Quantum Spectral Algorithms for Planted Clique

Deeksha Adil, Noah Fleming, Ian Mertz

1 Introduction

The Planted Clique problem has been intensely studied as a natural variant of the clique problem. It arose from the work of Karp [Kar76] who asked whether there was a polynomial time algorithm for finding the largest clique in an Erdős-Rényi random graph $G(n, \frac{1}{2})$. It is well known that with high probability the largest clique in a random graph has size $(2 + o(1)) \log n$ [GM75, BE76]. Simple heuristics find a clique of size $(1 + o(1)) \log n$ in polynomial time with high probability [AKS98]. However, there is no known algorithm that finds a clique of size $(1 + \varepsilon) \log n$ in polynomial time, for constant $\varepsilon > 0$. The Planted Clique problem is a natural variant of this problem, introduced by Kurcera [Kuc95] and Jerrum [Jer92]. Instead of attempting to recover a large clique in a random graph, one is asked to distinguish between the following two cases which occur with equal probability:

1. The input is a random graph $G \sim G(n, \frac{1}{2})$, or
2. The input is a random $G \sim G(n, \frac{1}{2})$ with a clique of size k added to a subset of its vertices.

This problem has been extensively studied but with limited algorithmic success. Based on the observation that with high probability the largest clique in a $G \sim G(n, \frac{1}{2})$ is of size $(2 \pm o(1)) \log(n)$, there is a quasi-polynomial time algorithm for any k. Furthermore, when $k = \Omega(\sqrt{n})$, computing the second highest eigenvalue of the adjacency matrix is sufficient to find a planted clique with high probability [AKS98]. When $k = \Omega(\sqrt{n \log n})$, the edge density of the clique is so large compared to the non-clique vertices that the greedy algorithm in which one simply picks the vertices of largest degree dominates. However, when k lies between $(2 \pm o(1)) \log(n)$ and $\Omega(\sqrt{n})$, polynomial-time algorithms seem far out of reach of current techniques.

Because of the lack of algorithmic progress, recent work has focussed on showing evidence of the hardness of Planted Clique. Because of average-case nature of this problem it seems unlikely that one can obtain evidence of the hardness of Planted Clique from standard complexity assumptions such as $\mathsf{P} \neq \mathsf{NP}$. Therefore, a line of work, beginning with Jerrum [Jer92], seeks to rule out natural families of algorithm for this problem. Jerrum showed that for $k = o(\sqrt{n})$ a natural family of polynomial-time Markov Chains fails to find a planted k-clique. Feldman et al. [FGR$^+$17] generalized this result to hold for any statistical query algorithm (for a bipartite-version of the planted clique problem), which capture many of the algorithms used in practice such as moments-based methods and convex optimization techniques. Another prominent family of algorithms used in theory and practice is semi-definite programs. Feige and Krauthgamer [FK03] showed that no polynomial-time algorithm based on the hierarchy of semi-definite programs known as Lovász-Schrijver (LS_+) could solve the Planted Clique problem $k = o(\sqrt{n})$.

Recently, there has been significant effort to replicate the result of Feige and Krauthgamer for the stronger Sum-of-Squares semi-definite programming hierarchy [DM15, HKP$^+$18, MPW15, RS15]. This is motivated in part by recent work showing that for the large and prominent class of constraint satisfaction problems (CSPs), Sum-of-Squares provides the optimal approximation ratio of any polynomial time algorithm, provided that the popular Unique Games Conjecture is true [Rag08]. Furthermore, most of the semi-definite programs arising in practice can be phrased as a semi-definite program in the Sum-of-Squares hierarchy. Although the Planted Clique problem is not a CSP, negative results rule out a very natural and powerful family of algorithms, giving strong evidence for the hardness of this problem. This culminated in the work of Barak et al. [BHK$^+$16], who showed that any Sum-of-Squares based algorithm requires time $n^{\Omega(n)}$ to solve Planted Clique when $k = o(\sqrt{n})$.

While Planted Clique is not known to be hard for all polynomial-time algorithms, many of known classical algorithms can be phrased as one of the aforementioned paradigms. Thus this can be viewed as strong evidence

for believing that there is no classical (randomized) polynomial-time algorithm for Planted Clique. Indeed, the difficulty of Planted Clique is becoming a standard average-case hardness assumption [AW17, BBB$^+$13, ABW10].

Our main result is a polynomial-time spectral algorithm in the vein of [AKS98] for Planted Clique. Under an assumption, we show that this algorithm can be run on $O(\log n)$ qubits, an exponential improvement when compared to the number of bits needed to directly run the classical spectral algorithm of [AKS98]. We also show that a straightforward application of Grover's algorithm combined with the correct encoding of the problem gives a $\log^2 n$-qubit algorithm with a quadratic improvement in runtime over the best known algorithm classical algorithm for $k = n^\delta$ for $\delta < 1/2$ (which we detail in Appendix A).

1.1 Prior work

Following this work, the authors became aware of work of Ta-Shma [TS13] which gives a quantum logspace algorithm for finding any eigenvalue of a matrix, and thus for solving planted clique for the same range of parameters as our work. In fact, at a high level this algorithm is nearly the same as ours; Section 5 outlines their main algorithm, which is identical to Algorithm 2. The key departure is our use and analysis of [LMR14], which necessitates Assumption 16 but leads to (in our estimation) a clean algorithm without too many technical details. By contrast Section 4 of [TS13] circumvents the work of [LMR14] (which appeared the following year) by using a non-trivial line of work on decomposing Hamiltonian operators.

