Finding Planted Clique in Erdős–Rényi Random Graphs: Improving previous methods and expanding applications

Megan Sochinski
Megan.Sochinski@Colorado.EDU

Follow this and additional works at: https://scholar.colorado.edu/honr_theses

Part of the Discrete Mathematics and Combinatorics Commons, Other Mathematics Commons, and the Probability Commons

Recommended Citation
https://scholar.colorado.edu/honr_theses/1623

This Thesis is brought to you for free and open access by Honors Program at CU Scholar. It has been accepted for inclusion in Undergraduate Honors Theses by an authorized administrator of CU Scholar. For more information, please contact cuscholaradmin@colorado.edu.
Finding Planted Cliques in Erdős–Rényi Random Graphs

Improving previous methods and expanding applications

Thesis Written By: Megan Sochinski

Thesis Directed By: Dr. Sean O’Rourke, Department of Mathematics

Honors Council Representative: Dr. Magdalena Czubek, Department of Mathematics

Outside Representative: Dr. Rafael Frongillo, Department of Computer Science

A thesis presented for consideration for Latin honors for the degree of Bachelors of Mathematics

Department of Mathematics
University of Colorado, Boulder
April 9, 2018
Abstract
In this paper, we will discuss new methods for finding planted cliques within Erdős–Rényi random graphs. An Erdős–Rényi random graph is a graph with \( n \) vertices, where each vertex is connected to each other vertex with some probability \( p \), independent of all other choices. The planted clique problem asks us to find the most efficient way to find a planted clique in an Erdős–Rényi random graph. A planted clique is a secretly chosen set of vertices in the graph that are purposefully connected with edges added to the graph until all of the selected vertices are connected. There are many other similar problems which have been posed and examined, regarding finding patterns in random graphs or matrices. This paper will begin briefly by discussing previous methods for finding planted cliques in Erdős–Rényi random graphs. It will then present two algorithms for finding planted cliques, one which finds any number of disjoint cliques, and another which finds only one planted clique. Disjoint implies that none of the points can be included in more than one clique. The paper will then prove theorems that state that these algorithms will find such planted cliques, under certain assumptions on some parameters. These theorems are the main results of the paper, and the algorithms can be run in polynomial time. Finally, the paper will discuss the improvements made using these algorithms over previous methods, such as the fact that it is no longer necessary to know the size of a planted clique to find it, and suggest areas of future research.
Contents

1 Main Results and Algorithms  3

2 Proof of Theorem 1.1  4
   2.1 Converting the graph to an adjacency matrix and calculating the eigenvalues and eigenvectors.  5
   2.2 Examining the eigenvalues and eigenvectors of the adjacency matrix  6
   2.3 Finding an estimate of k  8
   2.4 Using the eigenvector to make an estimate for S  11
   2.5 Clean up of the algorithm  13

3 Case with only one clique  14

4 Simulations  14

5 Conclusion  16

A Example Code  18

List of Figures

1 Proportion of successful trials for $p = .3$, $k = 7\sqrt{np}$, $n = \{100, 125, 150, 175, 200, 225, 250, 275\}$ and 100 trials at each value of $n$.  15

2 Proportion of successful trials for $p = .5$, $k = 6\sqrt{np}$, $n = \{100, 125, 150, 175, 200, 225, 250, 275\}$ and 100 trials at each value of $n$.  16

List of Algorithms

1 Finding a hidden clique within a random graph  4

2 Finding multiple disconnected hidden cliques within a random graph  5
1 Main Results and Algorithms

In an Erdős–Rényi random graph, there are \( n \) vertices, where any two vertices are possibly connected by an edge with a probability \( p \) independent of other choices. A planted clique is formed when some number of vertices in the graph are deliberately made into a clique by adding connections between the chosen vertices until all of them are connected to each other. One question that has been examined in the past is the planted clique problem, where the challenge arises from attempting to find a planted clique as efficiently as possible within an Erdős–Rényi random graph. The most rudimentary way to find such a clique is to have a computer examine all subsets of vertices in the graph to determine if that subset is a clique, but this method requires exponential calculations as \( n \) gets larger, so it is an impractical method of solving the problem. In 1998, Alon, Krivelevich, and Sudakov published a paper examining how to find potential planted cliques within these Erdős–Rényi random graphs using an algorithm that looked at the adjacency matrix of the random graph and used that matrix’s eigenvalues and eigenvectors to determine an estimate for the clique \([4]\). Although this method reduces the calculations needed to a polynomial level of work, it still has some restrictions. For example, the size of the clique needs to be known in order to find it, the probability of the vertices being connected is also known and restricted to \( \frac{1}{2} \), and the method can only find one planted clique. There has also been a lot of research about using eigenvalues and eigenvectors in order to find patterns in random graphs, for example, stochastic block models tend to create graphs with communities, a group which is like a clique, but just has a high density of edges and is not required to have all the vertices connected \([1]\). For examples of previous work, see any of the following: \([1][2][3][5][7][8][10][11][12][13][15][17]\). The paper aims to expand on current methods by reducing the amount of information required to find a planted clique and by being able to find multiple cliques in the same graph, assuming certain restrictions are met on some parameters.

