



HAL
open science

Dimensionality reduction using graphs

David Dobas, Jakub Rada, Theo Michel

► **To cite this version:**

David Dobas, Jakub Rada, Theo Michel. Dimensionality reduction using graphs. KAIST Graph Machine Learning, Jun 2023, Daejeon, South Korea. hal-04175042

HAL Id: hal-04175042

<https://hal.science/hal-04175042>

Submitted on 1 Aug 2023

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

#19: Dimensionality reduction using graphs

David Dobas, Jakub Rada, Theo Michel
20236035, 20236065, 20236005

Problem statement

- **Dimensionality reduction**
 - Reduce dimension of data while preserving the structure of data, such as clustering or closeness of points
 - Can be used for
 - Visualisation (reduction to 2D or 3D)
 - Preprocessing for ML algorithms
 - State of the art algorithms
 - PCA – powerful and fast, cannot capture complex manifolds
 - UMAP, t-SNE – powerful and fast, complex to understand

1

Our goal

- **Our research question**
 - Can we capture the structure of data in graph?
 - Use graph as a compact representation of the data
 - Can be used to compress data
 - Can we use the obtained graph to reduce dimensionality?
 - We can use graph embedding methods
- **Advantages of our approach**
 - Easy to understand algorithm
 - Able to capture complex manifolds

2

Graph building

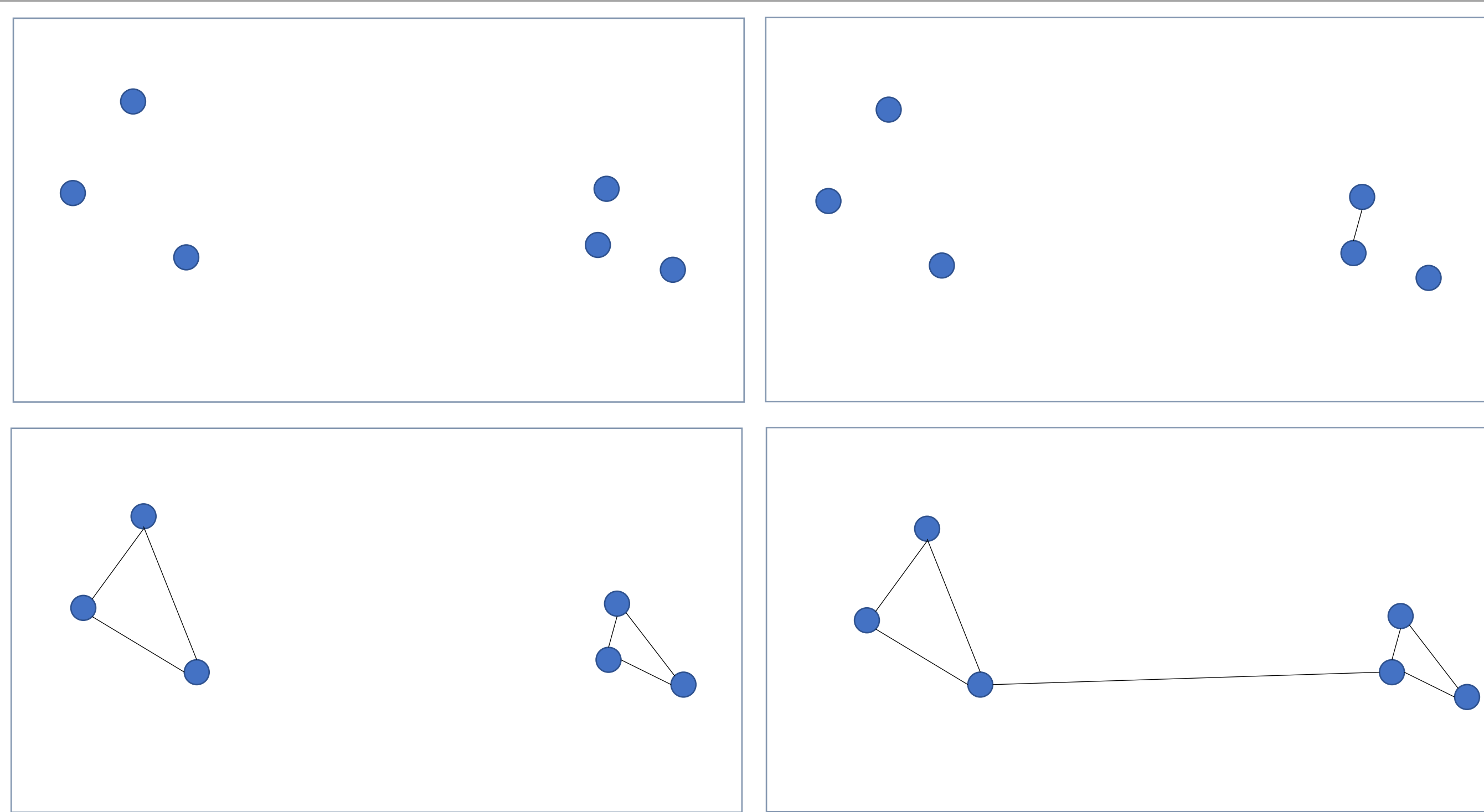
- **Cheapest builder**
 - Every datapoint is represented by node
 - We compute pairwise distances between all datapoints
 - Points with the shortest distances are connected by edges, edges added one by one until the graph is connected
 - Edge weights are added using the distances as

$$w = \frac{1}{d^k}$$

- k is a hyperparameter
- **Spanning tree**
 - We can use spanning tree, weights same as cheapest builder

