HCD Simulations Write Up

Audrey Fu Lab

02 May, 2024

Contents

1 Data Simulation

1.0.1 Simulating the network

We adopt a top-down approach to simulate hierarchical networks, considering various simulation parameters such as graph sparsity, noise, and the architecture of the super-level graph(s), including small-world, scale-free, and random graph networks (Watts and Strogatz 1998; Barabási and Bonabeau 2003).

Our simulations focus on basic hierarchies comprising one or two hierarchical layers. Two-layer networks mirror classical community detection on graphs, where our aim is to recover the true community labels from a given graph. Meanwhile, three-layer networks present a more intricate scenario, where the bottom layer of the hierarchy contains two levels of community structure. Here, the top level corresponds to the nodes at the uppermost layer of the hierarchy, and the middle level consists of communities nested within the top-level communities. The objective with these networks is to identify both sets of community partitions.

In each hierarchy, for fully connected networks, we initiate by simulating n_{top} top-level nodes, adhering to a directed small-world, random graph, or scale-free network architecture (Watts and Strogatz 1998; Barabási and Bonabeau 2003). In cases where the network is disconnected, we simply simulate n_{top} disconnected nodes. For networks with three hierarchical layers, we then generate a subnetwork of n_{middle} nodes from each top-layer node, adhering to the network structure utilized at the top level. If the network is fully connected, we apply a probability p_{between} to the nodes from different top-level communities being connected.

The final step in all hierarchies is to generate the nodes in the observed (bottom) layer of the hierarchy. For each top-layer or middle-layer node, we generate a subnetwork of n_{bottom} nodes under the same subnetwork structure as the previous layers, and we apply a probability p_{between} for nodes from different communities to share an edge.

1.0.2 Simulating gene expression

Once we simulate a hierarchical graph, we utilize this hierarchy to generate the node-feature matrix, which depicts the expression of N genes across p samples. Here, N denotes the number of nodes in the observed (bottom) layer of the hierarchy, and its range is governed by $a^{\ell+1} < N < a \times b^{\ell}$, where ℓ signifies the number of hierarchical layers.

We simulate the node-feature matrix using the topological order the observed level graph. We start by generating the features of nodes that have no parental input. We refer to these nodes as *origin* nodes. All origin nodes are simulated from a normal distribution with mean 0 and standard deviation σ. All other nodes are simulated from a normal distribution centered at the mean of their parent nodes and with standard deviation σ.

2 Datasets

We consider three sets of hierarchical networks which represent varying difficulty levels for inference:

1. **Complex networks** - used for final simulation assessment - **Table ?? - ??** .
2. **Intermediate networks** - used for investigative model tuning and performance assessment - **Table** 5 .
3. **Simple networks** - used for code implementation and debugging - **Table ??**.

3 Application to Intermediate Networks

A comprehensive overview of the intermediate networks is presented in **Table** 5 . These networks are structured as three-layered systems, each characterized by small-world, scale-free, or random graph architectures. In contrast to the more intricate networks featured in the **Complex Networks** dataset, the intermediate networks exhibit a comparatively simpler configuration. Specifically, each network comprises 5 super layer nodes, 15 middle layer nodes, and approximately 300 bottom layer nodes. Our primary focus in utilizing this dataset is to examine the performance of the Hierarchical Community Detection (HCD) method when applied to three-layer networks. The smaller scale of these networks facilitates a more in-depth analysis of the detected communities within the middle and upper layers of their hierarchical structures.

We apply the HCD method to each network separately using three options for the input graph corresponding to the nodes at the observed layer of the hierarchy:

- The input graph is the true graph
- The input graph is the correlation matrix of the simulated gene expression
- The input graph is the correlation matrix of the simulated gene expression wherein correlations weaker than 0.2 are disregarded and set to zero
- The input graph is the correlation matrix of the simulated gene expression wherein correlations weaker than 0.5 are disregarded and set to zero
- The input graph is the correlation matrix of the simulated gene expression wherein correlations weaker than 0.7 are disregarded and set to zero

We also explore various combinations of weighting the loss function across each of the aforementioned input graphs. In all cases, we ensure that the predicted number of communities in the middle or top levels of the hierarchy aligns with the ground truth of the simulation.