The main results of this paper include two algorithms that can be used to find planted cliques within Erdős–Rényi random graphs. Algorithm 1 will find, with probability tending to 1 as \( n \) approaches infinity, any single planted clique as long as \( k \), the size of the clique, is larger than \( C\sqrt{np} \) and \( p \) is between \( \frac{\ln n}{n} \) and \( \frac{1}{2} \). Algorithm 2 will find up to \( m \) cliques, as long as the size of the smallest clique is at least \( C\sqrt{np} \), and each larger clique is at least the size of the previous clique plus \( C\sqrt{i\sqrt{np}} \), so that \( k_i \geq C\sqrt{i\sqrt{np}} \), \( p \) holds to the same restrictions as in the previous case, \( m \leq \sqrt{\frac{\frac{n}{16C^2\ln(n)}}{\ln(n)}} \), and the cliques are disjoint, so no vertex is shared between any two planted cliques. \( C \) is an absolute constant.

**Theorem 1.1** Given an Erdős–Rényi random graph \( G \) with vertices indexed by the set \( \{1, \ldots, n\} \), and probability \( \frac{\ln(n)}{n} \leq p \leq \frac{1}{2} \), and \( S_1, \ldots, S_m \) perfectly disjoint cliques of sizes \( k_1, \ldots, k_m \) respectively, where \( k_m \geq C\sqrt{np} \), \( k_i \geq k_{i+1} + C\sqrt{i\sqrt{np}} \), and \( 1 \leq m \leq \sqrt{\frac{np}{16C^2\ln(n)}} \), where \( C \) is some absolute constant, Algorithm 2 will find \( m \) sets \( Q_i \), with \( i \in \{1, \ldots, m\} \), where \( Q_1 = S_1, \ldots, Q_m = S_m \) within \( G \), with
Algorithm 1 Finding a hidden clique within a random graph

Require: A graph $G$ on the vertex set $\{1, ..., n\}$ and a probability $\frac{\ln^5(n)}{n} \leq p \leq \frac{1}{2}$

1. Transform $G$ into a signed adjacency matrix $B$ using the formula:
   
   \[
   \text{For any } i, j \in \{1, ..., n\}, b_{ij} = \begin{cases} 
   1 - p & i \sim j, \text{or } i = j \\
   -p & i \not\sim j 
   \end{cases}
   \]

2. Calculate largest eigenvalue, $\lambda_1(B)$, of $B$ and its corresponding eigenvector $v_1(B)$.

3. Define $\hat{k} = \frac{\lambda_1(B)}{1 - p}$

4. Define $T = \{i : |(v_1(B))_i| \geq \epsilon\}$, where $\epsilon = \frac{1}{2\sqrt{k}}$

5. For any $r \in \{1, ..., n\}$, define the neighborhood of $r$ to be $N(r) = \{t : t \sim r\}$, and $Q = \{r : |N(r) \cap T| \geq 0.65\hat{k}\}$

6. Output: $Q$

Corollary 1.2 Given an Erdős–Rényi random graph $G$ with vertices indexed by the set $\{1, ..., n\}$, and probability $\frac{\ln^5(n)}{n} \leq p \leq \frac{1}{2}$, and a planted clique $S$ of size $k \geq C\sqrt{np}$, where $C$ is some absolute constant, Algorithm 1 will find, with probability tending to 1 as $n$ approaches infinity, a set $Q = S$ of size $k$ within $G$, given that $S$ is the only sufficiently large clique in the graph.

This paper will start by proving the results of Theorem 1.1, then it will show that Corollary 1.2 follows directly from Theorem 1.1. It will examine some computer simulations of the algorithms in practice. Finally, it will discuss future areas of research that should follow this work.

2 Proof of Theorem 1.1

The proof of this theorem requires that each section of the algorithm be analyzed individually. As such, the next several sections will break down the proof of this theorem.
Algorithm 2  Finding multiple disconnected hidden cliques within a random graph

Require: A graph $G$ on the vertex set $\{1, ..., n\}$ with $m$ up to $\sqrt{\frac{3np}{16C^2\ln(n)}}$ disjoint cliques and a probability $\frac{\ln(n)}{n} \leq p \leq \frac{1}{2}$

1. Transform $G$ into a signed adjacency matrix $B$ using the formula:
   
   $\text{For any } i, j \in \{1...n\}, b_{ij} = \begin{cases} 1 - p & i \sim j, \text{or } i = j \\ -p & i \not\sim j \end{cases}$

2. Let $\lambda_1(B), \lambda_2(B), ..., \lambda_n(B)$ be the ordered eigenvalues of $B$, such that $\lambda_1(B)$ is the largest and $\lambda_n(B)$ is the smallest. Consider the $m$ largest, $\lambda_1(B), ..., \lambda_m(B)$

3. Define $\hat{k}_i = \frac{\lambda_i(B)}{1 - p}$

4. Define $T_i = \{j : |(v_i(B))_j| \geq \epsilon\}$, where $\epsilon = \frac{1}{2\sqrt{\hat{k}_i}}$

5. For any $r \in \{1...n\}$, define the neighborhood of $r$ to be $N(r) = \{t : t \sim r\}$, and $Q_i = \{r : |N(r) \cap T| \geq .655\hat{k}_i\}$

6. Output: $Q_1, Q_2, ..., Q_m$

2.1 Converting the graph to an adjacency matrix and calculating the eigenvalues and eigenvectors.

In this step, the random graph is converted into an adjacency matrix, denoted $B$ using the following function:

$\text{For any } i, j \in \{1,...,n\} : b_{ij} = \begin{cases} 1 - p & i \sim j, \text{or } i = j \\ -p & i \not\sim j \end{cases}$

This adjacency matrix encodes the state of the random graph by using two different values, one when two points are connected to each other, and another when they are not connected. These values are chosen for computational ease later in the proof.

