Detecting Anomalies in Graphs

D.B. Skillicorn
School of Computing
Queen's University
skill@cs.queensu.ca

January 2007
External Technical Report
ISSN-0836-0227-
2007-529

Department of Computing and Information Science
Queen's University
Kingston, Ontario, Canada K7L 3N6

Document prepared January 10, 2007

Abstract

Graph data represents relationships, connections, or affinities. Innocent relationships produce repeated, and so common, substructures in graph data. We present techniques for discovering anomalous substructures in graphs, for example small cliques, nodes with unusual neighborhoods, or small unusual subgraphs, using extensions of spectral graph techniques commonly used for clustering. Although not all anomalous structure represents terrorist or criminal activity, it is plausible that all terrorist or criminal activity creates anomalous substructure in graph data. Using our techniques, unusual regions of a graph can be selected for deeper analysis.

Detecting Anomalies in Graphs

Abstract: Graph data represents relationships, connections, or affinities. Innocent relationships produce repeated, and so common, substructures in graph data. We present techniques for discovering anomalous substructures in graphs, for example small cliques, nodes with unusual neighborhoods, or small unusual subgraphs, using extensions of spectral graph techniques commonly used for clustering. Although not all anomalous structure represents terrorist or criminal activity, it is plausible that all terrorist or criminal activity creates anomalous substructure in graph data. Using our techniques, unusual regions of a graph can be selected for deeper analysis.

1 Graph Data

Most data analysis focuses on *attributed data*, records, perhaps representing individuals or transactions, each of which contains values for a set of attributes. In contrast, graph data consists of nodes with pairwise relationships between them. Nodes may represent individuals, telephone numbers, bank accounts, locations, and other kinds of entities, while the pairwise links represent associations between them [3]. The links are naturally weighted to describe the strength of the association or affinity between any two nodes.

There are four natural settings in which graph data is important:

1. It is the connections among objects, rather than their properties, that is most significant. For example, many countries have large cities, but the construction of the interstate highway network substantially changed the social fabric of the U.S. The importance of connections among individuals is also captured by the saying that "it is not what you know, but who you know that matters".

2. It is easy to capture pairwise affinities between pairs of nodes, but much harder to capture global properties of the system of interest. For example, in an online bookstore, it is easy to see that *this* customer purchased *that* book, but much harder to see that two customers have similar taste in books.

3. Attributed data of different kinds needs to be combined in a reasonable way. For example, collaborative filtering data can be combined with demographic data about customers, and content data about books to produce a graph whose nodes are the rows of any table and whose edges connect every two occurrences of identical values that appear in different tables (that is, values that would be used in a database join).

4. Attributed data is high-dimensional in appearance, but is known to lie in a low-dimensional manifold, so direct analysis is likely to fail because of the dimensionality. Objects are connected when they lie close to each other in the high-dimensional space, so that the graph data captures the low-dimensional structure. Analysis starting from the graph representation is likely to produce better results. For example, a chemical molecule may be described by the three-dimensional location of each of its atoms, but the molecular structure does not have nearly as many degrees of freedom as this representation suggests.

Terrorism and law-enforcement datasets are primarily of the first three kinds. The purpose of surveillance, for example, is to establish *connections* between people, places, and objects. It is also

natural to want to extend observed pairwise connections between, say, individuals to more global relationships such as command and control structure. Attributes collected *about* people can also create connections *between* them; for example, a dataset that includes addresses can be used to create links between individuals who may have stayed at the same address, but at different times.

Several kinds of analysis of graph data have been developed. Most try to extract global properties or structure from the local properties and structure implied by each link connecting two end-point nodes. The first, and oldest, is a family of techniques called social network analysis that look at graphs as representing primarily relationships between people. Hence these graphs reflect the flow of power, influence, or information, and can be analyzed to find which people are facilitators, blockers or accumulators of such flows. In terrorism settings, such analysis techniques can be used to find key personnel or key communicators, but with the important assumption that the graph contains only (or primarily) members of a terrorist group [5, 7].

A second kind of analysis is to cluster the nodes, based on the structure of the graph rather than similarities among the attributes of the nodes. Graph clustering is based on finding good cuts that separate the graph into subgraphs that are well-connected internally, sparsely connected to other subgraphs, and of reasonable size [11].

A third kind of analysis is to search for occurrences of particular subgraph structures within graph data, a kind of graph-based information retrieval. This approach has had some success in investigation of money laundering and insurance fraud, where particular techniques correspond to particular configurations of businesses, bank account, telephone numbers and people [6].

A fourth kind of analysis, comparatively recent, is to predict new edges in a graph, based on its existing structure. The idea is to find pairs of unconnected nodes that are somehow close or similar in a global sense although, obviously, they are not close in the local sense. Edge prediction is useful in a number of settings. In a dynamic environment, for example coauthorship, it can predict edges that are likely to appear soon [8]. In collaborative filtering, recommending a product to a user is exactly the prediction of a new edge in a graph connecting users to products [4]. In terrorism settings, applications include predicting a relationship between two people that has not been captured by surveillance, or has been actively concealed but is nevertheless real. For example, the leaders of two arms of an organization may never communicate directly with each other but may nevertheless become detectably connected because of the density of paths between them.

The contribution of this paper is to present techniques that enable a fifth kind of analysis, the discovery of anomalous subgraphs within a large graph. In a graph that captures human behavior, anything that is normal should be represented many times within the graph. Abnormal behavior of any kind is likely to result in substructures in the graph that are unusual. Not all abnormal behavior is suspicious, but all suspicious behavior should be abnormal. Hence, finding the abnormal substructures selects those parts of the graph to which further analysis should be applied. This avoids the problems associated with looking for particular substructure: it captures unusual subgraphs even if they have not been thought of, and it finds subgraphs that would have matched a known pattern except that one or more of the required edges was not captured. We illustrate these techniques applied to some necessarily small example graphs.

2 Spectral graph analysis

Spectral analysis of graphs is based on their eigenvector structure, and is a powerful way of understanding the global structure present within a graph. The natural representation of a graph is in

terms of its weighted adjacency matrix. Given a graph with n nodes, this matrix has n rows and n columns, and its ijth entry is the weight associated with the edge between node i and node j. If nodes i and j are unconnected, the ijth entry is 0. We will assume that weights are positive, and larger magnitudes correspond to greater pairwise affinities.

Many data-analysis techniques assume implicitly that data inhabit a geometric space in which proximity corresponds to similarity. In such a geometric space, the distance between two objects remains the same, no matter what other objects are present in the space, since similarity depends only on the attribute values of the objects concerned. For graph data, this is no longer true. In a graph space, the non-local affinity between two nodes depends, potentially, on *all* of the other nodes since it depends on all of the paths between them. Adding or removing a single node can change the non-local affinities for all of the other pairs of nodes. This makes graph spaces difficult to work with.

It is attractive to map a graph space into a geometric space, and then examine the graph's properties there. However, such an *embedding* is difficult to construct for two reasons. The first is that the graph space described by the adjacency matrix is inside-out with respect to the geometric space implied by the same matrix. For example, consider an unweighted adjacency matrix, where the ijth entry is 1 for nodes that are connected. Consider a row that contains a thousand 1's, and rows that contain only a few 1's that overlap with the thousand. In the implicit geometric space, the row with the thousand 1's is much further from the origin than the other rows, and may even appear as an outlier. However, in the graph space the row with the thousand 1's is central, since it connects all of the other nodes. Hence the mapping from the graph space to a geometric space must place nodes that are important in the graph sense near the origin, reflecting their importance in the geometric space.

The second difficulty is that proximity in the geometric space must reflect non-local affinity in the graph space, but in what sense? In other words, how is local affinity to be extended to non-local affinity? There are several possibilities and, of course, which one to use depends on the problem domain. The most obvious extension is the so-called Dijsktra distance, based on the shortest path between the two unconnected nodes. However, in many settings it is natural to consider unconnected nodes the same number of steps apart as more similar if they are connected by *many* short paths. It may also sometimes to be important to discount the effect of a few high-degree nodes that may distort the relationships among the other nodes [2]. For example, in a graph whose nodes are U.S. politicians and cities, adding a node for Washington may make it harder, rather than easier, to discover interesting connections among politicians based on geography. Each choice of extension of local structure to non-local structure implies a different mapping of graph space to geometric space.

