PySGN

A Python Package for Constructing Synthetic Geospatial Networks

Boyu Wang *University at Buffalo*
Andrew Crooks *University at Buffalo*
Taylor Anderson *George Mason University*
Andreas Züfle *Emory University*

2025 AAG Annual Meeting, Detroit, MI

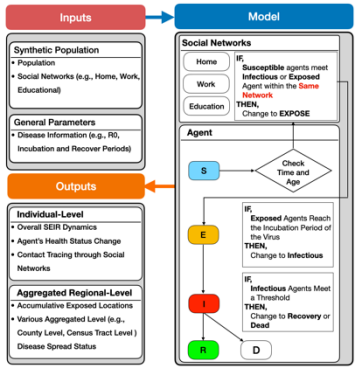
University at Buffalo The State University of New York

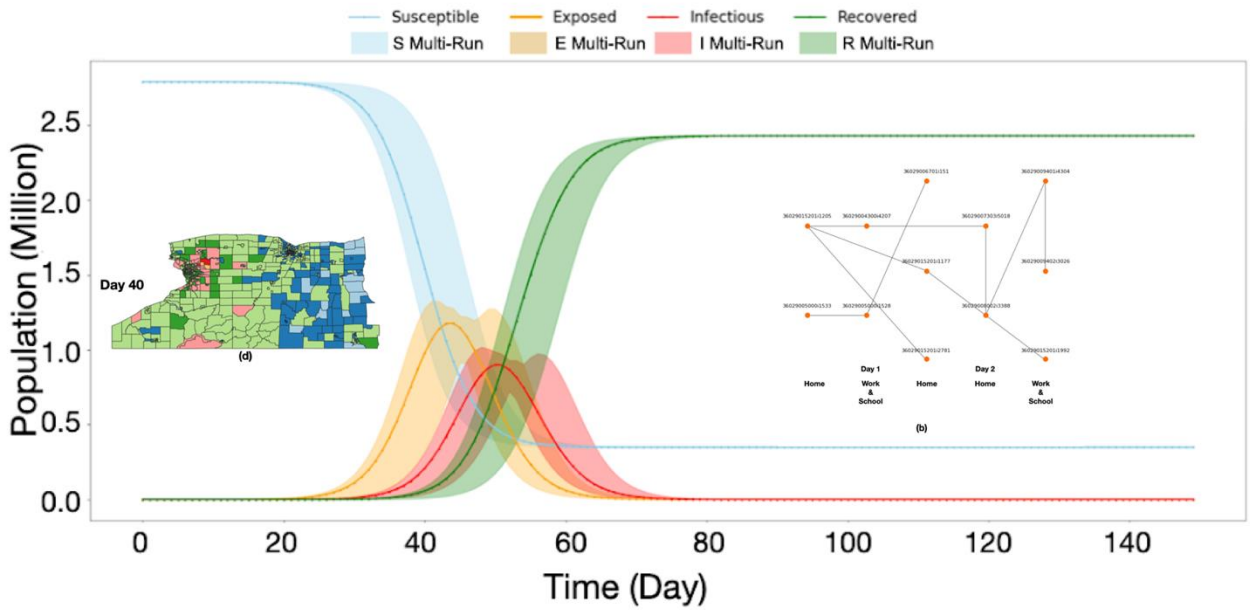
Introduction

- Synthetic geospatial networks integrate geographic information with connectivity, enabling realistic simulations of spatial phenomena across various domains.
- They are often used as null models in hypothesis testing within network analysis.
- Synthetic geospatial networks are also essential pieces in simulations such as disease modeling, pedestrian/traffic modeling, social interactions, and so on.
- PySGN extends classical network models with spatial considerations, producing synthetic geospatial networks that mirror real-world spatial characteristics.