Note: $B$ is a symmetric, square matrix. The values of $i$ and $j$ are bounded by 1 and $n$, since they cannot take any value that is not represented in the original random graph. Therefore, $B$ is an $n$ by $n$ matrix. $b_{ij} = b_{ji}$ for all $i, j \in V$, since if $i \sim j$, then $j \sim i$ and so forth.

Lemma 2.1 $B$ can be rearranged using a permutation matrix $P$, so that $PBPT^T$ represents the matrix where the cliques are represented by a group of matrices on the diagonal where all entries are $1 - p$, and the rest of the matrix is randomly filled with either $-p$ or $1 - p$ according to their random connections. The eigenvalues and eigenvectors of $PBPT^T$ are the same as the eigenvalues and
eigenvectors of $B$, up to rearrangement of entries in the eigenvectors.

Proof: In order to switch the rows of $B$, the permutation matrix $P$ is simply the identity matrix with the rows switched where the rows of $B$ need to be switched. $P^T$ will do the same to the columns of $B$. Therefore a $P$ can be written such that $P(B)P^T$ will rearrange the matrix $B$ such that the diagonal matrices starting at the top left corner represent only edges that are part of a clique. From this point forwards, this proof will assume that $B$ has been rearranged as such.

2.2 Examining the eigenvalues and eigenvectors of the adjacency matrix

Note: The matrix $B$ can be separated into two matrices. The first of these is $A$ which should include only the entries representing edges in any of the cliques. It should be a block diagonal matrix with $m$ matrices with entries $1 - p$ following the diagonal, and all other entries being 0. The other matrix, $E$, should have zeros in the same place as the diagonal matrices from $A$, and the other entries should be either $1 - p$ or $-p$ according to the edges in the Erdős–Rényi random graph. It should be obvious that the matrix $A$ plus the matrix $E$ is equivalent to $B$, since they contain disjoint parts of $B$ with zeros in all other locations.

Theorem 2.2 The non-zero-eigenvalues of $A$ are $k_i(1 - p)$.

Proof: $A$ is rank $m$, because there are only $m$ different linearly independent columns of $A$ (containing only entries of $1-p$), so $A$ has only $m$ linearly independent columns. Since $A$ is rank $m$, it only has $m$ non-zero eigenvalues.

Claim: Those eigenvalues are $k_i(1 - p)$, with eigenvectors of $(0, ..., 0, \frac{1}{\sqrt{k_i}}, ..., \frac{1}{\sqrt{k_i}}, 0, ..., 0)$ with $k_i$ entries being $\frac{1}{\sqrt{k_i}}$ and the remaining entries being 0.

For any eigenvalue of $A$, $A \ast v_i$ (the eigenvector corresponding to the eigenvalue $\lambda_i$) is equal to $\lambda_i v_i$. $A$ is a matrix of the form $\text{diag}(C_1, C_2, ..., C_m, 0)$, where $C_1, ..., C_m$ are square matrices of size $k_1, ..., k_m$ respectively, representing the $m$ cliques in the graph.

For any clique, the eigenvector $(0, ..., 0, \frac{1}{\sqrt{k_i}}, ..., \frac{1}{\sqrt{k_i}}, 0, ..., 0)$ will disregard any entries from other cliques and only multiply the entries in the clique by $\frac{1}{\sqrt{k_i}}, ..., \frac{1}{\sqrt{k_i}}$, meaning $A \ast v_i$ is equal to:

$(0, ..., 0, \frac{1}{\sqrt{k_i}} \ast k_i \ast (1 - p), ..., \frac{1}{\sqrt{k_i}} \ast k_i \ast (1 - p), 0, ..., 0)$,

since all the non-zero values will be multiplied by $(1 - p)$ $k_i$ times.

$\lambda_i v_i$ is also equal to $(0, ..., 0, \frac{1}{\sqrt{k_i}} \ast k_i \ast (1 - p), ..., \frac{1}{\sqrt{k_i}} \ast k_i \ast (1 - p), 0, ..., 0)$, so these eigenvalues and eigenvectors are correct.

Lemma 2.3 $\|E\| \leq 3\sqrt{np}$ with probability $1 - o(1)$.

Proof: To prove, we use Theorem 1.4 from [16]. First, check that $E$ meets all the requirements.

First, $|e_{ij}| \leq K$ for some $K$. Clearly, all points are less than 1, since they are either $1-p$, $-p$, or 0.
Next, all values not on the diagonal have an expectation of 0. This is true since all values are either 0 or random variables with the probability distribution:

\[ e_{ij} = \begin{cases} 1-p & i \sim j, \text{or} \\ -p & i \not\sim j \end{cases} \]

Therefore the expected value is

\[ E[e_{ij}] = (1-p)(P[e_{ij}] = 1-p) + (-p)(P[e_{ij}] = -p) = (1-p)(p) + (-p)(1-p) = 0 \]

Next, check that \( \text{Var}(e_{ij}) \leq \sigma^2 \), for all \( i,j \).

The variance of each \( e_{ij} \) is either this value or zero.

