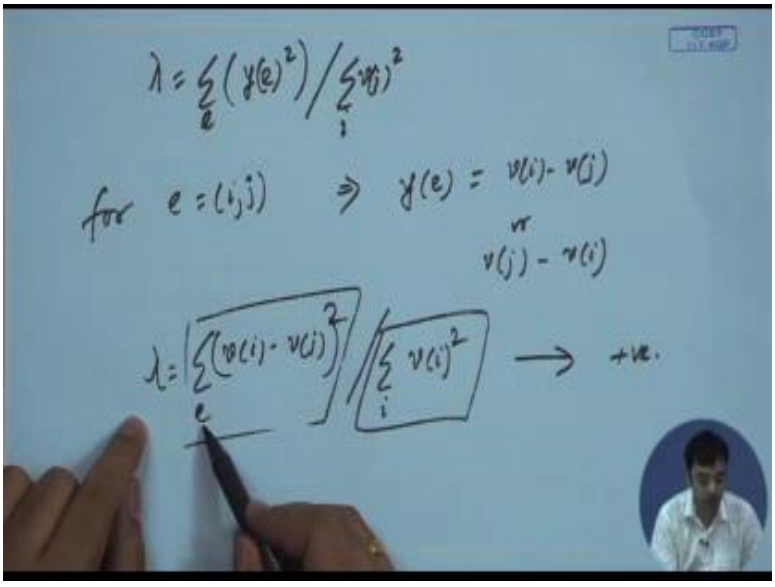
Complex Network: Theory and Application
Prof. Animesh Mukherjee
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur

Lecture - 17
Community Analysis – VI

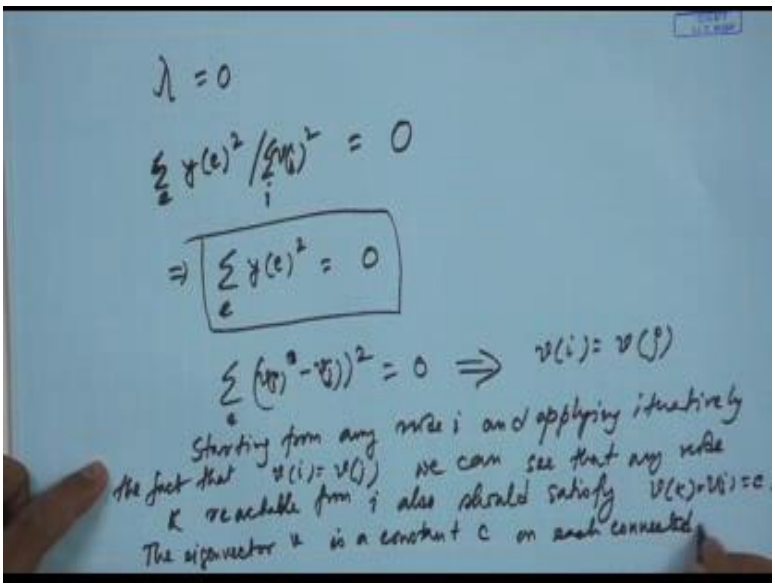
So, in the last lecture we have been looking into some of the spectral properties of the graph in order to culminate to the definition or the algorithm for spectral by section. So, we have proved some properties. Now, we will continue with the last few properties that we have to still prove.

(Refer Slide Time: 00:44)



So, we have seen that lambda comes to an expression, which is sum over all the edges v (i) minus v (j), where v (i) and v (j) are the entries in the eigenvector whole square divided by sum of v (i) whole square. So, this is a square divided sum of squares divided by another sum of squares, which we have seen that it is positive and therefore, we have proved till property 5. So, we have shown that lambda boils down to an expression which is non negative. Now, we have to look into this specific property 6, which is a very important property. In order to show property 6, we assume that lambda is equal to 0.