Another consequence of [TS13] is a classical $O(\log^2 n)$ space algorithm for finding eigenvalues, due to work of van Melkebeek and Watson [MW12] which implies that $\mathsf{BQL} \subseteq \mathsf{SPACE}(\log^2 n)$. Thus while our results show an exponential speedup over the direct translation of [AKS98], they show at most a quadratic improvement over the best classical algorithms. As of the time of this writing the authors are unaware of any $o(\log^2 n)$ space algorithm for finding the eigenvalues of a matrix, even a Hermitian norm 1 operator with eigenvalues separated by $n^{-\Omega(1)}$. Here we note that while follow-up works of [TS13] give classical logspace algorithms for finding eigenvalues, they only apply to Hermitian norm 1 operators where there exist constant separations in the eigenspectrum [Dor14, DSTS17].

2 Definitions and Preliminaries

We begin by formally defining the Planted Clique problem, which is defined using the following two distributions.

Definition 1 (Erdős-Rényi Random Graph)**.** An Erdős-Rényi random graph G is sampled from the distribution $G(n, p)$ by including each of the $\binom{n}{2}$ possible edges (v_i, v_j) with probability p.

Definition 2 (Planted Clique Distribution)**.** A random graph is sampled from the distribution $G(n, p, k)$ by first sampling $G \sim G(n, p)$ and then choosing a random subset S of k vertices of G and *planting* a clique on them. That is, for every $v_i \neq v_j \in S$, the edge (v_i, v_j) is added to G.

With these distributions in hand, we are ready to define the Planted Clique problem; we will state it for the standard parameter $p = 1/2$.

Definition 3 (Planted Clique)**.** Given as input a graph G, determine (with high probability) whether $G \sim G(n, \frac{1}{2})$ or $G \sim G(n, \frac{1}{2}, k)$, each of which occur with probability $1/2$.

Because we will be interested in quantum algorithms for the Planted Clique problem, we quickly review some useful properties of quantum states. Throughout this work, we will use lower-case greek letters, such as ψ, ρ, σ, to represent quantum states. We will use upper-case greek letters, such as Σ, P to denote the registers that these quantum states occupy. Throughout this work, it will be convenient to take advantage of the density matrix representation of quantum states.

Definition 4 (Density Matrix)**.** A Density Matrix is a matrix that describes the state of a system. It is defined as,

$$\rho = \sum_i p_i |\psi_i\rangle\langle\psi_i|,$$

where p_i denotes the probability that the system is in the pure state $|\psi_i\rangle$.

Alternatively, density matrices are matrices which satisfy the following conditions:

1. They are Hermitian.
2. They are positive semi-definite.
3. Their trace is 1.

Recall that matrices σ are evolved by conjugation $U\sigma U^\dagger$, where U is either a unitary matrix. Furthermore, σ may be evolved by more general operations, known as admissible operations. An *admissible operation* (also known as a completely positive trace preserving operation) is any operation D that can be written in the form

$$D(\sigma) = \sum_{j=1}^{\ell} A_j \sigma A_j^\dagger,$$

for matrices A_j satisfying $A_j A_j^\dagger = \mathbb{I}$, where $\mathbb{I}$ is the identity matrix.

3 Towards a Space-Efficient Quantum Algorithm for Planted Clique

Our space-efficient quantum algorithm for Planted Clique is inspired by the spectral algorithm of Alon, Krivelevich, and Sudakov [AKS98]. For intuition, we first present the classical polynomial-time spectral algorithm for planted clique. We defer the analysis to the next section.

Theorem 5. *Let G be a graph sampled either from $G(n, 1/2)$ or $G(n, 1/2, k)$ for $k = \Omega(\sqrt{n})$. Then the following algorithm distinguishes which distribution G was sampled from:*

Algorithm 1 Planted-Clique-Detector

1: **procedure** PCLIQUE($G = (V, E)$)
2: Compute λ_2, the second largest eigenvalue of the adjacency matrix of G.
3: If $\lambda_2 = (\frac{1}{2} \pm \frac{1}{4})k$, return *planted*, otherwise return *random*
4: **end procedure**

The algorithm is quite simple, but the major factor contributing to the space usage is in computing the second largest eigenvalue in step 1. Turning to quantum algorithms it is known that given a unitary matrix U one can recover its eigenvalues (in superposition) via an algorithm called *phase estimation* [Kit95]. Given U and an eigenvector $|u_j\rangle$ with eigenvalue $e^{-i\lambda_j}$, the phase estimation algorithm writes the t most significant bits of λ_j in binary to an auxilliary t-qubit register. More generally, for a unitary U with eigenbasis $\{|u_j\rangle\}_j$ and corresponding eigenvalues $e^{-i\lambda_j}$, and some error parameter δ,

$$PE_{U,t,\delta}\Big(\sum_j \alpha_j |u_j\rangle, |0^t\rangle\Big) = \sum_j (\alpha_j \pm \delta)|u_j\rangle|\lambda_j[1]\lambda_j[2]\dots\lambda_j[t]\rangle$$

where $\lambda_j[i]$ is the ith most significant bit of λ_j.

