Network Machine Learning

Guanhua Chen

Mentor: Ronan Perry

Motivation

Networks have become a common data structure with which to understand many of the concepts of every day life.

In the research field, we also encounter data with network structure, such as modelling communities of neurons in the brain.

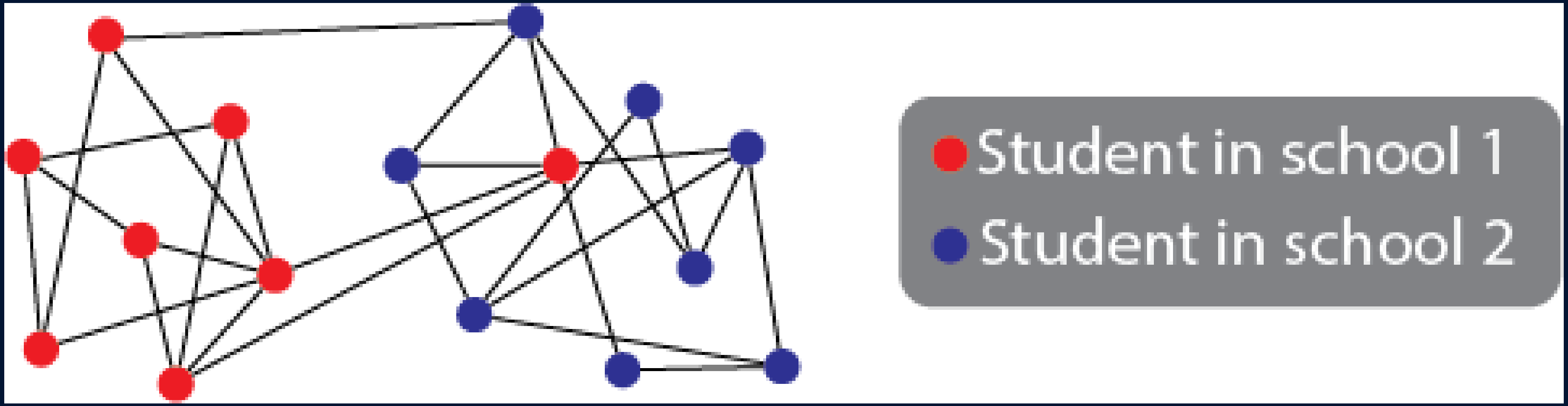
DRP Goals:

(1) learn about what is the Network Machine Learning and what is the difference between network-valued data and tabular structure data.

(2) Find an interesting dataset in the network structure and then do some hands-on machine learning towards it through Jupyter Notebook.

What is a network?

A network is a set of:
- Nodes (e.g. people)
- Edges between nodes (e.g. friendships)
- (Optional) information on the nodes and edges (e.g. labels)

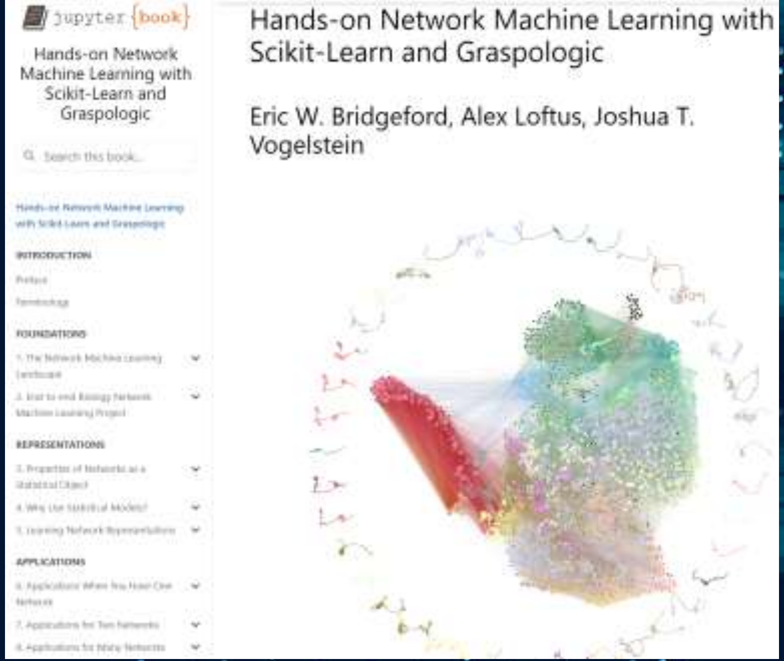


The textbook: “Hands-on Network Machine Learning with Scikit-Learn and Graspologic”

Its goal is to give us the concepts, the intuition, and the tools we need to actually implement programs capable of learning from network data.

