Math 126 Problem 18: Finding Cliques

Spring 2027

Submission Instructions

Please upload your work to Gradescope by **11:59pm on Monday, April 12, 2027 (Pacific time).**

- You are encouraged to discuss ideas and work with your classmates. However, you *must acknowledge* your collaborators at the top of each solution on which you collaborated with others and you *must write* your solutions independently.
- Your solutions to theory questions must be written legibly, or typeset in LaTeX or markdown.
- I recommend that you write your solutions in a Python notebook using Google Colab.
- You should submit your solutions as a *single PDF* via the assignment on Gradescope.

Grading: The point of the problem set is for *you* to learn. To this end, I hope to disincentivize the use of LLMs by *not* grading your work for correctness. Instead, you will grade your own work by comparing it to my solutions. This self-grade is due the Friday *after* the problem set is due, also on Gradescope.
Original notebook: Google Colab.

Finding Cliques

A common task in data mining is to identify large *cliques* in a graph. For example, in social networks, large cliques can be indicators of fraudulent accounts or networks of accounts designed to promote certain content. In this problem, we consider a spectral heuristic for finding a large clique based on the top eigenvector of the graph adjacency matrix $\mathbf{A}$:

- Compute the leading eigenvector $\mathbf{v}_1$ of $\mathbf{A}$.
- Let $i_1, \ldots, i_k \in \{1, \ldots, n\}$ be the indices of the k entries in $\mathbf{v}_1$ with the largest absolute value.
- Check if nodes $i_1, \ldots, i_k$ form a k-clique.

We will analyze this heuristic on a natural random graph model. Specifically, let G be an Erdős-Rényi random graph: we start with n nodes, and for every pair of nodes (i, j), we add an edge between the pair with probability $p < 1$. To simplify the math, also assume that we add a self-loop at every vertex i with probability p. Then, choose a fixed subset S of k nodes to form a clique. Connect all nodes in S with edges and add self-loops. We will argue that, for sufficiently large k, we can expect the heuristic above to identify the nodes in the clique.

Part A: Expected Adjacency

Let $\mathbf{A}$ be the adjacency matrix of a random graph generated as above. What is $\mathbb{E}[\mathbf{A}]$? Then write $\mathbb{E}[\mathbf{A}]$ in terms of its eigenvalues and eigenvectors (you don't need to find the exact algebraic expressions for these!).

Hint: Argue that, up to multiplying by a constant, any eigenvector $\mathbf{v}$ corresponding to a non-zero eigenvalue must have $v_i = 1$ for all $i \notin S$ and $v_i = \alpha$ for all $i \in S$, where α is a constant.

Part B: Random Adjacency

To prove the algorithm works, it is possible to use a matrix concentration inequality to argue that the top eigenvector of $\mathbf{A}$ is close to that of $\mathbb{E}[\mathbf{A}]$. Instead of doing that, let's verify things experimentally. Generate a graph G according to the prescribed model with $n = 900$, $k = |S| = 30$, and $p = 0.1$. Compute the top eigenvector of $\mathbf{A}$ and look at its 30 largest entries in magnitude. What fraction of nodes in the clique S are among these 30 entries? Repeat the experiment 100 times and report the average fraction recovered.