(Refer Slide Time: 01:24)

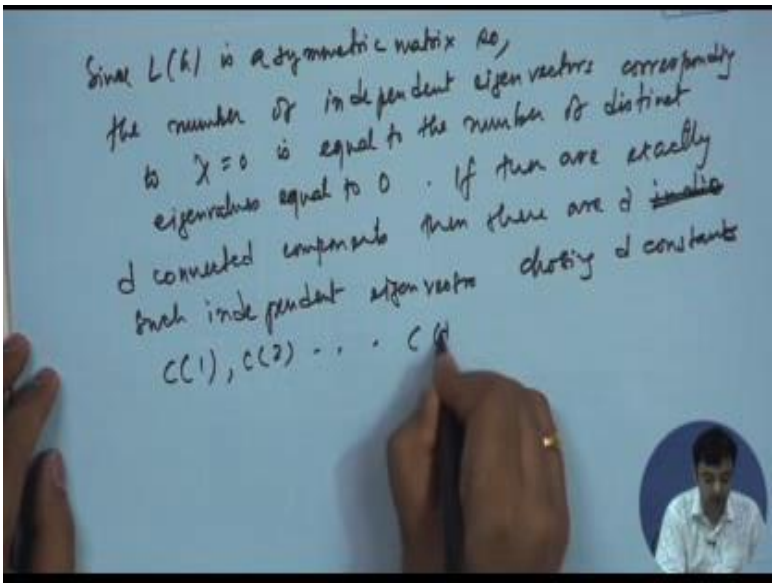


So, let us assume that lambda is equal to 0, then we have the expression sum over all edges y (e) whole square by v (i) whole square sum over all I, this value is actually equal to 0. If lambda is equal to 0, we have actually evaluated the expression for lambda like this which is some of e (y) is whole square by some of i v (i) whole square in the last lecture.

So, you can write that these expression actually will be equal to 0, if lambda is equal to 0; that means, essentially some of e y (e) whole square is equal to 0, since some of i v (i) whole square is a positive scalar. So, this is what you boil down to. Now, this means that essentially. So, this you can further write as v (i) minus v (j) the whole square is equal to 0 that is what we have seen last day. So, this will tell you that v (i) is nothing, but equal to v (j) now, starting from any node i and applying iteratively the fact that v (i) is equal to v (j). We can see that any node k reachable from i also should satisfy v (k) is equal to v (i). So, since v (i) minus v (j) whole square is equal to 0 then applying this property iteratively. So, v (i) is equal to v (j), v (j) is equal to v (k) and in this way v (k) is equal to v (m) and so on and so forth up to a point. So, on a connected component everything that is reachable from i, we will have a similar eigenvector entry.

So, that means, the eigenvector v is a constant c on each connected component. So, basically this comes from the assumption that lambda is equal to 0. So, lambda equal to 0 means v (i) minus v (j) whole square sum over all edges is equal to 0. So, this will give you v (i) is equal to v (j), similarly if you put, if you replace j with k then you will have v (i) is equal to v (k) because v (j) will become equal to v (k) then that means, v (i) will become equal to v (k) similarly you will keep on continuing. So, if j is connected to k then there is an edge between j and k. So, then that lambda will also be equal to 0; that means, v (i) will be equal to v (k). In this way v (i) will be equal to the next node until and analysis you have reach the end of a particular component. So, for every individual component you will have a separate eigenvalue equal to 0. So, there will be as many eigenvalues equal to 0 as many there are components in the graph.

(Refer Slide Time: 05:32)

