Distance based networks

Adrián Pauer

Task 1

In Task 1, we reproduced the method proposed by Keiser-Hilbert paper and implemented the corresponding adjacency matrix construction. The implementation utilized a single loop. In Task 1.2, we obtained the resulting networks and their adjacency matrices, which are presented in Figure 1.

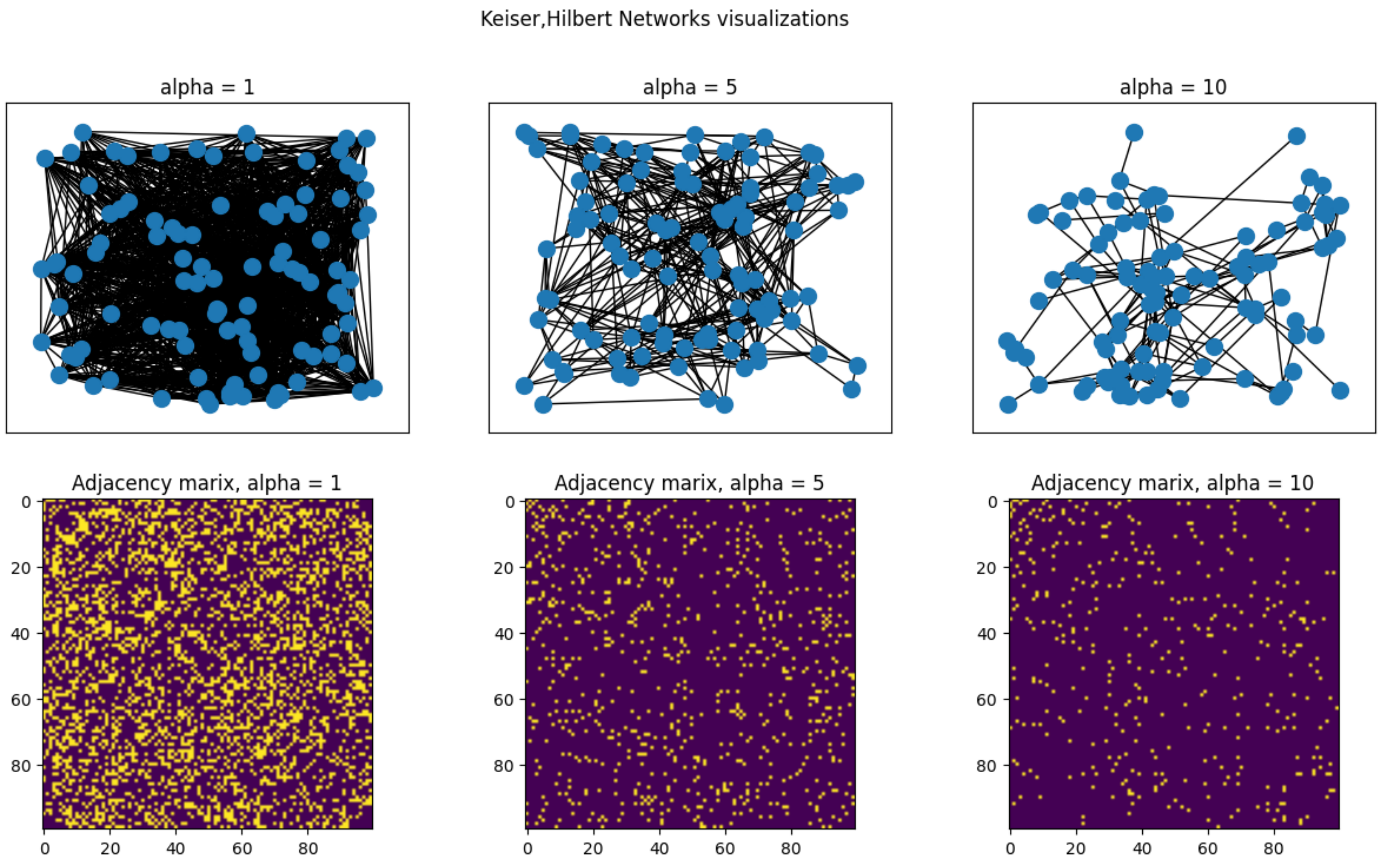

Figure 1:

From Figure 1, it can be observed that a large number of connections between nodes are formed for small values of the parameter α. As α increases, the resulting network becomes progressively more sparse.

In Task 1.3, we performed a parameter scan of α over the interval $[0.1, 100]$ using 100 sampling points. The resulting graph is shown in Figure 2.