It covers the fundamentals of network machine learning, focusing on developing intuition on networks, and doing while paired with relevant Python tutorials.

Many example figures shown in this presentation are from this textbook.



Hands-on Network Machine Learning with Scikit-Learn and Graspologic

Hands-on Network Machine Learning with Scikit-Learn and Graspologic

INTRODUCTION

Preface

Terminology

FOUNDATIONS

1. The Network Machine Learning Landscape

2. End-to-end Biology Network Machine Learning Project

REPRESENTATIONS

3. Properties of Networks as a Statistical Object

4. Why Use Statistical Models?

5. Learning Network Representations

APPLICATIONS

6. Applications When You Have One Network

7. Applications for Two Networks

8. Applications for Many Networks

Hands-on Network Machine Learning with Scikit-Learn and Graspologic

Eric W. Bridgeford, Alex Loftus, Joshua T. Vogelstein

What is network machine learning

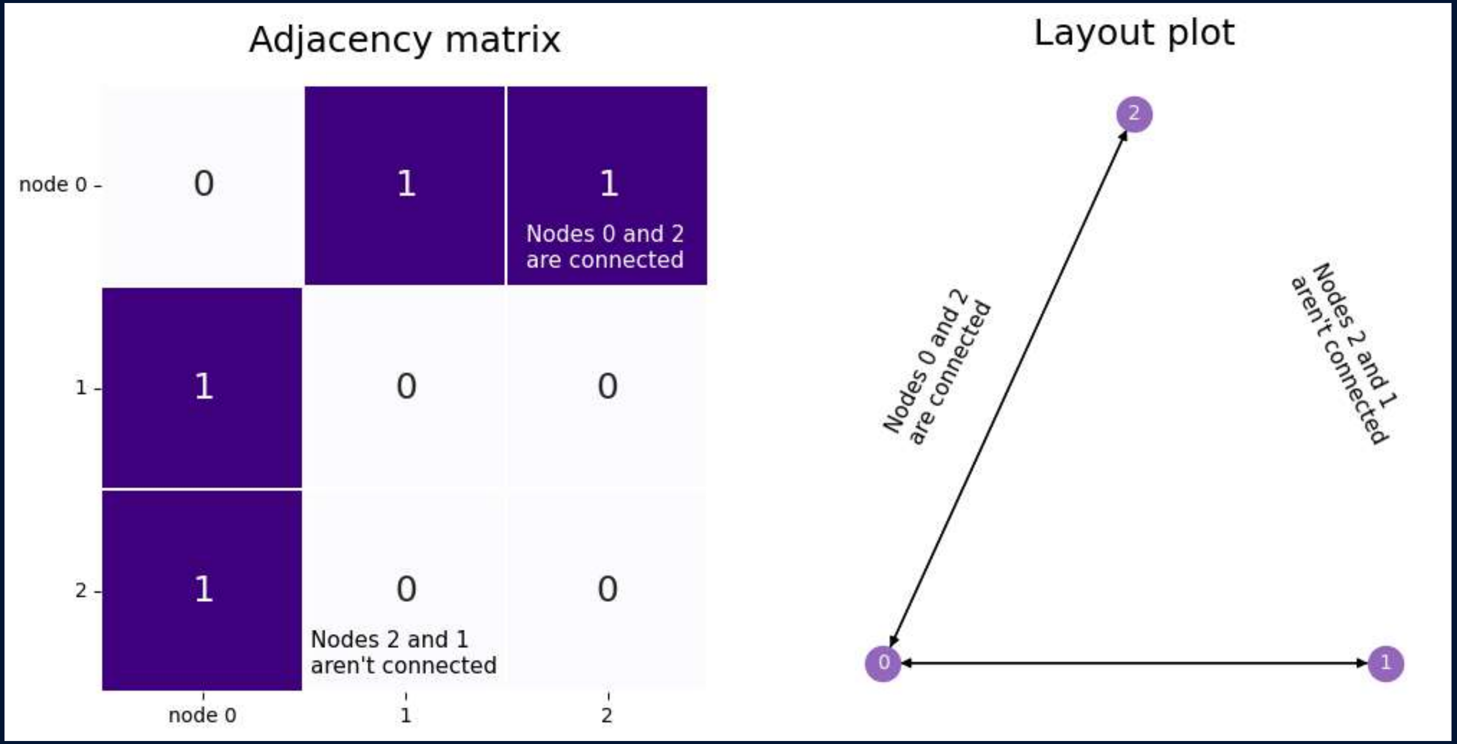
The network has information (given by the edges), but traditional statistics and machine learning are not meant for networks.