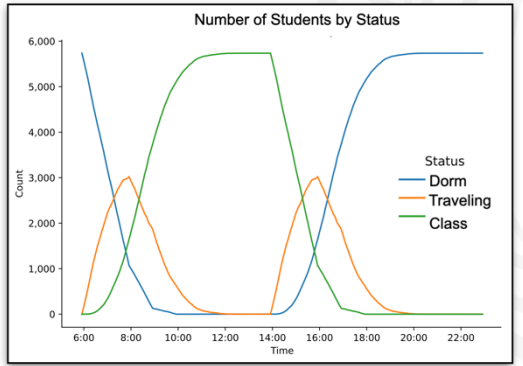
Introduction

Disease Modeling

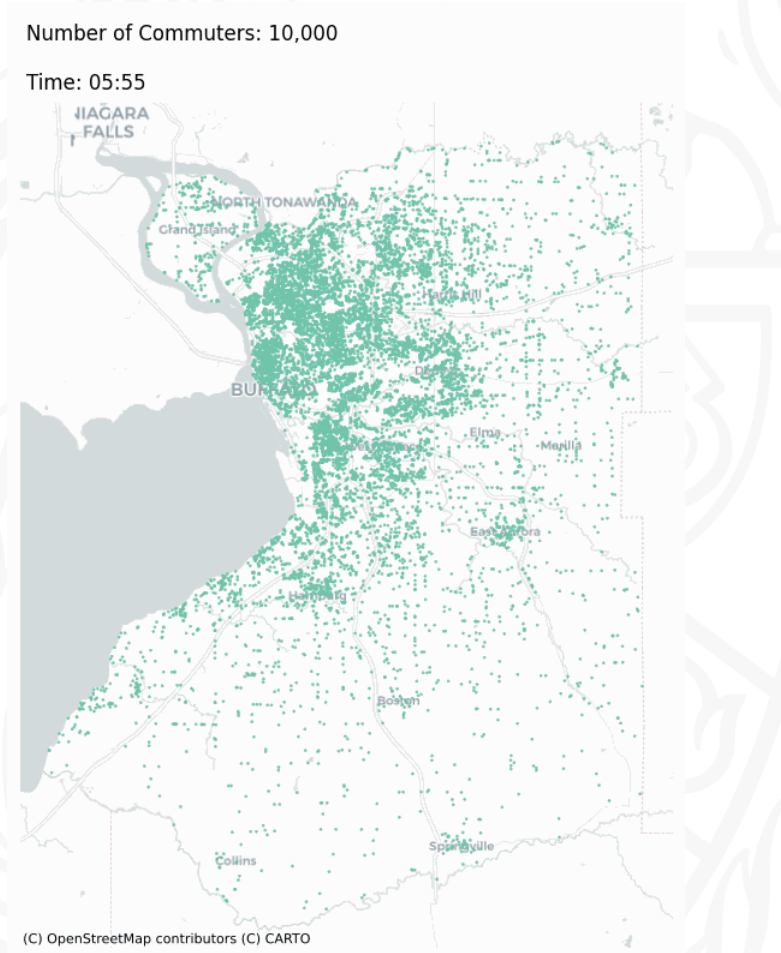




Pedestrian Modeling

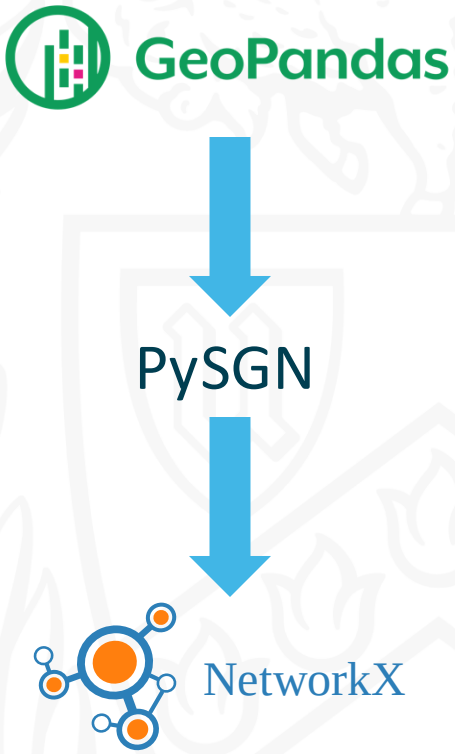


Traffic Modeling



Package Overview

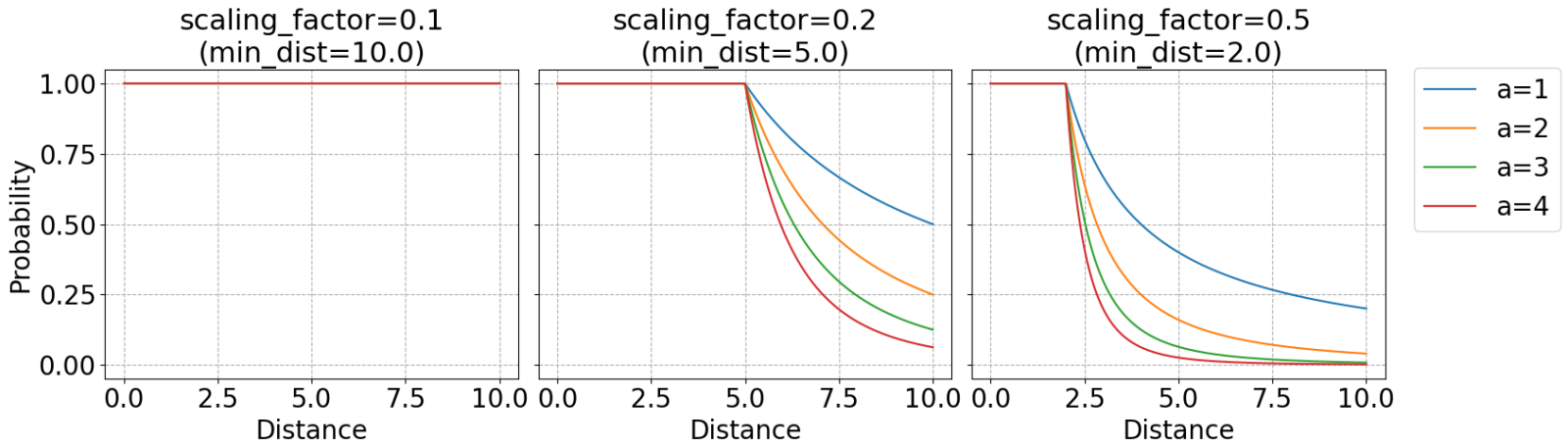
- PySGN: a Python package for constructing synthetic geospatial networks.
- It extends three classical network models by incorporating the locations of nodes:
  - Geospatial Erdős-Rényi
  - Geospatial Barabási–Albert
  - Geospatial Watts-Strogatz
- Offers an easy-to-use API, code examples, and documentations.



Model: Geospatial Erdős–Rényi

- Classical Erdős–Rényi: each edge added with a constant probability p
- Geospatial Erdős–Rényi: edge probability decreases with edge length d, controlled by minimum edge length min_dist and decay exponent a:

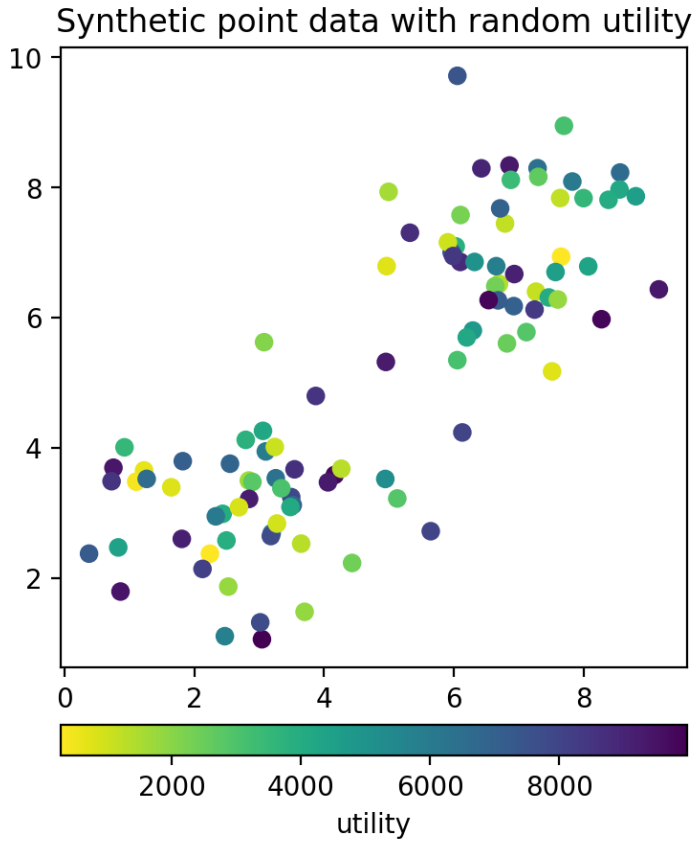
$$p(d|a, \text{min\_dist}) = \min\left(1, \left(\frac{d}{\text{min\_dist}}\right)^{-a}\right)$$

