Assignment 3

Community Detection
Submitted By: Nirmalya Gayen, Sr. No. 19464

Q1.

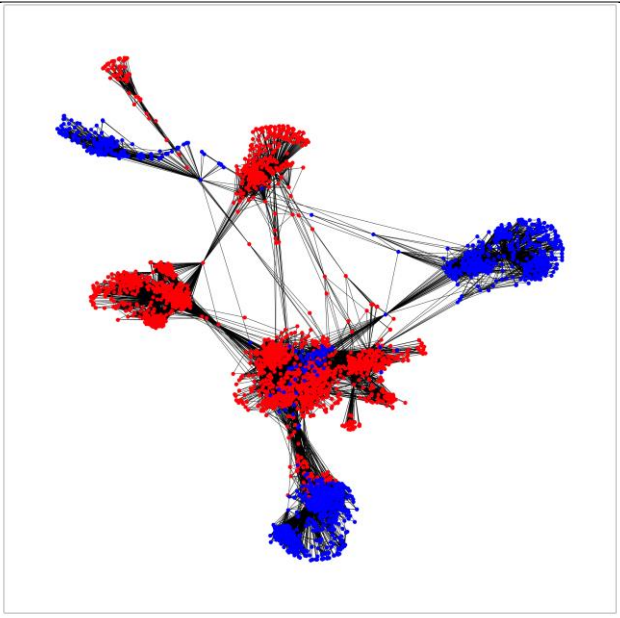
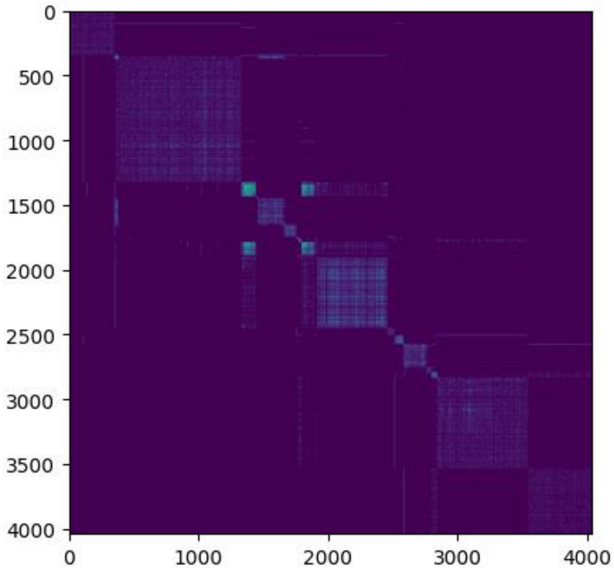
Steps:

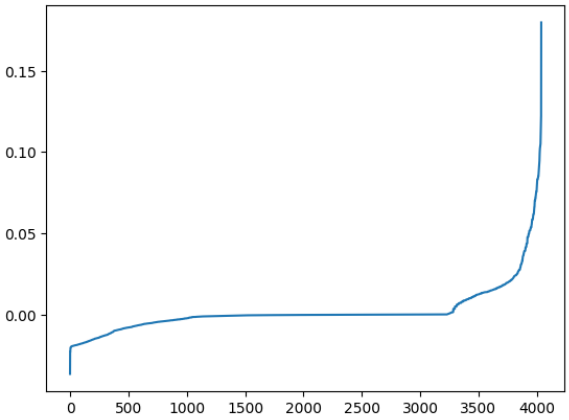
- **Map Vertex to Index**: For a given set of edges, I find the map of vertices -> index, as sometimes some nodes in the middle can be missing in the edge list, and sometimes the starting vertex might not be zero.
- **Adjacency Matrix**: Next, I find the adjacency matrix.
- **Disconnected Components**: I find the disconnected components in the graph. As they can easily be founded and are separate components. Next, I run "Spectral Decomposition" on all the disconnected components.
  - Fidelity Vector: Find Fidelity Vector for the component using the Normalized Laplacian matrix.
  - Stopping Criterion: This section is discussed later.
  - Dividing the Nodes: Based on the sign of the corresponding node, we separate the nodes into two separate clusters. Assign the least node number as the cluster number.
- Return (Fidelity Vector, Adjacency Matrix and Graph Partition)

Results:

FB: (facebook_combined.txt)

