Name: Rakesh Gopal Kavodkar

UnityID: rgopalk

Student ID: 200066020

Paper: EVENT DETECTION IN TIME SERIES OF MOBILE COMMUNICATION GRAPHS

**Algorithm**:

The paper discusses an algorithm for the detection of change points or anomalies in a time varying graph. The dataset that the paper discusses is of mobile messaging traffic in a city over a period of several months. The anomalies here are defined as a sudden change in the behavior or multiple nodes.

The algorithm uses(constructs with the edge data) a dataset which is a three dimensional matrix (N X T X F) where N is the nodes, T is the time and F is a set of features (although the algorithm uses 12 different features, we use just 3 for our implementation). We run the algorithm individually for each of the features, there using a dataset of T X N in each run. Once we have this matrix, we compute the correlation matrix (correlation between every pair of nodes in the dataset over a period of time). Once we have the correlation matrix, we compute the eigenvectors for that matrix, but retain just the principal vector for further use in the algorithm. We get the principal eigenvectors in each time window. Then we calculate the current eigen behavior, u(t), and the typical eigen behavior, r(t-1). Once we have these, we compute their dot product. If the value is higher the calculated threshold, then we consider that as a change point. Threshold is calculated as **median + 3 \* moving\_average**

**Features used in this implementation**: *degree of the node, clustering coefficient of the node* and *number of edges in the egonet of the node.*

**How could your algorithm be improved in order to be more robust and possibly identify more anomalies while not producing false positive?**

* While calculating the Z value, we skip the first *W’* nodes since we do not have any prior *W’*  nodes for the initial ones. The algorithm could be tweaked to include these values as well
* The algorithm use the previous values of eigenvectors as a standard to compare the current eigenvector.
* The algorithm requires a decent amount of time series data to make the sliding window implementation more accurate

**Could it be improved in terms of time and memory complexity?**

* The data matrix for a given day is highly sparse since not all nodes are connected on a given time. We could use something less memory intensive like a adjacency list or store only the active nodes on a given day and change the algorithm accordingly
* A lot of time is spent in reading the data from a large number of files. The data could be stored in a single file with the day information as well thereby reducing the amount of individual IO operation
* The entire set of eigenvectors are computed (using the igraph library) but only the principal vector is considered. This is a huge waste of time and space

**What kind of anomalies could your algorithm have trouble detecting? Why?**

* As pointed out in the above section, while calculating the Z values we skip the first *W’* values since we atleast *W’* values to compute the typical eigen behavior, r(t-1). Any anomaly in the skipped period is most certainly missed.

**Is your algorithm parameterized?**

Yes, the algorithm takes the *window size* and the *W’* values as parameters. As per the paper, we input the values as 7 and 5 respectively.

**Is the algorithm Randomized?**

No, there is no random component in the algorithm

**Does it start from a seed set of vertices? how does it affect your performance?**

No, it doesnt use a seed set of vertices. The algorithms considers the nodes in order of the time series

**What percentage of the total number of input graphs did your algorithm identify as**

**anomalous?**

The values in the table describes the fraction of nodes that are anomalous

* Enron dataset

|  |  |  |
| --- | --- | --- |
| Degree | Clustering Coefficient | Edges in the egonet |
| 0.1653543 | 0.05643045 | 0.156168 |

* Enron Noempty dataset

|  |  |  |
| --- | --- | --- |
| Degree | Clustering Coefficient | Edges in the egonet |
| 0.1612466 | 0.05420054 | 0.1476965 |

* Reality mining voices

|  |  |  |
| --- | --- | --- |
| Degree | Clustering Coefficient | Edges in the egonet |
| 0.03703704 | 0.2222222 | 0.2222222 |

**What does it say about sensitivity of the algorithm?**

The algorithm is sensitive to sliding window size, as mentioned in the paper. With a size of 7, it seems to perform better than with the size 10

