Assignment 3: Community Detection

Subhojyoti Khastagir
SR. no.: 04-04-00-10-51-21-1-19446
M. Tech. - CSE
Department of Computer Science and Automation
subhojyotik@iisc.ac.in

1 Introduction

Community detection is the method of finding out subgroups or communities who share a more stronger connection among themselves than with the other members of a whole community. The community can be of people sharing same locality, language, culture or beliefs, or it can be of plants and animals sharing common characteristics, or something altogether different like proteins sharing common structures etc.

2 Data

We are provided with two datasets, one consists of interaction data obtained from Facebook, and another of anonymous bitcoin transaction data. They are formatted as two columns of node IDs which represent an interaction. In terms of network graphs, the rows represent an edge between the two nodes in the two columns.

3 Community Detection Techniques

Let us take a look at two methods of detecting communities in a graph. Detecting communities is a very complex problem and hence we mostly use approximation algorithms like Karger's Algorithm for finding minimum cut sets, Spectral Decomposition for top down partitioning of nodes in a graph and Louvain Algorithm for finding the dendogram structure of a given graph. Here we will look at Spectral Decomposition technique and Louvain Algorithm.

3.1 Spectral Decomposition Technique

In the spectral Decomposition technique, we use the Laplacian matrix to find the value of a cut and we minimize this cut value. The Laplacian is defined as the difference of degree matrix and the adjacency matrix.

Let (V, E) be the given graph. Let (V_1, V_2) represent a partition of the vertex set. We assign values $+1$ to nodes in the set V_1 and -1 to nodes in the set V_2.

$$L = G - A$$

where L represents the Laplacian matrix, G represents the degree matrix and A the adjacency matrix.

The size of the cut can be given by

$$\text{cut}(V_1, V_2) = \frac{s^T L s}{4}$$

We have to solve this minimization problem, which is NP hard. Hence we assign arbitrary real values (x) instead of ± 1 to the nodes in V_1 and V_2. This modification along with some other constraints like $\sum x = 0$, etc and the use of lagrangian multipliers to solve the minimization, reduces the minimization problem to the eigenvector problem

$$Lx = \lambda x$$

After sorting the eigenvalues, the eigen vector corresponding to the minimium eigenvalue does not satisfy the constraints hence we use the eigenvector corresponding to the second smallest eigenvalue to determine the partition. This is also known as the Fiedler vector.

3.2 Louvain Algorithm

This is another approximation algorithm which uses a concept known as modularity to determine partitions in a bottom up manner. The intuition behind modularity is very simple. It is a value between -0.5 to +1 which indicates that a community is well connected among themselves and less connected with other communities when the value is higher.

It runs in two phases. In phase 1 we try shifting individual nodes from a community to another and calculate the change in modularity occurring due to that change. If the shift increases the modularity, we keep the change otherwise we do not change the community. In phase 2, we merge the communities found in phase 1 to form a smaller graph representing the communities with single nodes.

4 Implementation

4.1 Q1. Spectral decomposition one iteration

To implement the first question, I created an adjacency matrix from the given list of edges. Using this adjacency matrix, I created the degree matrix, and used these two parameters to come up with the Laplacian matrix.

The eigenvalues as well as the eigenvectors can be found out with the help of library functions, and the corresponding Fiedler vector can be obtained.

The plots of Fiedler vector, adjacency matrix and graph partition are given in the following figures: 1 and 2: Sorted Fiedler vectors of Facebook and bitcoin datasets repectively, 3 and 5: Adjacency matrix after one iteration of spectral decomposition on Facebook and bitcoin datasets respectively, 9 and 10: Graphs after one iteration of spectral decomposition on Facebook and bitcoin datasets respectively

4.2 Q2. Repeated spectral decomposition with stopping criterion

One iteration of spectral decomposition divides the graph into two partitions, hence to obtain more communities, the function for the previous question has to be executed iteratively or recursively until the desired number of communities is obtained.

In the script, I have set a predetermined minimum size of community, beyond which a community will not be subdivided. This is an arbitrary choice, hence I have tried executing the script with different values for this hyperparameter. The results of running on the Facebook dataset are provided below.