There are two obstructions to using this algorithm directly on A. First, A is an adjacency matrix and thus far from unitary. We solve this problem by defining $U = e^{-i\rho}$, where $\rho = \frac{1}{n^2}(A + n\mathbb{I})$. U will be unitary as long as ρ is Hermitian, which follows if ρ is a Hamiltonian and positive semidefinite matrix. Hamiltonian follows from the fact that both A and $\mathbb{I}$ are Hamiltonian matrices, and PSDness is shown in section 4.3. Most importantly, the eigenvalues of U still give us a way to distinguish $G(n, 1/2)$ from $G(n, 1/2, k)$ as in the original algorithm, which we state formally in the following lemma (to be proven in section 4.3):

Lemma 6. *Given an adjacency matrix A for a graph G that is either sampled from $G(n, 1/2)$ or $G(n, 1/2, 100\sqrt{n})$, let $\rho := \frac{A+n\mathbb{I}}{n^2}$. Then with probability at least 0.99*

- $\lambda_1(\rho) \geq \frac{5}{4n}$
- $|\lambda_3(\rho)| \dots |\lambda_n(\rho)| \leq \frac{1}{n} + \frac{1}{n^{3/2}}$
- *if* $G \sim G(n, 1/2)$*, then* $|\lambda_2(\rho)| \leq \frac{1}{n} + \frac{1}{n^{3/2}}$*; if* $G \sim G(n, 1/2, \sqrt{n})$*, then* $|\lambda_2(\rho) - (\frac{1}{n} + \frac{50}{n^{3/2}})| \leq \frac{25}{n^{3/2}}$

We make a note of classical approaches making use of Lemma 6. It was proved in [Dor14] that testing whether or not there exists an eigenvalue within $\alpha = \Omega(1)$ distance of a given ℓ can be done in L, but for $\alpha = O(n^{-c})$ it is conjectured that this problem requires $\Omega(\log^2 n)$ space classically, although proving this would imply $\mathsf{L} \neq \mathsf{P}$. Since $|\lambda_2(\rho) - \lambda_3(\rho)| \leq \frac{24}{n^{3/2}}$ regardless of the distribution G was sampled from, this falls into the regime when $\alpha = O(n^{-c})$.

The second obstruction is that to recover the tth most significant bit of the eigenvalues of U we need to be able to implement U^{2^t}. Removing this obstruction is much trickier, and so we take this as a black box assumption for the time being and return to it in a later section.

Assumption 7. For every t and $\delta > 0$ there exists a quantum circuit taking as input a vector $|\sigma\rangle$ and outputting $U^{2^t}|\sigma\rangle$ with probability $1 - \delta$ in time $O((2^t)^2/\delta \cdot \mathsf{poly}(n))$ and auxiliary space $O(\log n)$. Furthermore, this auxiliary space is unentangled from the qubits containing $U^{2^t}|\sigma\rangle$, and therefore can be reused after the computation.

Before presenting the algorithm we will sketch it at a high level. We will mimic the behavior of Algorithm 1 by using phase estimation. Since we cannot calculate an eigenvector of U in small space, we choose a random state instead, as a random state will be a superposition over all n eigenvectors $\{|u_j\rangle\}_j$ of U. Applying phase estimation we get the same superposition $\{|u_j\rangle\}_j$ over all eigenvectors of U, but now for each eigenvector we have its corresponding eigenvalue λ_j stored in an auxiliary register to some high precision. Because the eigenvalues are stored in superposition, all we can do to obtain any individual eigenvalue is to measure this register, which returns each eigenvalue λ_j with some probability. By choosing a random vector we hope that the probability of choosing each eigenvalue, and in particular the second eigenvalue λ_2, is close to uniform. Then, because our procedure is efficient we can simply iterate polynomially many times to ensure that with high probability we measure λ_2 at some point in the computation. At each stage we test to see if we've measured λ_2 by simply interpreting the binary value as a number, using Lemma 6 to know what value of λ_2 to look for, namely appproximately $1/n + 50/n^{3/2}$. Note that this value will only appear if the input graph contains a planted clique, and otherwise all eigenvalues (including λ_2) are far away from this target value. Our algorithm fails if a) we fail to measure λ_2 in the planted clique case, or b) the errors that come from phase estimation cause us to either see a false positive or not recognize our target value and get a false negative. Both of these errors will be bounded by $o(1)$, and so our algorithm succeeds with probability much greater than $2/3$. Finally note that we reset all of our registers in between iterations (one iteration being initialize a random vector, perform phase estimation, and measure the eigenvalue register), and so our space usage is only dependent on what we need to do these three steps, the only nontrivial one being phase estimation which we can do efficiently by Assumption 7.

We now present our main algorithm in full:

Algorithm 2 Quantum-Planted-Clique-Detector

procedure DIST($G = (V, E)$)
  define $\rho := \frac{A + n\mathbb{I}}{n^2}$, define $t := \frac{3}{2}\log n$, and define $\delta = 1/n^3$
  for n^2 iterations **do**
    Let Σ and Λ be registers on $\log n$ and t qubits respectively, initially all zeroes
    Apply H to each qubit in Σ and measure Σ
    Use Assumption 7 to apply $PE_{U:=e^{-i\rho},t}(\Sigma, \Lambda)$
    Measure Λ and check if the result is $\frac{1}{n} + \frac{50}{n^{3/2}} \pm \frac{25}{n^{3/2}}$; if so then output *planted*
  end for
  output *random*
end procedure

Theorem 8 (Main Theorem)**.** *Under Assumption 7, with probability at least* 0.98 *Algorithm 2 distinguishes between* $G \sim G(n, 1/2)$ *and* $G \sim G(n, 1/2, 50\sqrt{n})$ *in time* $\mathsf{poly}(n)$ *and space* $O(\log n)$*.*