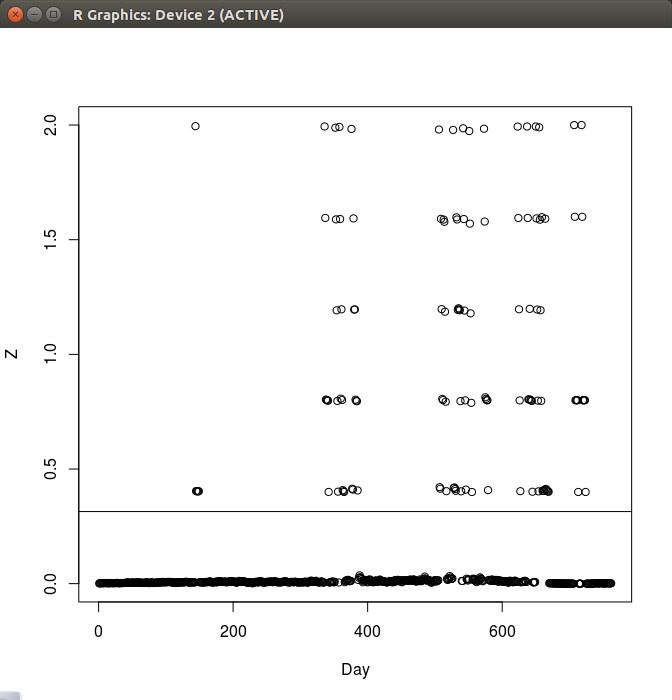
**Are the detected time points very near to each other or very spread out**?

The time points are spread out

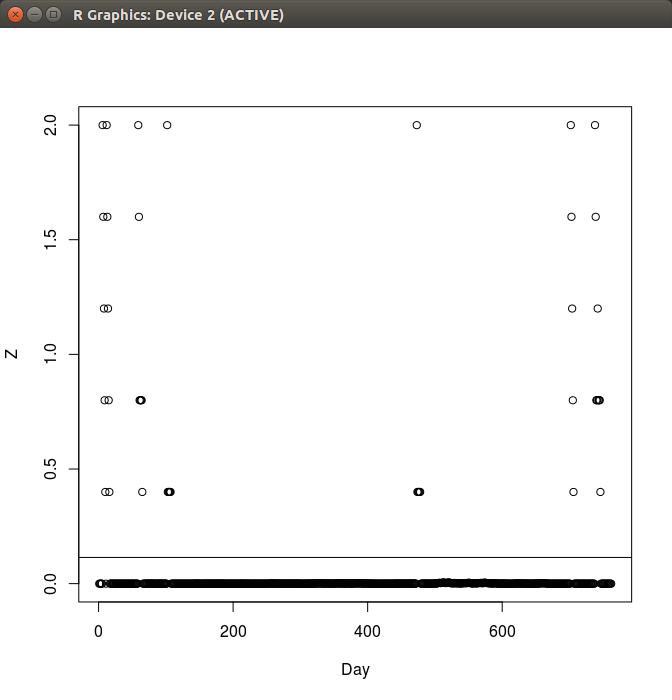
**Compare the anomalous time points to normal time points?**

The plots for the 3 different types of features on 3 different data sets are shown below