There are many different types of network problems/methods:
(1) single or multiple networks
(2) Networks with/without attributes
(3) (statistical) model-based or non-model based

One solution: we can adapt the network to a more traditional format: tabular structure. Then we can then apply regular techniques from statistics and machine learning.

Adjacency Matrix

Adjacency Matrix: a square matrix with a row and column for each node. A 1 in the matrix at row *i* and column *j* means an edge in the graph between nodes *i* and *j*.



Adjacency matrix

	node 0	1	2
node 0	0	1	1
1	1	0	0
2	1	0	0

Nodes 0 and 2 are connected

Nodes 2 and 1 aren't connected

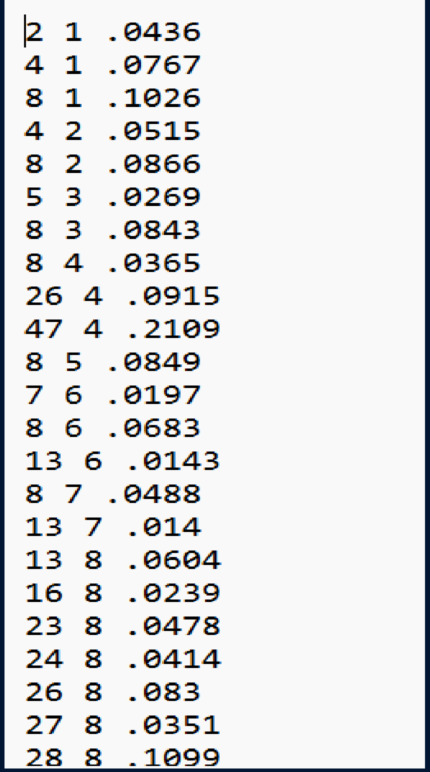
Layout plot

Nodes 0 and 2 are connected

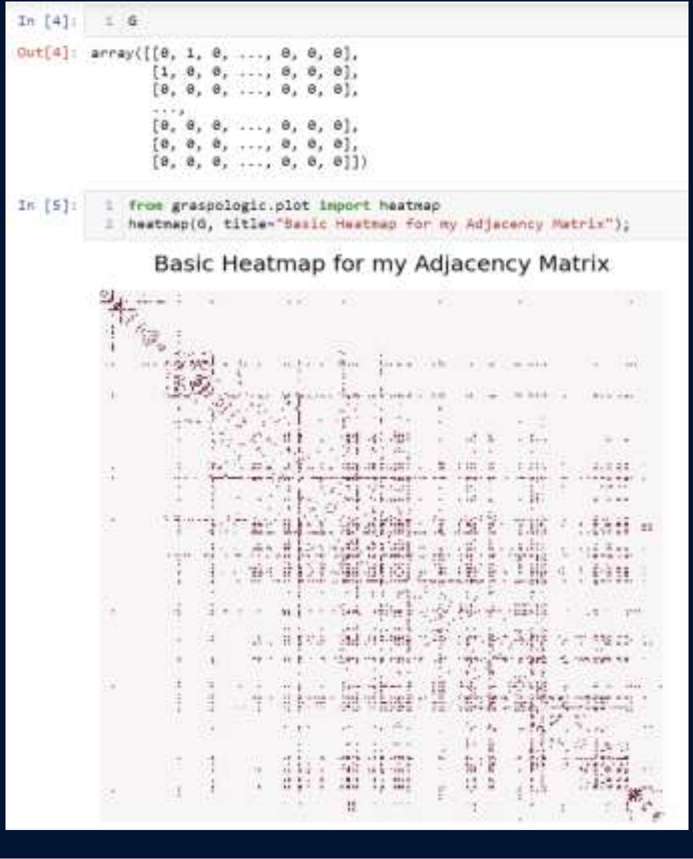
Nodes 2 and 1 aren't connected

My dataset: the adjacency matrix

I found network data about the US Airline in 1997. The first two columns are nodes, which indicate there is an edge between those two edges. The third column indicates the weight of the edge, which might be the frequency. I convert the data into an adjacency matrix.



```
2 1 .0436
4 1 .0767
8 1 .1026
4 2 .0515
8 2 .0866
5 3 .0269
8 3 .0843
8 4 .0365
26 4 .0915
47 4 .2109
8 5 .0849
7 6 .0197
8 6 .0683
13 6 .0143
8 7 .0488
13 7 .014
13 8 .0604
16 8 .0239
23 8 .0478
24 8 .0414
26 8 .083
27 8 .0351
28 8 .1099
```