Proof. Let $|\sigma\rangle = \sum_j \alpha_j |u_j\rangle$ be the state in Σ before applying Phase Estimation, where $\{u_j\}_j$ is the eigenbasis of U. After applying Phase Estimation, with probability $1 - n^{-3}$ the joint state of the Σ and Λ registers is $\sum_j \alpha_j |u_j\rangle|\lambda_j[1]\dots\lambda_j[t]\rangle$, and after measuring Λ we get $|\lambda_j[1]\dots\lambda_j[t]\rangle$ for some j. Let λ be the interpretation of the binary string measured in Λ as a binary decimal value. By our assumption with $t = 3/2 \log n$,

$$|\lambda - \lambda_j| \leq \frac{1}{n^{3/2}}$$

Therefore by Lemma 6 with probability 0.99 the only way $\lambda = \frac{1}{n} + \frac{50}{n^{3/2}} \pm \frac{25}{n^{3/2}}$ is if $j = 2$ and $G \sim G(n, 1/2, k)$. Lastly we show that we will measure λ_2 at some point in our algorithm. We write $|u_2\rangle = \sum_{i \in \{0,1\}^{\log n}} \beta_i |i\rangle$. Recall that $|\sigma\rangle$ was initialized to $|i\rangle$ for a randomly chosen string $i \in [\log n]$, and so for each $|i\rangle$ the probability that $|\sigma\rangle = |i\rangle$ is exactly $1/n$ by construction. Therefore the probability of measuring $|\lambda_2\rangle$ is

$$\sum_{i \in \{0,1\}^m} |\beta_i|^2 \langle i|\sigma\rangle = \frac{1}{n} \sum_{i \in \{0,1\}^{\log n}} |\beta_i|^2 = \frac{1}{n}$$

and so after n^2 rounds the probability of having not observed λ_2 is $e^{-\Omega(n^2)}$ by Chernoff bounds. In each round by Assumption 7 phase estimation fails with probability at most $1/n^3$. Thus our total probability of failure after n^2 rounds is at most $0.01 + n^2/n^3 + n^2 e^{-\Omega(n^2)} \ll 0.02$.

We now analyze the time and space usage of our algorithm. We use Assumption 7 to apply Phase Estimation with $t = \frac{3}{2} \log n$ and $\delta = 1/n^3$ in time $O((2^t)^2/\delta \cdot \mathsf{poly}(n)) = \mathsf{poly}(n)$ and space $O(\log n)$. We have n^2 rounds each running Phase Estimation. We require $\log n$ qubits to store σ, $\frac{3}{2} \log n$ qubits to store λ, as well as the additional $O(\log n)$ qubits for Assumption 7. After each iteration of the for loop of this algorithm, we will reuse each of these registers by simply observing them and then classically setting them back to the all-zeroes state. Thus our space usage is $O(\log n)$ as claimed. □

4 Analysis of Spectral Guarantees

In this section, we discuss the spectral distribution of the adjacency matrix of the graph in both, the planted and non-planted case. This will in turn give us a simple classical algorithm (Algorithm 1) that distinguishes between the two cases (it can also be extended to recovering the clique by just looking at the second eigenvector). The distribution of the eigenvalues will further be used in the analysis of our quantum algorithm. We begin by analysing the classical algorithm.

4.1 Spectral Algorithm for Recovering Planted Cliques

Algorithm 1 looks at the second eigenvalue and distinguishes between the graphs with and without a planted clique with high probability. To recover the clique when there is one, we just need to look at the k largest entries in the second eigenvector. In this section we will see why we care about the second eigenvector and eigenvalue. We will also look at the spectral distribution when there is a planted clique.

In order to see why the second eigenvector distinguishes our clique let us first look at the expected adjacency matrix of the graph. Let $\hat{A}$ denote the expected adjacency matrix of G. The vertices are indexed so that the first k vertices correspond to the ones belonging to the clique.

$$\hat{A} = \begin{bmatrix} 1 & \dots & 1 & 1/2 & \dots & 1/2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & \dots & 1 & 1/2 & \dots & 1/2 \\ 1/2 & \dots & 1/2 & 1/2 & \dots & 1/2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1/2 & \dots & 1/2 & 1/2 & \dots & 1/2 \end{bmatrix}$$

Note that the above matrix has only 2 different type of rows and thus has rank 2. In other words $\hat{A}$ has only 2 non zero eigenvalues. It is shown in [Rou17] that these are roughly $n/2$ and $k/2$ (to be exact they are $n/2+o(n)$ and $k/2+o(1)$). The corresponding eigenvectors are approximately

$$u_1 = (1, 1, \ldots, 1)$$
$$u_2 = (n-k, \ldots n-k, -k, \ldots -k).$$

The eigenvector u_2 above has $n-k$ on the vertices corresponding to the clique and $-k$ on the other vertices. If our adjacency matrix was like this, we could exactly distinguish the clique vertices from the other vertices by looking at the entries of the second eigenvector. Of course, the actual adjacency matrix differs from the expected adjacency matrix, but it was shown in [Rou17] that it doesn't differ by too much when $k = \Omega(\sqrt{n})$. We now show, using Chernoff bounds, that the largest and second largest eigenvalues of the actual adjacency matrix are indeed close to $n/2$ and $k/2$ with high probability.

