TSKS33 Hands-On Session 6

Fall 2024

Version of this document: November 5, 2024

Preparation Work (to be Done Before Coming to the Lab)

- Study the course material and the chapters on random network models (Erös-Renyi, configuration model, and small worlds) in the textbook.

Objectives

The purposes of this lab exercise are to

- experimentally examine a number of networks and their structural properties to gain improved understanding of the theory of complex networks
- experiment with a graph learning algorithm (graphical Lasso), to appreciate the power of complex networks analysis tools when applied to big data

Task 1: Recognize Structural Properties of Networks

Eight networks are given and stored in edge-list format in the files `g1.NET`, `g2.NET`, ..., `g1.NET`. Visualizations using the `sfdp` layout engine are also available in the files `g1.png`, ..., `g8.png`.

To analyze degree distributions and clustering coefficients, you can use the Python script `py_analysis.py`. The results of the script can be imported into any software of choice for further processing. The files `G-clustering-N.csv` have one line per node with three columns: node ID, degree and local clustering coefficient. The files `G-degree-N.csv`

contains the degree distribution (first and second column) and the cumulative degree distribution (first and third column). The Python script also plots the degree distribution (saved in `Deg-Dist-N.png`) and the clustering coefficient (saved in `Deg-Clust-N.png`).

Questions to answer, for each network:

- Describe the structural properties of the network: degree distribution, clustering coefficient and its dependence on the node degree.
- What kind of network is it likely?
  - an Erdös-Renyi (Poisson) network - if so, with what average degree?
  - a scale-free network – if so, with what degree exponent?
  - a regular graph?
  - something else, if so what (to the extent anything can be said)?

All networks in this task are synthetically generated, none of them are from real data.

Task 2: Structural Balance

Investigate whether structural balance seems to hold in the who-trusts-whom online social network of the consumer review site `Epinions.com`[1], by sampling triangles at random, and checking whether they are balanced or not.

The network is directed but we ignore the directions of the links. A small fraction of the links will have inconsistent signs (+ in one direction, − in the other) and we ignore those links.

To sample triangles at random and look at the signs of their edges, you may use the program `structbal.py`.

Questions to answer:

1. Go through the program `structbal.py` and explain how it works, line by line.
2. Is the sampling of triangles in `structbal.py` uniform or not? If not, in what way is it biased?
3. How large fraction of the links, approximately, are positive and negative, respectively?
4. From a quick visual inspection of the output from `structbal.py`, does structural balance seem to hold in this network?

[1] From the SNAP dataset description [1]: This is a who-trust-whom online social network of a general consumer review site `Epinions.com`. Members of the site can decide whether to "trust" (+) or not trust (-) each other. All the trust relationships interact and form the Web of Trust which is then combined with review ratings to determine which reviews are shown to the user.

Task 3: Small-world phenomena

Use the program `small_world.py` to examine the effective diameter for

(a) A circle network with node degree 4 and 1000 nodes

(b) A Watts-Strogatz network based on a circle network with node degree 4, re-wiring probability 0.01, and 1000 nodes

(c) A Watts-Strogatz network based on a circle network with node degree 4, re-wiring probability 0.1, and 1000 nodes

(d) A scale-free network with $\gamma = 2.2$ and 1000 nodes

(e) The Amazon co-purchasing network (from the SNAP datasets)

The program also examines the effective diameter after random rewiring. The degree distribution is also plotted for some of them.

For (a)–(d), increase the number of nodes 100 times to see what happens. Create the following two tables with the generated values (the increment factor is the quotient between the two diameters):

Without rewiring			
Network	Effective Diameter (N=1000)	Effective Diameter (N=100 000)	Increment Factor
Circle	?	?	?
WS 0.01	?	?	?
WS 0.1	?	?	?
Scale-free	?	?	?

After rewiring			
Network	Effective Diameter (N=1000)	Effective Diameter (N=100 000)	Increment Factor
Circle	?	?	?
WS 0.01	?	?	?
WS 0.1	?	?	?
Scale-free	?	?	?

What conclusions can you draw for the networks (a)–(d), are the corresponding worlds small or not, or even"ultra-small"? In each of the four cases, what causes the small-world property (to the extent it is present): is it the fact that the degree distribution is heavy-tailed (well approximated by a power law), or is it some other property of the network? What can you say about (e)?

Task 4: Degree correlations

The program `analyze_networks.py` measures the degree correlation coefficient, ρ_k and plots the degree correlation function, dcf(k) for different networks. It also measures ρ_k and plots dcf(k) after random rewiring of the network.

