Spectral Graph Partitioning

Complex Network Theory

Tutorial – 6/12/2021

During this lab, we will implement the three variants of the spectral partitioning algorithm that we saw during the lesson, and we will test these variants to assess their accuracy and limitations.

Q1) We will test our methods on some random **simple** graph. Namely:

- This graph has 1000 nodes.
- It is composed of two blocks of 500 nodes.
- The probability for two nodes of a same block to be linked is 0.05.
- The probability for two nodes of different blocks to be linked is 0.01.

Write a Matlab function to generate the adjacency matrix of such a random graph. Observe this matrix using the function `spy`.

We will now create a Matlab function that encapsulates the spectral partitioning algorithm variants. This function takes as an input the graph adjacency matrix, and returns the signed indicator of the discovered set. For all the algorithm variants, we will need to implement the two following steps:

Q2) Building the graph Laplacian.

Q3) Computing the Fiedler vector of this Laplacian.

To get a better understanding of how the spectral algorithm works, it is useful to visualise the Fiedler vector in such a simple case.

Q4) In the function, implement the visualisation of the Fiedler vector with its coordinates sorted in ascending order. What do you observe?

The variants that we are going to implement are the three techniques for building the set S evoked in the slide 7 of the lesson, namely the zero threshold, the median threshold, and the sweep cut method.

Q5) Adapt the function signature and format the code to implement the possibility for a user to choose one or the other variants.

Q6) In the Matlab function, implement the zero threshold method. The resulting cut must appear on the Fiedler vector plot.

Q7) In the Matlab function, implement the median threshold method. The resulting cut must appear on the Fiedler vector plot.

Q8a) In the Matlab function, implement the sweep cut method. The resulting cut must appear on the Fiedler vector plot.

Q8b) Implement the visualisation of the isoperimetric ratio values computed when applying this variant, along with the Fiedler vector.

Now we would like to observe how the methods behave on some graphs.

Q9) Create other random graphs by modifying the probability of nodes to be linked, and observe and comment the partitioning returned by the different variants (consistency of the blocks, Cheeger's inequality, etc.).

Q10) Create other random graphs by modifying the number of blocks within the random graph. Observe and discuss the partitioning returned by the different variants (consistency of the blocks, Cheeger's inequality, etc.).

Bonus: Now we would like to observe what happens for two unbalanced blocs.

B1) Adapt your code from Q1 so that the random graph is composed of two blocks of size 666 and 334.

B2) What do you observe when applying the variants on this graph?