Finally, check that \( \sigma \geq C' \sqrt{n} \ln^2(n) \), where \( C' \) is an arbitrary constant.

This is true since \( \sigma = \sqrt{p(1-p)} \geq \sqrt{p} \sqrt{\frac{1}{2}} \geq \frac{\sqrt{\ln^2(n)\sqrt{2}}}{\sqrt{n}} \geq \frac{C' \ln^2(n)}{\sqrt{n}} \).

This is true as long as \( \sqrt{\frac{1}{2} \ln^2(n)} \geq C' \) as \( n \to \infty \), which is true since \( \ln^2(n) \to \infty \) as \( n \to \infty \).

Therefore, Theorem 1.4 from [16] can be used; and \( \|E\| \leq 2\sigma \sqrt{np} + C(k\sigma)^\frac{1}{2} \ln(n) \) with probability \( 1-o(1) \).

Therefore, with high probability, \( \|E\| \leq 2\sqrt{p(1-p)} \sqrt{n} + C(1-p)^{\frac{1}{4}} n^{\frac{1}{4}} \ln(n) \)

\[ \leq 2\sqrt{np} + \sqrt{np} \ln(n) \leq 3\sqrt{np} \]

This is true since \( \ln(n) \leq \frac{C}{\ln(n)} \sqrt{np} \), so \( C \sqrt{np} \ln(n) \leq \frac{C^2}{\ln(n)} \sqrt{np} \leq \sqrt{np} \) as \( n \) approaches infinity.

\[ \frac{C}{\ln(n)} \sqrt{np} \geq \frac{(\ln(n))^{\frac{1}{2}} C}{\ln(n)} \]

\[ \geq \frac{(\ln(n))^{\frac{1}{2}} C}{\ln(n)} \]

\[ \geq \ln(n) \]

In the first step, the inequality is bounded by the minimum value of \( \sqrt{np} \). The second step is true for \( n \) large enough such that \( \ln(n) \leq (\ln(n))^\frac{1}{10} \).

**Lemma 2.4** If \( k_m \geq C \sqrt{np} \) and \( k_i \geq k_{i+1} + C \sqrt{np} \), then \( \lambda_m(B) \geq \|E\| \), and \( \lambda_i(B) \geq \lambda_{i+1}(B) \)

**Proof**: Weyl’s Inequality [14] states that \( |\lambda_i(A + E) - \lambda_i(A)| \leq \|E\| \) with
probability $1 - o(1)$, where all eigenvalues are ordered from largest to smallest. Therefore $|\lambda_i(B) - k_i(1-p)| \leq 3\sqrt{np}$ with probability $1 - o(1)$, if $i \leq m$ and $|\lambda_i(B)| \leq 3\sqrt{np}$ if $i > m$. Therefore, the smallest non-zero eigenvalue, $\lambda_m(B)$, can be easily distinguished from all $\lambda_i$ where $i > m$, since $|\lambda_m(B) - k_m(1-p)| \leq 3\sqrt{np}$, and $|\lambda_{m+1}(B)| \leq 3\sqrt{np}$. Since $k_m \geq C\sqrt{np}$, we can assume for some $C$ that $k_m(1-p) \geq 3\sqrt{np}$. Since $k_m(1-p) \geq 3\sqrt{np}$, and $|\lambda_m(B) - k_m(1-p)| \leq 3\sqrt{np}$, then $\lambda_m(B) > 3\sqrt{np}$. And since $\lambda_m(B) > 3\sqrt{np}$, and $|\lambda_{m+1}(B)| \leq 3\sqrt{np}$, $\lambda_m(B) > \lambda_{m+1}(B)$.

Similarly, any eigenvalue can be distinguished from the next largest eigenvalue as long as $k_i(1-p) \geq k_{i+1}(1-p) + C\sqrt{i}\sqrt{np}$ because $\lambda_i(B) > \lambda_{i+1}(B)$. The values will only become easier to distinguish since the factor of $\sqrt{i}$ will only drive them farther apart. Therefore, as long as the parameters are met, all the eigenvalues will be easily distinguished and there is no risk of confusing any two eigenvalues.

### 2.3 Finding an estimate of $k$

**Theorem 2.5** For $\hat{k}_i = \frac{\lambda_i(B)}{1-p}$, $|\hat{k}_i - k_i| \leq .2k_i$.

**Proof:** $\hat{k}_i$ is an estimate for $k$ using $\lambda_i(B)$ The eigenvalue is $k_i(1-p)$, so $\hat{k}_i$ is calculated as $\frac{\lambda_i(B)}{1-p}$.

By Theorem 23 from [14], If $E$ is $(C_1, c_1, \gamma)$ - concentrated, and $C_1, c_1, \gamma > 0$, and $A$ is rank $r$, then for $1 \leq j \leq r : \lambda_j(A) - t \leq \lambda_j(A + E) \leq \lambda_j(A) + tr^\gamma + 2\sqrt{\lambda_j(A+E) - j\lambda_j(A+E)}$. In this case, we will use $t = \sqrt{i}\ln(n)$.

First, it is necessary to show that the condition is met. Since $E$ is a $n \times n$ symmetric matrix and all values are either $0$ or iid with expectation $0$, and all values are less than $1$, $E$ is $(C_1, c_1, \gamma)$-concentrated, with $C_1 = 2$, $c_1 = \frac{1}{2}$, and $\gamma = 2$ [14]. Therefore, this theorem can be applied.