Given a adjacency matrix, A, the corresponding Laplacian matrix is

$$L \;=\; D - A;$$

where D is the diagonal matrix whose iith entry is the (weighted) sum of the ith row (equivalently, column) of A. In other words, the off-diagonal entries are negations of the corresponding entries in A, and each row and column sums to zero.

The walk Laplacian, L_w, is obtained from the Laplacian by dividing each row by the diagonal entry, so that the diagonal entries are now all 1, and the off-diagonal entries are negative numbers whose magnitudes are smaller than 1. The off-diagonal entries in the walk Laplacian can be thought of as the negatives of transition probabilities for a walk in the graph.

The walk Laplacian corresponds to an embedding of the graph space in an n-dimensional geometric space, where each row corresponds to a point in the space, and Euclidean distances represent the local and non-local affinities of nodes. The extension of local affinities to non-local ones is based on both the length of paths and the number of paths connecting pairs of nodes. If the edges are regarded as wires, and the edge weights are regarded as inverse electrical resistances, the non-local affinities are the inverses of the effective resistances between each pair of nodes [10, 11]. The negation of the entries of the walk Laplacian ensures that the mapping correctly turns the structure inside-out.

The geometric space described implicitly by the walk Laplacian has dimensionality n, so it is awkward, in practice, to work in this space. Typically an eigendecomposition is applied to the walk Laplacian, which can then be truncated to leave k components that approximate the global structure well. For conventional graph analysis, it is the k eigenvectors associated with the k *smallest* eigenvalues that are retained.

The walk Laplacian has rank no greater than $n - 1$ (since the connectivity of the last node is completely determined by that of the all of the others). Therefore the smallest eigenvalues is zero and, in fact, the number of connected components is determined by the number of zero eigenvalues.

The $n \times k$ matrix that results from truncation can be used to address all four of the problems described in the introduction: clustering can be done using standard techniques in the k-dimensional space; there are deep connections between the magnitudes of the smallest eigenvalues and large-scale properties of the graph; particular subgraphs can be discovered in the style of latent semantic indexing; and edge prediction can be done by looking for the smallest (Euclidean) proximities of unconnected nodes in the k-dimensional space.

Every eigenvector associates a value with each node of the graph, and the magnitudes and signs of these values reveal much of the structure of the graph. In other words, each eigenvector is a mapping from the nodes of the graph to the real number line. A special case is the mapping associated with the $n - 1$st eigenvector. This mapping has the strong property that, if a real value within the image is chosen, the nodes mapped to smaller values and the nodes mapped to larger values both form connected subgraphs of the graph. In other words, selecting a point in the image splits the graph into two clusters. Choosing zero as the split point gives a normalized cut of the graph, so that the size of the two clusters is balanced.

The eigenvector associated with the smallest non-zero eigenvalue represents a kind of standing wave on the graph, in which the nodes at one 'end' are up (greater than zero) and the nodes at the other 'end' are down. Eigenvectors associated with slightly larger eigenvalues represent standing waves of greater complexity, in which smaller connected regions are up and down. This is not a metaphor – the reason why the matrix describing the graph is called a Laplacian is because it is a discrete form of the Laplace-Beltrami operator [1].

3 Beyond clustering – deeper analysis of graph substructures

Our approach is to carry out the eigendecomposition of the representation matrix, but not the truncation. Instead we retain and examine all $n - 1$ eigenvectors. Well-connected substructures of the graph are associated with eigenvectors whose eigenvalues have small magnitudes. Cliques or near-cliques in the graph tend to be visible as clusters described by such eigenvectors, even if they are of small size. A single small clique or near-clique is an anomalous structure, since it represents a set of objects, perhaps people, that are much more closely related than average.

We use a singular value decomposition, rather than an eigendecomposition both because implementations produce singular values in descending order, and because it justifies the following argument. Suppose that we forget that the walk Laplacian was constructed from a graph, and think of it as the matrix of attributed data. Then the eigenvector entries associated with eigenvalues of large magnitude are large for rows of the matrix that have unusual correlation with other rows. Rows that are similar to many other rows are placed so that they have positive dot products with as many other rows as possible; and rows that are dissimilar to many other rows are placed so that they have negative dot products with as many other rows as possible; and the net effect is that rows of both kinds plot close to the origin. The eigenvectors associated with large eigenvalues therefore select nodes of the graph whose neighborhoods are unusual. Such nodes are also anomalous structures in the graph, since they represent an object, perhaps a person, who has an unusual set of relationships.

Thus, if we consider the columns (eigenvectors) of the U matrix that results from SVD, we have two ways to interpret them. If we start from the right-hand end, then the eigenvectors first describe regions of the graph that are sparsely connected to each other (clusters), but as we move leftwards across the columns we find descriptions of regions that are locally smaller and smaller, with more and more extensive connections between them. Eventually, the leftmost eigenvectors describe small regions that are well-connected with each other.

If we start from the left-hand end, then the eigenvectors first describe nodes whose neighborhoods are unusual. As we move rightwards across the columns, we find descriptions of regions of increasing size whose neighborhoods are decreasingly unusual. Eventually, the rightmost eigenvector describes large regions whose neighborhoods are not unusual. These two descriptions are, of course, describing the same thing from different perspectives.

Now consider the eigenvectors in the middle of the U matrix. Such eigenvectors describe standing waves of small intensity, that is involving few nodes or with small amplitudes; and sets of nodes with moderately unusual neighborhoods. Such regions correspond to interesting, typically small, substructures in the graph, and so to regions that may be anomalous.

4 Detection techniques

Since we are working with an $n \times n$ matrix, the U matrix from an SVD, and n is typically large, it is not trivial to find ways to understand the results of the decomposition.

Each column of the matrix is a characteristic vector of a standing wave on the nodes of the graph, with positive values corresponding to 'up' in the standing wave and negative values to 'down'. Plotting a column provides some information about the sizes of the regions involved in each standing wave. Plotting the mean of the absolute value of the columns of U also provides a way to detect which columns (eigenvectors) may be interesting. The minimum points on such a plot suggest interesting eigenvectors involving only a small number of nodes and/or small vibrational 'energy'.

Another way to examine substructures in the graph is to plot the nodes using, as coordinates two (usually adjacent) eigenvectors. The positions of the points therefore reflect non-local affinity. If the edges of the graph are superimposed, then local affinity can also be represented. If non-local affinity extends local affinity smoothly, then nodes that are strongly connected should plot close together. For example, for well-behaved graphs, the plot based on the smallest and second-smallest nonzero eigenvalues often produces a reasonable planar drawing of the graph. In such a plot,

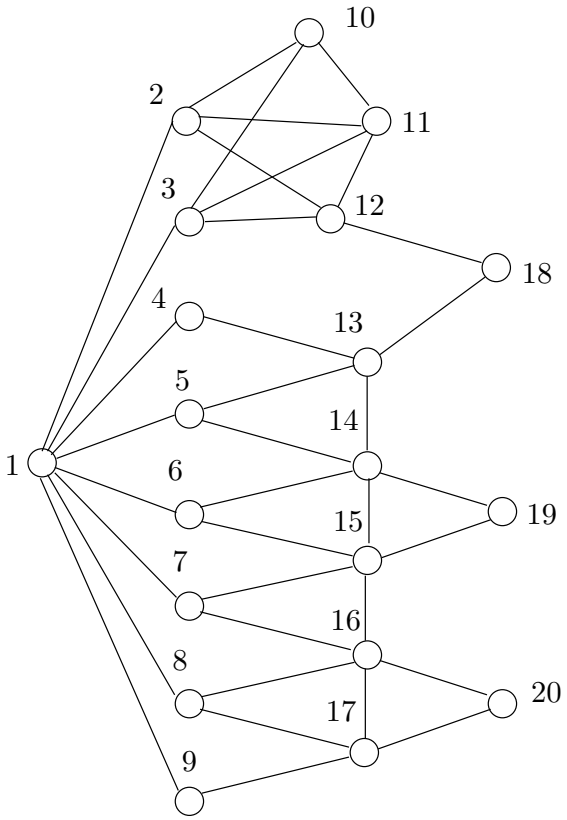

Figure 1: A small example graph

distance from the origin is also significant, so nodes that plot far from the origin are typically the more interesting. The outer product matrices for each column/row can also reveal some of the substructure of the graph.