Min size	3	4	5	10	20	50	100	200	500
Time (seconds)	5.70	5.66	5.72	4.87	4.22	3.39	3.34	3.32	3.42
Communities	42	39	38	28	15	8	7	6	4

The results of running spectral decomposition on the bitcoin dataset are given below.

Min size	3	4	5	10	20	50	100	200	500
Time (seconds)	6.36	6.31	6.48	5.58	6.42	6.07	6.25	5.64	6.02
Communities	55	55	54	45	33	21	12	9	4

4.3 Q3. Visualization of adjacency matrix

After obtaining the Fiedler vector, I sorted it and created the adjacency matrix for the obtained ordering. Libraries like matplotlib can then be used to visualize the adjacency matrix.

Figures 4 and 6 shows the sorted adjacency matrix after running spectral decomposition on Facebook and bitcoin datasets respectively

In the adjacency matrix the communities can be seen as distinct blocks.

4.4 Q4. Louvain algorithm

After one iteration of the Louvain algorithm, the following adjacency matrices were obtained:

Figures 7 and 8 show the adjacency matrix obtained after one iteration Louvain algorithm on the Facebook and bitcoin datasets respectively.

5 Analysis and inferences

5.1 Q5. Best decomposition criterion of nodes into communities

For spectral decomposition, the script will not proceed for further subdivisions if one iteration of decomposition does not yield two separate communities. In other words, if one iteration of spectral decomposition cannot yield two sub-communities, it will not send that community for further decomposition.

This might result in isolated vertices forming a community of their own in some cases, hence I have used a criterion based on the size of the subdivided community. When the division of a community yields sub-communities smaller than a predetermined number, it will not divide that community. The predetermined size of the community can be chosen arbitrarily. Some results of the relation between the minimum community size and the number of communities are provided in answer to question 2.

For the Louvain algorithm, the best decomposition criteria is given by the modularity value, which is in a way a measure of how good a community decomposition is, hence there is no explicit stopping criteria for Louvain algorithm. It can be repeated multiple times on the graph to obtain a dendogram, and from the dendogram, we can obtain a desired level of partitioning.

5.2 Q6. Running time

The running times of executing the spectral decomposition algorithm on an Intel Xeon Gold 6142 CPU (2.60GHz) with 188 GB of RAM is given in the answer to question 2. The running times for the Louvain algorithm after one iteration are given below

Dataset	Facebook	Bitcoin
Running time (seconds)	21.12	46.22
Components	703	1281

5.3 Q7. Comparison of the two algorithms

The better communities are given by the Louvain algorithm. This is because in the Louvain algorithm, the modularity is used as a decomposition criteria. To understand why this is useful, we need to first take a look at the intuition behind the modularity.

Modularity is a measure of how good a community is formed. This is measured based on the number of edges among the members of a community as well as outside the community. It basically compares the number of edges between the members of a community to the number of edges between members of a community and members outside the community. This is a good enough measure of how well a community is formed, hence intuitively it feels like this should be a better way of finding out communities.

6 Conclusion

Community detection is a difficult task, in fact an NP-hard problem, but approximation algorithms like Louvain helps us in making a good enough estimate of the communities found in a graph. Such analysis can lead to prediction or early detection of splitting up of a bigger community into smaller factions as was the case with the Zachary's Karate club.