To show that $|\hat{k}_i - k_i| \leq .2k_i$, it is necessary to show both that $\hat{k}_i \geq .8k_i$ and that $\hat{k}_i \leq 1.2k_i$.

**Proof that $\hat{k}_i \geq .8k_i$:** To prove, use the lower bound from Theorem 23 from [14]:

$$\lambda_j(A) - t \leq \lambda_j(A + E) \Rightarrow k_i(1-p) - t \leq \hat{k}_i(1-p) \Rightarrow k_i - \frac{t}{1-p} \leq \hat{k}_i$$

To show that the above statement implies that $\hat{k}_i \geq .8k_i$, all that is necessary to show is that $\frac{t}{1-p} \leq .2k_i$

$$2.2k_i \geq 2C\sqrt{np} \geq \frac{2C(\ln(n))^{\frac{1}{2}}}{1-p}(1-p) \geq \frac{1C(\ln(n))^{\frac{1}{2}}}{1-p} \geq \frac{\sqrt{i}\ln(n)}{1-p} = \frac{t}{1-p}.$$

The last inequality holds for $n$ large enough, which is sufficient for the purposes of this proof.
Proof that $\hat{k}_i \leq 1.2k_i$: To prove, use the upper bound from Theorem 23 from [14]

$$\lambda_j(A + E) \leq \lambda_j(A) + tr^\frac{1}{2} + 2\sqrt{\lambda_j(A + E)} + j\frac{\|E\|^2}{\lambda_j(A + E)}^2$$

$$\hat{k}_i(1 - p) \leq k_i(1 - p) + \sqrt{i}\ln(n)m + 2\sqrt{\frac{\lambda_j(A + E)^2}{k_i(1 - p)} + \frac{i(3\sqrt{np})^3}{k_i(1 - p)^2}}$$

$$\hat{k}_i \leq k_i + \frac{\sqrt{i}\ln(n)m^\frac{1}{2}}{1 - p} + 2\sqrt{i}\frac{9np}{k_i(1 - p)^2} + \frac{27(\sqrt{np})^3}{k_i^2(1 - p)^3}$$

$$\leq k_i + \frac{m\ln(n)m^\frac{1}{2}}{1 - p} + 2\sqrt{i}\frac{9np}{C\sqrt{i}\sqrt{np}(1 - p)^2} + \frac{27(\sqrt{np})^3}{(C\sqrt{i}\sqrt{np})^2(1 - p)^3}$$

$$\leq k_i + 2\ln(n)(m^{1 + \frac{1}{2}}) + \frac{72\sqrt{np}}{C} + \frac{216\sqrt{np}}{C}$$

To show that $\hat{k}_i \leq 1.2k_i$, it is necessary to show that $2\ln(n)(m^{1 + \frac{1}{2}}) + \frac{72\sqrt{np}}{C} + \frac{216\sqrt{np}}{C} \leq .2k_i$.

First, show that $2\ln(n)(m^{1 + \frac{1}{2}}) \leq \frac{\sqrt{np}}{C}$.

$$4\ln(n)(m^{1 + \frac{1}{2}}) = 4\ln(n)(m^{\frac{3}{2}}) \leq 4\ln(n)(\sqrt{\frac{np}{16C^2\ln^2(n)}})^{\frac{3}{2}} = \frac{\sqrt{np}}{C}$$

Next, show that $\frac{289\sqrt{np}}{C} \leq .2k_i$. This is true as long as $C > 87$.

$$\frac{289\sqrt{np}}{C} \leq .2k_i \iff$$

$$72\sqrt{np} \leq .2C^\frac{1}{2}\sqrt{np} \iff$$

$$72 \leq .2C^\frac{1}{2} \iff$$

$$C > 87$$

It is possible that a lower value of $C$ will also work, but in order to prove that this algorithm works as $n$ approaches infinity, we just need to show that for some value of $C$, the requirements are satisfied. Now we have shown that $k_i \leq 1.2k_i$ and that $\hat{k}_i \geq .8k_i$, so we have successfully shown that $|\hat{k}_i - k_i| \leq .2k_i$.

Finally it is important to show that the probability this is true tends to zero as $n$ approaches infinity.

The probability of success of the lower bound for any particular clique is $1 - 2C_19^i e^{-c_1(\frac{m}{n})}$ [14], so we must show that the probability of the union of the
complements is $1 - o(1)$.

$$P(\bar{E}_1 \cup \ldots \cup \bar{E}_m) \leq \sum_{i=1}^{m} 2C_1 9^i e^{-c_1(\frac{l}{r})}$$

$$\leq \sum_{i=1}^{m} 4e^{i \ln(9)} e^{-\frac{1}{2}(\frac{\sqrt{7 \ln(n)}}{16})}$$

$$\leq \sum_{i=1}^{m} 4e^{-i(\frac{1}{16}(\ln(n))^2 - \ln(9))}$$

Using a geometric sequence we can see that if $e^{-i(\frac{1}{16}(\ln(n))^2 - \ln(9))} < 1$, then

$$\sum_{i=1}^{\infty} 4e^{-i(\frac{1}{16}(\ln(n))^2 - \ln(9))} = \frac{e^{-i(\frac{1}{16}(\ln(n))^2 - \ln(9))}}{1 - e^{-i(\frac{1}{16}(\ln(n))^2 - \ln(9))}}.$$

$$\sum_{i=1}^{m} 4e^{-i(\frac{1}{16}(\ln(n))^2 - \ln(9))} \leq \frac{4e^{-i(\frac{1}{16}(\ln(n))^2 - \ln(9))}}{1 - e^{-i(\frac{1}{16}(\ln(n))^2 - \ln(9))}}$$

$$\lim_{n \to \infty} \frac{4e^{-i(\frac{1}{16}(\ln(n))^2 - \ln(9))}}{1 - e^{-i(\frac{1}{16}(\ln(n))^2 - \ln(9))}} = 0$$