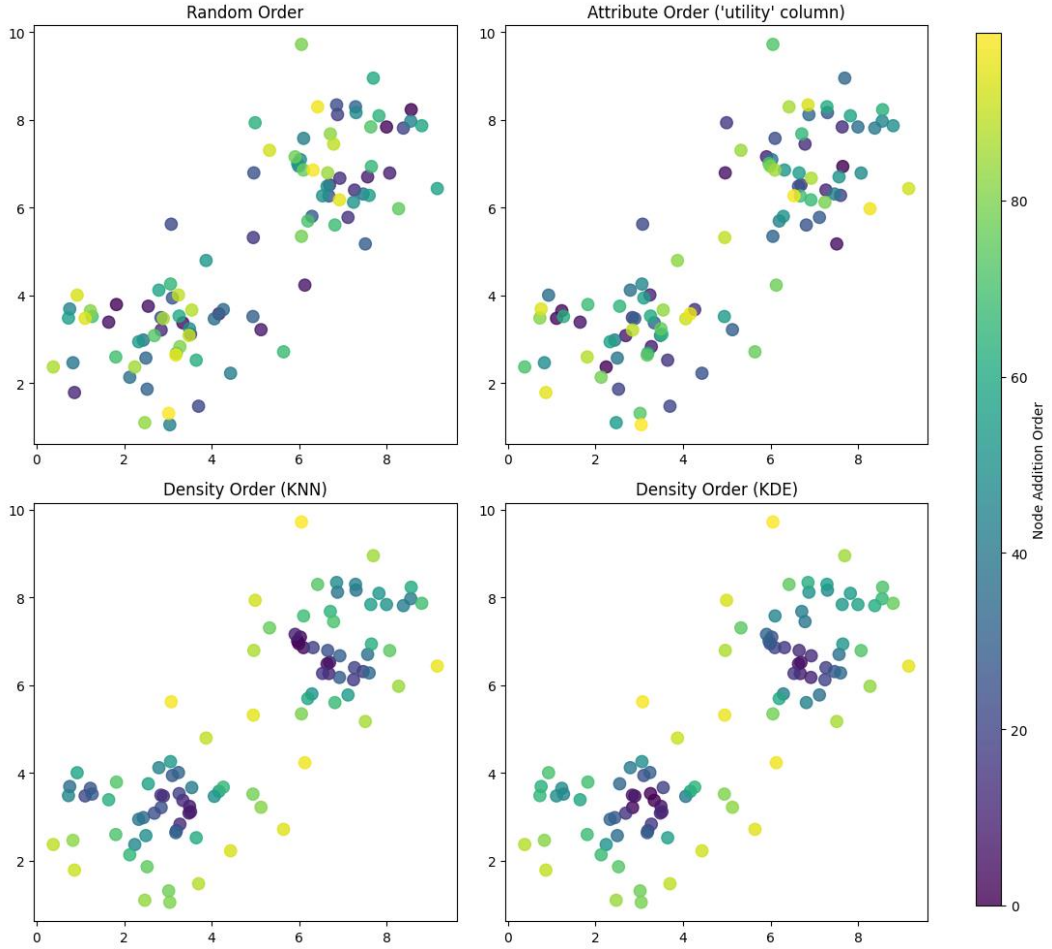


Model: Geospatial Barabási-Albert

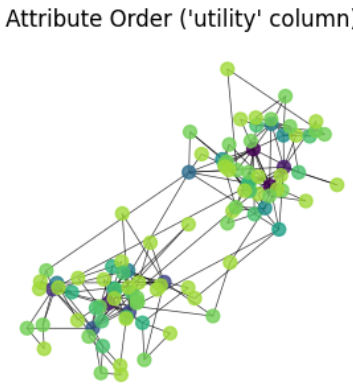
- Classical Barabási-Albert:
  - Nodes added sequentially
  - Probability of a new node to connect to an existing node i is $p_i \propto k_i$, the degree of node i
- Geospatial Barabási-Albert:
  - Nodes added sequentially
  - Probability of a new node to connect to an existing node i is $p_i \propto k_i \cdot \min\left(1, \left(\frac{d}{\text{min\_dist}}\right)^{-a}\right)$
  - Flexible node ordering: order in which nodes are added
    - random, by attribute value, by spatial density (KNN or KDE), or through user-defined functions