5 A small example

To make this concrete, consider the small graph shown in Figure 1 with unit edge weights. The plot of the eigenvectors corresponding to the two *smallest* non-zero eigenvalues are shown in Figure 2. Note that eigenvector 19 separates the almost clique involving nodes 2, 3, 10, 11, 12, and 18 from the rest of the graph. This ability to detect an interesting substructure is exactly the property that we want for terrorism datasets. Eigenvector 18 captures the extremes of the vertical structure: the nodes at the top and the bottom of the graph as it is drawn are at one end of the 18th dimension, while the more central nodes are at the other.

The plot of the eigenvectors corresponding to the two *largest* eigenvalues are shown in Figure 3. These plots describe local properties of individual nodes. For example, dimension 1 separates high-degree nodes, such as node 1, from low degree nodes, such as nodes 4 and 9.

We can see the standing waves in the example graph by plotting the columns of the eigendecomposition matrix. Figure 4 shows this for the eigenvectors 19 and 18.

The standing waves can be seen even more clearly by labelling the nodes of the original graph.

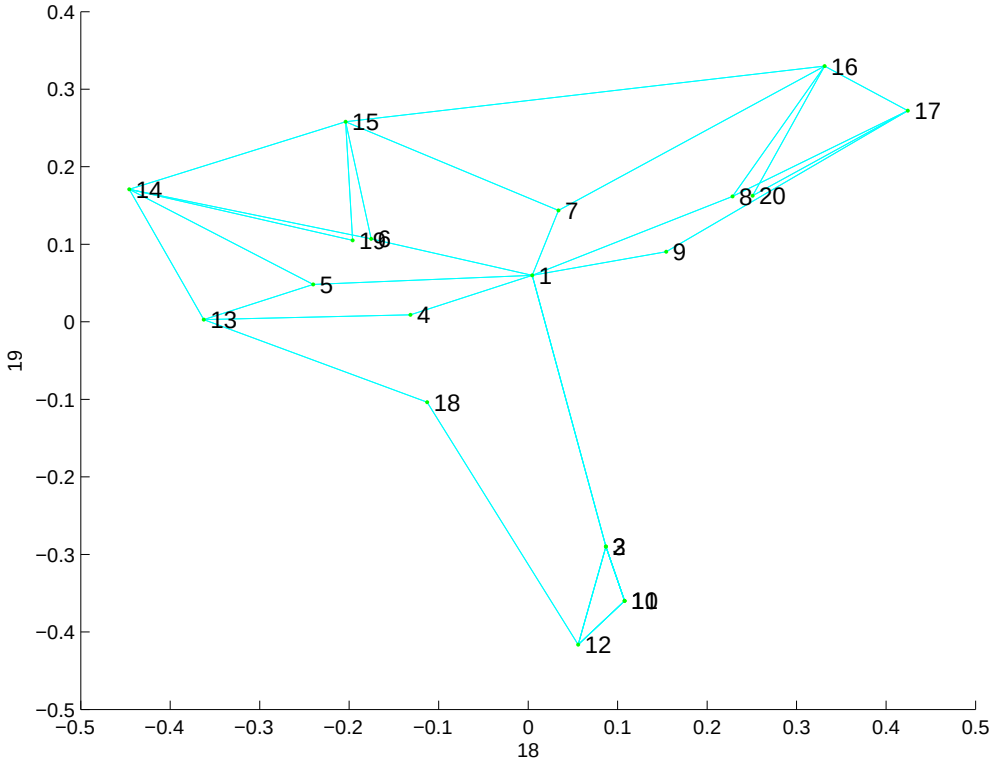

Figure 2: Graph nodes plotted according to the eigenvectors corresponding to the two smallest eigenvalues; each eigenvector classifies graph nodes into two classes, so each quadrant represents one of four classes based on the two attributes

Figure 5 shows the graph labelled according to eigenvectors 19 and 18. These figures show how the eigenvectors corresponding to small eigenvalues partition the graph into large regions with small cuts separating them.

The eigenvectors from the other end of the spectrum are shown in Figure 6. These figures show how the eigenvectors corresponding to large eigenvalues partition the graph into large unconnected regions with cuts through many edges between them. Note especially how eigenvector 1 partitions the graph into layers; and how this is orthogonal in some sense to the partitioning from eigenvector 19. Regions with large cuts are, of course, also regions with high degree nodes at their edges.

Figure 7 shows two small substructures detected by eigenvalues 8 and 12 respectively. In each case, a very small, unusual region of the graph is selected. Figure 8 shows another region with an interesting symmetry in the main part of the graph. Notice how, for these eigenvalues, most nodes have eigenvector values that are zero, so these nodes can be ignored.

6 A regular graph

As a slightly larger example, we consider the graph of Carbon_{60}, the so-called Buckyball, which forms a 3-dimensional sphere with pentagonal faces. Again, edges are given unit weights. A standard planar drawing of this graph is shown in Figure 9.

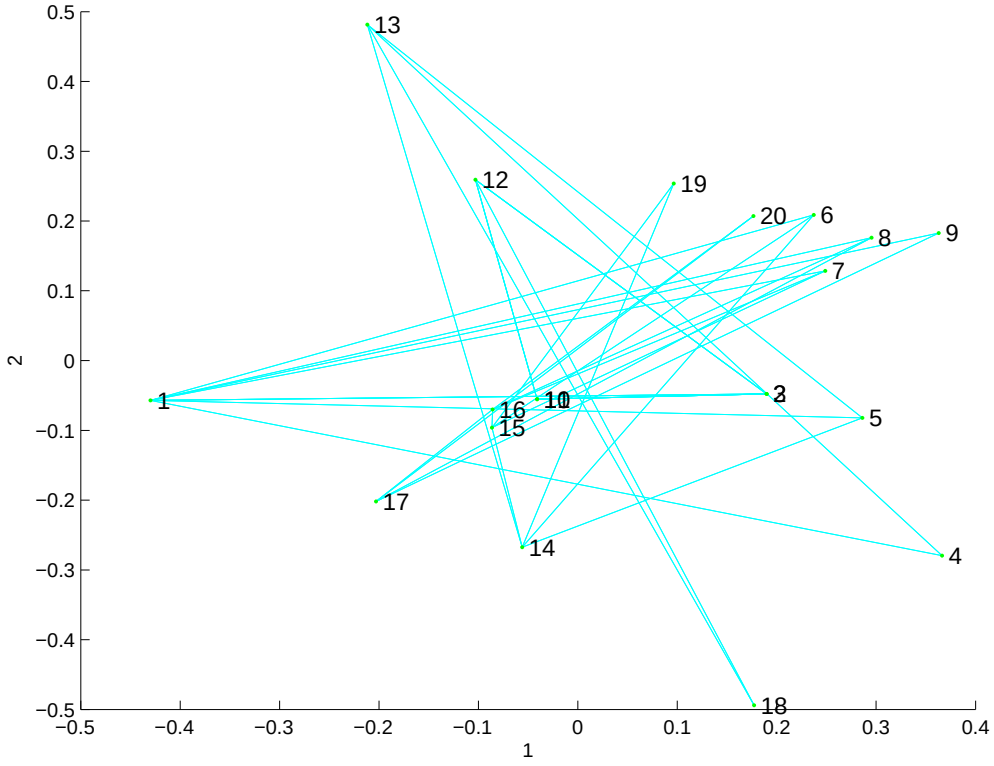

Figure 3: Graph nodes plotted according to the eigenvectors corresponding to the two largest eigenvalues

Even a small clique added to this structure is easily detectable. Figure 10 shows a plot of the eigenvectors corresponding to the two smallest nonzero eigenvalues. The three nodes of an inserted clique connecting nodes 1, 11, and 21 are all at one extreme of dimension 59. The corresponding plot of the 59th column of U is also shown in Figure 11.