- Connected Graph, 4039 nodes
- **Clusters**: 2 clusters created

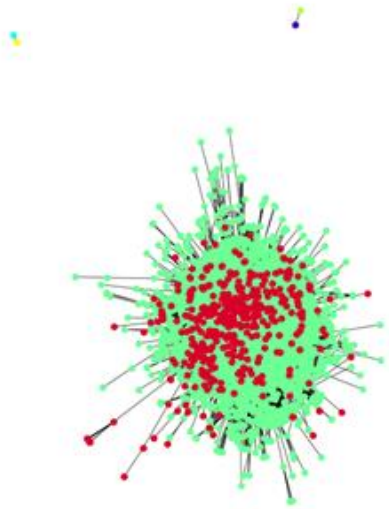
	
Graph of 2 cluster	Sorted adjacency matrix based on clusters



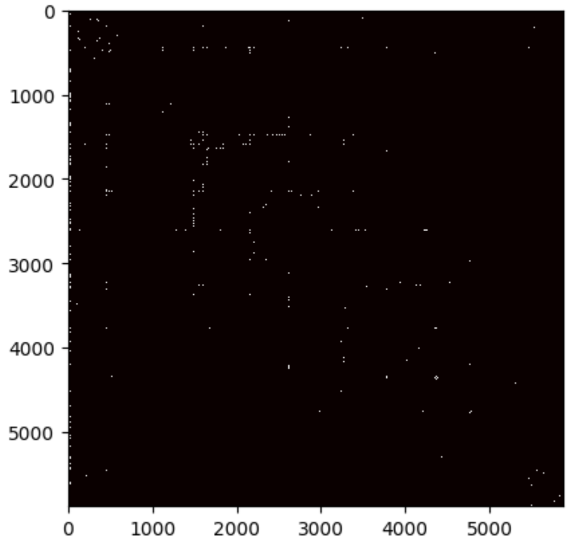
Sorted Fiedler Vector

BTC: (soc-sign-bitcoinotc.csv)

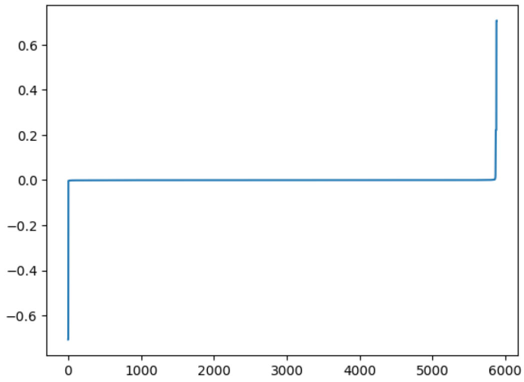
- 4 disconnected components with node size [5875, 2, 2, 2]
- **Clusters**: 8 clusters created (2 on the main graph)
- **Time**: 1min, 33sec