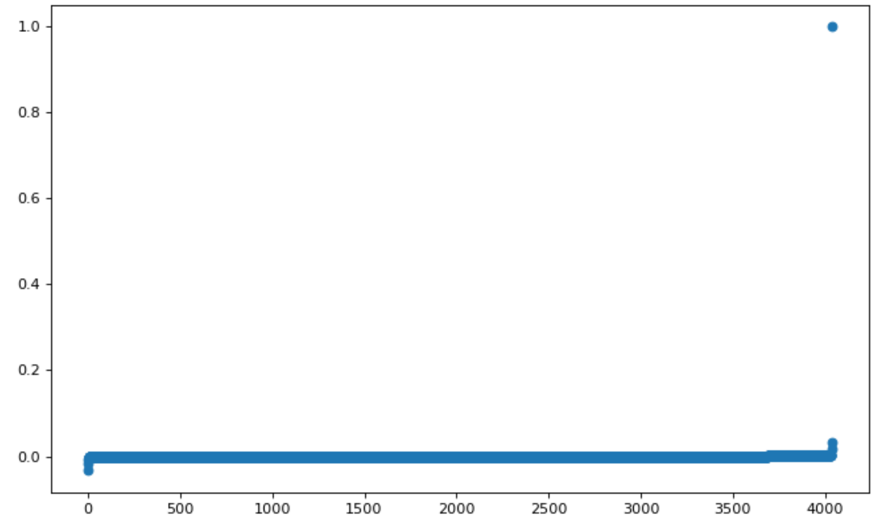

Figure 1: Sorted Fiedler vector after one iteration on Facebook dataset

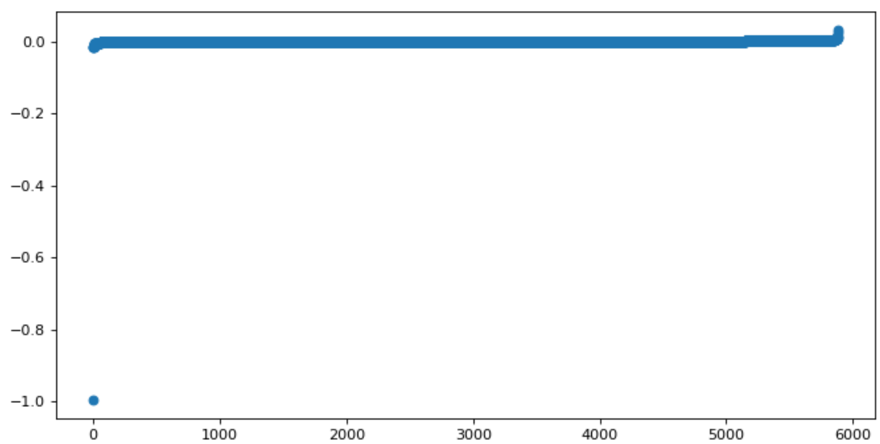

Figure 2: Sorted Fiedler vector after one iteration on bitcoin dataset

7 Figures

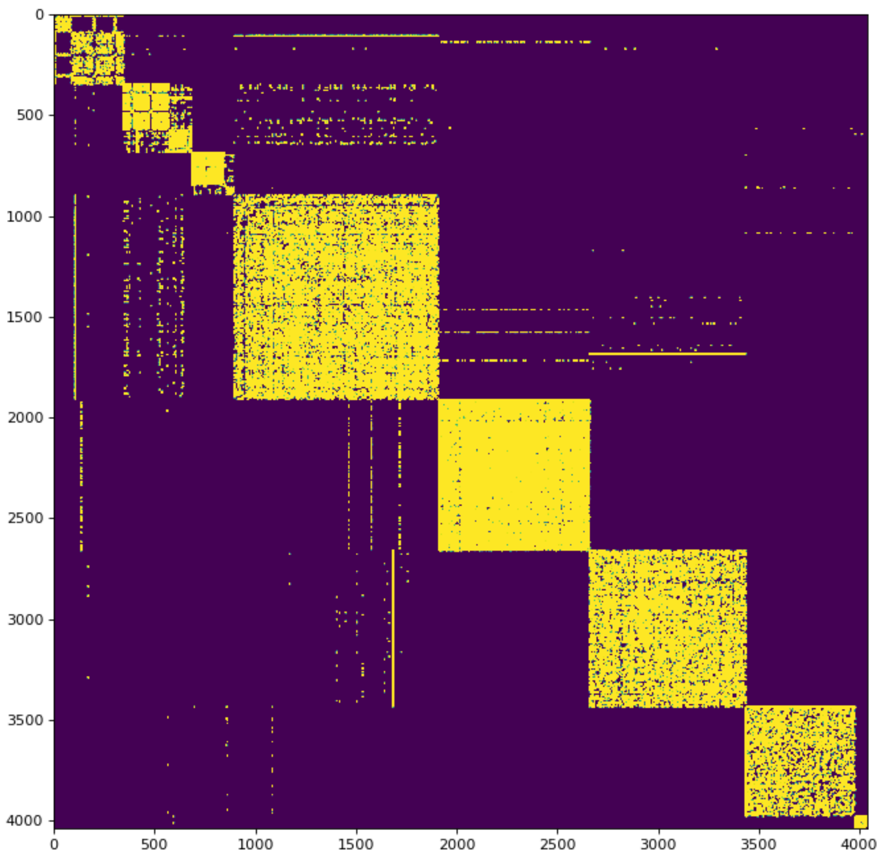

Figure 3: Adjacency matrix after one iteration of spectral decomposition on Facebook dataset