Figure 12 shows how two disjoint paths connecting node 1 to node 21 via node 4 and node 5 also creates an easily detectable pattern.

Figure 13 shows what happens when five extra edges are added to node 1, connecting it to nodes 11, 21, 31, 41, and 51. As expected, it is now the eigenvector corresponding to the largest eigenvalue that is important. Node 1 is extremal at one end of dimension 1 while the points to which it connects are extremal at the other end.

7 Real graph data

It has been our experience that real data is 'lumpier' than artificial data, so that analysis algorithms typically run better. We now use a dataset of movie ratings, selected from the MovieLens data [9], with 400 people rating 600 movies from 1 to 5, where 5 indicates strong approval and 1 indicates strong disapproval. A graph is built from this data by considering each individual and movie to be an object, and each non-zero rating to be an edge, producing a 1000×1000 matrix. Such a dataset is expected to contain interesting substructures already, so we examine it to see what can be found,

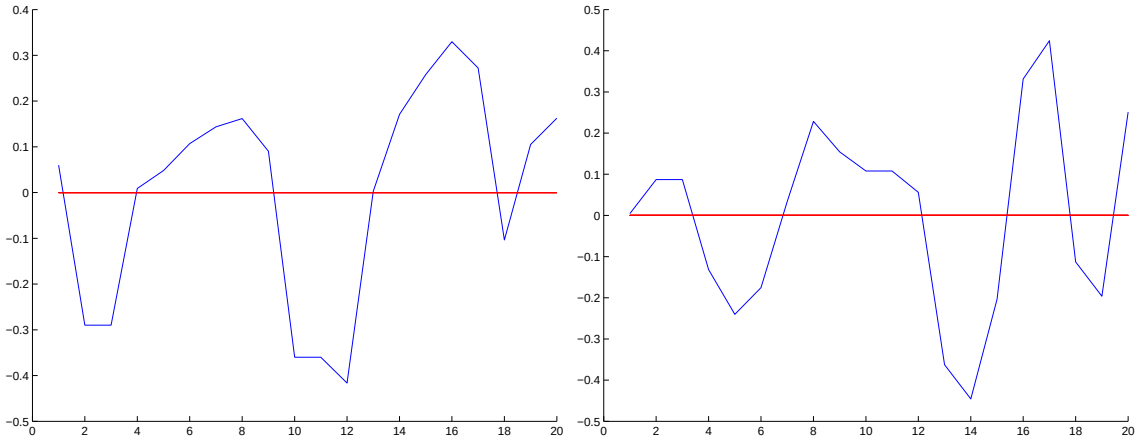

Figure 4: Eigenvectors 19 and 18 for the example graph – positive values are one cluster, negative values the other

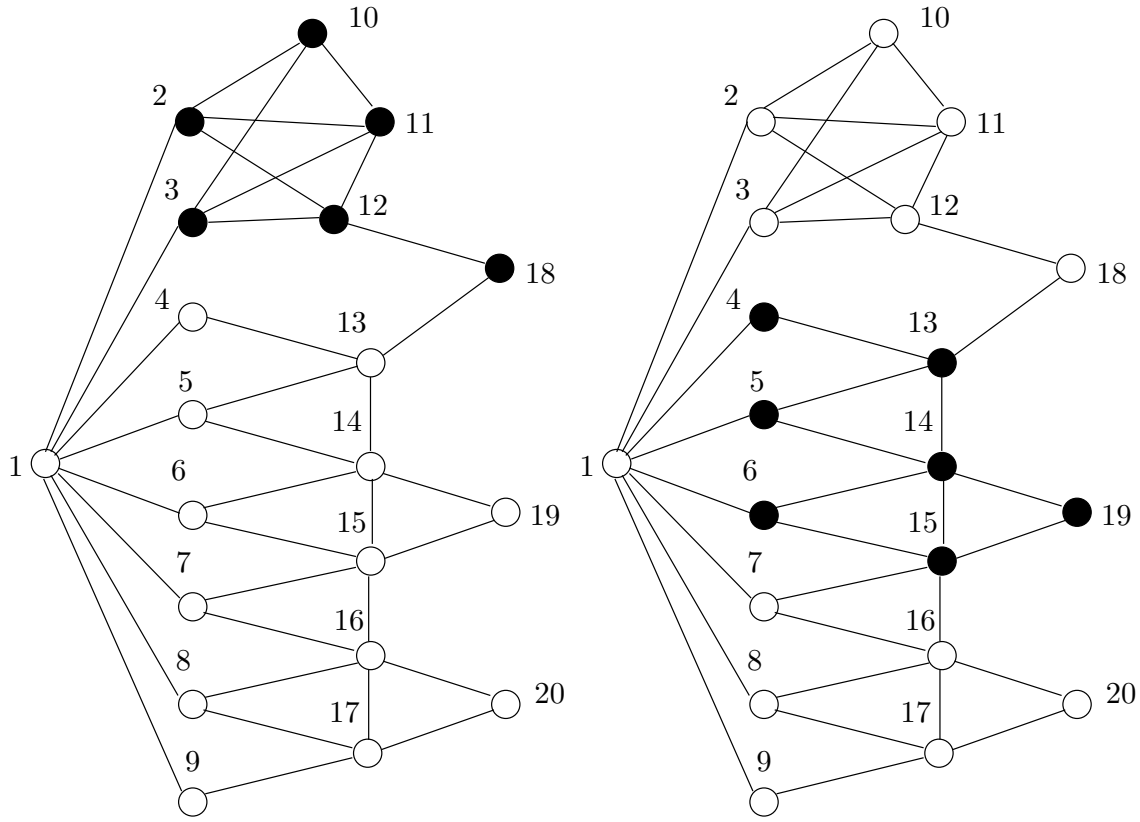

Figure 5: Example graph labelled from eigenvectors 19 and 18 (black = one cluster, white = the other cluster)