Questions to answer:

1. Go through the program `analyze_networks.py` and the function `degreecorr.py` and explain what each lines does.
2. Analyze the science collaboration network in Barabasi's dataset (`collaboration.edgelist.txt`) with and without random rewiring and interpret the result.
3. Analyze a $G(500, 100000)$ Poisson random network and interpret the result.

Task 5: Graph learning with the graphical Lasso

The objective of this task is to gain some understanding for what type of analysis that is possible with graph learning algorithms and how they can be used to discover the topology of complex networks from data. Specifically we will use the graphical Lasso solver in the Python library scikit-learn [2] on two real data sets. The solver uses a Lagrange multiplier formulation of the graphical Lasso problem, so the regularization parameter does not have the exact same meaning as in the course material.

Questions:

(a) *Social network from voting records in the United States congress.* The data consist of votes (yes/no) on 699 resolutions by 443 senators in the U.S. congress in 2019, that is $699 \times 443 = 309657$ votes in total. These data are open for anyone to download from the Internet.[2] The Python script `download-US-congress-votes.py` downloads the data and saves them to disk. As this takes a while, the downloaded files are also available under `/courses/TSKS33/ht2024/data/US-congress`. The file format is not compatible with all Python versions, so if you work outside of the LiU labs you may have to run the download script.

Use the program `congress.py` to analyze the voting data using graphical Lasso. The program takes a subset of 200 senators and constructs the network between them based on their voting patterns. Go through the program `congress.py` and explain what each lines does, including what all generated plots represent.

[2] `https://www.govtrack.us/congress/votes`

First inspect the raw votes visually ("spy" in Python). Can anything be said? Import the result of the graphical Lasso into Gephi and visualize the network using the ForceAtlas layout. Are there communities? There is no "ground truth" for the communities, but an expectation is that the clusters match well with the party affiliation of the senators. This affiliation is color coded as (red=republican, blue=democrat). Comment on the result. Experiment with different values of the regularization parameter.

Some heuristics are applied in the analysis. For example, occasional votes are missing and they are simply taken as "no". Also, since we have discrete-valued data the distribution of the data is relatively far from Gaussian. The $\boldsymbol{Z} + \boldsymbol{I}/3$ sample covariance suggested for binary data in [3], and used for example for the animal classification example in [4], is used to deal with this.

(b) *Gene profiles from The Cancer Genome Atlas (TCGA) database.* The data contain gene expression data for 20531 genes for 801 subjects with one out of five identified cancer types, including breast (BRCA), colon (COAD) and kidney (KIRC) [5]. These data can be downloaded from the Internet and used under a Creative Commons license.[3]

Use the program `TCGA.py` to analyze the gene profiles using graphical Lasso. The program takes a subset of subjects and reconstructs the connectivity between them. Layout the resulting network using Gephi with the ForceAtlas layout. The ground truth is color coded according to the five cancer types.

Also go through `TCGA.py` and explain how the program works.

Task 6:

You are done with the TSKS33 lab series, good job. Please tell the teaching assistant which part of the course and/or the lab series that you found most interesting, and/or if you have any suggestions for improvements.

Examination

- Individual oral examination takes place in class (computer lab). Students are also expected to be able to answer questions relating to the course material. All students must study the pertinent course material before coming to the lab.
- Before asking for oral examination, you have to collect all generated plots/figures and answers into a document, for example, in PowerPoint or LibreOffice. Additionally, the code should be open and ready to be run upon request. It is suggested that you have already ran the code once so that the output is in the terminal.

[3] `https://archive.ics.uci.edu/ml/datasets/gene+expression+cancer+RNA-Seq`

- If a reference/sample solution is provided, you should format your solution the same way.
- Collaboration on this homework in small groups is encouraged, but each student should individually demonstrate understanding of all tasks.

References

[1] J. Leskovec and A. Krevl, "SNAP Datasets: Stanford large network dataset collection," http://snap.stanford.edu/data, Jun. 2014.

[2] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[3] O. Banerjee, L. E. Ghaoui, and A. d'Aspremont, "Model selection through sparse maximum likelihood estimation for multivariate Gaussian or binary data," *Journal of Machine learning research*, vol. 9, no. Mar, pp. 485–516, 2008.

[4] H. E. Egilmez, E. Pavez, and A. Ortega, "Graph learning from data under Laplacian and structural constraints," *IEEE Journal of Selected Topics in Signal Processing*, vol. 11, no. 6, pp. 825–841, 2017.

[5] J. N. Weinstein, E. A. Collisson, G. B. Mills, K. R. M. Shaw, B. A. Ozenberger, K. Ellrott, I. Shmulevich, C. Sander, J. M. Stuart, C. G. A. R. Network *et al.*, "The cancer genome atlas pan-cancer analysis project," *Nature genetics*, vol. 45, no. 10, p. 1113, 2013.