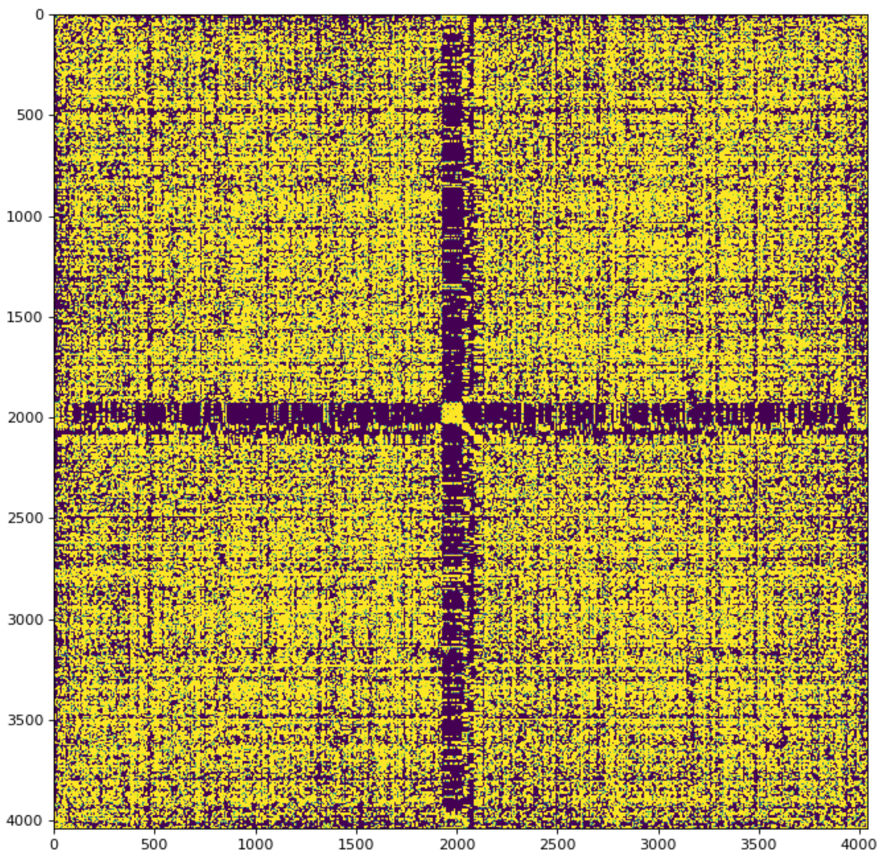

Figure 4: Adjacency matrix after spectral decomposition on Facebook dataset

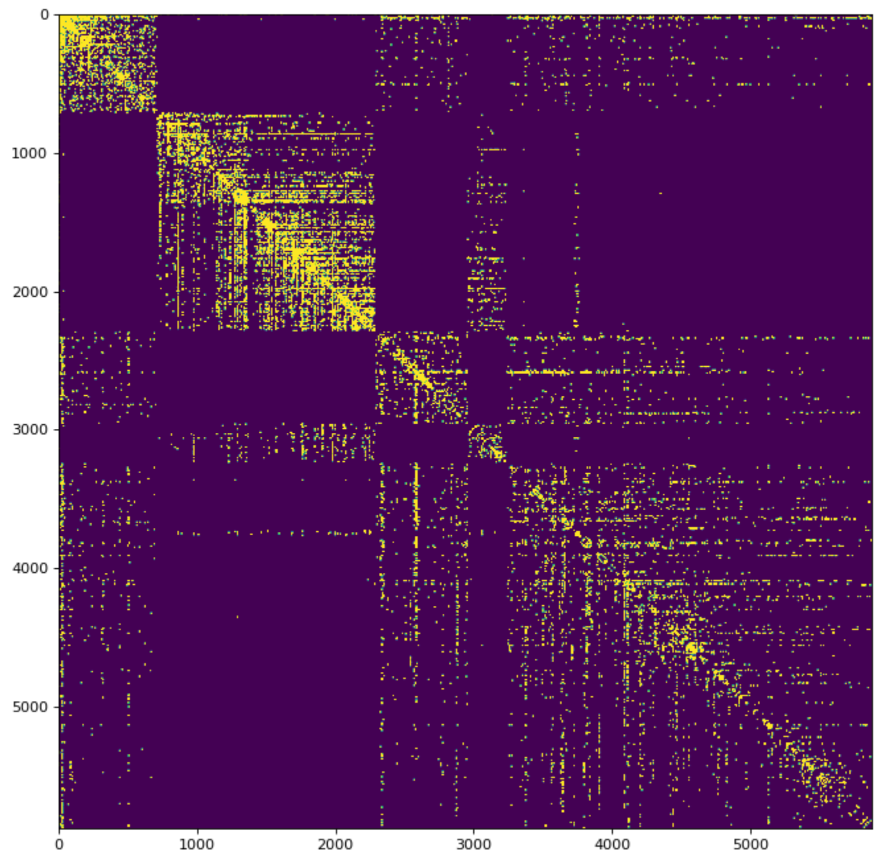

Figure 5: Adjacency matrix after one iteration of spectral decomposition on bitcoin dataset