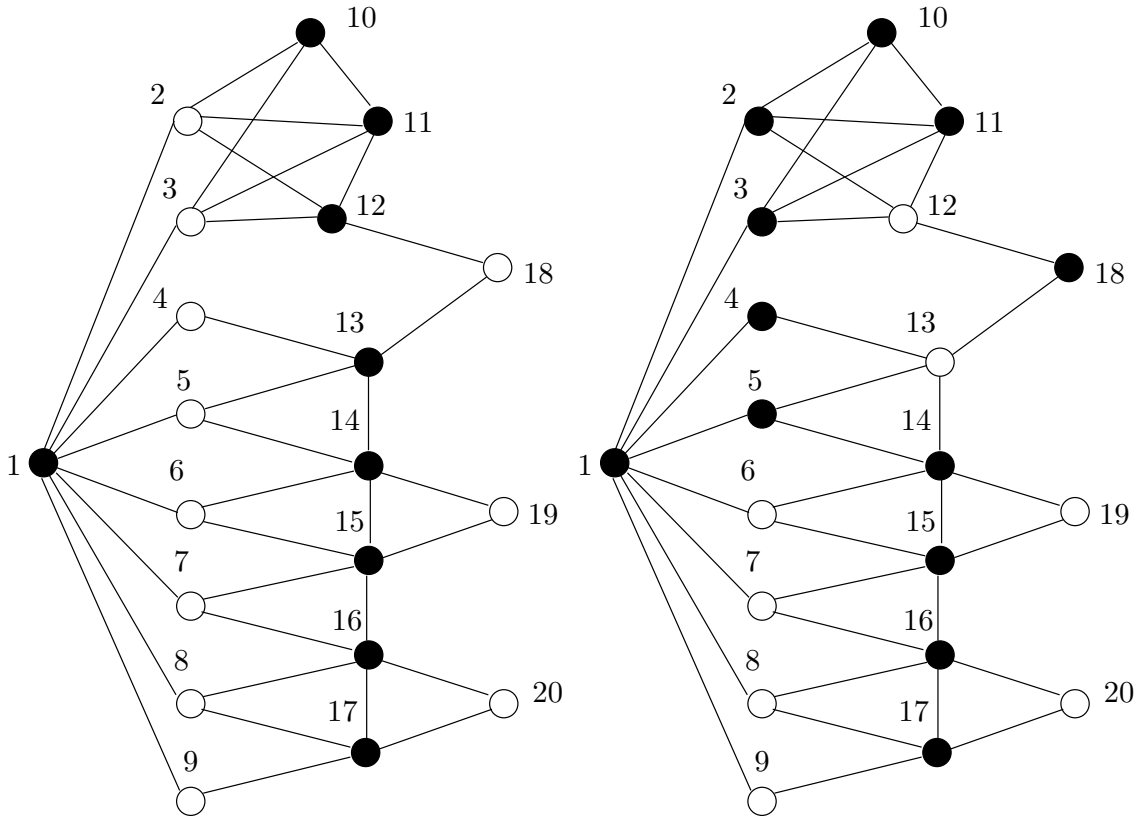

Figure 6: Example graph labelled from eigenvectors 1 and 2 (black = one cluster, white == the other cluster)

rather than inserting substructures.

Figure 14 plots the means of the absolute values of the eigenvectors of the walk Laplacian of this graph. It is obvious that interesting structure is to be found associated with the eigenvectors at each end, and a set of eigenvectors in the middle of the U matrix.

Figures 15–19 show a plot of the relevant eigenvector, and a plot of the graph for columns 50, 250, 500, 750, and 910 respectively. (There are many disconnected single-node components in this graph, which is why 910 rather than 1000 is used.)

There are clear differences between the substructures associated with components 250 and 750, which are dense and whose subgraphs contain much overlapping structure, and the other components that involve relatively few nodes, and whose subgraphs contain much simpler structures. The plot from component 50 indicates interesting single nodes; the plot from component 910 indicates an interesting set of cliques (note how the extremal nodes are connected to each other); while the plot from component 500 indicates a more complex structure involving relatively few nodes.

8 Conclusions

Graph data describes relationships or connections among people, places, and objects rather than describing their individual properties. It is also the natural way to describe situations where local

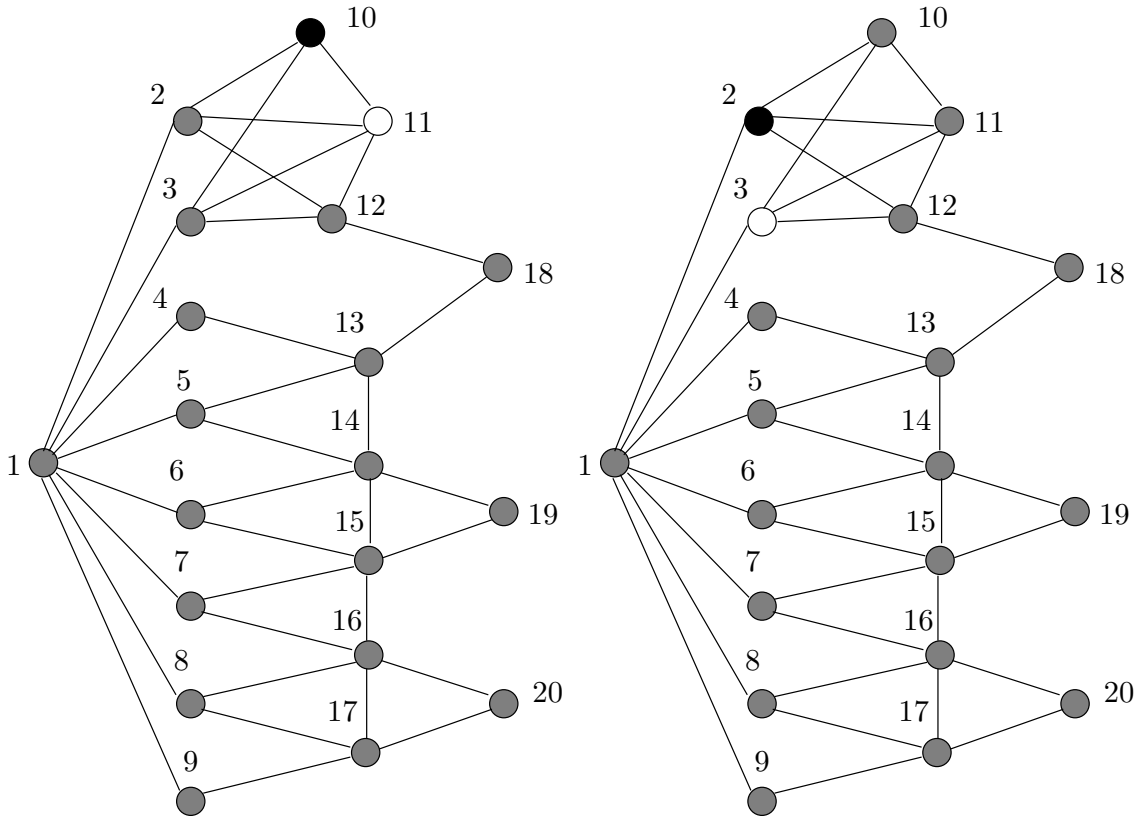

Figure 7: Example graph labelled from eigenvectors 8 and 12 (black = one cluster, white = the other cluster, gray = neutral)

information is readily available, but hard to integrate into a global picture.

In such data, substructures that correspond to the actions of ordinary people will be repeated often and so will be common. Because terrorists and criminals do unusual things they will tend to have unusual relationships, which will appear in graph data as unusual substructures. The problem of finding such unusual substructure has been treated as an information retrieval problem: given a particular substructure, find occurrences of it in a given graph. This is limited because it requires all possible attack substructures to have been thought of; and because it may fail to match a substructure because some part of it was not captured in the datasets.

We have presented an extension to spectral graph partitioning that can be used to find *all* unusual structures in a graph. In practice, some of such structures will be entirely innocent, so some post-analysis is still needed. However, in terrorism settings false positives are preferred to false negatives.

Weighted adjacency matrices corresponding to graph data are mapped to walk Laplacian matrices that are then transformed by eigendecomposition. Interesting substructure can be found in any component of the resulting matrix. Since this matrix is still $n \times n$, we have suggested some ways in which the significant columns can be detected. There is considerable room for developing more subtle analysis techniques, perhaps by considering the norms of the outer product matrices, or constructing better visualizations.

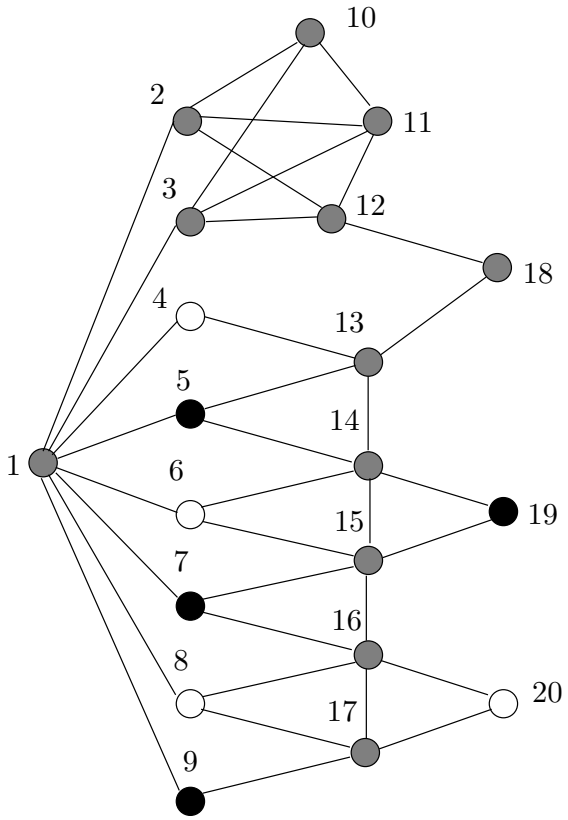

Figure 8: Example graph labelled from eigenvector 11 (black = one cluster, white = the other cluster, gray = neutral)

References

[1] M. Belkin and P. Niyogi. Laplacian eigenmaps and spectral techniques for embedding and clustering. In T. G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 14*, Cambridge, MA, 2002. MIT Press.