3.0.1 Evaluating performance

We evaluate the performance of our HCD method using three graph-based clustering metrics:

1. **homogeneity** evaluates the degree to which each predicted community contains only data points from a single true community, indicating how well the algorithm avoids mixing different groups. Thus, homogenity tends to be high if resolved communities contain only members of the same true community.
2. **completeness** assesses the extent to which all data points that belong to the same true community are correctly assigned to a single predicted community. Thus completeness is always high if all members of the same true communities end up in the same resolved community even if several true communities are allocated together.
3. **NMI** is a weighted average of the previous two metrics.

For each simulation, we configure the number of communities in the middle and upper layers of the hierarchy to match the true count in each layer. Then, we evaluate the community predictions of the Hierarchical Community Detection (HCD) algorithm at these levels against the actual communities using three metrics. As a baseline, we employ the Louvain method, which utilizes hierarchical graph partitioning to maximize modularity, resulting in a single set of resolved communities. These resolved communities may align with the middle, upper, or a combination of both layers in the true hierarchy. Thus, we compute the performance metrics of the communities identified by the Louvain method against the true communities at both the upper and middle levels of the hierarchy.

4 Results

Previously we apply our HCD method to all complex and intermediate datasets looking at various parameters such as the input graph and some initial test values for loss hyperparameters. Our previous findings are outlined in the lab report from 3/13/2024 and pointed out several issues that remain to be addressed.

1. An additional loss component is needed to help further reduce the tendency of HCD to combine smaller communities into super-communities.

We have since implemented a hierarchical adaptation of the Kmeans approach which aims to ensure members clustered to the same community have the smallest possible dissimilarity. See the latest update of the HCD pseudo code to view the specific mathematical details.

2. How should we assess performance when comparing the Louvain method and our HCD method.

- Louvain uses a heuristic hierarchical community partitioning approach to find the communities that maximize modularity within a local greedy search of the space of possible partitions. The predicted communities are the final community assignments that arise when no additional assignment changes can improve the modularity - when a local optimum of modularity has been achieved. This leads to a single set of community predictions which may represent either the top layer of the true hierarchy or it may represent one of the middle layers. It may also be a blend of middle and top layers.

5 Tables

Summary statistics for intermediate difficulty simulated networks.

Subgraph type	Connection type	Layers	Standard deviation	Nodes per layer
small world	disc	3	0.1	(5, 15, 300)
small world	full	3	0.1	(5, 15, 300)
scale free	disc	3	0.1	(5, 15, 300)
scale free	full	3	0.1	(5, 15, 300)
random graph	disc	3	0.1	(5, 12, 167)
random graph	full	3	0.1	(5, 12, 167)

Edges per layer

(0, 15, 354)

(10, 25, 300)

(0, 10, 966)

(10, 20, 300)

(0, 7, 133)

(10, 17, 167)

Subgraph probability

0.05

0.05

0.05

0.05

0.05

0.05

Sample size

500

500

500

500

500

500

Modularity (top)

0.799

0.715

0.78

0.751

0.791

0.665

Average node degree top

1.18

1.34

3.22

3.32

0.796

0.886

Avg connections within top communities

70.8

73.6

193.2

193.2

26.6

26

Avg. connections between top communities

0

1.7

0

1.5

0

0.9

Modularity (middle)

0.781

0.679

0.873

0.845

0.787

0.696

Average node degree middle

1.18

1.34

3.22

3.32

0.796

0.886

Avg connections within middle communities

20

20

61.333

61.333

9.667

9.667

Avg connections between middle communities

0.257

0.486

0.219

0.362

0.129

0.242

6 Figures

References

Barabási, Albert-László, and Eric Bonabeau. 2003. “Scale-Free Networks.” *Scientific American* 288 (5): 60–69.

Watts, Duncan J, and Steven H Strogatz. 1998. “Collective Dynamics of ‘Small-World’networks.” *Nature* 393 (6684): 440–42.