Lemma 9. *The largest eigenvalue λ_1 and the second largest eigenvalue of λ_2 of A are such that*

$$P\left[0.5\frac{n}{2} \geq \lambda_1 \leq 1.5\frac{n}{2}\right] \geq 1 - e^{-\frac{n}{16}}$$

$$P\left[0.5\frac{k}{2} \geq \lambda_2 \leq 1.5\frac{k}{2}\right] \geq 1 - e^{-\frac{k}{16}}$$

Proof. We treat λ_2 as a random variable with expectation $k/2 + o(1) \approx k/2$. Using Chernoff bounds we get

$$P\left[\lambda_2 \geq (1+0.5)\frac{k}{2}\right] \leq e^{-\frac{k}{16}}$$

and

$$P\left[\lambda_2 \leq (1-0.5)\frac{k}{2}\right] \leq e^{-\frac{k}{16}}.$$

The result now follows from a union bound. The same analysis follows for the largest eigenvalue as well. □

Thus, when we have a planted clique in our random graph, we get eigenvalues close enough to $n/2$ and $k/2$ with high probability. All other eigenvalues are similar to the no clique case [Rou17].

4.2 Graphs with no Planted Clique

Suppose we just have a random graph with no planted clique, i.e., our graph $G \sim G(n, 1/2)$. In this section we observe how the eigenvalues of the adjacency matrix of this graph are distributed. Let us first consider the expected adjacency matrix. The expected adjacency matrix of such a graph has all entries $1/2$ and has only one non-zero eigenvalue $\lambda_1 = n/2$. However, the actual adjacency matrix differs somewhat from the expected matrix. The following two lemmas bound bound the true distribution of the eigenvalues of the adjacency matrix. First, we have the following result from [CLV03].

Lemma 10. *In an Erdos-Renyi random graph $G(n,p)$, the highest eigenvalue of the adjacency matrix is almost surely greater than $max\{np, \sqrt{n}\}$*

The above lemma implies that the largest eigenvalue of our adjacency matrix is at least $n/2$ with high probability.

Our next result from [TVW10] is about the distribution of all other eigenvalues in an Erdos-Renyi graph. Before we state the result, we define what we mean by an Empirical Spectral Distribution.

Definition 11 (Empirical Spectral Distribution)**.** The empirical spectral distribution (ESD) of a matrix A is a one-dimensional function

$$F(x) = \frac{1}{n}|\{1 \leq j \leq n-1 : \lambda_j(A) \leq x\}|.$$

This can be thought of like the cumulative distribution function. The next lemma is in terms of the density function which when integrated over a range gives the ESD.

Lemma 12. *For $p = \omega(1/n)$, the empirical spectral distribution (ESD) of the matrix $\frac{1}{\sqrt{np(1-p)}}A$ converges in distribution to the semicircle distribution which has a density*

$$\rho(x) = \frac{1}{2\pi}\sqrt{4 - x^2}. \tag{1}$$

The above theorem implies that all but the largest eigenvalue of $\frac{2}{\sqrt{n}}A$ lie between -2 and 2. We can thus conclude that all but the largest eigenvalues of A lie between $-\sqrt{n}$ and $\sqrt{n}$. The following corollary sums the results on the distribution of eigenvalues.

Corollary 13. *Let $G = G(n, 1/2)$. If G does not have a planted clique, then the largest eigenvalue is at least $n/2$ and all other eigenvalues are between $-\sqrt{n}$ and $\sqrt{n}$ with high probability.*

4.3 Spectral Bounds for the Density Matrix

Theorems 5 and 8 follow directly from Lemma 6. We thus prove Lemma 6 in this section.

Proof. (of Lemma 6) Recall that we define

$$\rho := \frac{n\mathbb{I} + A}{n^2}$$

Note that ρ is a positive semi-definite matrix since for any real $n \times n$ matrix with entries $0, 1$, the eigenvalues can only lie between $-n$ and n. Here A is a real matrix with entries 0 or 1 the minimum eigenvalue is $-n$. Now $nI + A$ have eigenvalues $n + \lambda_i(A)$ and are thus non-negative implying ρ to be a positive semidefinite matrix. Now if G has a planted clique of size $k = 100\sqrt{n}$, from Lemma 9 we have that with high probability the top two eigenvalues from of ρ are:

$$\frac{5}{4n} \leq \lambda_1(\rho) \leq \frac{7}{4n}$$
$$\frac{1}{n} + \frac{25}{n^{3/2}} \leq \lambda_2(\rho) \leq \frac{1}{n} + \frac{75}{n^{3/2}}.$$

When G does not have a planted clique, then by Corollary 13, with high probability the top two eigenvalues satify,

$$\lambda_1(\rho) \geq \frac{3}{2n}$$
$$\lambda_2(\rho) \leq \frac{1}{n} + \frac{1}{n^{3/2}}.$$

As mentioned in the discussion in the previous sections (Lemma 13 for the no clique case and [Rou17] for the case with the clique), all other eigenvalues in both cases are less than $\frac{1}{n} + \frac{1}{n^{3/2}}$ in absolute value. □

5 The LMR Algorithm

In this section we prove Assumption 7, under a weaker assumption that one can efficiently implement $\rho := \frac{A+n\mathbb{I}}{n^2}$, the scaled adjacency matrix. For this, we will rely heavily on the algorithm of Loyd, Mohseni, and Rebentros [LMR14]. Phase estimation allows one to recover the eigenvalues of a unitary matrix U assuming that you are given the ability to apply controlled U^t for some $t \in \mathbb{N}$. Loyd, Mohseni, and Rebentros showed that if the unitary U is of the form $e^{-i\rho t}$ for some density matrix ρ, then it is possible to efficiently approximate $U^t\sigma(U^t)^\dagger$, given many copies of ρ. Here, approximate is in terms of the *trace distance* between matrices A and B,

$$\frac{1}{2}\|A - B\|_{tr}$$

where $\|A\|_{tr} = \sqrt{AA^\dagger}$.