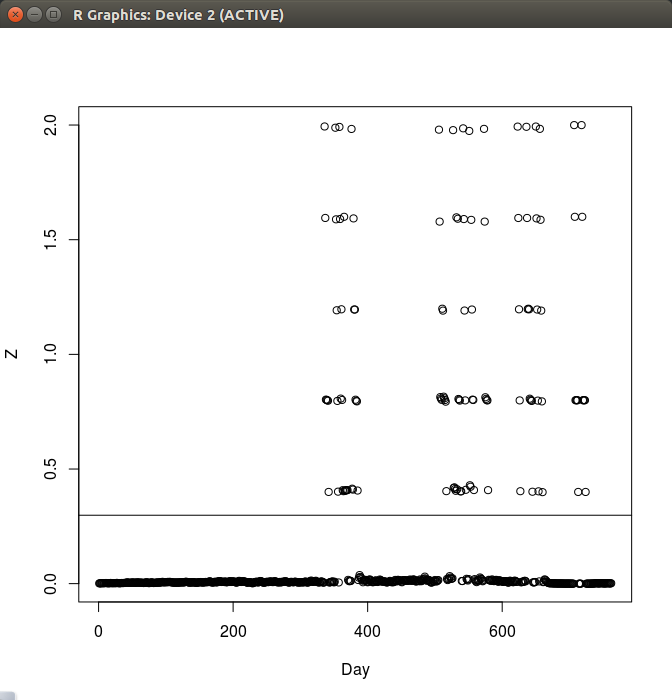
Enron - Degree



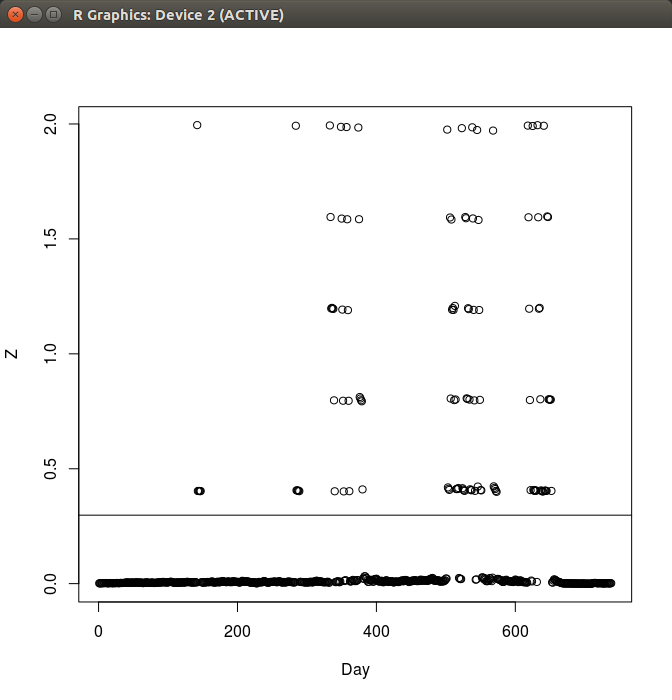
Enron - Clustering Coefficient



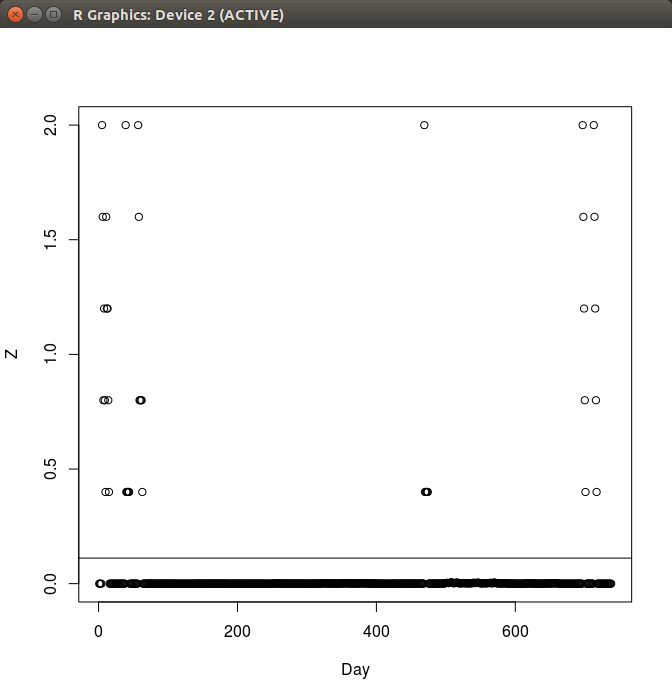
Enron - No. of edges in the egonet



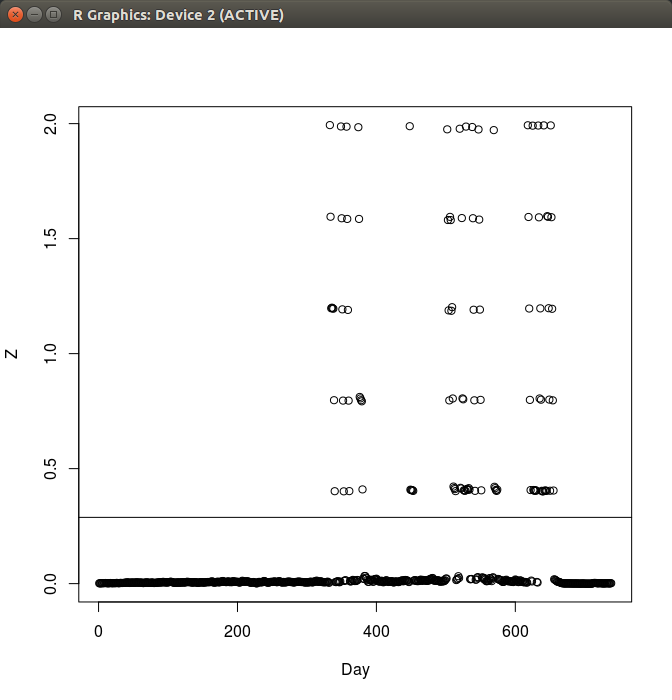
Enron\_noempty - Degree



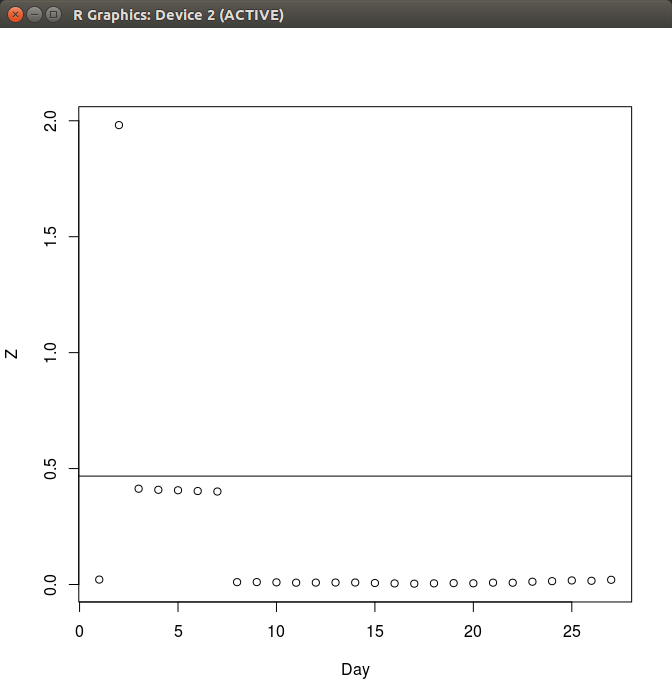
Enron\_noempty - Clustering Coefficient



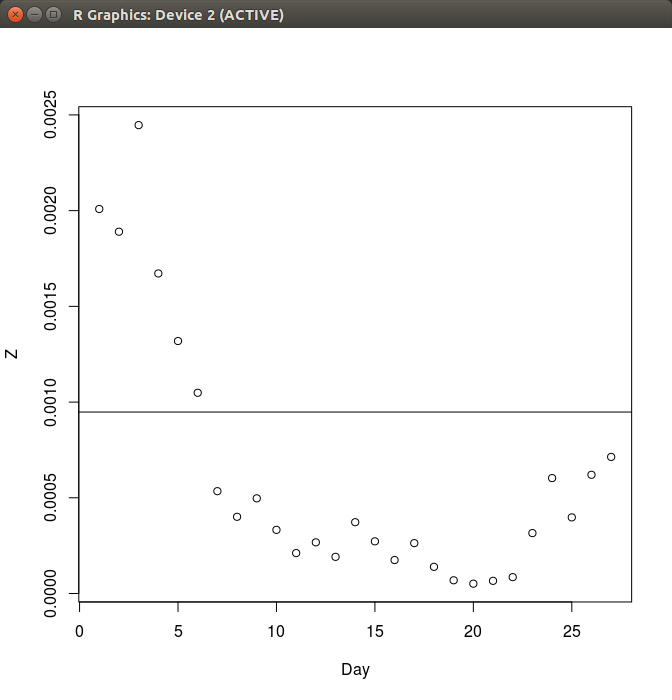
Enron\_noempty - No. of edges in the egonet