For the upper bound, the probability of success for any one clique is $1 - 2C_1 9^r e^{-c_1 r(\frac{l}{r})}$. Again, we must show that the probability of the union of the complements is $1 - o(1)$.

$$P(\bar{E}_1 \cup \ldots \cup \bar{E}_m) \leq \sum_{i=1}^{m} 2C_1 9^i e^{-c_1 r(\frac{l}{r})}$$

$$\leq \sum_{i=1}^{m} 4e^{2r \ln(9)} e^{-\frac{1}{2}(\frac{\sqrt{7 \ln(n)}}{16})}$$

$$\leq \sum_{i=1}^{m} 4e^{-r(\frac{1}{16}(\ln(n))^2 - 2 \ln(9))}$$

$$\leq m 4e^{-r(\frac{1}{16}(\ln(n))^2 - 2 \ln(9))}$$

$$\leq \frac{4n}{e^{r(\frac{1}{16}(\ln(n))^2) - 2 \ln 9}}$$

$$= \frac{4n 2^n (9)}{e^{r(\frac{1}{16}(\ln(n))^2)}}$$

$$= \frac{4 e^{n}(n) + 2n(9)}{e^{r(\frac{1}{16}(\ln(n))^2)}}$$
Now, we just have to verify that this term goes to zero as \( n \) approaches infinity. To do this, we will use L'Hôpital's rule.

\[
\lim_{n \to \infty} \frac{4e^{\ln(n) + 2\ln(9)}}{ce^{\frac{32}{\ln(n)^2}}} = \lim_{n \to \infty} \frac{\ln(n)}{\frac{1}{32}(\ln(n))^2} = \lim_{n \to \infty} \frac{n}{\ln(n)^2} = \lim_{n \to \infty} \frac{32}{n} = 0
\]

### 2.4 Using the eigenvector to make an estimate for \( S \)

**Theorem 2.6** The distance between the endpoints of \( v_i(B) \) and \( v_i(A) \) is less than or equal to \( \sqrt{2 - 2\sqrt{1 - \frac{36}{C^2}}} \) with probability \( 1 - o(1) \).

**Proof:** By the Davis Kahan inequality [6], for matrices \( A \) and \( B \),

\[
\sin(\langle v_i(A), v_i(B) \rangle) \leq \frac{4\|E\|}{\sigma},
\]

where \( \sigma = \min\{\lambda_i(A) - \lambda_{i+1}(A), \lambda_{i-1}(A) - \lambda_i(A)\} \) with probability \( 1 - o(1) \).

\[
\frac{12\sqrt{np}}{\min\{k_i(1-p) - k_{i+1}(1-p), k_{i-1}(1-p) - k_i(1-p)\}} \leq \frac{12\sqrt{np}}{C\sqrt{np}} = \frac{12}{C}
\]

Now use the law of cosines to show that the maximum distance is \( \sqrt{2 - 2\sqrt{1 - \frac{144}{C^2}}} \).

\[
c^2 = a^2 + b^2 - 2ab\cos(C)
\]

\[
\leq 1 + 1 - 2\sqrt{1 - \left(\frac{12}{C}\right)^2}
\]

\[
c \leq \sqrt{2 - 2\sqrt{1 - \frac{144}{C^2}}}
\]

Intuitively, since the angle between these two vectors is small, many of the entries should have similar values. Keeping in mind that the values of \( v_i(A) \) are all either \( \frac{1}{\sqrt{k_i}} \) or 0, isolating the larger values within \( v_i(B) \) should provide a decent estimate for the values in the clique.

**Theorem 2.7** The set \( T_i = \{j : |v_j(B)| \geq \epsilon\} \), where \( \epsilon = \frac{1}{2\sqrt{k_i}} \), shares at least 99% of values with \( S_i \).

**Proof:** It is necessary to show that \( |T_i \Delta S_i| \leq \frac{1}{100}k_i = \frac{1}{100} |S_i| \).

Do this by showing that each of \( |T_i - S_i| \) and \( |S_i - T_i| \) are less than \( \frac{1}{200}k_i \). In the case of proof \( u_j \) represents the \( j \)-th entry of \( v_i(B) \) and \( v_j \) represents the \( j \)-th entry of \( v_i(A) \), in the order of the calculated eigenvectors.
Proof of $|S_i - T_i|$

$$
|S_i - T_i| = \sum_{j : v_j = \frac{1}{\sqrt{\hat{k}_i}}, \ |u_j| < \frac{1}{2\sqrt{k_i}}} 1
\leq \sum_{j : v_j = \frac{1}{\sqrt{\hat{k}_i}}, \ |u_j| < \frac{1}{2\sqrt{k_i}}} 1
\leq \sum_{j : v_j = \frac{1}{\sqrt{\hat{k}_i}}, \ |u_j| < \frac{1}{2\sqrt{k_i}}} [2\sqrt{8}||u_j|| - v_j|\sqrt{k_i}]^2
\leq 4(0.8)k_i \sum_{j=1}^{n} |u_j - v_j|^2
= 3.2k_i ||v_i(B) - v_i(A)||^2
\leq 3.2k_i \left( 2 - 2\sqrt{1 - \frac{144}{C^2}} \right)
\leq \frac{1}{200} k_i
$$

To show that the last inequality is true, we simply need to solve for $C$. In this case, it is true as long as $C$ is greater than around 13.