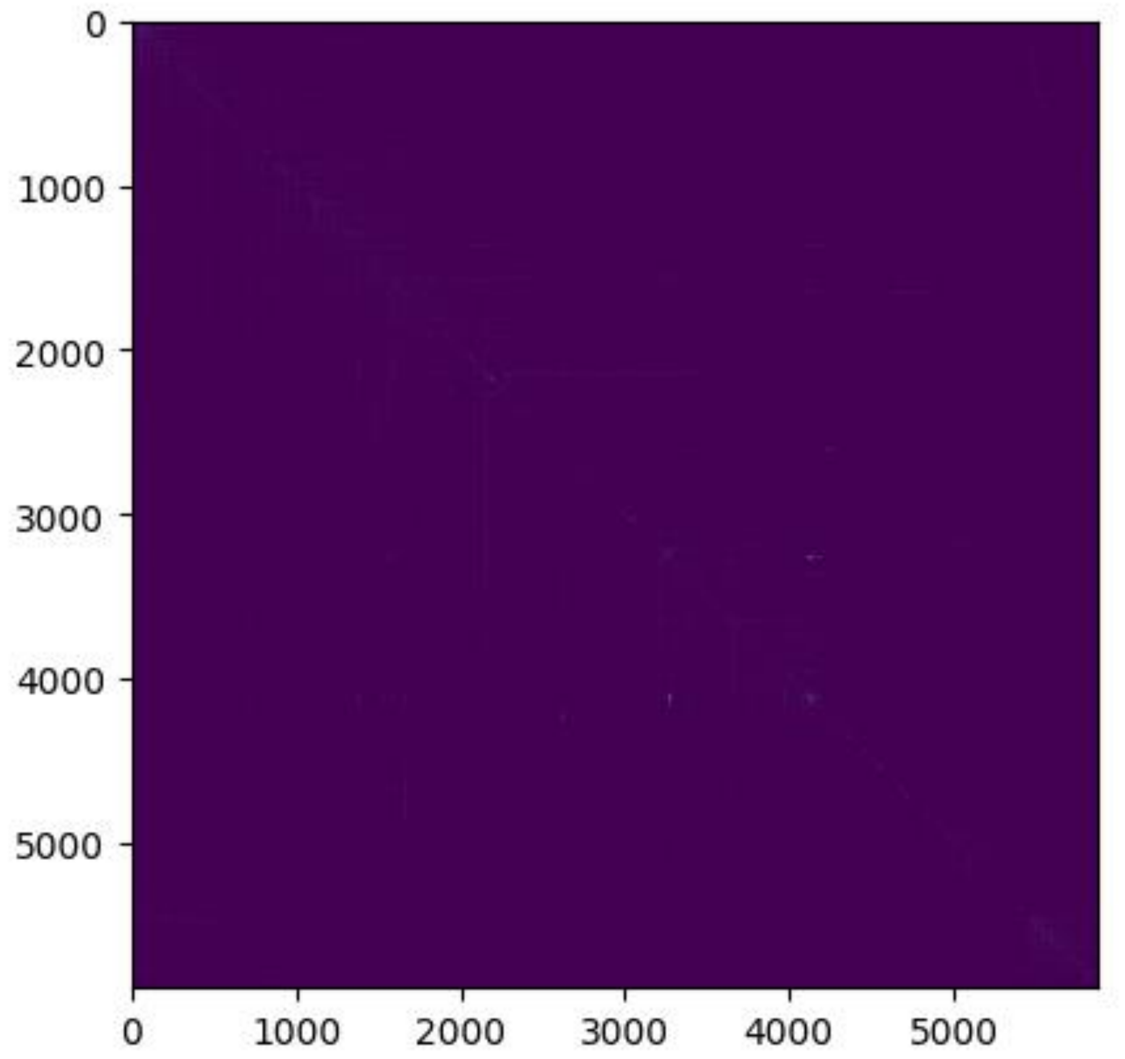
Graph of 2 main clusters



Sorted adjacency matrix based on clusters



Sorted Fiedler Vector



Sorted adjacency matrix based on clusters

Q2

Stopping Criterion:

There are 4 stopping criterions:

1. In **spectralDecomp_OneIter**
    a. **Sum of absolute(FiedlerVector) for two partitions**: Let the Fiedler Vector values of the +ve and -ve components are F^{+} and F^{-} then if $sum(F^{+}) - sum(abs(F^{-}) > 0.05$ stop the partition. As the components will be of very different sizes if not.
    b. **Number of Nodes < 2**: Before starting if number of nodes is less than 2 we stop it
    c. **Partition size zero**: If any of the partition (+ve or -ve) has no nodes stop the partition process.
2. In **spectralDecomposition**
    a. **Number of Returned Partitions**: if the number of returned partitions is zero stop and return graph_partition

Q3

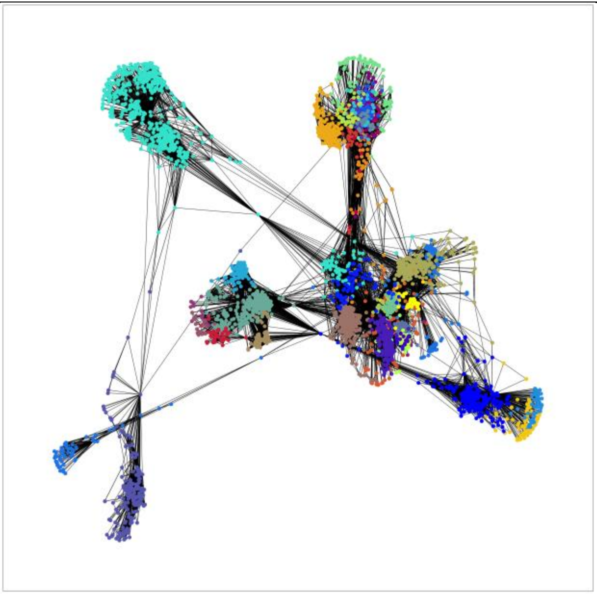
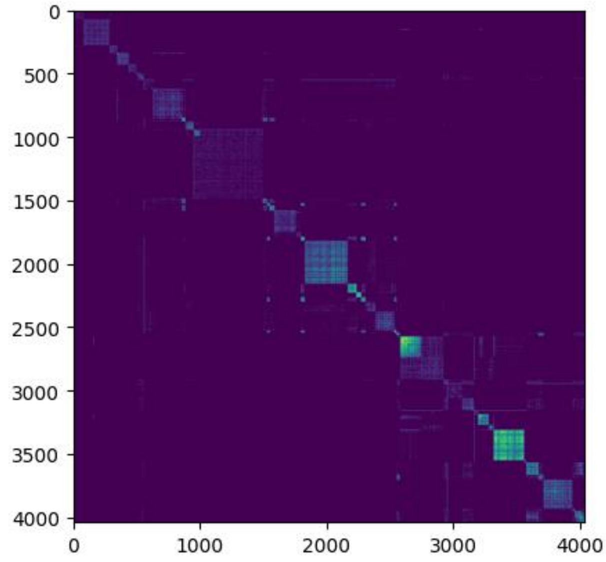
Steps:

- **Call spectralDecomp_OneIter**: Get the new components.
  - Stop if no new component is found and return the graph_partition
- **Create nodes_connectivity_list**: Create the new set of edge lists for the disconnected components.
- **Recurse**: Call itself on the new set of connectivity list.
  - **Map**: Keep a map of new component nodes -> old nodes index
  - **Merge**: Merge the returned graph_partition
- **Return**: Return the graph_partition.

Result:

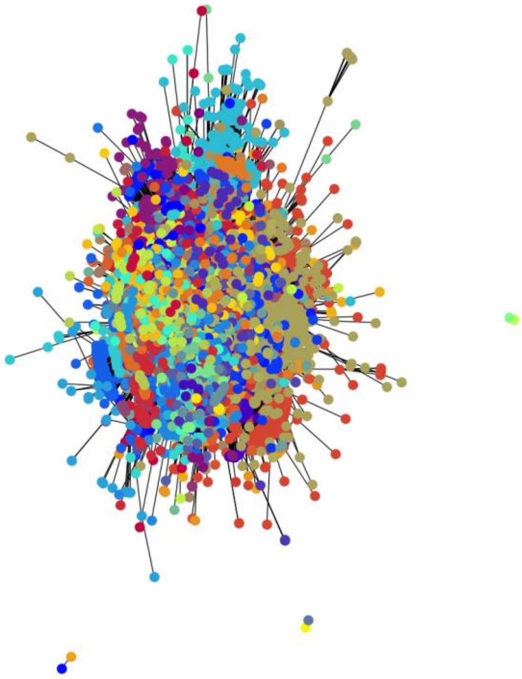
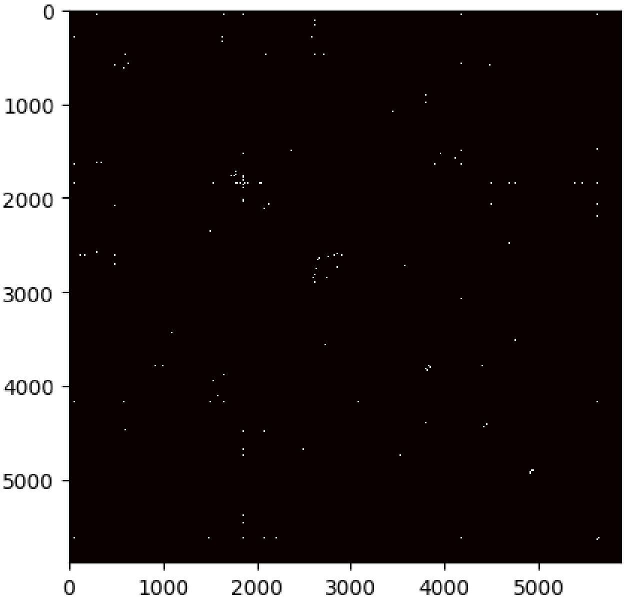
FB: (facebook_combined.txt)