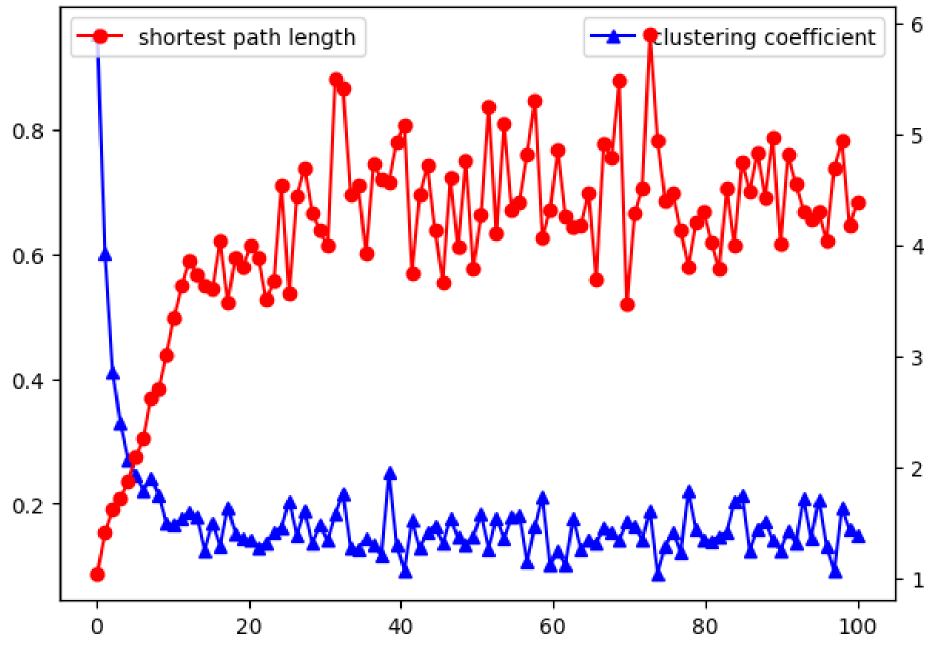

Figure 2: Parameter scan for cl. coefficient and sh. path for Keiser,Hilbert network

Task 1.4: A small-world network is obtained for small values of the parameter α. In this regime, the network exhibits a high clustering coefficient and a small average shortest path length. However, as shown in the results, for example at $\alpha = 0.1$, the clustering coefficient is close to 1 and the average shortest path length is approximately 1. These values are characteristic of an almost fully connected network. Therefore, for approximating a real-world network, a more appropriate choice of α is at the point where the two curves intersect.

Task 2

In Task 2, we implemented an improved adjacency matrix construction, as well as an optimized, faster version of the algorithm.

Task 2.1 : After implementing the improved version, we measured the execution times for two different values of N:

N	Time (seconds)
100	0.1935
1000	1575.7494

Table 1: Execution time for different values of N

From table 1 it is clear that implementation is really time consuming. After executing the algorithm, we obtained the following visualization of networks:

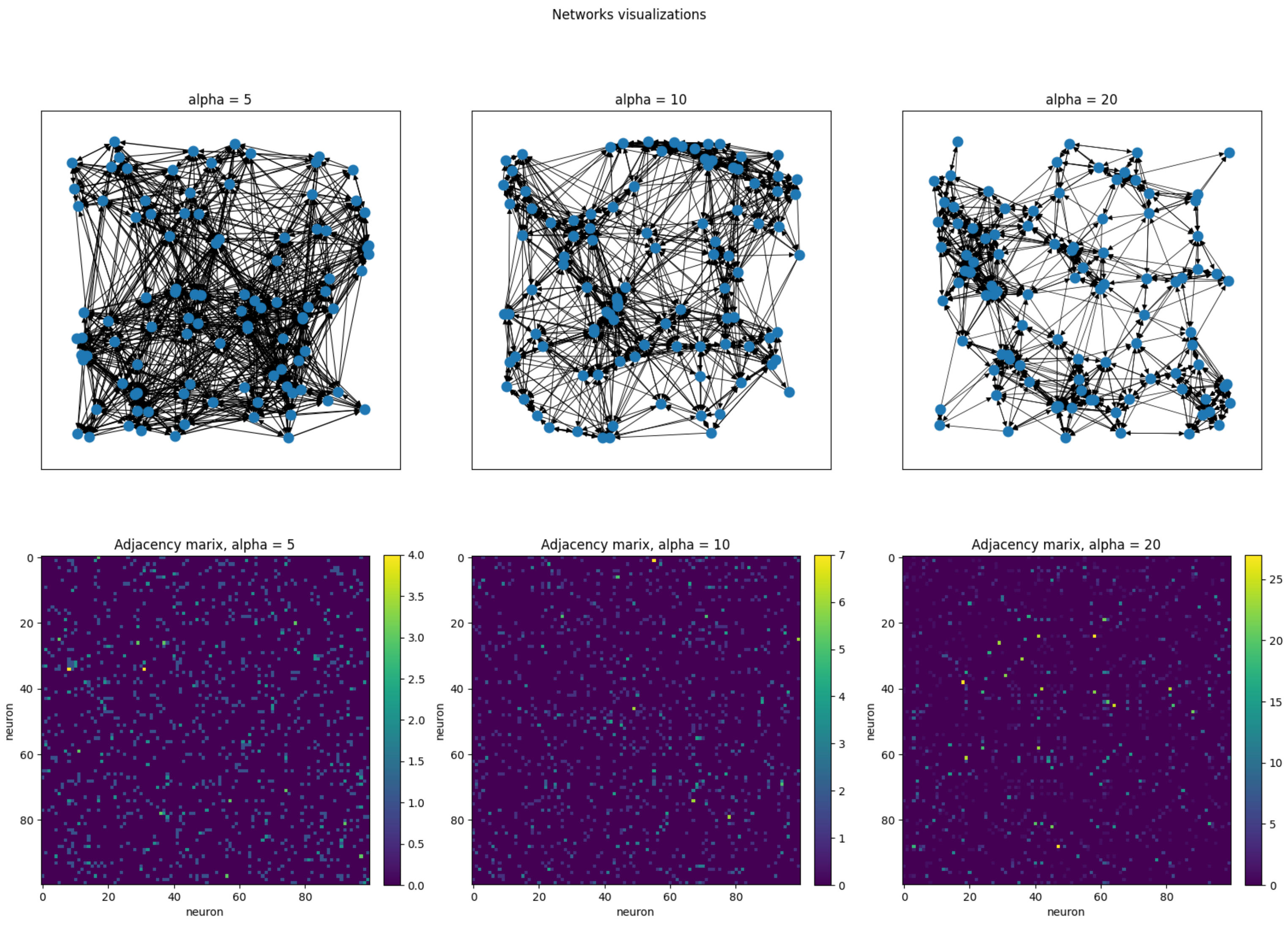

Figure 3: Improved algorithm networks visualization

In the original version, we used a basic loop while constructing the network. The idea behind the faster implementation was based on the observation that the required number of distinct edges needed to achieve the target density is known in advance. However, sampling edges without replacement does not assign weights to the edges. Therefore, to mimic the behaviour of the original algorithm, we sampled the edges $N(N-1)\rho$ with replacement and subsequently applied a loop to reach the desired density.

In addition, we set an upper limit on the edge weight equal to its expected value. This approach resulted in a significant improvement in computational performance. We used this version only in task 2.3 for fast computation of network statistics.

Task 2.3:

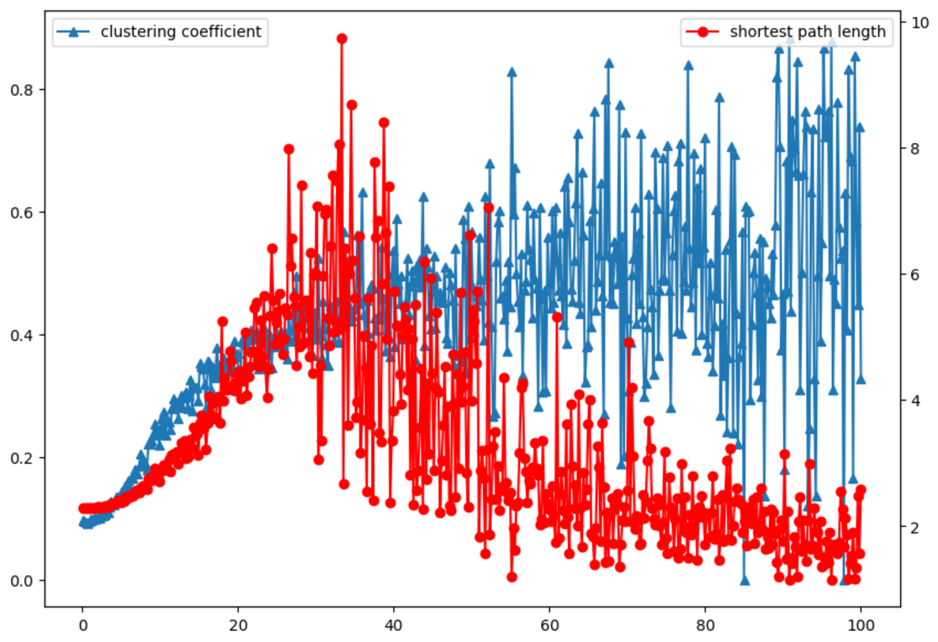

Figure 4: Parameter scan for cl. coefficient and sh. path for improved network algorithm