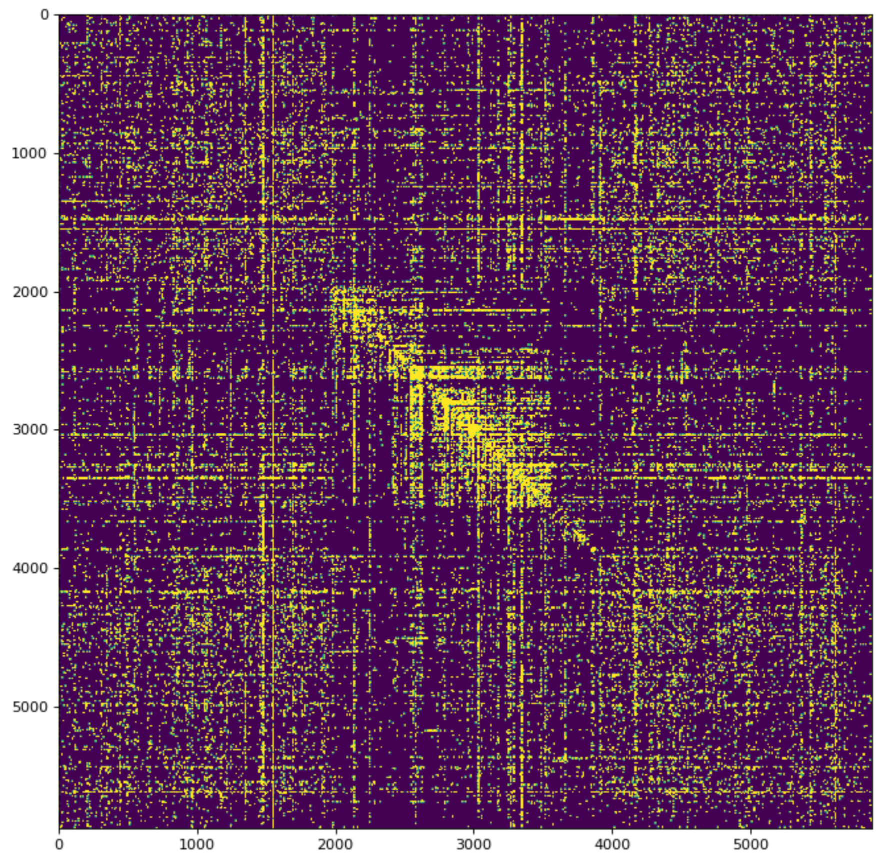

Figure 6: Adjacency matrix after spectral decomposition on bitcoin dataset

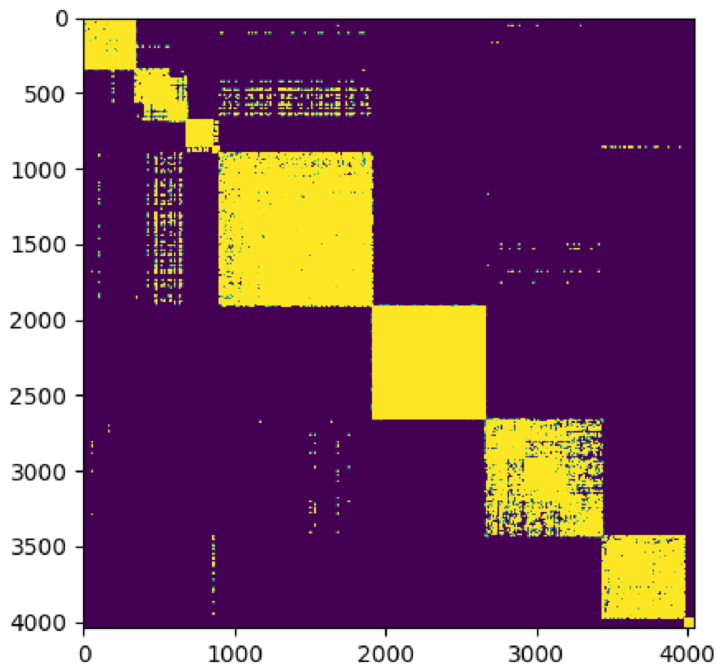

Figure 7: Adjacency matrix