Theorem 14. *([LMR14]) Let σ, ρ be two unknown states. There is a quantum algorithm that takes $\sigma \otimes \rho^{\otimes \ell}$ and outputs $\tilde{\sigma}$ such that*

$$\frac{1}{2}\|e^{-i\rho t}\sigma e^{i\rho t} - \tilde{\sigma}\|_{tr} \leq \delta,$$

where $\ell = O(t^2/\delta)$ in running time $O(t^2/\delta)$.

For clarity in describing the algorithm, we will denote by Σ the registers that hold the state σ and by P_i the registers that hold the ith copy of ρ. Furthermore, we will denote the contents of register P_i by ρ_{P_i} The algorithm will use two operators. The first is the *partial trace operator*, which is analogous to marginalizing in probability theory. For a composite Hilbert space $\mathcal{H}_\Sigma \otimes \mathcal{H}_{P_i}$ the partial trace of P_i is the linear extension of the map

$$Tr_{P_i}\left[\alpha_\Sigma \otimes \beta_{P_i}\right] = Tr[\beta]\alpha_\Sigma, \tag{2}$$

where $Tr[]$ is the standard matrix trace. Formally, for a basis $\{|u_i\rangle\}$ of Σ and a basis $\{|v_i\rangle\}$ of P_i, and a state $\gamma_{\Sigma,P_i} = \sum_{i,j,k,l} p_{i,j,k,l}|u_i\rangle\langle u_j| \otimes |v_k\rangle\langle v_l|$ over $\mathcal{H}_\Sigma \otimes \mathcal{H}_{P_i}$

$$Tr_{P_i}\left[\gamma_{\Sigma,P_i}\right] = c_{i,j,k,l}|u_i\rangle\langle u_j|\langle v_k|v_l\rangle.$$

The partial trace is known to be an admissible operator. That is, we marginalize to only look at the Σ qubits, ignoring P_i. The second operator is the *swap operator* which swaps the values of two registers (or sets of registers). Define S_j to be the gate which swaps the contents of register A with register B_i;

$$S_j(\alpha_\Sigma \otimes \beta_{P_j}) = \beta_\Sigma \otimes \alpha_{P_j}.$$

Note that S_j is a unitary because $S_j^2 = I$. The LMR algorithm proceeds by repeatedly applying a *partial swap* between the register Σ originally holding σ and the registers P_j holding ρ. This partial swap is computed by the operator

$$e^{-iS_j\varepsilon} = (\cos\varepsilon)I - i(\sin\varepsilon)S_j$$

The LMR algorithm proceeds as follows:

Algorithm 3 LMR

1: **procedure** LMR($\sigma \otimes \rho^{\otimes \ell}$)
2: Let $\ell = O(t^2/\delta)$ and let $\varepsilon = \delta/t$
3: **for** $j \in [\ell]$ **do**
4: Let σ'_Σ be the state in register Σ.
5: $Tr_{B_j}[e^{-i\varepsilon S_j}(\sigma'_\Sigma \otimes \rho_{P_j})e^{i\varepsilon S_j}]$.
6: **end for**
7: **end procedure**

We defer the reader to Appendix B of the paper of Kimmel et al. [KLL+17] for the analysis of this algorithm, and that it does indeed produce a density matrix which is at most δ away $e^{-i\rho t}\sigma e^{i\rho t}$ in trace distance.

Approximating Controlled Powers of U The LMR algorithm alone is not enough for phase estimation. Indeed, phase estimation required the ability to apply *controlled* powers of U. Kimmel et al. [KLL+17] show that this can be achieved without altering the LMR algorithm, by instead by replacing ρ with $\tilde{\rho} := |1\rangle\langle 1| \otimes \rho$, noting that

$$e^{-i\tilde{\rho}t} = e^{-i(|1\rangle\langle 1|\otimes\rho)t} = |0\rangle\langle 0| \otimes \mathbb{I}_\Sigma + |1\rangle\langle 1| \otimes e^{-i\rho t},$$

where $\mathbb{I}_\Sigma$ is the identity matrix acting on the registers holding ρ. Therefore, this leaves the LMR algorithm unchanged.

Efficient Implementation A second useful observation of Kimmel et al. [KLL+17] is that the (potentially expensive) partial swap operation $e^{-iS_j\varepsilon}$ can actually be implemented efficiently, in particular, using $O(t^2/\delta \cdot \log dim(\mathcal{H}_\Sigma))$ single-qubit controlled swap gates.

5.1 LMR with Space Conservation

The LMR algorithm is designed for the case when both σ and ρ are unknown. We show that if ρ is known ahead of time, and furthermore that copies of ρ can be implemented efficiently, then the LMR algorithm can be run using only a single ρ-register P.

Lemma 15. *Assume there exists a quantum algorithm A that takes in the state $|0^{2\log n}\rangle$ and outputs the density matrix ρ in polynomial time and logarithmic space. Then there is a quantum algorithm that takes as input $\sigma \otimes \rho$ and outputs $\tilde{\sigma} \otimes \rho'$ such that*

$$\frac{1}{2}\|e^{-i\rho t}\sigma e^{i\rho t} - \tilde{\sigma}\|_{tr} \leq \delta,$$

running in time $O(t^2/\delta)$ and logarithmic space.