[2] M. Brand. A random walks perspective on maximizing satisfaction and profit. In *SIAM International Conference on Data Mining*, pages 12–19, 2005.

[3] T. Coffman, S. Greenblatt, and S. Marcus. Graph-based technologies for intelligence analysis. *CACM*, 47(3):45–47, March 2004.

[4] F. Fouss, A. Pirotte, J.-M. Renders, and M. Saerens. Random-walk computation of similarities between nodes of a graph, with application to collaborative recommendation. *IEEE Transactions on Knowledge and Data Engineering*, 2006.

[5] M. Girvan and M.E.J. Newman. Community structure in social and biological networks. *Proceedings of the National Academy of Science*, 99(12):7821–7826, 2002.

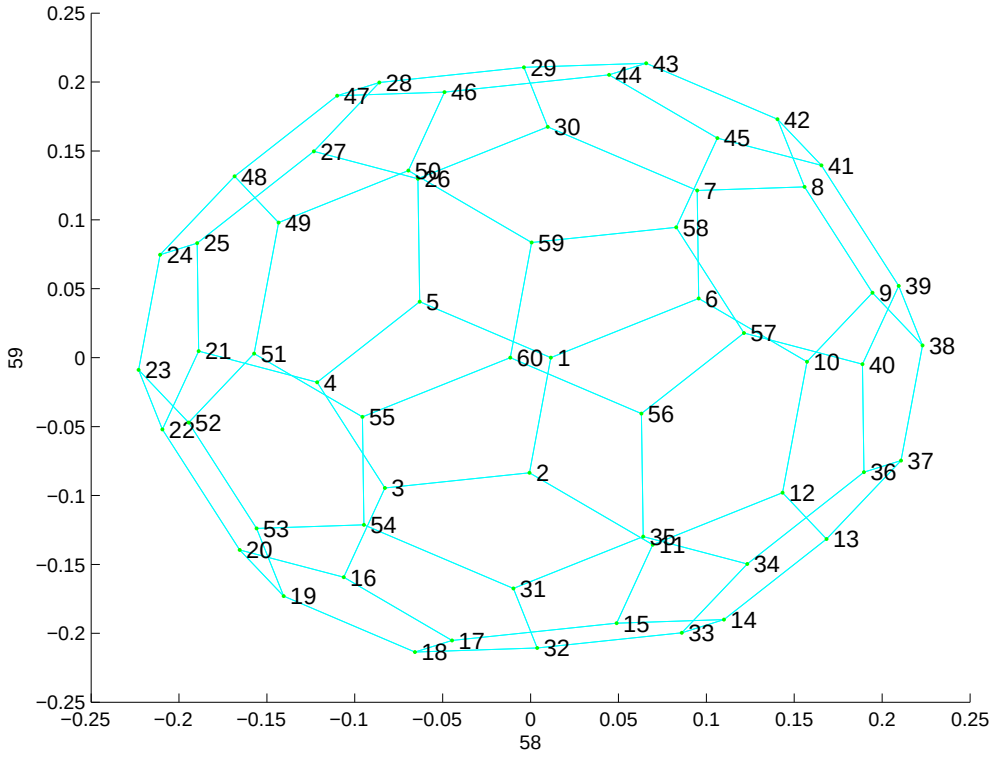

Figure 9: Standard planar drawing of the Buckyball graph

[6] D. Jensen and J. Neville. Data mining in social networks. Invited presentation to the National Academy of Sciences Workshop on Dynamic Social Network Modeling and Analysis, November 2003.

[7] V.E. Krebs. Mapping networks of terrorist cells. *Connections*, 24(3):43–52, 2002.

[8] D. Liben-Nowell and J. Kleinberg. The link prediction problem for social networks. In *Proceedings of the twelfth international conference on Information and knowledge management*, pages 556–559, 2003.

[9] B. N. Miller, I. Albert, S. K. Lam, J. A. Konstan, and J. Riedl. MovieLens unplugged: Experiences with an occasionally connected recommender system. In *IUI'03: Proc. 8th Int. Conf. on Intelligent User Interfaces*, pages 263–266, Miami, Florida, USA, 2003. ACM Press.

[10] M. Saerens, F. Fouss, L. Yen, and P. Dupont. The principal component analysis of a graph and its relationships to spectral clustering. In *ECML 2004*, 2004.

[11] U. von Luxburg. A tutorial on spectral clustering. Technical Report 149, Max Plank Institute for Biological Cybernetics, August 2006.

