

Assignment 1:

Estimating a quantity from free probability theory within random graphs

2WB50 – Stochastic Simulation

Jaron Sanders

February 6, 2023

1 Introduction

Network science is an academic field which studies complex networks such as telecommunication networks, computer networks, biological networks, cognitive and semantic networks, and social networks, considering distinct elements or actors represented by nodes (or vertices) and the connections between the elements or actors as links (or edges). Network science draws on theories and methods from many fields, including e.g. from fields that work on *random graphs* and *random matrices*.

From a mathematical perspective, random graphs are used to answer questions about the properties of typical networks. Probability theorists in particular routinely seek *concentration inequalities* for different properties of random graphs. Concentration inequalities namely provide bounds on how a random variable deviates from some value (typically, its expected value). The law of large numbers of classical probability theory is one example: it states that sums of independent random variables are, under very mild conditions, close to their expectation with large probability.

1.1 Recent, state-of-the-art developments

New concentration inequalities for the spectral norm

$$\|S_n\|_2 := \max_{\|v\|_2 \neq 0} \frac{\|S_n v\|_2}{\|v\|_2} = \lambda_{\max}(S_n) \quad (1)$$

of a sum of dependent matrices

$$S_n := \sum_{i=1}^n X_i \quad (2)$$

were recently found. Here, $\|v\|_2 = (\sum_{i=1}^d |v_i|^2)^{1/2}$ denotes the Euclidean norm of a real vector $v = (v_1, \dots, v_d)$; $\lambda_{\max}(S_n)$ denotes the largest eigenvalue of the matrix $S_n = (S_{n,i,j})_{i=1,j=1}^d$; ¹ and the X_1, \dots, X_n can be any sequence of random, real, symmetric $d \times d$ matrices (not necessarily independent). These matrices could e.g. be *adjacency matrices* of random graphs (which you will study after Section 3).

Specifically, it has been identified that *free probability theory* allows one to derive tight concentration inequalities for the spectral norm in (1). A so-called *free model* $S_{n,\text{free}}$ can be associated to S_n , and a quantity $\|S_{n,\text{free}}\|$ then appears within tight concentration inequalities for $\|S_n\|_2$. For instance, these concentration inequalities imply that

$$\mathbb{E}[\|S_n\|_2] \leq \|S_{n,\text{free}}\| + \varepsilon_d \quad (3)$$

for some (asymptotically) small ε_d .

Now, don't worry! We will neither ask you to study free probability theory nor to construct a free model. Doing that is quite nontrivial: it would require knowledge of C^* -algebras. But when given a mathematical relation such as

$$\|S_{n,\text{free}}\| = \max_{\eta \in \{+1, -1\}} \inf_{Z \succ 0} \lambda_{\max}(Z^{-1} + \eta \mathbb{E}[S_n] + \mathbb{E}[(S_n - \mathbb{E}[S_n])^T Z (S_n - \mathbb{E}[S_n])]), \quad (4)$$

you *can* estimate $\|S_{n,\text{free}}\|$ using basic simulation techniques. Here, the infimum is taken over all positive definite matrices;² B^{-1} denotes the *matrix inverse* of a matrix B , which is a matrix such that $B^{-1}B = I_d$;³ C^T denotes

¹A value $\lambda \in \mathbb{R}$ is an eigenvalue of a matrix A if there exist a vector v_λ such that $Av_\lambda = \lambda v_\lambda$.

²A real matrix $Z \in \mathbb{R}^{d \times d}$ is *positive definite* denoted by $Z \succ 0$, if and only if $z^T Z z > 0$ for every nonzero real column vector $z \in \mathbb{R}^d$.

³Here, I_d denotes the *identity matrix* of size $d \times d$.

the *transpose* of a matrix C , whose entries satisfy $(C^T)_{i,j} = C_{j,i}$; and for any random matrix E , $\mathbb{E}[E]$ denotes the matrix with entries $(\mathbb{E}[E_{i,j}])_{i,j}$.