The algorithm is simple: we reuse the same ρ-register P for each iteration of the LMR subprocedure, and after the iteration is over we "reset" P to the all-zeroes state by measuring it and then making the necessary transformation. Because we don't use the result of measuring P for any other calculations, the density matrix σ will actually be treated as if P was left untouched and a new register containing ρ was introduced into the system. With this analysis in hand we perform LMR as usual, and for completeness we state the algorithm now:

Algorithm 4 Space-Conscious LMR

procedure LMR($\sigma \otimes \rho$)
  Let $\ell = O(t^2/\delta)$ and let $\varepsilon = \delta/t$.
  for $j \in [\ell]$ **do**
    Let σ'_Σ be the state in register Σ.
    $Tr_{P_j}[e^{-i\varepsilon S_j}(\sigma'_\Sigma \otimes \rho_P)e^{i\varepsilon S_j}]$.
    Reset registers P to $|0\rangle$ with algorithm A.
    Load $P \leftarrow \rho$
  end for
end procedure

Proof. (of Lemma 15) First, note that the time of our algorithm is $O(\ell) \cdot TIME(A)$ and the space is $2 \cdot 2\log n + SPACE(A)$, which are polynomial and logarithmic in n respectively. Thus we prove that $\frac{1}{2}\|e^{-i\rho t}\sigma e^{i\rho t} - \tilde{\sigma}\|_{tr} \leq \delta$. Consider the jth execution of the inner subroutine, where σ^j and ρ^j are the states of Σ and P (respectively) after applying the partial trace operation. Note that by definition the matrix σ^j after marginalizing away ρ^j is the expectation over all measurement outcomes of P of the posterior matrix σ^j. Thus, measuring P still leaves the Σ register with the expected density matrix σ^j, and so when we reset P the joint state of the system is indistinguishable from any other measurement outcome on P, and so Σ holds the actual marginal density matrix from the jth execution of the inner subroutine of Algorithm 3. The lemma then holds by the same analysis as before (see appendix B of [KLL$^+$17]). □

5.2 Phase Estimation using LMR

Finally, we prove Assumption 7, using the space-conscious LMR algorithm from Lemma 15. We rely on the following weaker assumption:

Assumption 16. Let $\rho := \frac{A+c\mathbb{I}}{Tr(A+c\mathbb{I})}$ be the state in Algorithm 2, where A is the adjacency matrix of the input graph. We assume that there is a procedure that begins with $|0\rangle$ and produces $\tilde{\rho} := |1\rangle\langle 1| \otimes \rho$ is in time $\mathrm{poly}(n)$ and uses an additional $O(\log n)$ qubits, starting in the state $|0\rangle$, and which are not entangled with $\tilde{\rho}$ at the end of the algorithm (and so can be replaced with $|0\rangle$ for the next iteration).

We note that setting a register to $|0\rangle$ is one of DiVincenzo's necessary criteria for a quantum computer [DiV00], and so this is not necessary to assume as part of Assumption 16.

Lemma 17. *Phase estimation on unitary $U = e^{-i\rho}$ can be performed to precision κ and failure probability $O(\delta/\kappa)$ in time $O(t_{\tilde{\rho}}/\kappa\delta)$ using $O(dim(U) + s_{\tilde{\rho}})$ auxiliary qubits, provided that Assumption 16 holds. Here $t_{\tilde{\rho}}$ and $s_{\tilde{\rho}} = O(\log n)$ are the time and additional qubits needed by the algorithm for Assumption 16.*

This was first proven in Kimmel et al. [KLL$^+$17] when the standard LMR algorithm is used, we restate it here for completeness.

Proof. Estimating an eigenvalue of U up to precision κ requires obtaining the $\log(1/\kappa)$ bits of that eigenvalue. To apply Kitaev's phase estimation [Kit95] one needs to apply controlled-U^t for $t = 2^i$ and $i \in [0, 1/\kappa]$. Because this is geometric, we need to apply controlled-U $O(1/\kappa)$ times. This can be done by repeated application of Lemma 15 (for $t = 1$), using the same space. Let δ be the bound in the trace distance for Lemma 15 (which can be chosen), then the total error in trace distance of the final state is $O(\delta/\kappa)$.

Because every application of Lemma 15 (for $t = 1$) requires time $O(t_{\tilde{\rho}}/\delta)$, the runtime of the algorithm is $O(1/\kappa\delta)$. Finally, by Lemma 15, each application of U can be done in place, and therefore, the space required to hold σ and ρ is $O(\log dim(U))$, with the additional space $s_{\tilde{\rho}}$ that we are allowing for the construction of $\tilde{\rho}$. □

6 Conclusions and Open Problems

This paper initiates the study of the Planted Clique problem in the setting of quantum computation. The main contribution is an algorithm which, under a mild assumption, requires only a logarithmic number of qubits, a quadratic improvement over the classical spectral algorithm. As well, we show that for $k = n^\delta < \sqrt{n}$, a simple application of Grover's Algorithm gives a slight improvement in the running time over the most efficient classical algorithm for this range of k. We conclude by mentioning several open problems:

1. The first immediate problem left over by this work is to remove Assumption 16. That is, to show that there is an algorithm that efficiently implements $\tilde{\rho}$, the controlled scaled adjacency matrix.