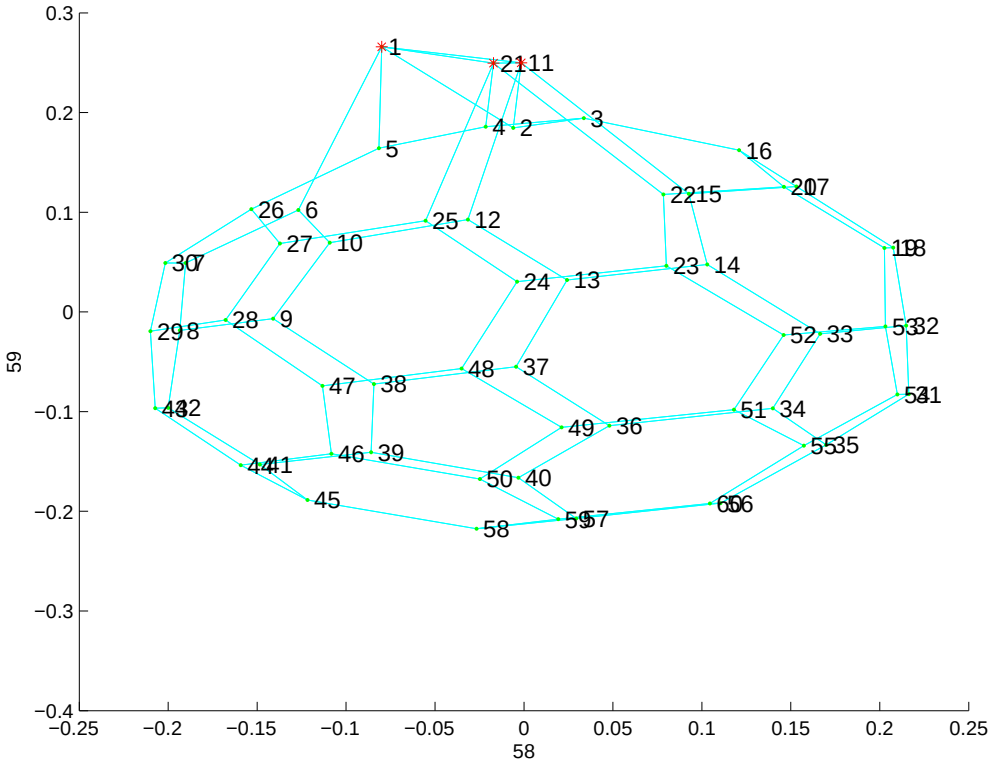

Figure 10: Buckyball with inserted clique, points labelled with red stars

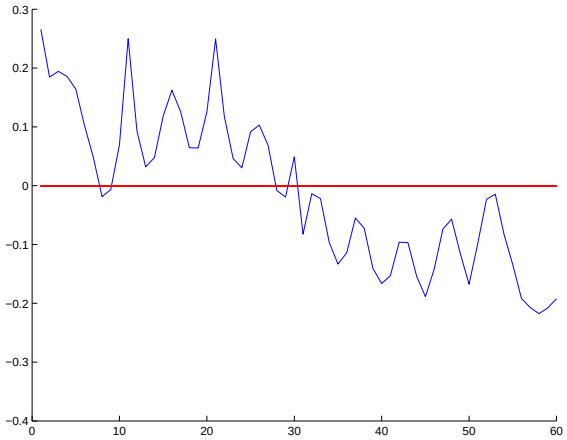

Figure 11: Plot of 59th column of U

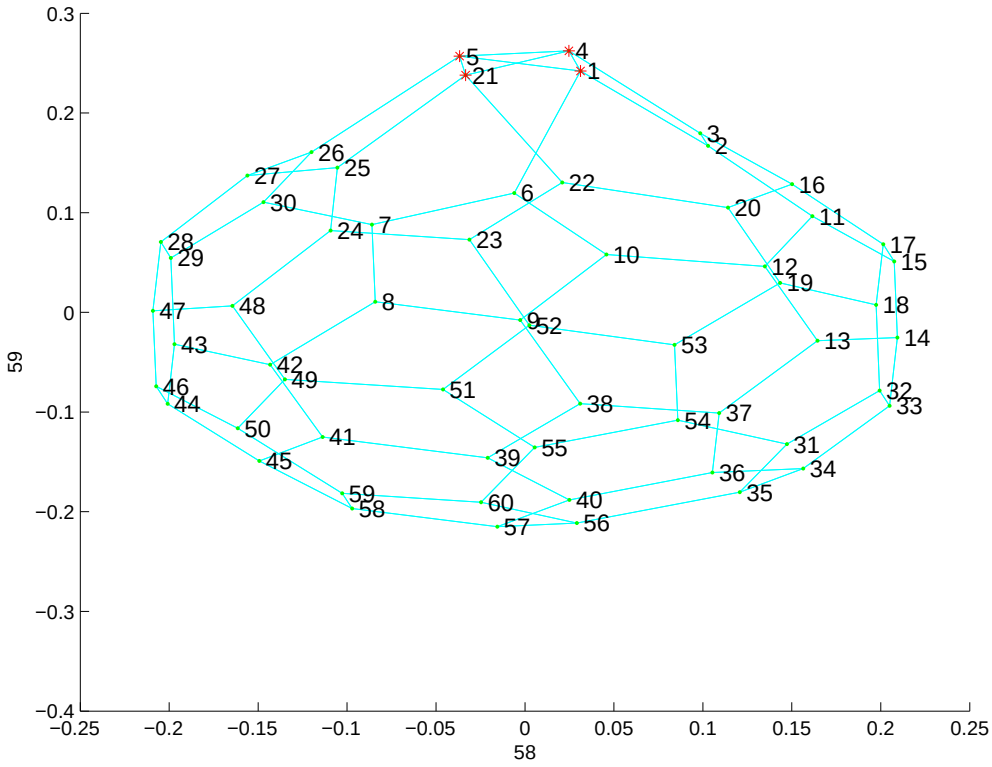

Figure 12: Buckyball with inserted paths, points labelled with red stars

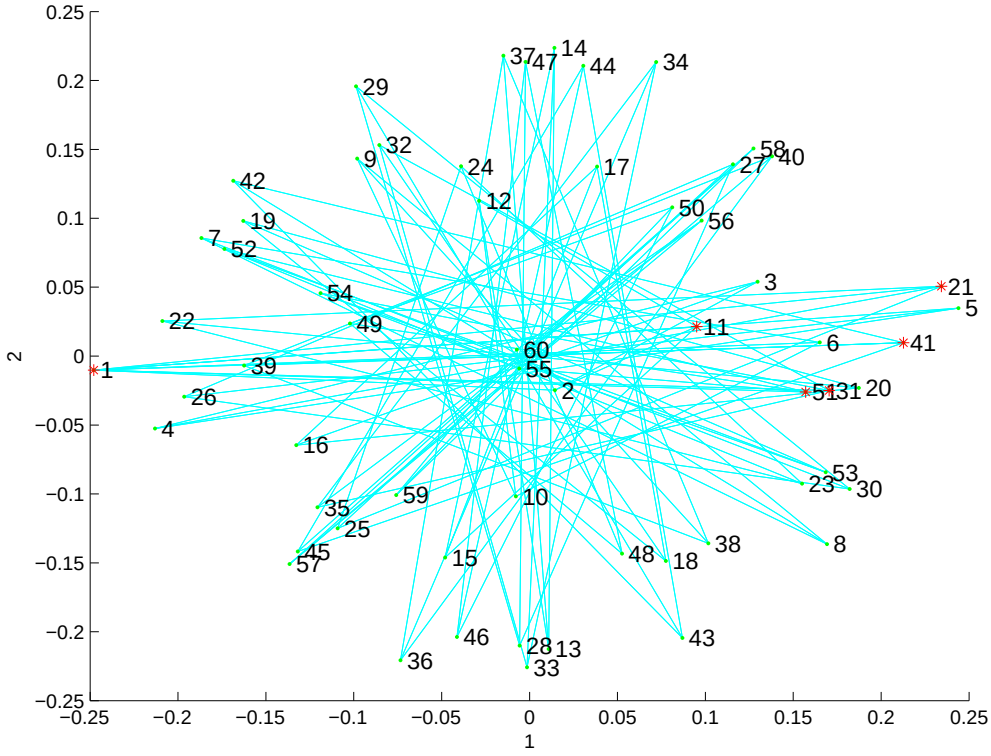

Figure 13: Buckyball with inserted high-degree node 1, points at the ends of added edges labelled with red stars

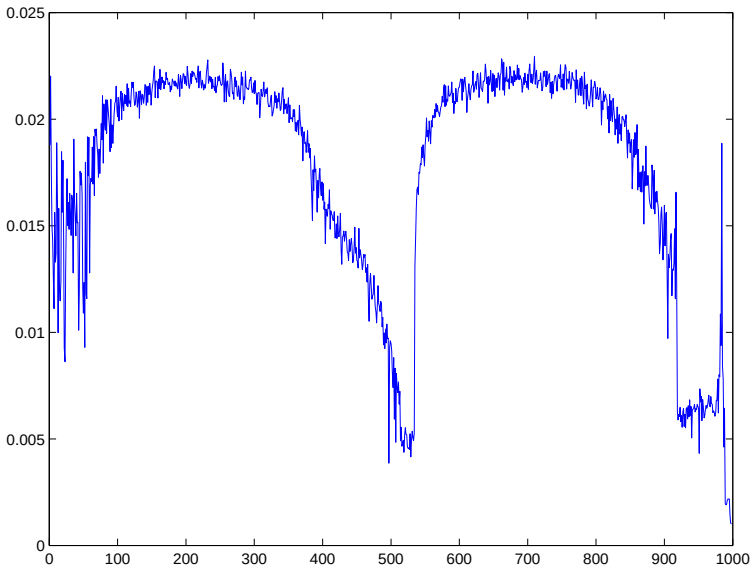

Figure 14: Plot of the means of the absolute values of each column of U. Low values indicate components that involve few nodes, or deviate only slightly from zero, or both. Such components are likely to be interesting