Remark. Throughout this assignment, you may use a Python library that can represent matrices as data objects, that can do elementary algebraic operations with matrices such as summing and multiplying, that can calculate eigenvalues, and that can numerically invert matrices. We recommend NUMPY.

2 Objective

As an exercise into generating random variates and Monte Carlo methods, you will estimate eigenvalue distributions and $\|S_{n,\text{free}}\|$ for so-called Erdős–Rényi random graphs.

2.1 Related literature

This assignment takes inspiration from recent research. In particular, we have looked at *F. Lehner*, “Computing norms of free operators with matrix coefficients,” 1999, and *A.S. Bandeira, M.T. Boedihardjo, R. van Handel*, “Matrix concentration inequalities and free probability,” 2021. Chapter 1 of *J.A. Tropp*, “An introduction to matrix concentration inequalities,” 2014 discusses further history and other applications.

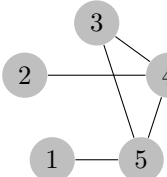
3 Model: Adjacency matrices of Erdős–Rényi random graphs

Assume that a size $d \in \mathbb{N}_+$, and map $p : \mathbb{N}_+ \rightarrow (0, 1)$ are given.

We will let $A \in \{0, 1\}^{d \times d}$ denote an *adjacency matrix* of an *undirected Erdős–Rényi random graph* $\mathcal{G}(d, p(d))$. This means that for $i > j$,

$$A_{i,j} = A_{j,i} \stackrel{(d)}{=} \text{Bernoulli}(p(d)), \quad (5)$$

and that for $i \in \{1, \dots, d\}$, $A_{i,i} = 0$. Here, we understand that two *vertices* $i, j \in \{1, \dots, d\}$ are *connected by an edge* if and only if $A_{i,j} = 1$. For example,

$$\text{the adjacency matrix } A = \begin{pmatrix} 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 & 0 \end{pmatrix} \quad \text{encodes the undirected graph} \quad (6)$$


In other words: each edge in an Erdős–Rényi random graph $\mathcal{G}(d, p(d))$ is present with probability $p(d)$ independently of all other edges.

3.1 Visualization

Exercise 1. Code a function that given an adjacency matrix A , visualizes the associated graph. You may use a Python library to visualize graphs. An option could be NETWORKX, maybe in combination with PLOTLY.

Exercise 2. Code a function that given d and $p(d)$, visualizes a histogram of the eigenvalues of the centered matrix $A - p(d)(J_d - I_d)$. Here, J_d denotes the *all-ones matrix* of size $d \times d$. You may use a Python library to calculate eigenvalues and histograms. We recommend NUMPY and MATPLOTLIB.

3.2 Estimation

From here on out, we will assume that $X_i \stackrel{(d)}{=} A$ for $i = 1, \dots, n$.

Exercise 3. Code a function that, given $n \in \mathbb{N}_+$, calculates a numerical estimate of the expectation $\mathbb{E}[S_n]$ via simulation. We will refer to such estimate from now on as $\hat{\mu}_n$. Validate your function by comparing $\hat{\mu}_n/n$ to the fact that $\mathbb{E}[S_n]/n = p(d)(J_d - I_d)$. You may not use a Python library for estimating means or covariances.

Exercise 4. Code a function that, given $n \in \mathbb{N}_+$, a positive definite matrix Z , and estimate $\hat{\mu}_n$, calculates an estimate of the expectation $\mathbb{E}[(S_n - \hat{\mu}_n)^T Z (S_n - \hat{\mu}_n)]$. We will refer to such estimate from now on as $\hat{\nu}_n$. You may not use a Python library for estimating means or covariances.