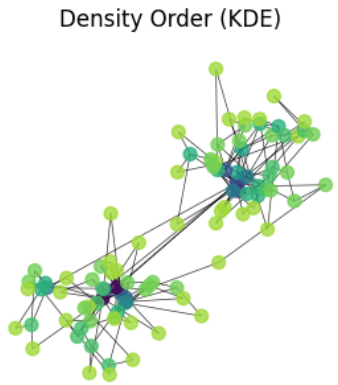
Model: Geospatial Barabási-Albert

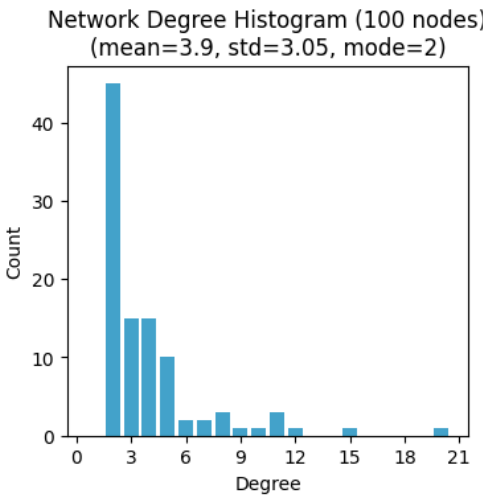


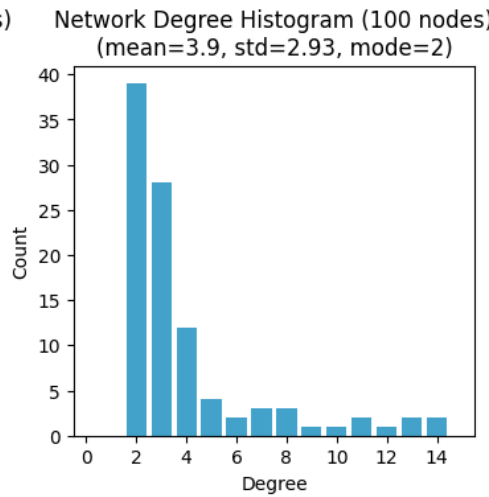


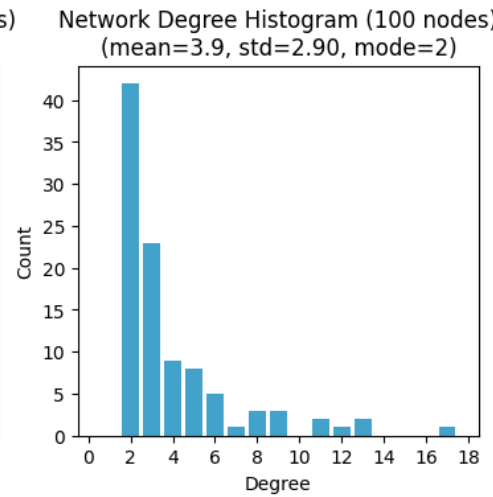
Model: Geospatial Barabási-Albert

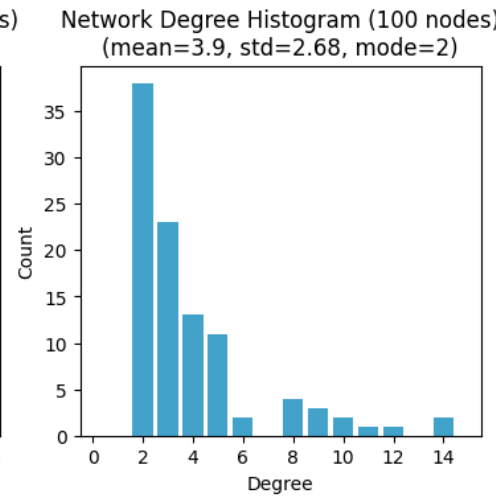










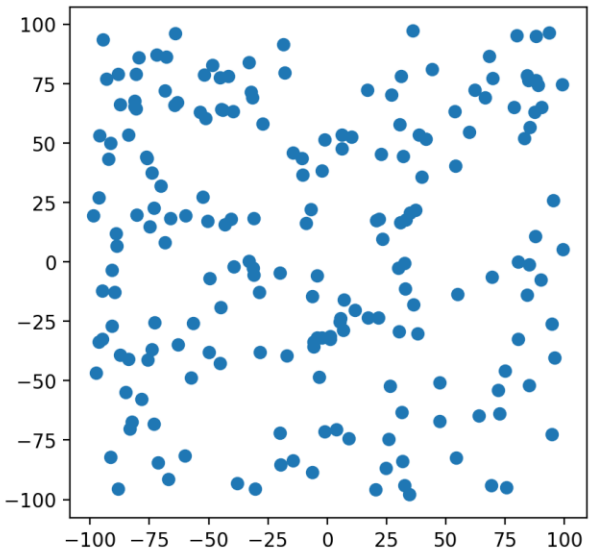


Model: Geospatial Watts-Strogatz

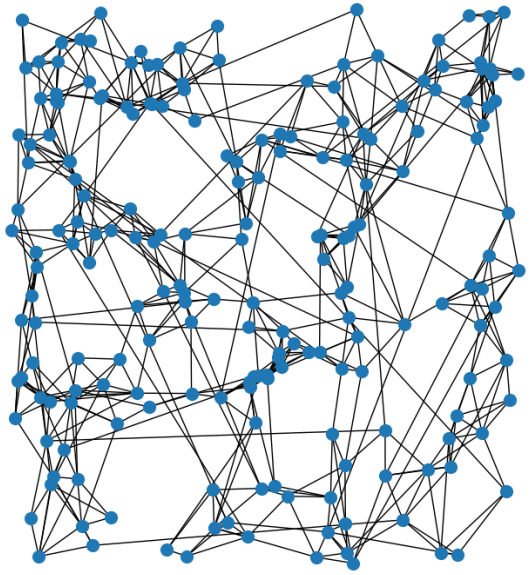
- Classical Watts-Strogatz:
  - Create a ring lattice with each node connected to its k neighbors
  - Rewire each edge with probability p to a random node
- Geospatial Watts-Strogatz:
  - Connect each node with its k nearest neighbors
  - Rewire each edge with probability p to a random node
  - The target node is selected with probability $p(d|a, \text{min\_dist}) = \min\left(1, \left(\frac{d}{\text{min\_dist}}\right)^{-a}\right)$

