Generating Very Wide Small-world Bipartite Networks

Ross Pantone

June 2, 2020

1 Introduction

In this document, we present an efficient method of generating very wide small-world bipartite networks. Our method also allows the user to approximately set the sparsity of the network.

2 Algorithm

Recall that distance-dependent networks are (generally) small-world networks. See section (4.2) of this paper.

Consider a bipartite network with n_1 nodes in one class and n_2 nodes in the other class. Label the first class of nodes as $\{x_1, x_2, \cdots, x_{n_1}\}$ and the second class of nodes as $\{y_1, y_2, \cdots, y_{n_2}\}$.

For $i \in \{1, 2, \cdots, n_1\}$ and $j \in \{1, 2, \cdots, n_2\}$, we define the probability of there being an edge between x_i and y_j by

$$P(x_i, y_j) = \alpha \left(|i - j + 0.5\,|n_2 - n_1|| + \beta \right)^{-\lambda},$$

where $\alpha, \beta, \lambda \in \mathbb{R}^+$. Note that we are using the indices to define a distance.

To create a bipartite network of class sizes n_1 and n_2 with desired sparsity s, first calculate the corresponding number of nonzero values by $N' = n_1 n_2 (1 - s)$. Let N be a discrete random variable defining the total number of edges in a bipartite network configured using the method described in these notes. See that

$$\mathbf{E}[N] = \sum_{i,j} P(x_i, y_j).$$

Fix α and β. Hence, to form a network with an approximate number of nonzero entries N', we must solve

$$
\begin{aligned}
N' &= \sum_{i,j} P(x_i, y_j) \\
&= \sum_{i,j} \alpha \left(|i - j + 0.5\,|n_2 - n_1|| + \beta \right)^{-\lambda}
\end{aligned}
$$

for λ. In our current experiments, we use $\alpha = 1$ and $\beta = 1$.

When varying λ and making N' unknown, see that N' decreases monotonically. This fact enables us to efficiently solve for λ (when λ is back to being unknown and N' is known). We use a simple binary search with initial lower bound $\lambda_{\min} = 10^{-5}$, initial upper bound $\lambda_{\max} = 5$, and error $\epsilon = |N' - \sum_{i,j} P(x_i, y_j)|$. We cease the binary search when $\epsilon < 1$.

Using our newly calculated λ, we generate probability matrix $\mathbf{P} \in \mathbb{R}^{n_1 \times n_2}$, where $\mathbf{P}_{ij} = P(x_i, y_j)$. We then generate a reference matrix $\mathbf{A} \in \mathbb{R}^{n_1 \times n_2}$, where $\mathbf{A}_{ij} = \text{Uniform}[0, 1]$. We then generate a Boolean mask matrix $\mathbf{M} \in \mathbb{R}^{n_1 \times n_2}$ by

$$
\mathbf{M}_{ij} = \begin{cases} 0, & \mathbf{P}_{ij} < \mathbf{A}_{ij} \\ 1, & \mathbf{P}_{ij} \geq \mathbf{A}_{ij} \end{cases}.
$$

This mask is the final connectivity matrix for our bipartite small-world network.

In practice, for very large networks (20,000+ nodes), we produce these masks and store their sparse COO representations sequentially. This allows us to generate bipartite small-world networks with 100,000+ nodes.