3

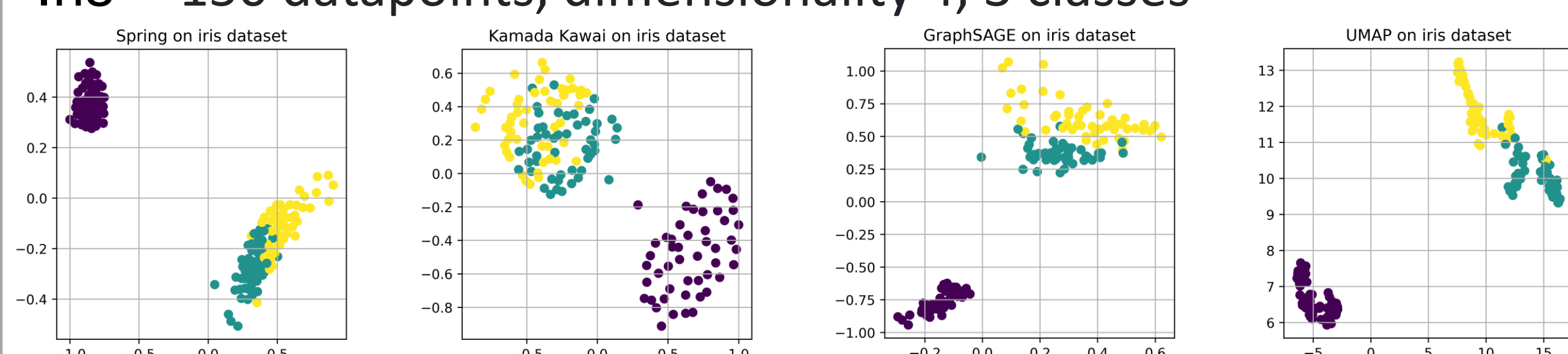
Graph building – cheapest builder



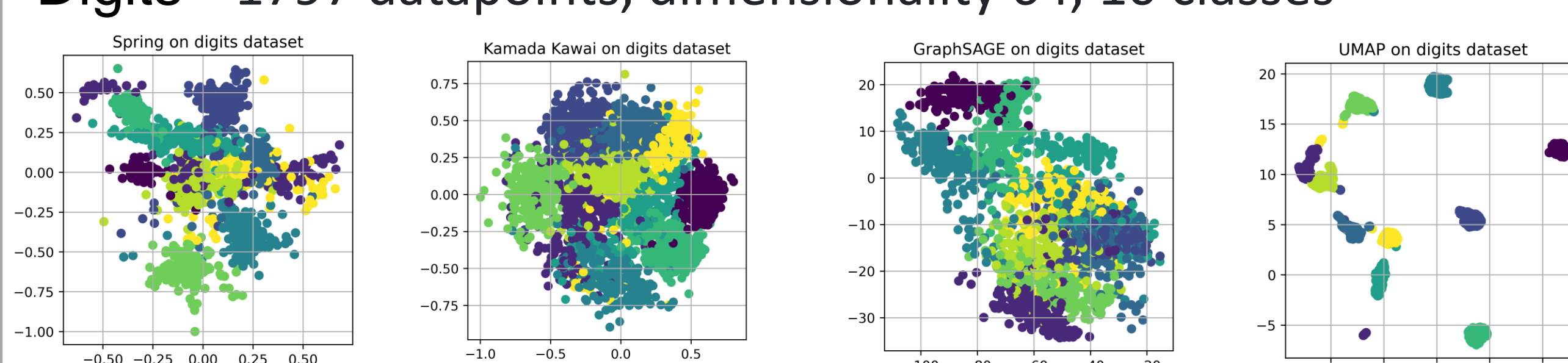
4

Visual results

Iris – 150 datapoints, dimensionality 4, 3 classes



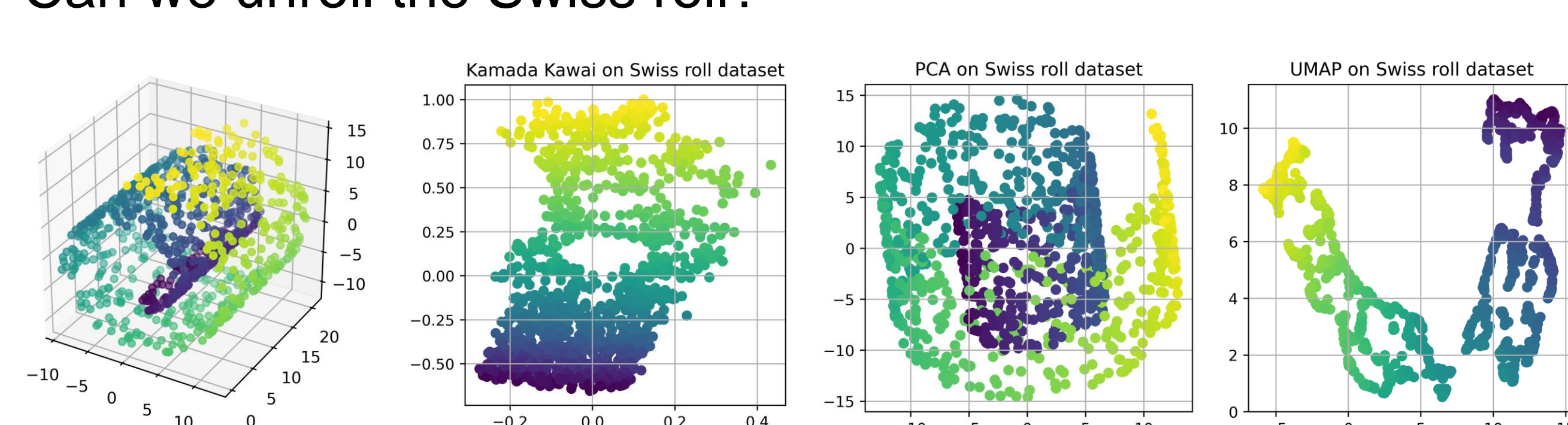
Digits - 1797 datapoints, dimensionality 64, 10 classes



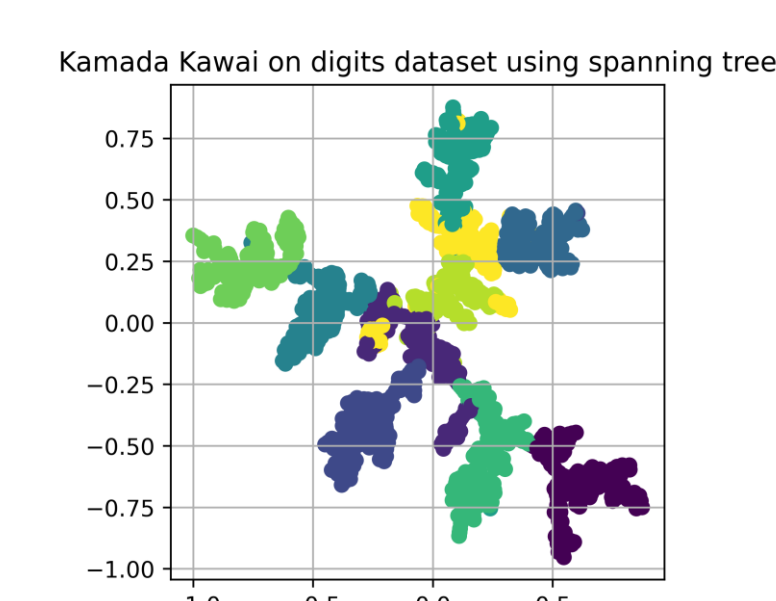
5

Visual results

Can we unroll the Swiss roll?



How do the embeddings look using spanning tree?



6

Quantitative results – digits dataset

| Algorithm | Dimension | Trustworthiness | Continuity | Computation time |
|-------------------------|-----------|-----------------|---------------|------------------|
| Cheapest + Spring | 2 | 0.9182 | 0.9784 | 1m 49.7s |
| Cheapest + Kamada Kawai | 2 | 0.8728 | 0.9661 | 8m 39.0s |
| Spanning + Kamada Kawai | 2 | 0.9875 | 0.9859 | 2m 5.0s |
| Cheapest + GraphSAGE | 2 | 0.8462 | 0.9530 | 34m 22.6s |
| PCA | 2 | 0.8591 | 0.9646 | 0.2s |
| UMAP | 2 | 0.9913 | 0.9927 | 7.2s |
| Cheapest + Kamada Kawai | 10 | 0.9949 | 0.9937 | 11m 8.1s |
| Spanning + Kamada Kawai | 10 | 0.9951 | 0.9867 | 57m 38.1s |
| Cheapest + GraphSAGE | 10 | 0.9611 | 0.9774 | 24m 40.5s |
| PCA | 10 | 0.9979 | 0.9990 | 0.3s |
| UMAP | 10 | 0.9961 | 0.9944 | 7.8s |

7

Summary

- We used simple algorithms for graph building and still obtained great results
- Our algorithm is able to compete with state-of-the-art algorithms in terms of trustworthiness and continuity
- However, we can not compete in computational time
- Also, we use all pairwise distances, which is not feasible for big datasets
- None of our methods is universally best on every dataset
- That creates a room for improvements

8