Proof of $|T - S|$ This follows from similar logic:

$$
|T_i - S_i| = \sum_{j : v_j = 0, \ |u_j| > \frac{1}{2\sqrt{k_i}}} 1
\leq \sum_{j : v_j = 0, \ |u_j| > \frac{1}{2\sqrt{k_i}}} 1
\leq \sum_{j : v_j = 0, \ |u_j| > \frac{1}{2\sqrt{k_i}}} \left[ ||u_j||2\sqrt{1.2k_i} \right]^2
\leq 4(1.2)k_i \sum_{j=1}^{n} |u_j - v_j|^2
= 4.8k_i ||v_i(B) - v_i(A)||^2
\leq 4.8k_i \left( 2 - 2\sqrt{1 - \frac{144}{C^2}} \right)
\leq \frac{1}{200} k_i
$$
where the last inequality also holds with $C > 13$.

### 2.5 Clean up of the algorithm

**Theorem 2.8** The set $Q_i = \{ r : |N(r) \cap T_i| \geq .655k_i \}$, for all $i \in \{1, ..., m\}$ where $N(r)$ is the set of all points connected to an arbitrary point $r \in V$, is equivalent to $S_i$ with probability $1-o(1)$ for all $i \in \{1, ..., m\}$.

**Proof:** There are two cases, where $r \in S_i$ and where $r \notin S_i$.

**Proof when $r \in S_i$:**

$|N(r) \cap T_i| \geq k_i - .2k_i - .01k_i \geq .79k_i$

**Proof when $r \notin S_i$:**

$||N(r) \cap T_i| - |N(r) \cap S_i|| \leq \frac{1}{100}k_i$, so use $|N(r) \cap S_i|$. This is the number of vertices $r$ is connected to in $S$. Although this is not known, since $S$ is not known, if we can prove that this is less than $.51k_i$, we know that $|N(r) \cap T_i| < .655k_i < .79k_i$, and that this barrier will accurately distinguish between vertices in the clique $S_i$ and vertices not in the

---

{
\begin{align*}
\frac{1}{k_i} \sum_{j=1}^{k_i} Z_j &= E\left(\frac{1}{k_i} \sum_{j=1}^{k_i} Z_j\right) + O(\epsilon), \Rightarrow \\
\sum_{j=1}^{k_i} Z_j &= E\left(\sum_{j=1}^{k_i} Z_j\right) + O(\sqrt{k_i} \log(n)), \Rightarrow \\
|N(r) \cap S_i| &\leq .5k_i + O(\frac{1}{100}k_i),
\end{align*}

So $|N(r) \cap S_i| \leq .51k_i$. This is true, since the expected number of edges to any $k_i$ vertices in the random graph for $r \neq k_i \cdot k_i$, so the most it could be is $.5k_i$, due to the restrictions on $p$.

Since $|N(r) \cap S_i| \leq .5k_i$, then $|N(r) \cap T_i| \leq .52k_i$ because we have already shown that $|S_i \Delta T_i| \leq \frac{1}{100}k_i$.

Therefore, we can show that $.52k_i < .655k_i < .79k_i$, and that this barrier will accurately distinguish between vertices in the clique $S_i$ and vertices not in the
According to Hoeffding’s Inequality [9] the probability of success for any one clique is $1 - 2e^{-2\log^2(n)}$. We will show that the sum of the complements tends to zero as $n$ goes to infinity:

$$P(\bar{E}_1 \cup \ldots \cup \bar{E}_m) \leq \sum_{i=1}^{m} 2e^{-2\ln^2(n)}$$

$$\leq m2e^{-2\ln^2(n)}$$

$$\leq n2e^{-2\ln^2(n)}$$

$$= \frac{2n}{e\ln^2(n)}$$

Now, we will take the limit of this term as $n$ approaches infinity:

$$\lim_{n \to \infty} \frac{2e^{\ln(n)}}{e^{\ln^2(n)}} = \lim_{n \to \infty} \frac{e^{\ln(n) + \ln(2)}}{e^{\ln^2(n)}} = \lim_{n \to \infty} \frac{n + \ln(2)}{\ln^2(n)} = 0$$

Since the probability of the theorem failing goes to zero as $n$ approaches infinity, the probability of success is $1 - o(1)$

### 3 Case with only one clique

It is easy to simplify this theorem in the case that there is only one planted clique. In that case, the algorithm can be run with $m = 1$, and the required restrictions are a bit more lenient. Corollary 1.2 follows directly from Theorem 1.1. Some of the bounds can be brought in to be more strict in the case with only one clique, but the previous proof is sufficient to show that Algorithm 1 will find one planted clique as long as the restrictions are met.

### 4 Simulations

In order to test the validity of the algorithm, several simulations were run. The general methods for constructing the simulations were as follows:

1. Have a computer generate a random symmetric matrix of size $n$ with values of $1 - p$ and $p$, for inputed values of $n$ and $p$.  