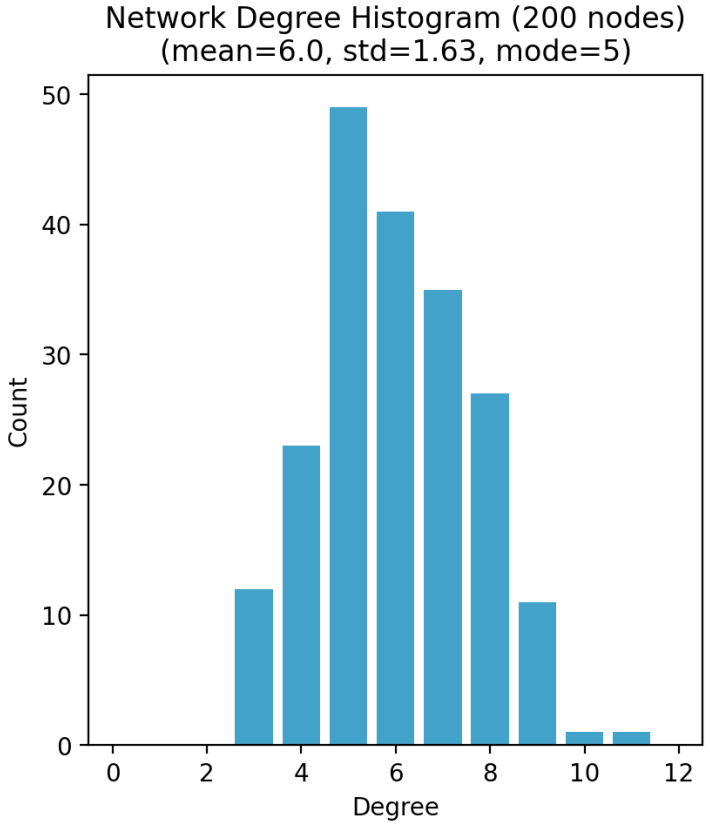
Model: Geospatial Watts-Strogatz

Synthetic point data with locations

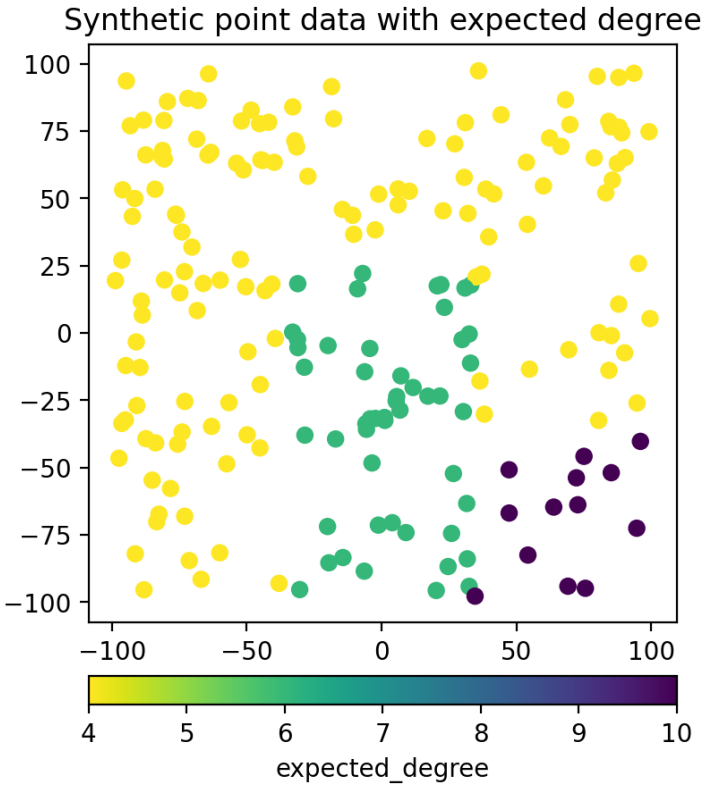


Geospatial Watts-Strogatz Network



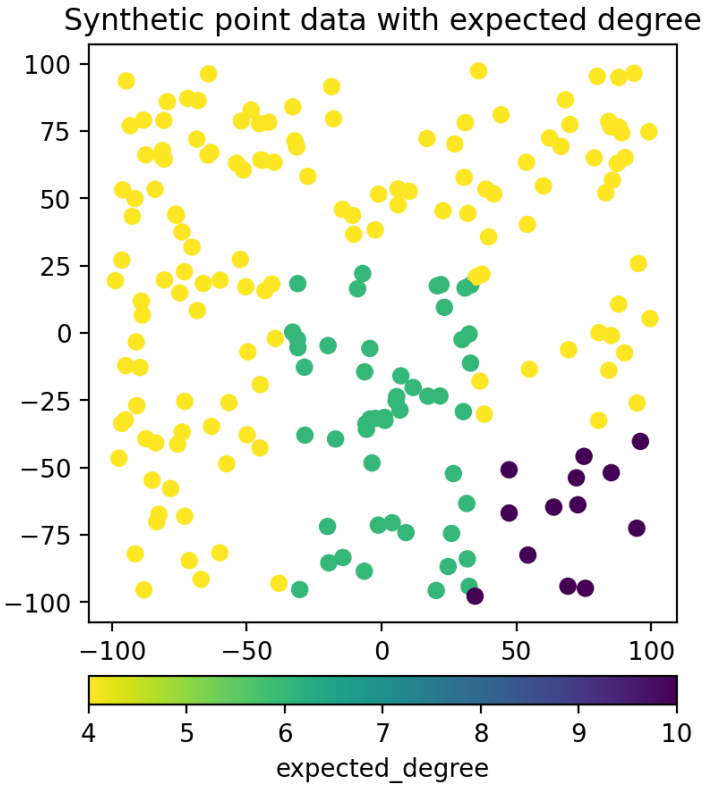


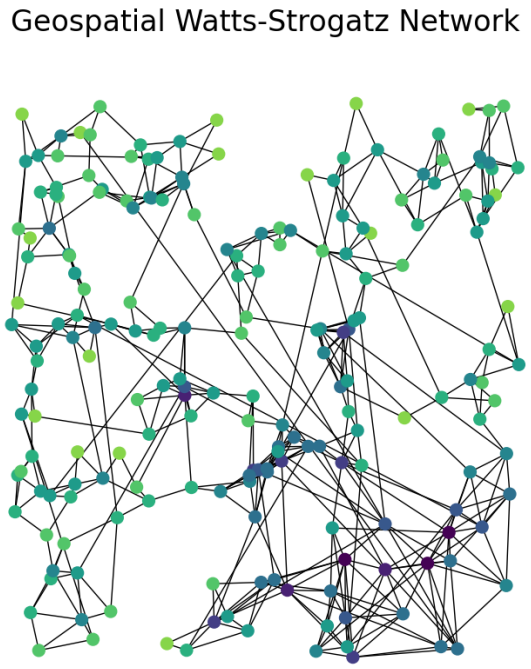
Model: Geospatial Watts-Strogatz

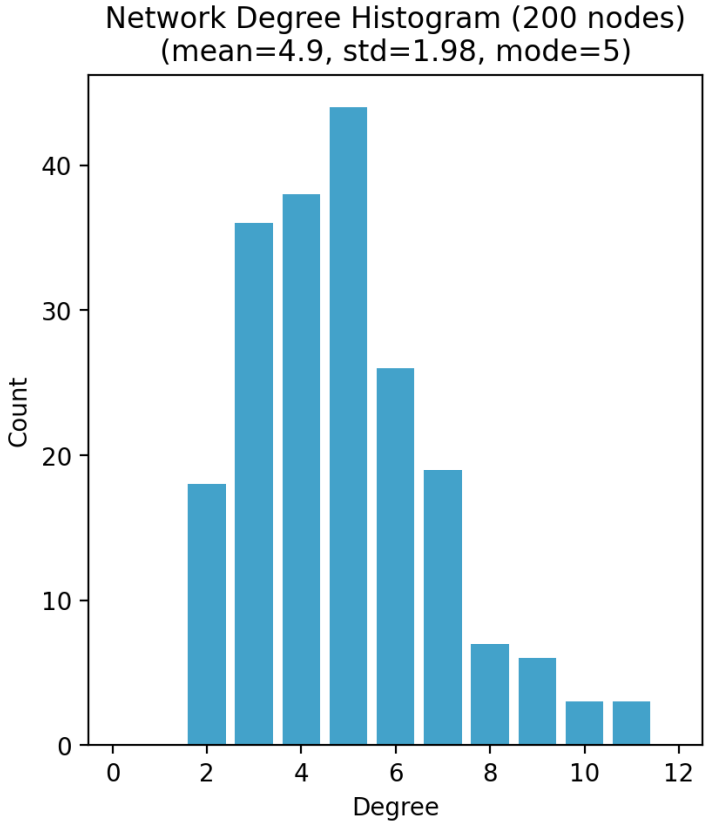


	geometry	expected_degree
0	POINT (1.084 -31.419)	6
1	POINT (-41.849 78.248)	4
2	POINT (-28.621 -12.754)	6
3	POINT (-52.381 27.293)	4
4	POINT (-72.945 -25.463)	4

Model: Geospatial Watts-Strogatz







Code Examples

Install `pysgn` using pip:

```
$ pip install pysgn
```

Import functions and load data:

```
import geopandas as gpd
from pysgn import (
    geo_erdos_renyi_network,
    geo_barabasi_albert_network,
    geo_watts_strogatz_network,
)
```

Load data using GeoPandas:

```
gdf = gpd.read_file('your_gis_data.shp')
```

```
graph = geo_erdos_renyi_network(
    gdf,
    a=3,              # optional
    scaling_factor=0.5, # optional
)

graph = geo_barabasi_albert_network(
    gdf,
    m=3,
    node_order='utility', # optional
    a=3,                  # optional
    scaling_factor=0.5,   # optional
)

graph = geo_watts_strogatz_network(
    gdf,
    k=4,
    p=0.1,
    a=3,              # optional
    scaling_factor=0.5, # optional
)
```