after one iteration of Louvain algorithm on Facebook dataset

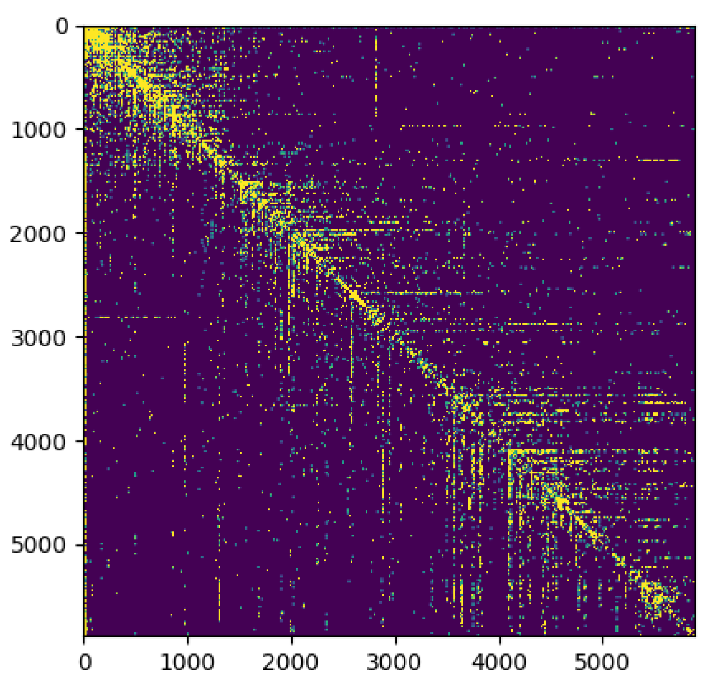

Figure 8: Adjacency matrix after one iteration of Louvain algorithm on bitcoin dataset