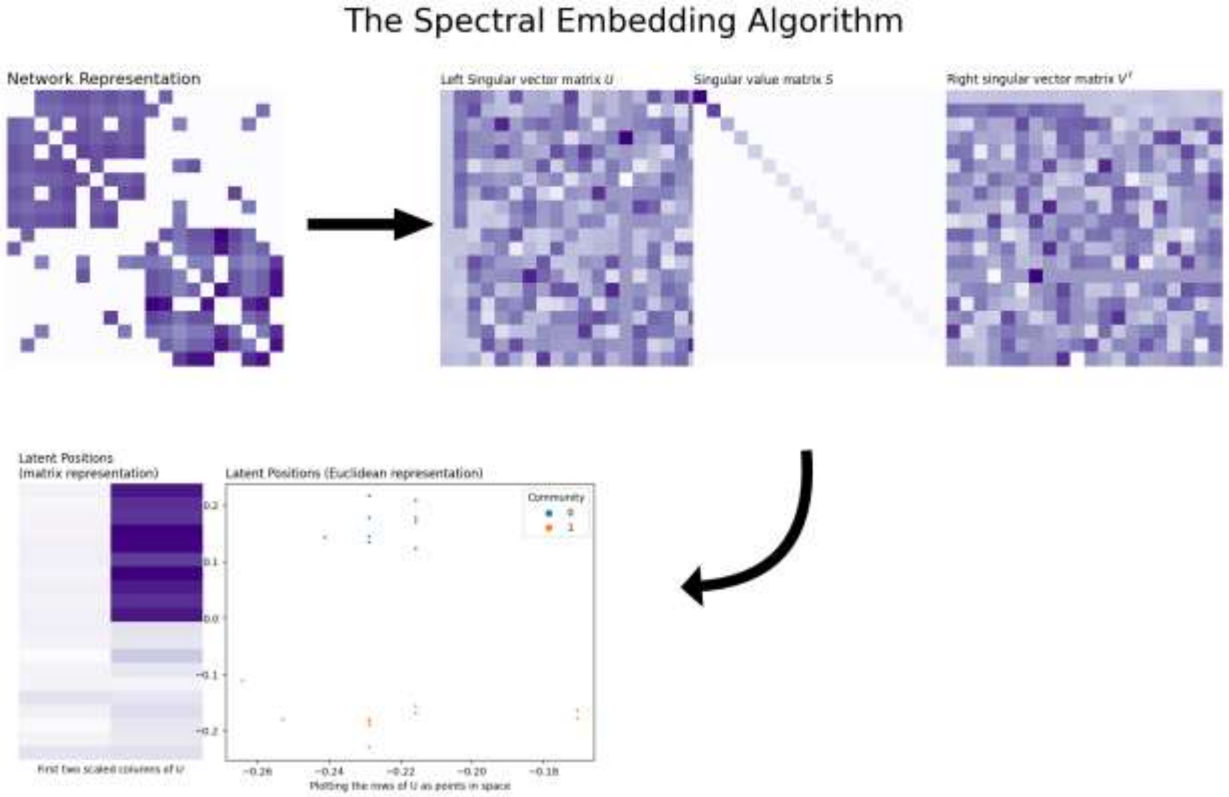


```
In [4]:  G
Out[4]:  array([[0, 1, 0, ..., 0, 0, 0],
                [1, 0, 0, ..., 0, 0, 0],
                [0, 0, 0, ..., 0, 0, 0],
                ...,
                [0, 0, 0, ..., 0, 0, 0],
                [0, 0, 0, ..., 0, 0, 0],
                [0, 0, 0, ..., 0, 0, 0]])

In [5]:  from graspologic.plot import heatmap
         heatmap(G, title="Basic Heatmap for my Adjacency Matrix");
```