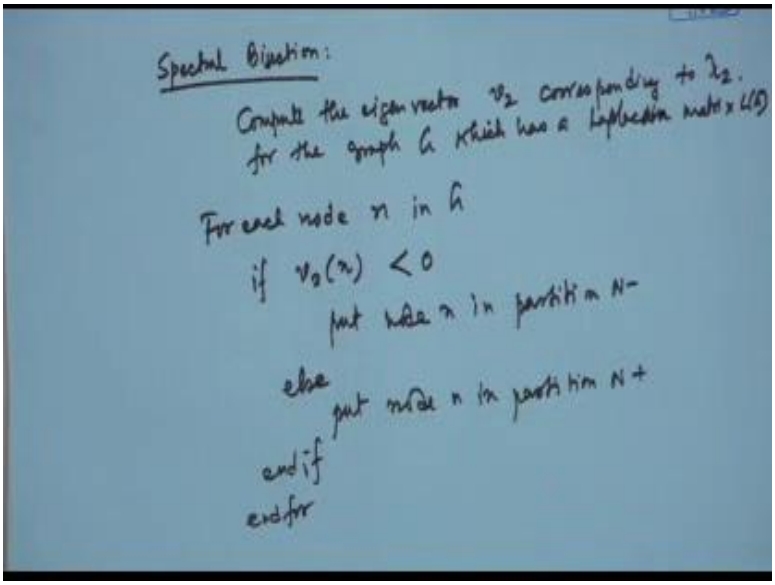


Basically, what I want to say is that since L (G) is a symmetric matrix, the number of independent eigen vectors corresponding to lambda equal to 0 is equal to the number of distinct eigenvalues equal to 0. So, if there are exactly d connected components then there are d such independent eigenvectors, choosing d constants. So, each eigenvector as we have discussed will be a constant like c choosing c (1), c (2) up until c (d). So, each lambda is equal to 0, each eigenvalue equal to 0 will define an eigenvector of constant entries and that will be defined on a particular component.

Now, if there are more than 1 0 lambda values; that means, there are more than 1 connected component in the graph that means, lambda 2 has to be non-zero if there is only 1 single connected component in the graph. So, that is why we ensure the condition that if the graph has to be connected then the lambda 2 value has to be non-zero.

So, that is why lambda 2 actually encodes the information whether the graph is disconnected into modules or whether it is a single module and that is why this is used so heavily in identifying communities based on spectral methods. Now, given all this properties we will use the value of lambda 2 in order to identify the partitions. So, the algorithm is very, very simple; spectral bisection.

(Refer Slide Time: 08:33)



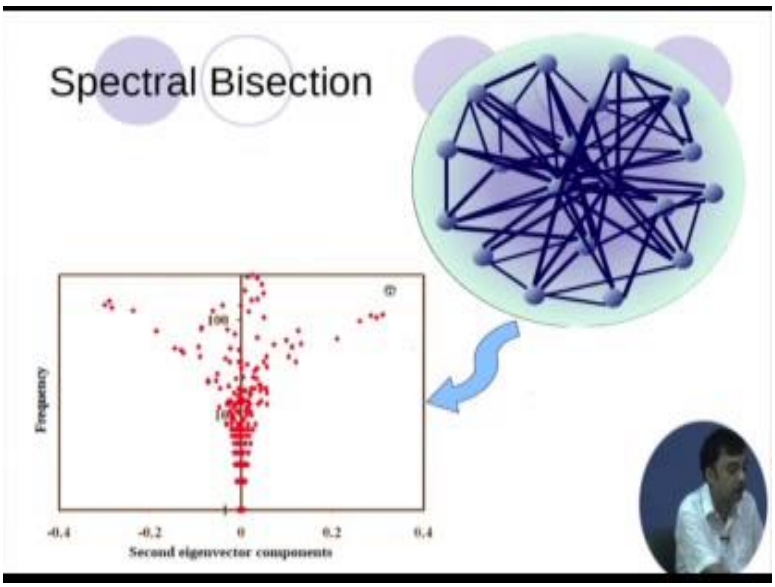
Compute the eigenvector v 2 corresponding to lambda 2 for the graph G which has laplacian matrix L (G). For each node n in G, if v 2 of n is less than equal to 0 put node n in partition n minus else put node n in partition n plus end. So, basically end if and then you have end for. So, basically what you are doing is you are leveraging the idea of the fact that lambda 2 encodes whether there are multiple components in the graph or not. In general, if you have the principle eigenvector will be all positive entries.

So, therefore, the second eigenvector corresponding to the eigenvalue lambda 2 the second smallest eigenvector corresponding to lambda 2, all its entries cannot be positive some of them has to be positive, whereas some of them has to be negative because why because we have already seen that since L (G) is a symmetric matrix; that means, that all its eigenvectors are orthogonal to each other and since they are orthogonal to each other their dot product should be equal to 0 and therefore, all the entries in the vector corresponding to lambda 2 cannot be positive, since all the entries in vector corresponding to lambda 1 is positive.