Exercise 5. Code a function that, given a positive definite matrix Z , integer η , and estimates $\hat{\mu}_n, \hat{\nu}_n$, calculates $\lambda_{\max}(Z^{-1} + \eta\hat{\mu}_n + \hat{\nu}_n)$. You may use a Python library to calculate eigenvalues and matrix inverses. We recommend `NUMPY`.⁴

4 Sampling positive definite matrices

Next, you will need to randomly generate positive definite matrices Z_1, \dots, Z_m . The reason is that

$$\|S_{n,\text{free}}\| \leq \max_{\eta \in \{+1, -1\}} \min \left\{ \lambda_{\max}(Z_1^{-1} + \eta \mathbb{E}[S_n] + \mathbb{E}[(S_n - \mathbb{E}[S_n])^T Z_1 (S_n - \mathbb{E}[S_n])]), \dots, \lambda_{\max}(Z_m^{-1} + \eta \mathbb{E}[S_n] + \mathbb{E}[(S_n - \mathbb{E}[S_n])^T Z_m (S_n - \mathbb{E}[S_n])]) \right\} \quad (7)$$

with probability one. Furthermore, we can expect that the right-hand side tends to $\|S_{n,\text{free}}\|$ as $m \rightarrow \infty$ if any positive definite matrix has a positive probability of being generated.

We will tell you how to generate positive definite matrices using the *Wishart distribution* $\text{Wishart}_d(\Sigma, d)$. These distributions are of great importance in the estimation of covariance matrices in multivariate statistics. Here, $\Sigma \in (0, \infty)^{d \times d}$ can be any real, positive definite matrix with strictly positive entries.

Exercise 6. Code a function that, given d and Σ , computes the following: for $i = 1, \dots, d$, generate random vectors $G_i = (g_i^1, \dots, g_i^d)^T \stackrel{(d)}{\equiv} \text{MultivariateNormal}_d(0, \Sigma)$, and return $Z = \sum_{i=1}^d G_i G_i^T$. Then $Z \stackrel{(d)}{\equiv} \text{Wishart}_d(\Sigma, d)$. You may use a Python library to draw samples from a multivariate normal distribution; you may not use a Python library to draw directly from a Wishart distribution.

We will also challenge you to come up with your own method that randomly generates positive definite matrices. Feel free to be inspired by outside sources.

Exercise 7. Describe and implement your own method of generating positive definite matrices, that is distinct from the method described in Exercise 6 based on Wishart distributions. Except for generating random numbers from $\text{Uniform}[0, 1]$, you may not use a Python library for other random variate generation.

Hint. A matrix is positive definite if and only if all of its eigenvalues are strictly positive.

5 Deliverable

Combine what you have learned in Sections 1 to 4, and write a program that:

1. Given d, p , visualizes a histogram of the eigenvalues of an adjacency matrix of an Erdős–Rényi random graph.
2. Given d, m, n, p, Σ , estimates a good upper bound for $\|S_{n,\text{free}}\|$ using Exercise 6.
3. Given d, m, n, p , estimates a good upper bound for $\|S_{n,\text{free}}\|$ using Exercise 7.

Also program several Python tests that:

1. Check (mathematical) validity / accuracy of the functions that you programmed in Exercises 3–7.

Then:

- a. For $d = 20$ and $q = 0.3$, visualize an Erdős–Rényi random graph with $p(d) = q$.
- b. For $d \in \{100, 1000\}$ and $q \in \{0.2, 0.5, 0.8\}$, visualize histograms of eigenvalues of the matrices $A - p(d)(J_d - I_d)$, where each time $A \in \{0, 1\}^{d \times d}$ is the adjacency matrix of an Erdős–Rényi random graph with $p(d) = q$.
- c. For $d = 10, n = 1, q = 0.7, \Sigma = I_d$, conduct an experiment to choose a suitable sample size m for Tasks 2 and 3 (those based on Exercises 6 and 7), when the X_1, \dots, X_n are adjacency matrices of Erdős–Rényi random graphs with $p(d) = q$. Comment on which implementation is more efficient as a function of m .
- d. For $d = 10, m = 100, \Sigma = I_d, n \in \{1, 10, 100\}, q \in \{0.05, 0.15, \dots, 0.95\}$, create a 3×9 table displaying your upper bound for $\|S_{n,\text{free}}\|$ with confidence intervals as a function of (n, p) , when the X_1, \dots, X_n are adjacency matrices of Erdős–Rényi random graphs with $p(d) = q$.⁵

