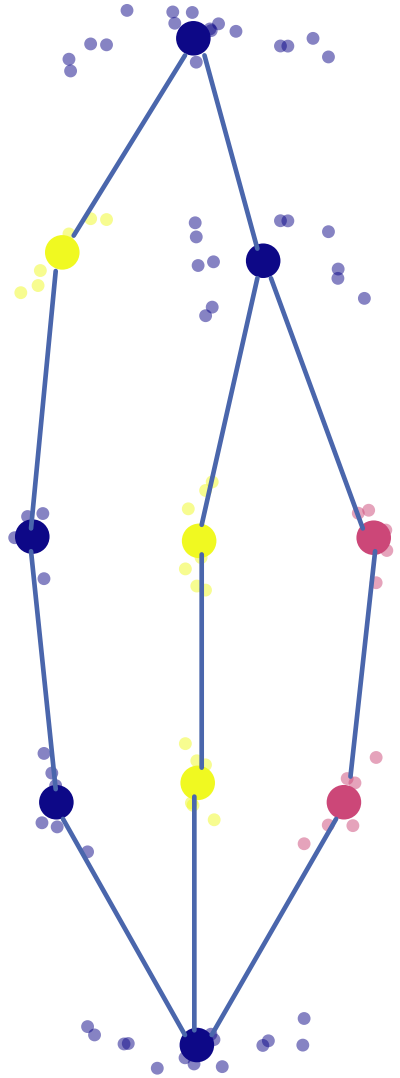


# Recall the Mapper Algorithm structure

Pulled back cover, refined by local clustering



# Recall the Mapper Algorithm structure



Connect the clusters to form the **nerve complex** of the refined covering, and we get the **Mapper complex**.

# Why do we expect the result to tell us anything about the data source?

- Stability
  - If we vary the exact cover chosen, we get the same qualitative observations
  - Stability theorems and frameworks (Dey-Mémoli-Wang; Jeitziner-Carriére-Rougemont-Oudot-Hess-Brisken; Belchí-Brodzki-Burfitt-Niranjan; …)
- Nerve Lemma
  - If the refined cover is *good* (ie all intersections of sets of cover elements are topologically simple [ie acyclic… contractible… etc]), then the Nerve Complex is equivalent (homotopy equivalent… quasi-isomorphic…) to the original shape.

# Stability

- In early Mapper applications, results were motivated by **stability under parameter variation**: that since the resulting shape is similar when the details of generating the cover or the clustering algorithm are modified, the result is saying *something* about the data source.

- Stability alone does not give us an actual connection between data source shape and the Mapper complex: it tells us that it is some kind of invariant, but not whether we can draw any further conclusions.

# Nerve Lemma

- The Nerve Lemma **does** provide a concrete connection between shapes: as long as the cover is nice enough (a **good cover**), the Mapper complex and the data source describe the same shape, up to some notion of equality. (different theorems provide different types of equality)
- V-J – Leshchenko: Mapper as it was usually used (probably still is…) does not actually **check** whether the resulting cover is nice enough: we are basically taking it on trust that the Nerve lemma comes close to applying.
  - V-J – Mukherjee: In order to run a hypothesis test for the Nerve lemma conditions, acyclicity testing needs multiple hypothesis correction; our paper provides methods for this.
- Failure to ensure a good cover can both add and remove topological features.
- Alvarado-Belton-Lee-Palande-Percival-Purvine: **ANY** (small enough) simplicial complex is the Mapper complex on a dataset if you can pick the lens function.

Focus of this talk:

The choice of **clustering algorithm** in the Mapper algorithm.

# Clustering Algorithms in Mapper
# And a little bit of history

- 2007: Gurjeet Singh proposes the Mapper algorithm in his PhD thesis.

  Singh goes on to found Ayasdi together with Gunnar Carlsson, and launch numerous academic collaborations. Ayasdi sells a data analysis platform based on Mapper.
- Ayasdi's internal research refines an adaptive clustering algorithm: Hierarchical clustering, followed by a heuristic for choosing a cut-off value within the first large gap in the dendrogram.
- 2011: Daniel Müllner and Aravindakshan Babu release the first independent open source implementation.
- Since then: Kepler-Mapper, TDAMapper [R], giotto-tda, tmap, …

- Since Ayasdi's clustering algorithm is proprietary, most later work uses more accessible algorithms: often scikit-learn, often k-Means or DBSCAN.

# DBSCAN

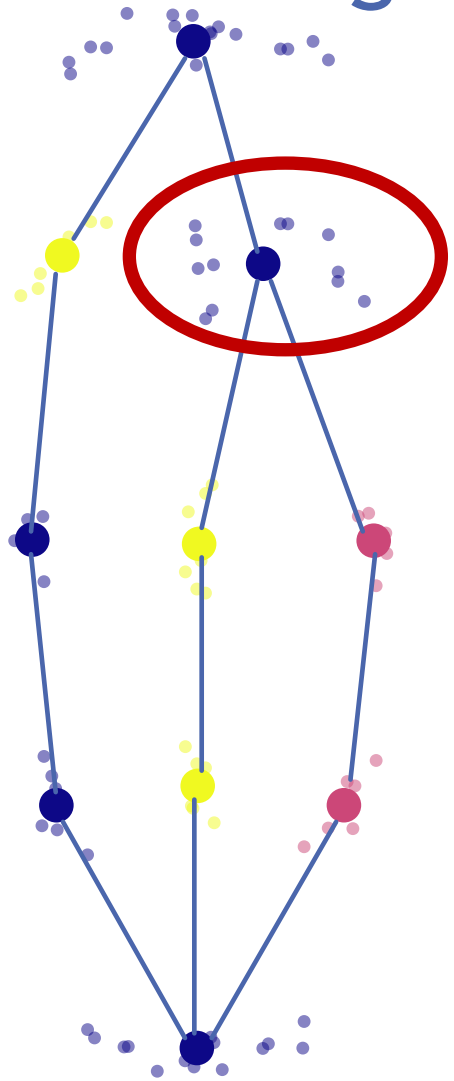
- The DBSCAN is an adaptive algorithm that picks the number of clusters from the data.
- It works by finding sufficiently large high-density sets of points, and then growing these core sets.
- Low-density points and outliers might not get assigned to any one actual cluster: DBSCAN returns one special group of unclustered points.
- To use DBSCAN in Mapper, you must handle the unclustered points.