Since, the principle eigenvector everybody is positive the second eigenvector everybody cannot be positive some of there will be positive whereas, the others will be negative these positive and the negative class actually defines the orientation of the nodes into 2 different clusters and that is how, it divide the network into 2 components and that is the idea that this spectra bisection algorithm exploits.

So, basically for every node end, you inspect the entries in the second eigenvector you try to see each entry in this second eigenvector. If you see that the entry corresponding to a node is negative then you put that node in the negative partition. If you see that it is positive then you put that node in the positive partition in this way you get 2 communities 2 broad communities. Now, you can iteratively further divide this communities in order to get further substructures and finely construct a construct a dendrogram as we have seen earlier.

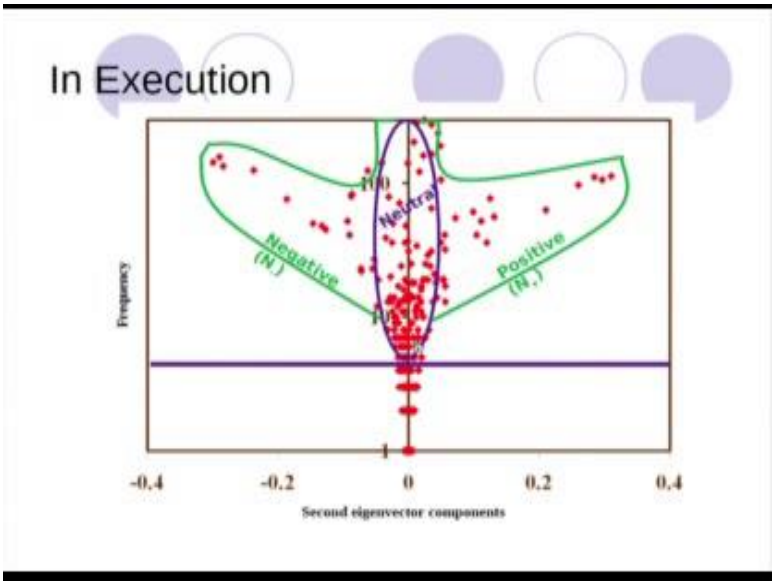
(Refer Slide Time: 12:10)



So, I will show you, to end this lecture I will show you in over the slides execution of spectral bisection on a given sample graph. Let us consider that this is our sample graph here. So, if you plot the components of the second eigenvector. So, these are like the v 2 ns that we have seen the entries in the second eigenvector. So, as I said some of them will be negative whereas, the others will be positive and the y axis here is basically the degree of the node if you plot the degree versus the second eigenvector entries you get a structural like this.

This actually defines very nicely 2 dominant classes, there is 1 which is a negative class here where the entries are predominately negative, there is 1 which is positive at the entries are predominately positive, but then there are a set of nodes which are very close to 0 this is what we call the neutral class.

(Refer Slide Time: 13:06)



So, the set here is basically the neutral class it is very difficult to identify whether they belong to positive or negative class because their eigenvector entries are very, very small whereas, you have one very clearly defined negative class and another very clearly defined positive class.

In this way what you can do so in this next step. So, in order to further refine your clustering. If you want to get further deeper levels of clusters what you can do is you can remove the nodes which are clearly in n minus, you can also remove the nodes which are clearly in n plus. So, you will get a smaller graph will get a smaller sub graph from the bigger graph which will consider only this nodes which are close to the neutral.

Now, you again re-compute the second eigenvector for that smaller graph and you again will get a positive class and the negative class then and the neutral class. So, then again remove the negative and the positive and with the neutral class reconstruct another create another sub graph and you keep on doing this procedure until and unless there are very, very less nodes left in the neutral class and if you backtrack this process you will finally, get the dendrogram or the hierarchical structure of the communities.

So, that actually brings us to the end of the spectral bi-sectioning method and also the community analysis series. So, in the next lecture we will take up the citation analysis which is like case study for all the things that we have studied so far.

Thank you very much.