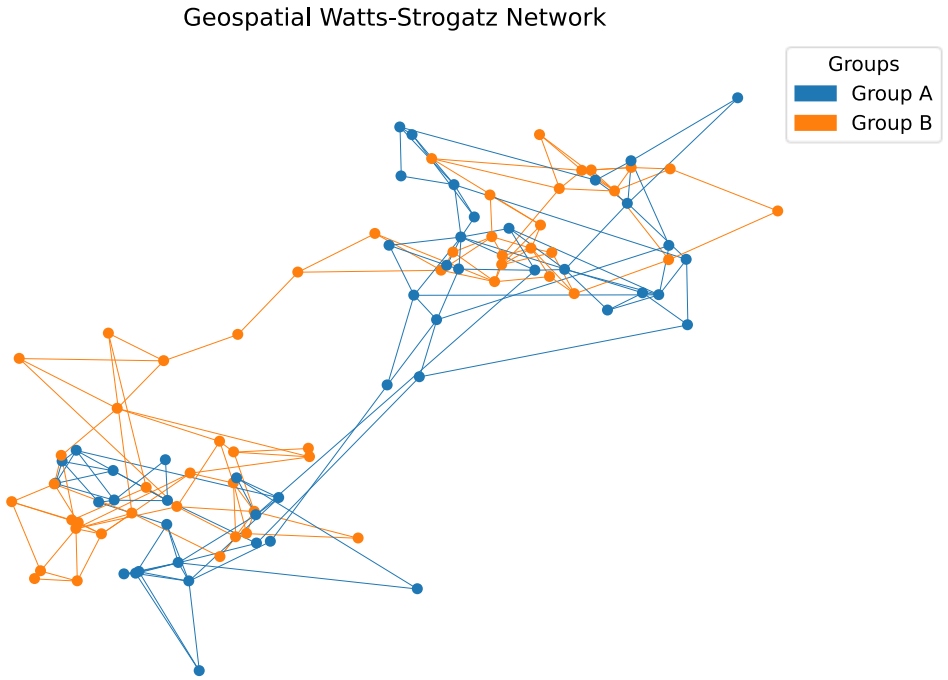
Code Examples

	utility	geometry	group
0	6579	POINT (3.6279 2.11581)	Group B
1	2582	POINT (1.87378 3.22788)	Group B
2	1560	POINT (5.09026 2.39971)	Group B
3	3365	POINT (1.66496 1.77868)	Group B
4	6191	POINT (3.81644 5.51679)	Group B

Set custom constraints on edges:

```
graph = geo_watts_strogatz_network(
    gdf,
    k=4,
    p=0.3,
    constraint=lambda u, v: u.group == v.group,
)
```

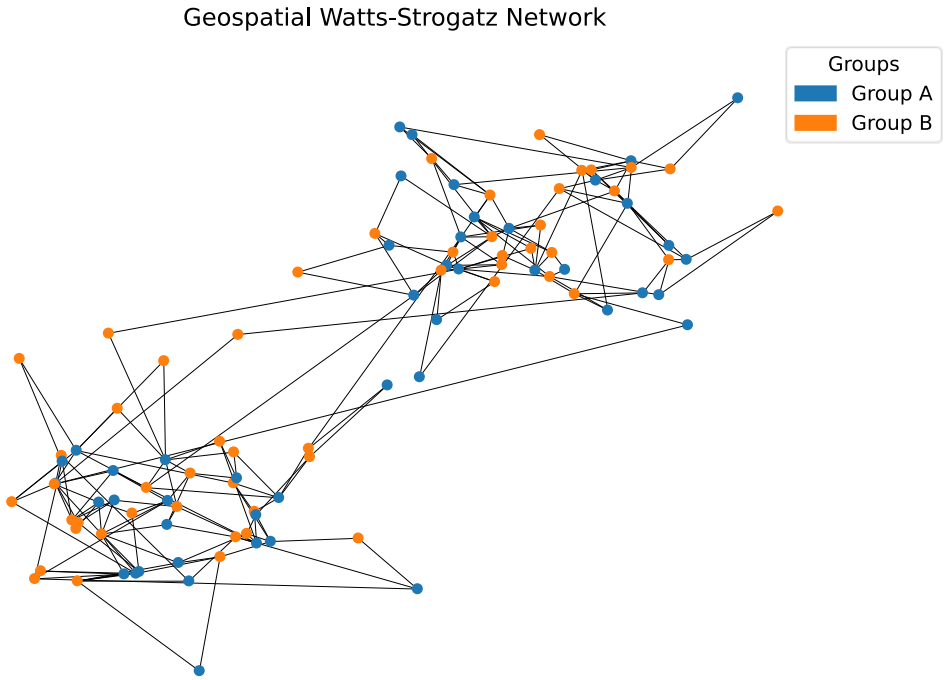
Code Examples



Set custom constraints on edges:

```
graph = geo_watts_strogatz_network(
    gdf,
    k=4,
    p=0.3,
    constraint=lambda u, v: u.group == v.group,
)
```