- I have checked the source-code of giotto-tda: they do it correctly.

# *k*-means

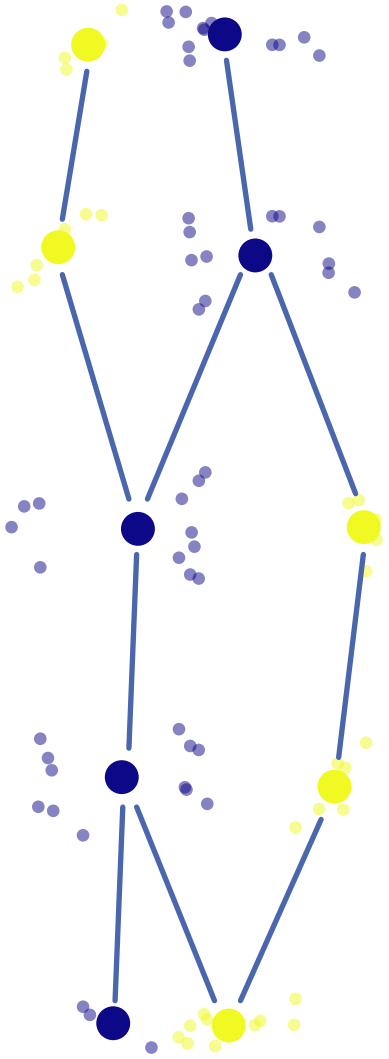
- *k*-means clustering starts by picking a prescribed number of clusters. Then the data is divided up into *k* cluster sets that optimize a penalty function that rewards small distances within clusters and long distances between clusters.
- *k*-means will return *k* clusters, regardless of whether or not the data allows for it.
- If *k* is too low, clusters will be merged together.
- If *k* is too high, clusters will be split apart.

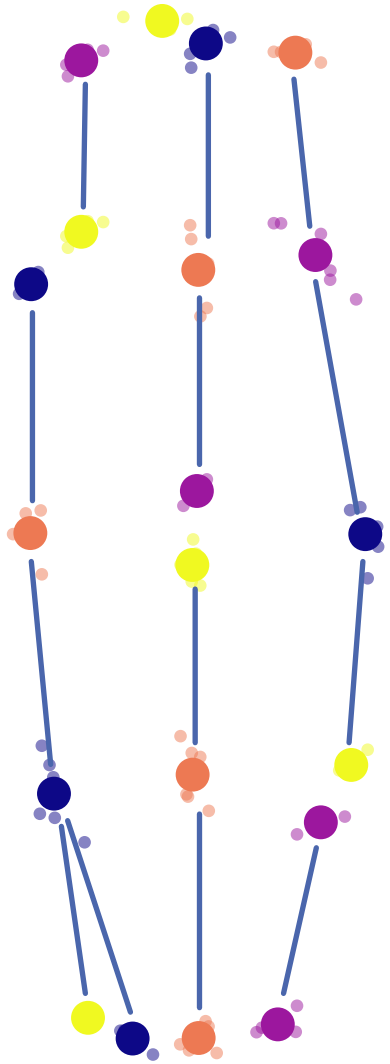
# Clustering Failures: DBSCAN



- We already saw one clustering failure in the example.
- This example used DBSCAN.

# Clustering Failures: 2-means, 4-means





“For clustering we use Agglomerative Single Linkage Clustering with the “cosine”-distance and 3 clusters.”

Kepler Mapper & NLP examples

“

```
# Define the simplicial complex
scomplex = mapper.map(lens, X,
  cover=km.Cover(n_cubes=15, perc_overlap=0.7),
  clusterer=sklearn.cluster.KMeans(n_clusters=2, random_state=3471))
```

”

Kepler-Mapper: Choosing a lens (Cancer-demo.html)

“

```
>>> # Use KMeans with 2 clusters
>>> graph = mapper.map(X_projected, X_inverse,
>>>     clusterer=sklearn.cluster.KMeans(2))
```

”

Kepler-Mapper help text for the main API command

“

```
mapper_algo = MapperAlgorithm( cover=CubicalCover( n_intervals=10,
overlap_frac=0.65 ), clustering=AgglomerativeClustering(10), verbose=False
)
```

”

tda-mapper documentation, digits dataset

# No Good Cover In Sight

- DBSCAN sometimes fails to produce good covers, and will — when used correctly — censor sparse parts of the data.
- *k*-means — or really any non-adaptive clustering algorithm — will silently produce unpredictably bad deviations from being a Good Cover.

- You may be getting stable results, but without clear relationships between the Mapper complex and your data source.

**Do not use non-adaptive clustering methods for Mapper.**

# Conclusion and Thanks

- Open Source Mapper implementations are a great addition to the field.
- But we **MUST** get better at our implementations, and at the advice we give by writing tutorials.


- Thanks to:
  - Gunnar Carlsson, for getting me interested in the first place
  - Simons Foundation, for travel support under grant # 961833
  - CUNY, for a long enough sabbatical that my brain is waking back up