---

4.5
2. Change the values of the first $k$ by $k$ values to $1-p$ according to an inputed $k$ value.

3. Run the algorithm to output $Q$, the estimate of the original clique.

4. Compare $Q$ with the original clique, to see how many correct values were included.

5. for each value of $n$ tested, test multiple times and record the proportion of completely successful trials.

In the appendix of this paper, copies of the code used will be listed.

The first graph shows the results from a simulation with one clique of a size obviously larger than necessary for success.

Figure 1: Proportion of successful trials for $p = .3$, $k = 7\sqrt{np}$, $n = \{100, 125, 150, 175, 200, 225, 250, 275\}$ and 100 trials at each value of $n$.

It should be clear from Figure 1 that the algorithm worked for all tested cases 100% of the time.

The second simulation run shows more interesting results. The graphs tested also hold only one planted clique, but the size is closer to the minimum sized clique for the algorithm to be successful.

In Figure 2, it is clear that the algorithm is still successfully retrieving a large part of the planted clique, since 85% or more of the trials run were 100% accurate at all levels of $n$. Another important observation is that as the tested $n$ grew larger, the correct proportion tended towards 100%. This is a good indication that this algorithm would be useful in finding planted cliques in large data pools, where the clique may be harder to find, and supports the claim that the algorithms will tend to be more accurate as $n$ approaches infinity.

The next simulation examined whether the algorithm worked for three separate planted cliques within the random graph. This process was similar, and clique
sizes were chosen to be near the limits of the algorithm.

Figure 2: Proportion of successful trials for $p = .5$, $k = 6\sqrt{np}$, $n = \{100, 125, 150, 175, 200, 225, 250, 275\}$ and 100 trials at each value of $n$.

5 Conclusion

This paper presents a similar solution to a previous algorithm for finding planted cliques within a Erdős–Rényi random graph model. Given an Erdős–Rényi random graph, where each vertex is connected to each other vertex with probability $p$, if a clique of size $k$ is hidden within the graph, it shows how it is possible to find the clique without having to look at every subset of the random graph. This algorithm is simply improving on ideas put forth by Alon, Krivelevich and Sudakov [4]. Their algorithm includes three main steps.

1. Calculating the eigenvalues and eigenvectors of the adjacency matrix of the graph

2. Finding the distinguished eigenvector and choose the $k$ values with the largest absolute values

3. Re-examine all vertices in the graph and only include vertices that are connected to at least 75% of the selected vertices.
The algorithm presented in this thesis is similar, but distinct in a few important ways. In the previous algorithm, it was assumed that $p$ was equal to $\frac{1}{2}$, and it was necessary to know $k$. For this algorithm, it is not necessary to know $k$. It is necessary to know $p$, but within a range, and not necessarily equal to $\frac{1}{2}$. This research is worthwhile because it expands our knowledge of graph theory and of planted cliques in particular. It also greatly increases a computer’s ability to find cliques within graphs created from data, since it requires far fewer calculations. There are, of course, many other improvements that could be made to expand our knowledge on this subject in the future. Some ideas for future exploration include:

1. Exploring methods for finding overlapping cliques, or cliques that share values
2. Examining whether the algorithm works for cliques with some vertices missing, and how many vertices can be removed before the clique cannot be found
3. Improving the lower bound for finding cliques, for both $k$ and $p$
4. Examining whether or not the clean-up step is needed
5. Improving the efficiency of the algorithm for use on a computer

As our knowledge of community detection expands, we will continue to improve methods for finding patterns within large amounts of data. This could make it easier for companies to more accurately recommend products you might like while searching online, or connecting you with people you know on social media.
A Example Code

```python
list3 = []
x = 8
for i in range (0,x):
    list2 = []
t=100
n=25*i+100
for i in range (0,t):
    C = ComplexField(500)
    R = RealField(1000)
    RR = RDF
    p=.5
    k=int(6*sqrt(n*p))
    M = MatrixSpace(RR,n,n)
    Bin = GeneralDiscreteDistribution([1-p,p])
    list1=[]
    for i in range (n^2):
        list1.append (1-p)
    B1 = M(list1)
    for i in range (n):
        for j in range (i+1,n):
    for i in range (k):
        for j in range (i+1,k):
    v= B1.eigenvectors_left()[0]
    k_hat = v[0]/(1-p)
    cutoff= 1/(2*sqrt(k_hat))
    v1= v[1][0]
    v2= copy(v1)
    for i in range (0,n):
        if abs(v1[i]) >= cutoff:
            v2[i]=1
        else:
            v2[i]=0
    sum = 0
    for i in range(0, n):
        if v2[i] == 0:
            sum = 0
        elif v2[i] == 1:
            for j in range(0, n):
                if B1[i][j] == 1-p:
                    sum = sum + 1
                if sum <= .655*k_hat:
                    v2[i] = 0
                if sum > .655*k_hat:
                    v2[i] = 1
    sum = 0
```
for i in range(0, len(v2)):
    if v2[i] == 1:
        sum = sum + 1
    if sum == k:
        list2.append (1)
    else:
        list2.append (0)
    sum = 0
for i in range (0,t):
    if list2[i] == 1:
        sum = sum + 1
    list3.append ((n,sum/t))
print list3
X=list_plot(list3,plotjoined=True)
show(X)
References