Basic Heatmap for my Adjacency Matrix

Converting the network to a tabular format

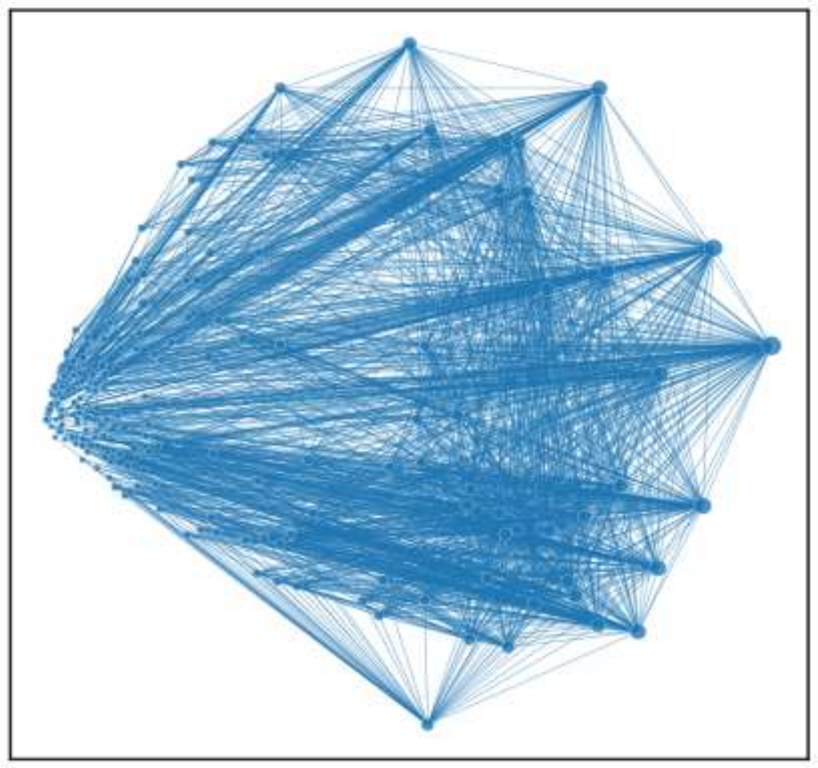
- Spectral Embedding estimates a tabular representation (called latent positions).
- We embed the adjacency matrix into a low dimensional representation.
- This is done using linear algebra (singular value decomposition)

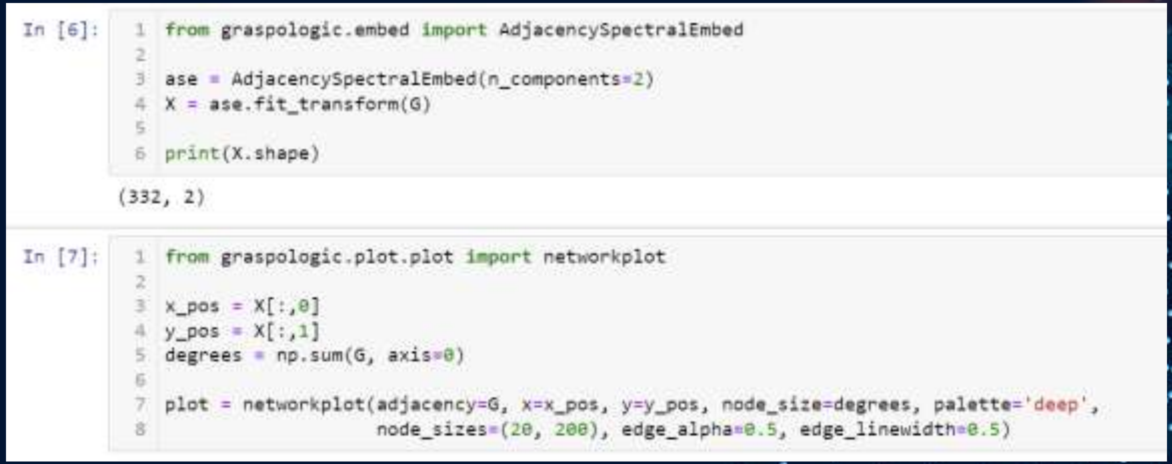

The Spectral Embedding Algorithm

Network Representation
Left Singular vector matrix U
Singular value matrix S
Right singular vector matrix V^T
Latent Positions
(matrix representation)
First two scaled columns of U
Latent Positions (Euclidean representation)
Community
● 0
● 1
0.2
0.1
0.0
−0.1
−0.2
−0.26
−0.24
−0.22
−0.20
−0.18
Plotting the rows of U as points in space

My dataset: learning an embedding

- I embedded the adjacency matrix
- We can plot the first two dimensions of this embedding and draw the graph