It can be observed that Figure 4 differs from the results obtained in the previous task. In this case, for small-world network, the clustering coefficient and the average shortest path length correspond to small and large values of the parameter α. Both curves have oscillatory behaviour.
Task 2.4:

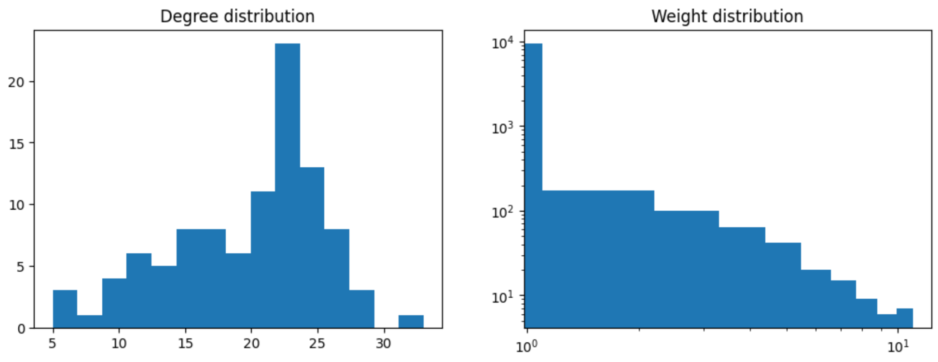

Figure 5: Degree and weight distribution for improved network

For degree distribution we can say its quite unimodal and most of nodes have degree around ten. Degree distribution is definitely not scale-free. For weight distribution most of connections have small weight and very few connections have big weight, indicating a heavy-tailed distribution. This suggests that behaviour of network is dominated by a small subset of strong connections.

Task 3

We displayed mouse cortex network on figure 6

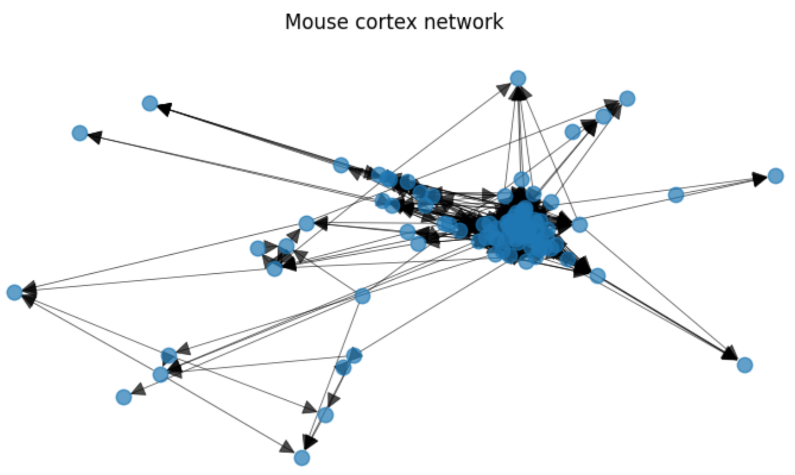

Figure 6: Mouse network visualization

Figure 6 shows that network has couple of isolated vertices and group of nodes which form connected component. We manually compared parameter scans from both algorithms and statistics obtained from mouse cortex network and choose α values which produced most similar network. We used $\alpha = 4.13636364$ for keiser, hilbert algorithm and $\alpha = 10$ for improved algorithm. We computed the relevant network statistics and compared the models in the following table and visualization