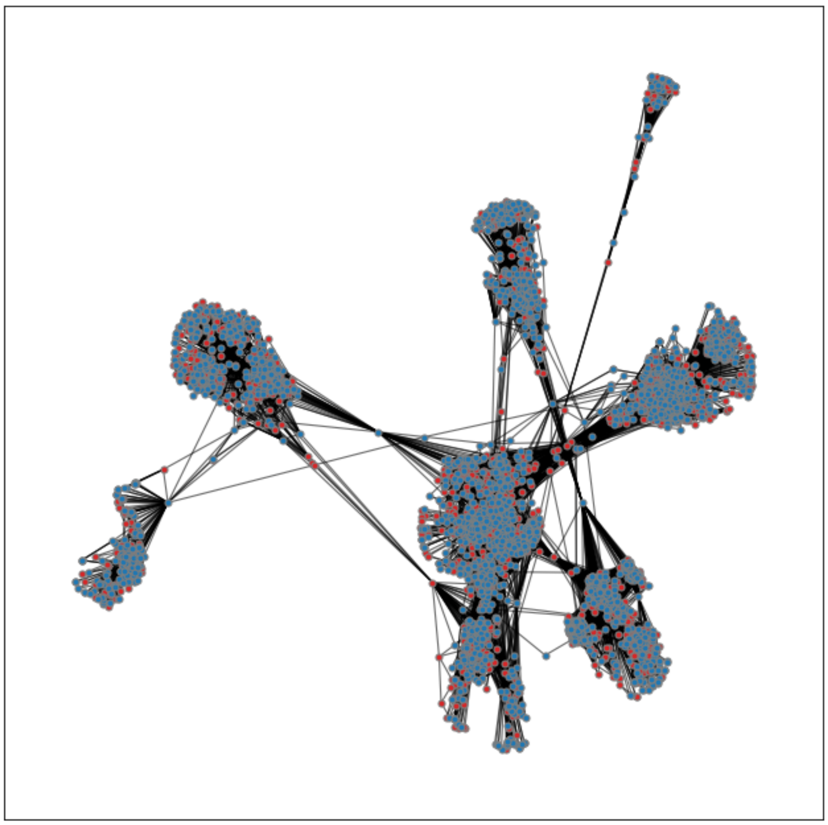

Figure 9: Graph of Facebook dataset after one iteration of spectral decomposition

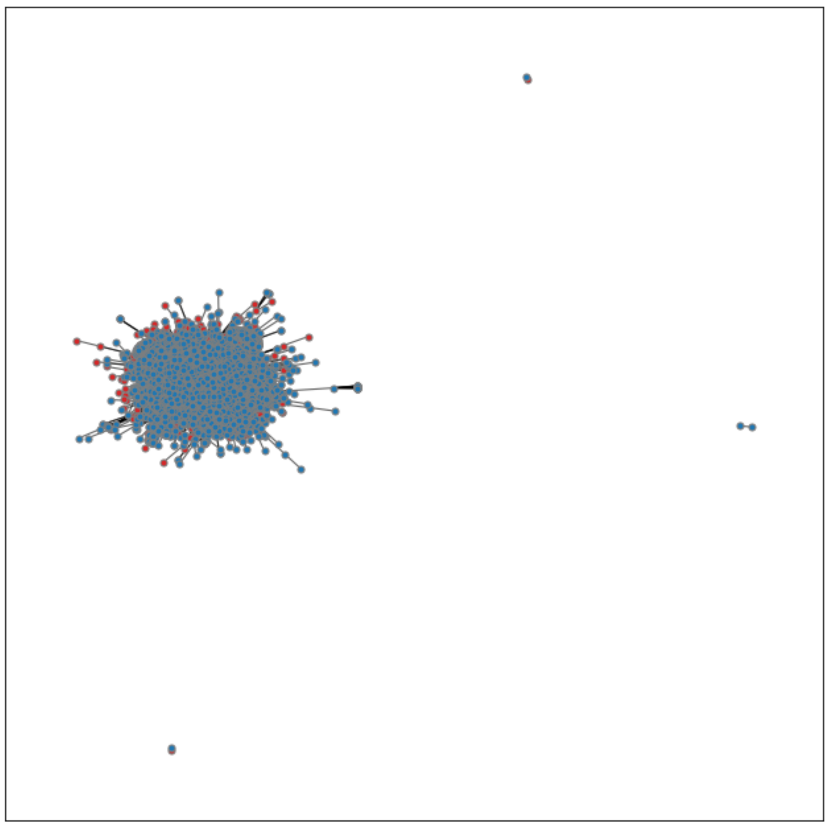

Figure 10: Graph of bitcoin dataset after one iteration of spectral decomposition