⁴If you would like to test your code, consider that the identity matrix I_d is an example of a positive definite matrix. Another example is $M^T M$, where M can be any real invertible matrix.

⁵When calculating a confidence interval or confidence band, you may at that moment use a Python library for calculating means or variances.

- e. Make sure that you create enough independent replications of your simulation so that all values within your tables have at least **two significant digits**, e.g. 1.2 ± 0.3 or 45 ± 6 . We consider *three* significant digits very impressive, e.g. 7.89 ± 0.01 , but this requires efficient programming. Here, the first number indicates your estimated sample mean and the second number indicates your confidence interval.
- f. For $d = 100$, let A be the adjacency matrix of an Erdős–Rényi random graph with $p(d) = (\ln d)^\alpha / d$. For $\alpha \in (0, 4)$, estimate $\mathbb{E}[\|A - p(d)(J_d - I_d)\|_2]$ and $\|(A - p(d)(J_d - I_d))_{\text{free}}\|$. Let us call your estimates $\hat{\xi}_\alpha$ and $\hat{\zeta}_\alpha$, respectively. Create $(\alpha, \hat{\xi}_\alpha)$ - and $(\alpha, \hat{\zeta}_\alpha)$ -plots with coinidence bands.
- g. Conduct one extra scientific experiment on either Erdős–Rényi random graphs or (sums of) random matrices. **You may choose what to investigate!** Try to consult a wide array of sources for inspiration / background, and cite several. Here are a few *suggestions*, which you may freely ignore:
 - depending on whether $p(d) < \ln d/d$, $p(d) = \ln d/d$, or $p(d) > \ln d/d$, a large Erdős–Rényi random graph ($d \rightarrow \infty$) can either have, or not have, one unique ‘giant component’;
 - surprisingly, the empirical eigenvalue distribution of a large random matrix is *almost always* semicircular, irrespective of the underlying distribution;
 - however, the empirical eigenvalue distribution of random matrices drawn from the Wishart distribution have a related but slightly different shape.

Write a report, preferably in L^AT_EX, and hand in a PDF detailing your implementation and findings. The report must contain at least the following:

- A brief motivation of the problem and a short description of the model.
- A detailed description of your implementations. Use pseudo-code to most effectively communicate your implementation within the report.
- Well-designed and clear plots and tables. The more scientifically interesting your plots are, the better!
- A brief discussion and conclusion based on your plots.
- Exact copy of your code in the Appendix. Code in the Appendix does not count towards a page limit.

5.1 Knockout criteria

Your report and code **must** meet the following requirements in order to be marked:

1. An electronic copy must be submitted in Canvas (PDF), on time.
2. It includes a title, author names, student numbers, and bibliography when citing.
3. It includes a paragraph in the Appendix, in which every student’s specific contributions are explained. In particular, tell us which percentage of (a) programming and (b) report writing was done by each student. Each student is expected to contribute significantly to both.
4. There is a hard page limit of **eight A4 pages**, single column. This excludes code in the Appendix, the bibliography, and project contribution statement.
Focus on what is important: it is about quality of content, not quantity.
5. Margins should be at least 1 cm, and font size at least 10 pt.
6. The overall presentation is clean / neat / orderly.
7. Your texts are legible, in English, and contain few spelling mistakes.

5.2 Grading

We will be judging for effort and you may report difficulties or even failure. The reporting however, has to be detailed and genuine. Your implementations / attempts have to be critically reported.