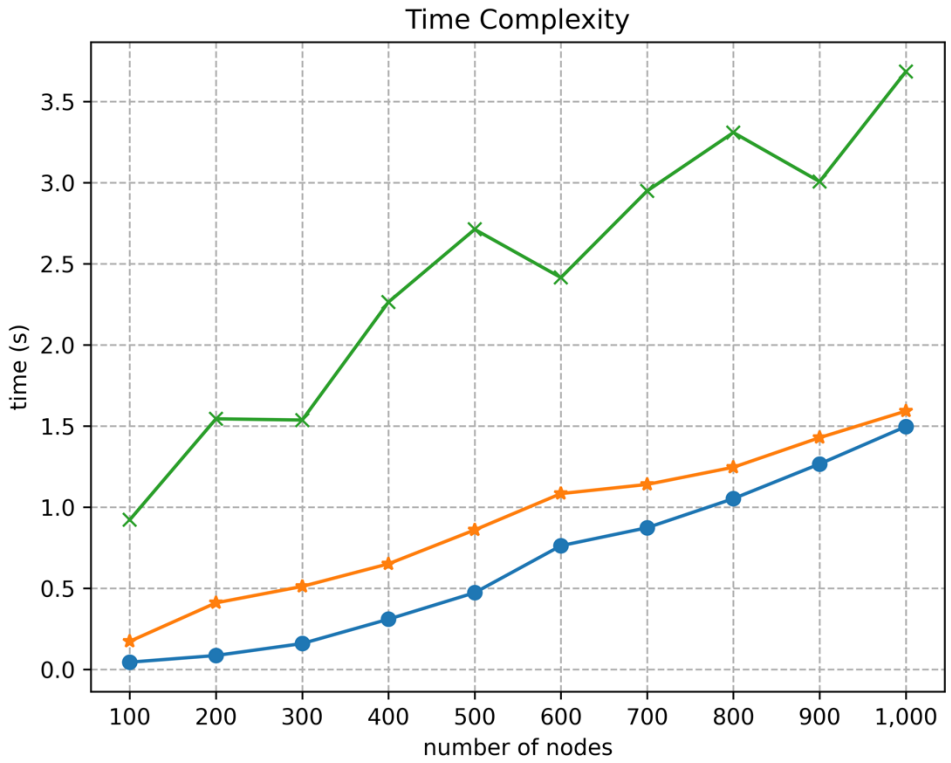
Code Examples

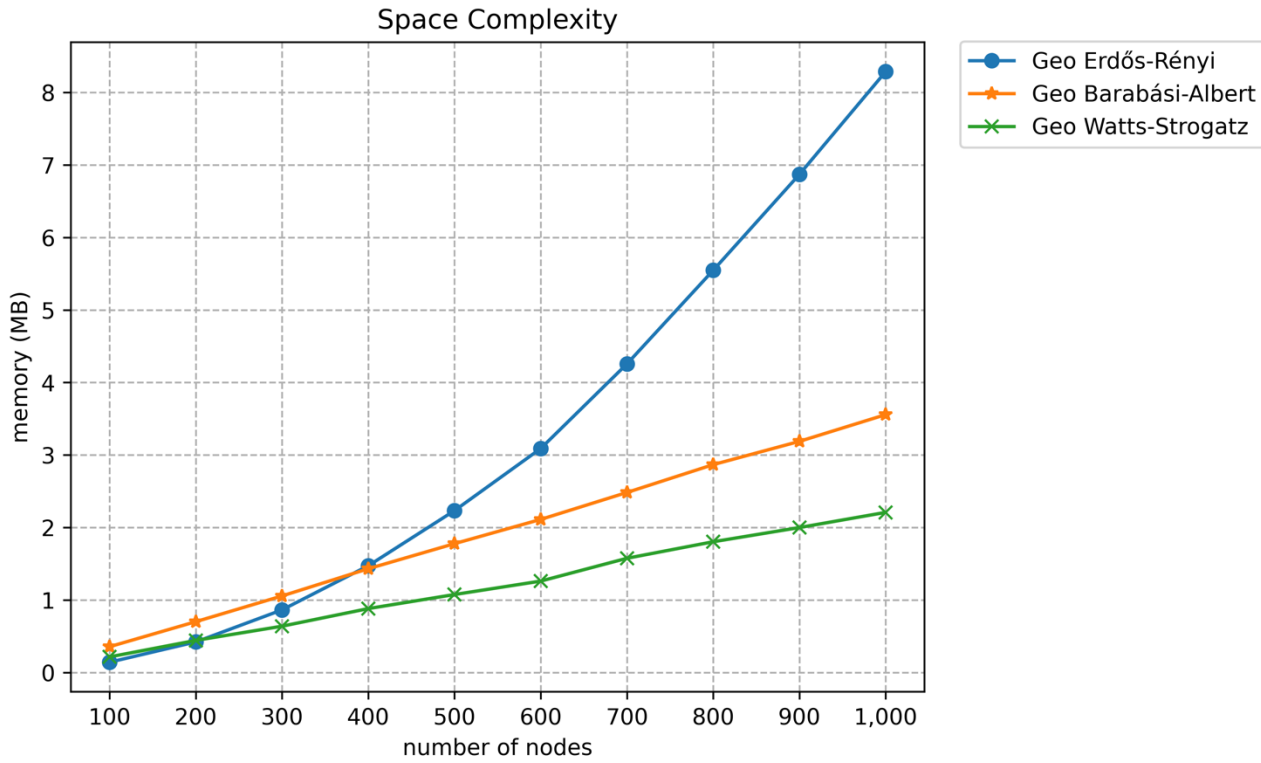


Set custom constraints on edges:

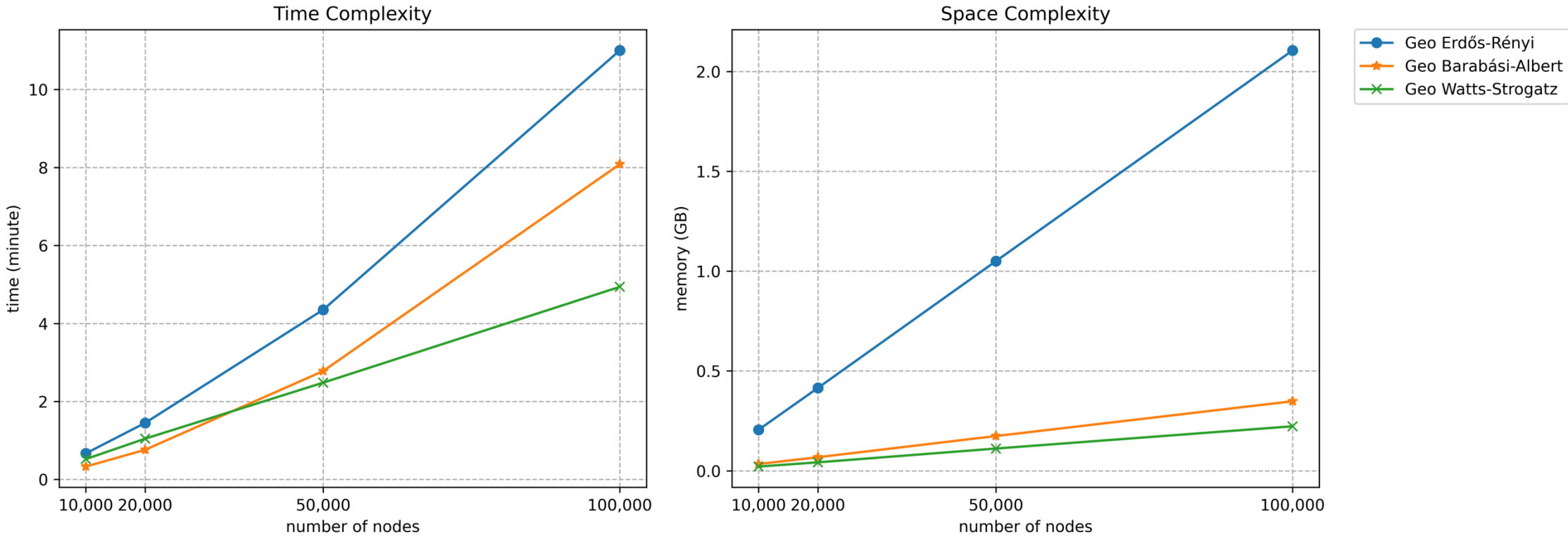
```
graph = geo_watts_strogatz_network(
    gdf,
    k=4,
    p=0.3,
    constraint=lambda u, v: u.group != v.group,
)
```