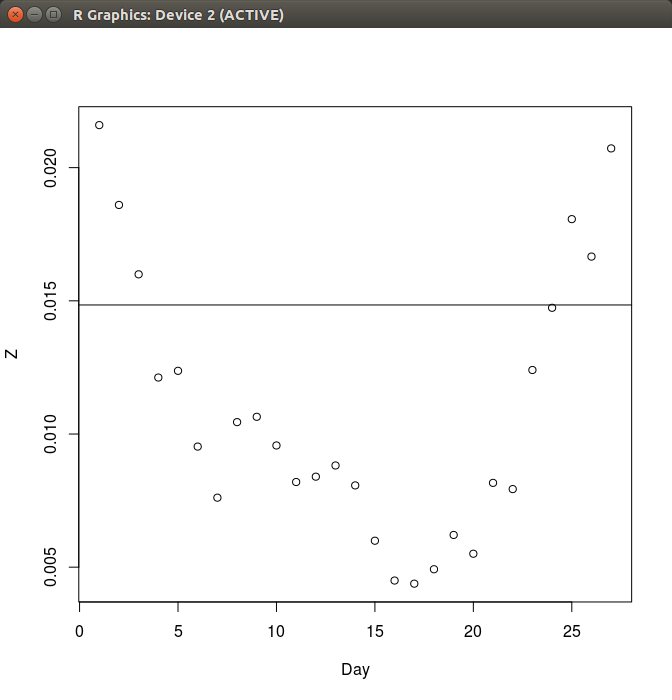
Reality mining - Degree



Reality mining - Clustering Coefficient



Reality mining - No. of edges in the egonet



**What differences do you see in graph structure?**

The anomalies are well separated from the rest of points. The value of Z is very high as compared to the ones without any anomaly. However, the distinction is not that *distinct* in case of the Reality Mining Voices dataset.