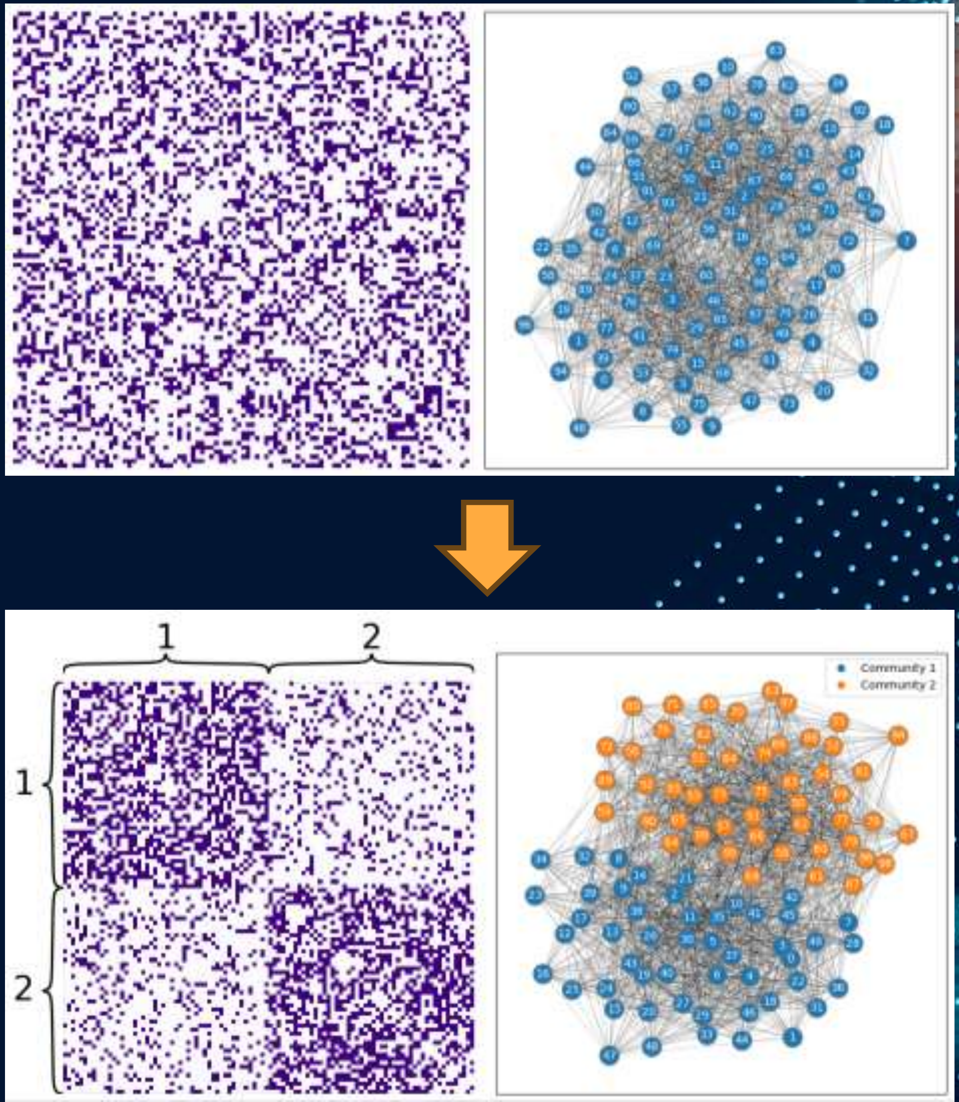
```
In [6]:  from graspologic.embed import AdjacencySpectralEmbed

         ase = AdjacencySpectralEmbed(n_components=2)
         X = ase.fit_transform(G)

         print(X.shape)

         (332, 2)

In [7]:  from graspologic.plot.plot import networkplot

         x_pos = X[:,0]
         y_pos = X[:,1]
         degrees = np.sum(G, axis=0)

         plot = networkplot(adjacency=G, x=x_pos, y=y_pos, node_size=degrees, palette='deep',
                            node_sizes=(20, 200), edge_alpha=0.5, edge_linewidth=0.5)
```

Clustering (tabular) data using K-means

The k-means clustering algorithm is a partitioning method used to group a set of data points into k clusters based on their similarity.

Clustering methods are meant for tabular data, and so we can use them after embeddings.

Example (right): data from two clusters is simulated. (Top) it is hard to see any clusters. (Bottom) a clustering algorithm learns communities.



My dataset: clustering the airports

I used K-means on the network adjacency matrix embeddings to cluster the airports into two communities.