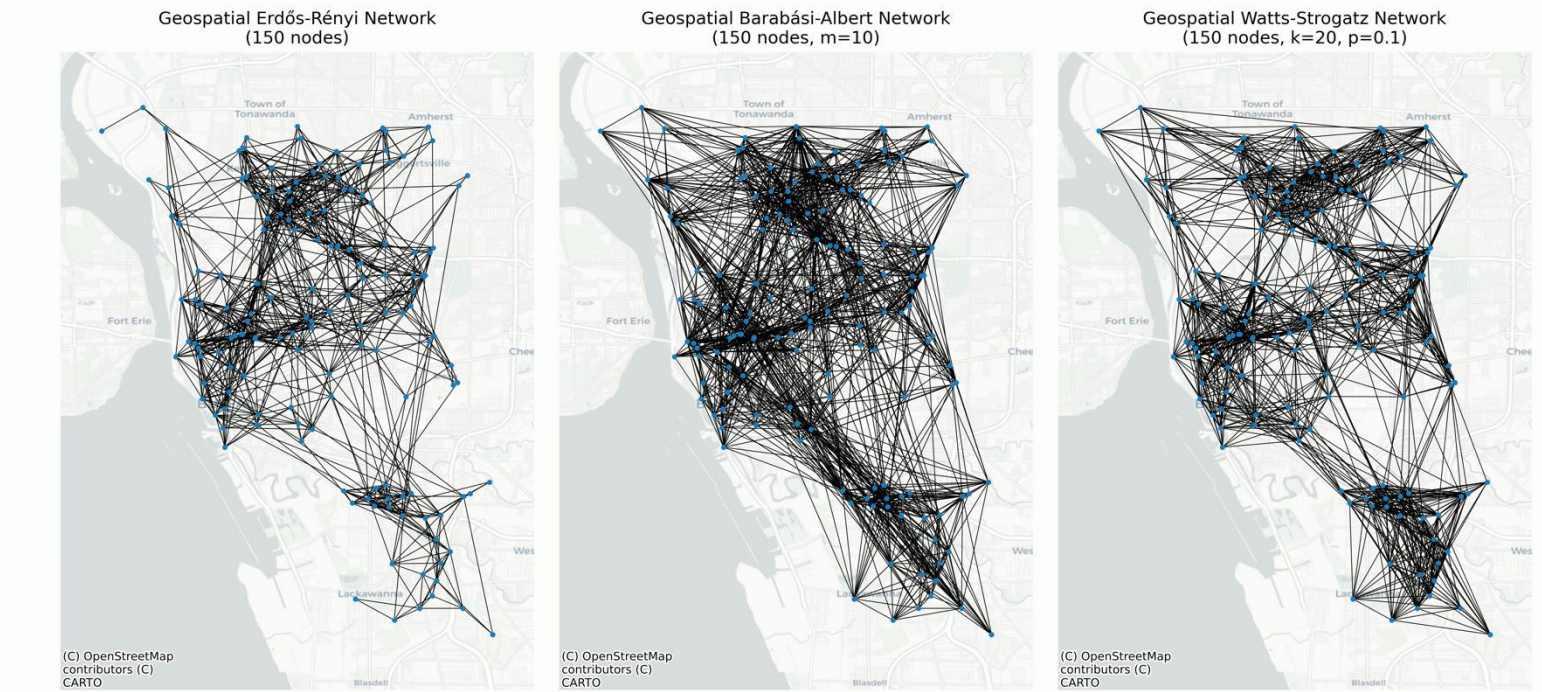
Performance: up to 1,000 nodes

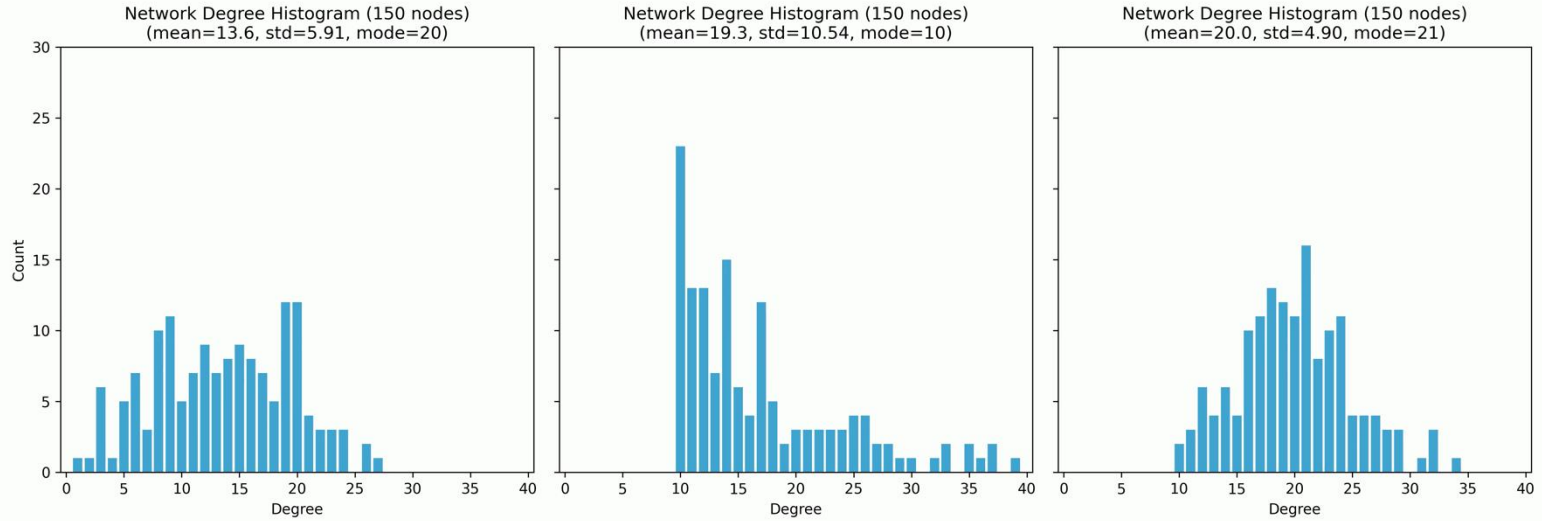




Performance: up to 100k nodes




Geospatial Erdős-Rényi Network
(150 nodes)
Geospatial Barabási-Albert Network
(150 nodes, m=10)
Geospatial Watts-Strogatz Network
(150 nodes, k=20, p=0.1)
Town of Tonawanda
Amherst
Fort Erie
Lackawanna
(C) OpenStreetMap
contributors (C)
CARTO


Network Degree Histogram (150 nodes)
(mean=13.6, std=5.91, mode=20)
Network Degree Histogram (150 nodes)
(mean=19.3, std=10.54, mode=10)
Network Degree Histogram (150 nodes)
(mean=20.0, std=4.90, mode=21)
Count
Degree

Conclusion

- PySGN provides a robust toolkit for generating synthetic geospatial networks by extending classical network models with spatial considerations.
- It integrates with the PyData ecosystem (e.g., GeoPandas, NetworkX), providing flexible API for custom constraints and various use-cases.
- Free and open-source! Source code available at: https://github.com/wang-boyu/pysgn

https://pysgn.readthedocs.io

Acknowledgement

The algorithms implemented in PySGN are based on the following work with several improvements and modifications, including bug fixes, performance enhancements, and additional features. We would like to thank the authors for their contributions to the field of synthetic geospatial network generation.

- Gallagher, K., Anderson, T., Crooks, A., & Züfle, A. (2023). Synthetic Geosocial Network Generation. In *Proceedings of the 7th ACM SIGSPATIAL Workshop on Location-based Recommendations, Geosocial Networks and Geoadvertising* (pp. 15-24).
- Alizadeh, M., Cioffi-Revilla, C., & Crooks, A. (2017). Generating and analyzing spatial social networks. *Computational and Mathematical Organization Theory*, 23, 362-390.

University at Buffalo The State University of New York

Thank you for listening!
Welcome comments, questions and suggestions.

bwang44@buffalo.edu
atcrooks@buffalo.edu
tander6@gmu.edu
azufle@emory.edu

@BoyuWang_
@AndyCrooks
@hellotaylora
@andreaszuefle

https://wang-boyu.github.io
https://gisagents.org
https://bit.ly/4bZmnPO
https://zuefle.org