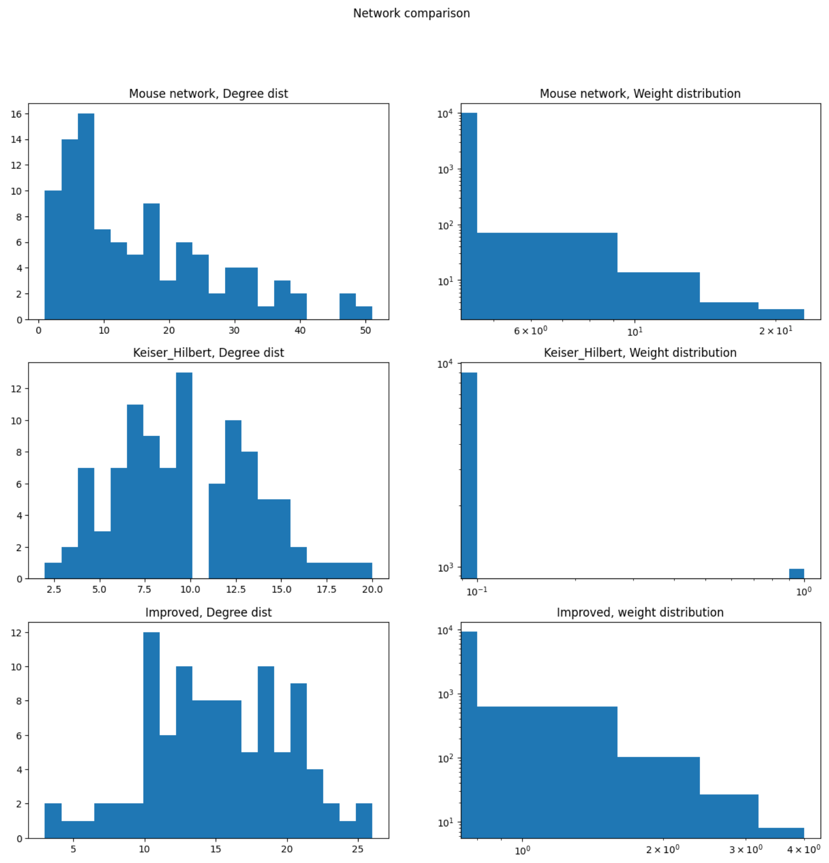

Figure 7: Comparison of degree and weight distribution of networks

Network	Clustering Coefficient	Avg. Shortest Path Length
Mouse	0.227850	2.508281
Keiser	0.145286	2.287879
Improved	0.235162	2.838485

Table 2: Comparison of network structural properties

From Table 2, we observe that the average shortest path length is quite similar across all three networks; therefore, both models perform well with respect to this metric. On the other hand, the clustering coefficient is lower for keiser-hilbert network compared to the mouse network. We also experimented with other values of the parameter α, but this typically resulted in a higher average shortest path length.

From figure 7 it is clear, that keiser-hilbert algorithm does not match weight distribution because it produces unweighted network. For improved algorithm we obtained similar distribution to mouse network but with lower weights. Degree distribution does not match in any of two models. In mouse network we have power law distribution, so to match it we need to perform some preferential attachment.

1 Preferential attachment

We tried to implement also economic preferential attachment. The algorithm was time-consuming so we decided not to do parameter scan. For parameters $N = 100, \alpha = 3, \gamma = 0.2$ we obtained above results.

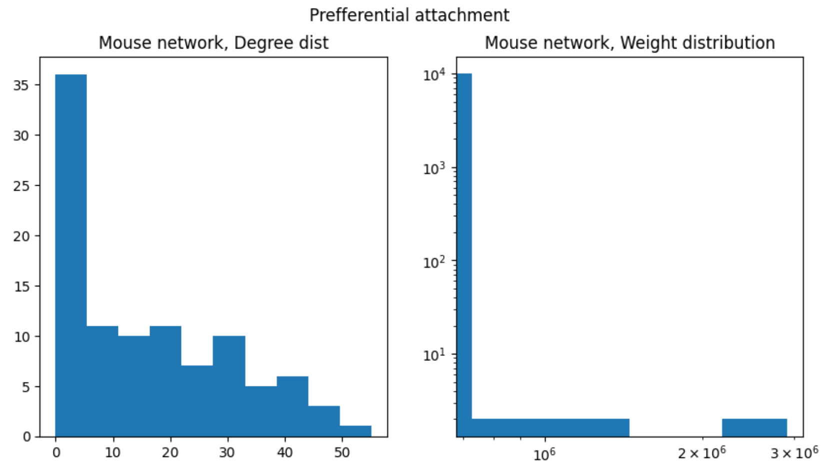

Figure 8: Degree and weight distribution from preferential attachment algorithm

Network resulted in scale free degree-distribution, which is similar to muse cortex network. At the other hand, we get for some edges really big weight which does not correspond reasonably to mouse network. The average shortest path is 2.16, and the clustering coefficient is 0.162 but after performing some parameter scan we can probably match mouse network.