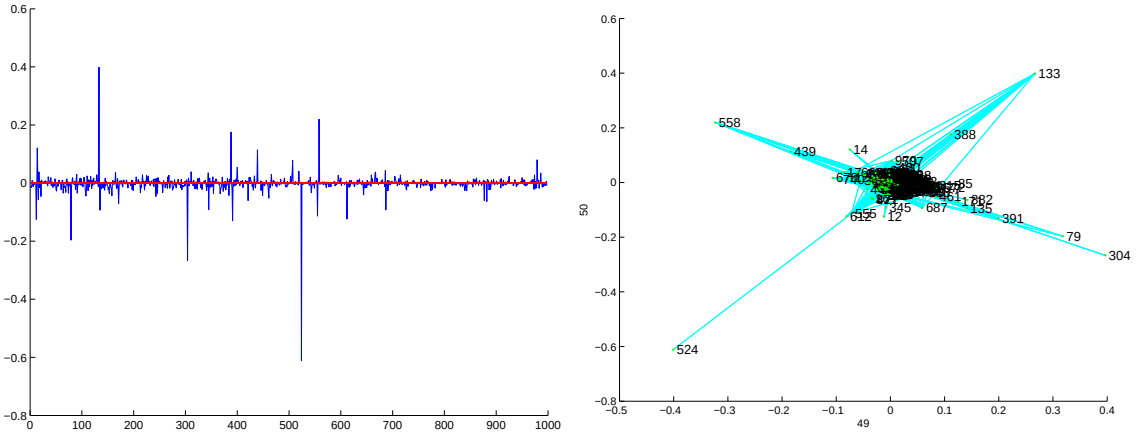

Figure 15: Eigenvector and graph plots for column 50 of the U matrix

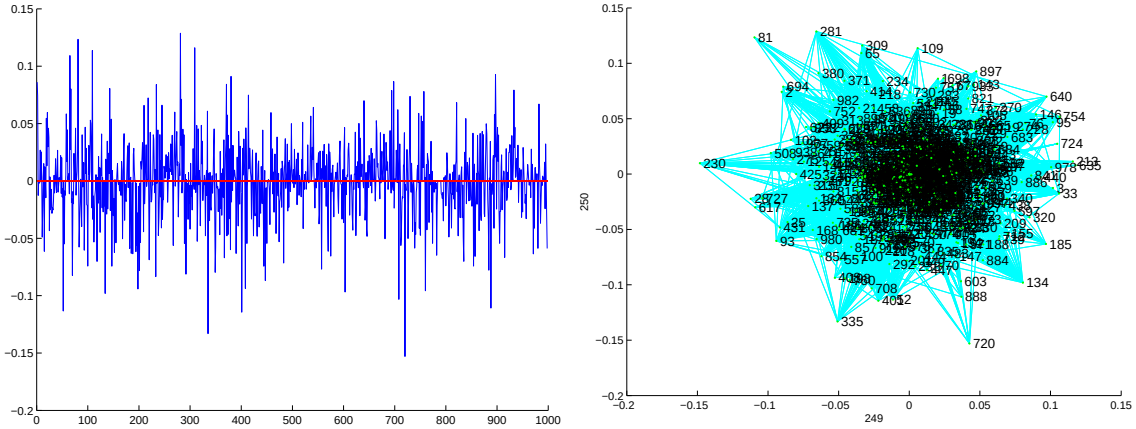

Figure 16: Eigenvector and graph plots for column 250 of the U matrix

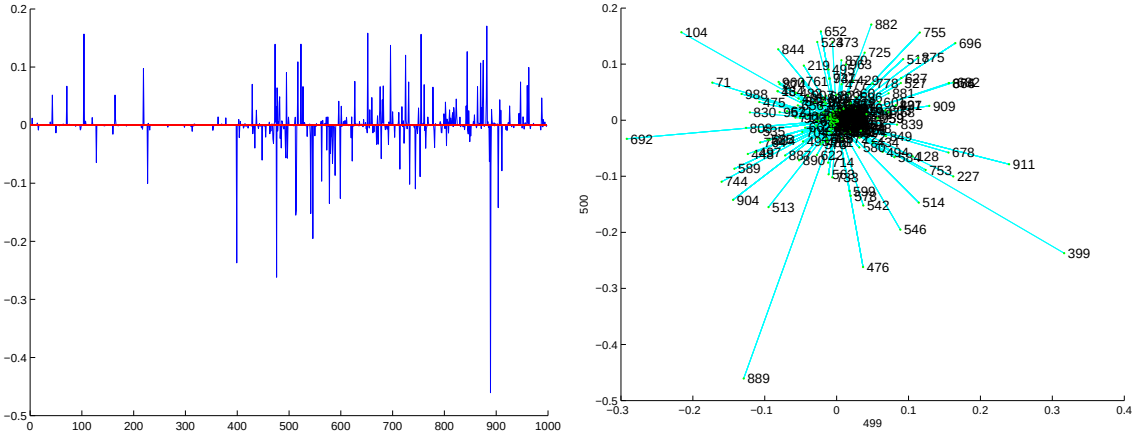

Figure 17: Eigenvector and graph plots for column 500 of the U matrix

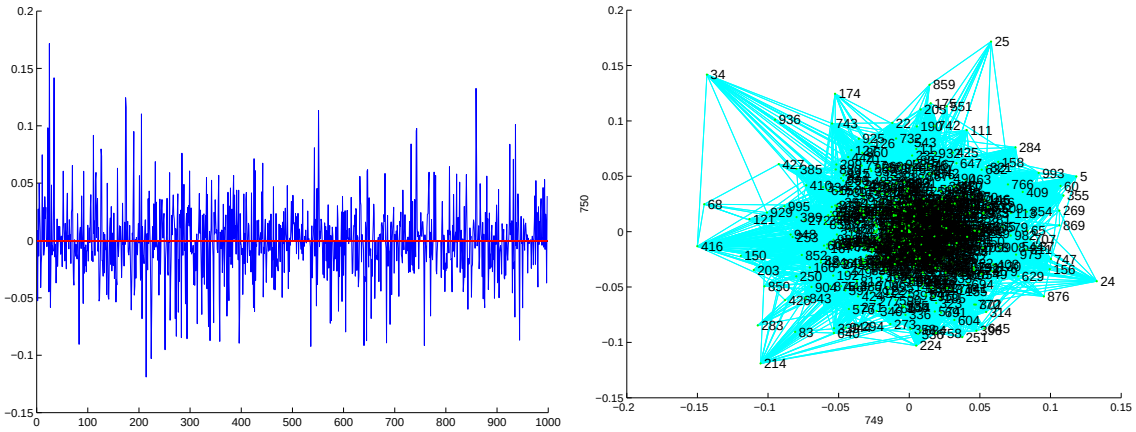

Figure 18: Eigenvector and graph plots for column 750 of the U matrix

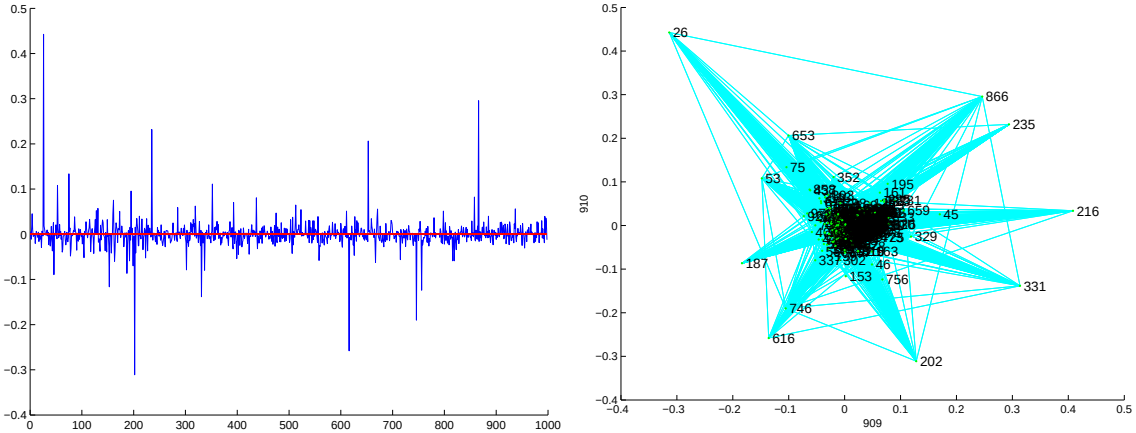

Figure 19: Eigenvector and graph plots for column 910 of the U matrix