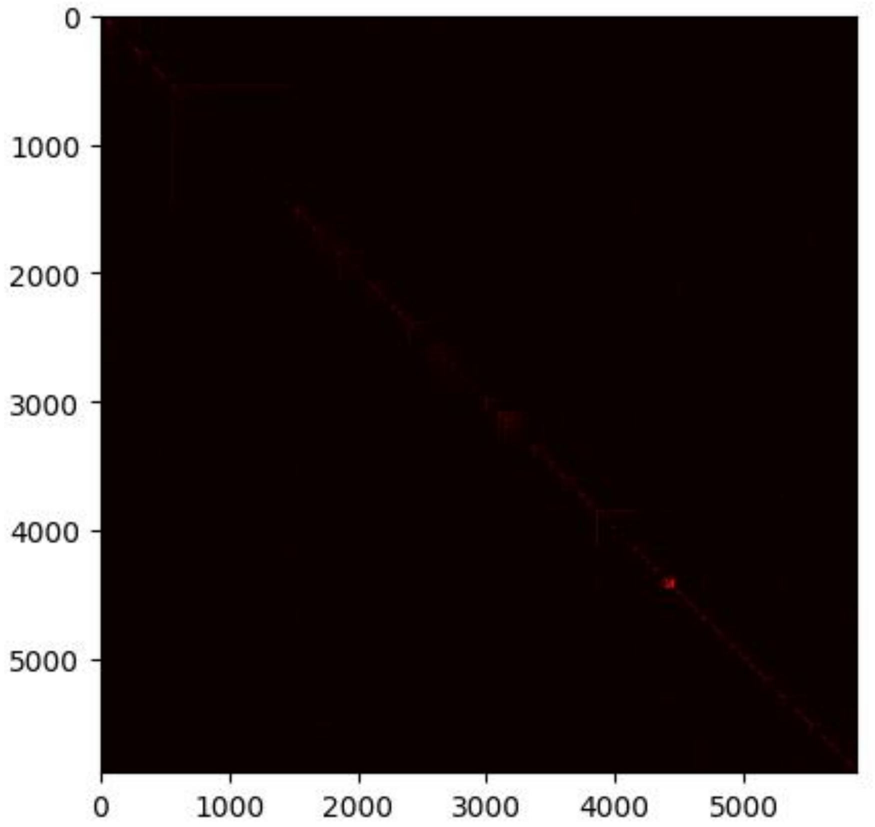
- Connected Graph, 4039 nodes
- **Cluster**: 40 clusters created
- **Time**: 1min, 59sec

	
Graph of 40 clusters FB data	Sorted adjacency matrix based on clusters (40 clusters) FB data

BTC: (soc-sign-bitcoinotc.csv)

- 4 disconnected components with node size [5875, 2, 2, 2]
- **Clusters**: 79 clusters created
- **Time**: 8min, 5.2sec

	
Graph of 79 clusters BTC data	Sorted adjacency matrix based on clusters (79 clusters) BTC data



Sorted adjacency matrix based on clusters (79 clusters) BTC data

Q4

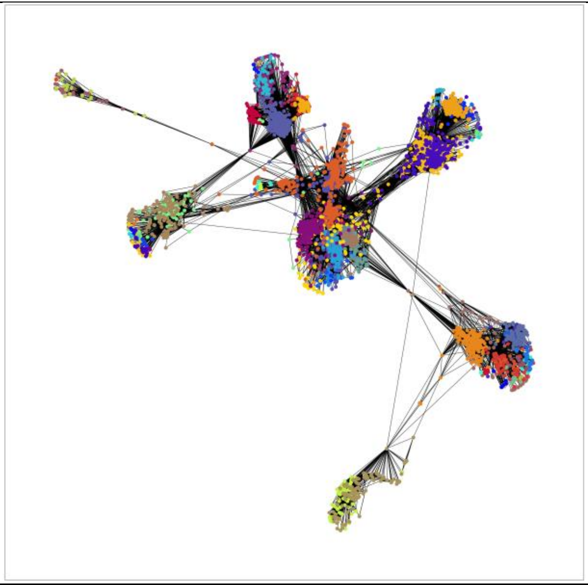
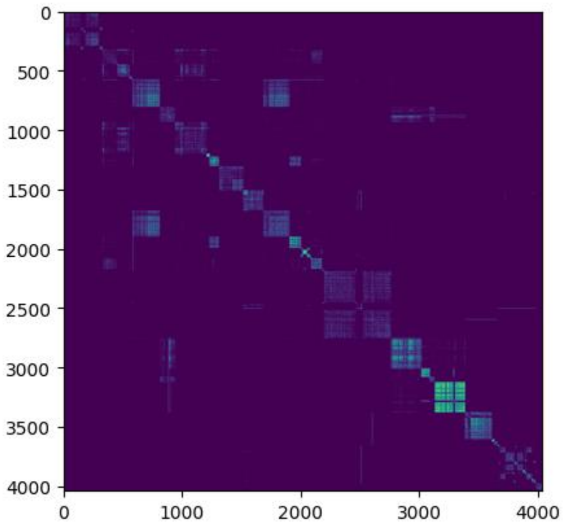
Louvain:

Steps:

- **Calculate Q Matrix**: Calculate Q matrix prior (it reduces time), else it also can be calculated on the go. (both has some problem as Q matrix in O(n^2) it takes almost 264MB RAM for BTC)
- For each i in set of Nodes:
    - For each c in clusters:
        - Calculate ∇Q
    - For max ∇Q we put the i -> c
- Repeat until we sum of chosen ∇Q is same.

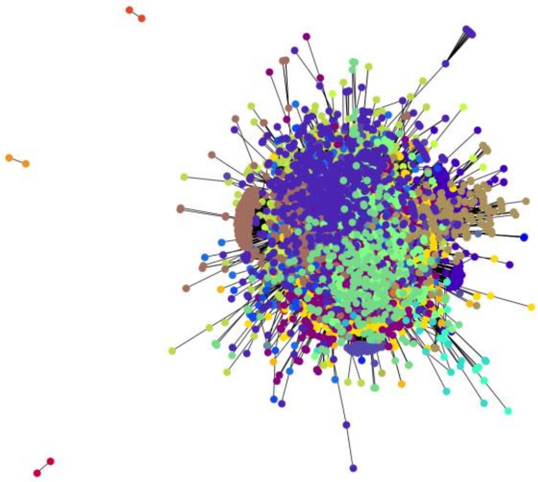
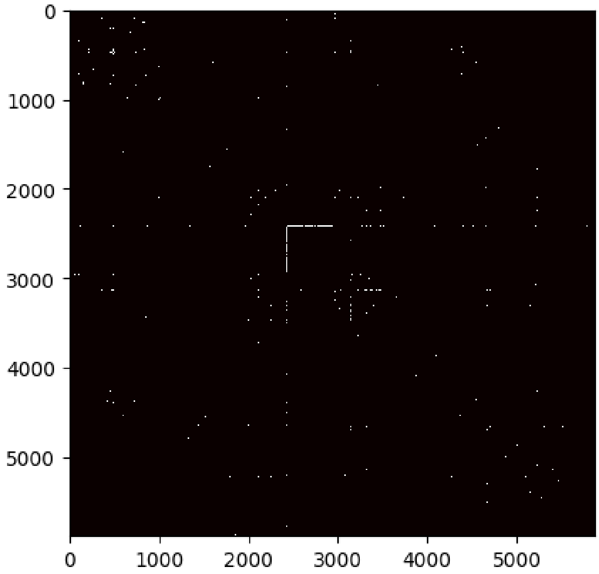
FB: (facebook_combined.txt)

- Connected Graph, 4039 nodes
- **Clusters**: 78 clusters created
- **Time**: 3min

	
Graph of 78 clusters FB data Louvain algorithm	Sorted adjacency matrix based on clusters (78 clusters) FB data

BTC: (soc-sign-bitcoinotc.csv)

- Graph with 5881 nodes
- **Clusters**: 31 clusters created
- **Time**: 33mins

	
Graph of 31 clusters BTC data	Sorted adjacency matrix based on clusters (31 clusters) BTC data

Q5

1. **Using Adjacency matrix**: Using the Sorted adjacency matrix based on the clusters we can see the **density of the cluster connectedness**. The points represent edge so density will show how good they are connected in between them. And they also should not have significant amount of connection outside.
2. **Eigenvalue of Fidelity Vector**: If the eigen value of the Fidelity vector is too low, that means the cluster is good.

These are two significant methods to recognize best decomposition.

Q6

	Spectral decomposition	**Louvain algorithm**
facebook_combined.txt	1 min, 59 sec	3 min
soc-sign-bitcoinotc.csv	8 min, 5.2 sec	33 mins

Q7

	Spectral decomposition	**Louvain algorithm**
Better clusters	Louvain algorithm is better than Spectral decomposition.	Louvain algorithm has a better result. As it is a purely automated process. We also can see the sorted adjacency matrix
Time	We can get big initial clusters 1st in Spectral decomposition as in many cases we might need only some of the 1st clusters	But in Louvain algorithm we need many iterations to get to big clusters.