2. Not only is the classical spectral algorithm able to distinguish the random graphs from random graphs with a planted clique (for $k \geq O(\sqrt{n})$), but it is able extract the planted clique by using the largest entries of the eigenvector corresponding to the second largest eigenvalue. For this range of parameters, extracting a planted clique using only $O(\log n)$ qubits remains an interesting open problem. While the posterior state of our algorithm actually holds an implicitly encoded copy of the eigenvector we want, it is unclear if we can recover the largest values even with multiple copies of the eigenvector. These values would have to be significantly bigger than the ones corresponding to non-clique vertices in order to recover even one vertex of the clique with high probability, which is true in the corresponding eigenvector of the *expected* adjacency matrix but seems unlikely to be true for the true adjacency matrix. Still, there may be a way to rotate the eigenvector towards the eigenvector of the expected adjacency matrix, or otherwise amplify the larger coefficients such that we could recover clique vertices with high probability.

3. Our improvement for $k = o(\sqrt{n})$ (see Appendix A) is a blind application of Grover's Algorithm. This ignores the significant amount of structure in the planted clique and the sub-cliques formed by them. Whether quantum algorithms are able exploit this structure is still unclear and an interesting avenue to explore. On one hand, a quantum algorithm for $k = o(\sqrt{n})$ would be a significant breakthrough, especially since planted clique is being used as a hardness result in cryptography [AW17,ABW10]. On the other hand, negative results (even for natural families of quantum algorithms) would give evidence that the hardness of planted clique is a reasonable assumption even for quantum computation, supporting its use as a hardness assumption.

References

[ABW10] Benny Applebaum, Boaz Barak, and Avi Wigderson. Public-key cryptography from different assumptions. In *Proceedings of the 42nd ACM Symposium on Theory of Computing, STOC 2010, Cambridge, Massachusetts, USA, 5-8 June 2010*, pages 171–180, 2010.

[AKS98] Noga Alon, Michael Krivelevich, and Benny Sudakov. Finding a large hidden clique in a random graph. *Random Struct. Algorithms*, 13(3-4):457–466, 1998.

[AW17] Aubrey Alston and Yanrong Wo. On the cryptanalysis via approximation of cryptographic primitives relying on the planted clique conjecture. *CoRR*, abs/1707.00078, 2017.

[BBB+13] Maria-Florina Balcan, Christian Borgs, Mark Braverman, Jennifer T. Chayes, and Shang-Hua Teng. Finding endogenously formed communities. In *Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2013, New Orleans, Louisiana, USA, January 6-8, 2013*, pages 767–783, 2013.

[BBHT] Michel Boyer, Gilles Brassard, Peter Hyer, and Alain Tapp. Tight bounds on quantum searching. *Fortschritte der Physik*, 46(45):493–505.

[BE76] Béla Bollobás and Paul Erdös. Cliques in random graphs. *Mathematical Proceedings of the Cambridge Philosophical Society*, 1976.

[BHK+16] Boaz Barak, Samuel B. Hopkins, Jonathan A. Kelner, Pravesh Kothari, Ankur Moitra, and Aaron Potechin. A nearly tight sum-of-squares lower bound for the planted clique problem. In *IEEE 57th Annual Symposium on Foundations of Computer Science, FOCS 2016, 9-11 October 2016, Hyatt Regency, New Brunswick, New Jersey, USA*, pages 428–437, 2016.

[CLV03] Fan Chung, Linyuan Lu, and Van Vu. Spectra of random graphs with given expected degrees. *Proceedings of the National Academy of Sciences*, 100(11):6313–6318, 2003.

[DiV00] David P DiVincenzo. The physical implementation of quantum computation. *Fortschritte der Physik: Progress of Physics*, 48(9-11):771–783, 2000.

[DM15] Yash Deshpande and Andrea Montanari. Improved sum-of-squares lower bounds for hidden clique and hidden submatrix problems. *CoRR*, abs/1502.06590, 2015.

[Dor14] Dean Doron. On the problem of approximating eigenvalues of undirected graphs in probabilistic logspace. 2014.

[DSTS17] Dean Doron, Amir Sarid, and Amnon Ta-Shma. On approximating the eigenvalues of stochastic matrices in probabilistic logspace. *computational complexity*, 26(2):393–420, Jun 2017.

[FGR+17] Vitaly Feldman, Elena Grigorescu, Lev Reyzin, Santosh Srinivas Vempala, and Ying Xiao. Statistical algorithms and a lower bound for detecting planted cliques. *J. ACM*, 64(2):8:1–8:37, 2017.

[FK03] Uriel Feige and Robert Krauthgamer. The probable value of the lovász–schrijver relaxations for maximum independent set. *SIAM J. Comput.*, 32(2):345–370, 2003.

[GM75] G. R. Grimmett and C. J. H. McDiarmid. On colouring random graphs. *Mathematical Proceedings of the Cambridge Philosophical Society*, 77(2):313324, 1975.

[HKP+18] Samuel B. Hopkins, Pravesh Kothari, Aaron Henry Potechin, Prasad Raghavendra, and Tselil Schramm. On the integrality gap of degree-4 sum of squares for planted clique. *ACM Trans. Algorithms*, 14(3):28:1–28:31, 2018.

[Jer92] Mark Jerrum. Large cliques elude the metropolis process. *Random Struct. Algorithms*, 3(4):347–360, 1992.

[Kar76] R. M. Karp. The Probabilistic Analysis of some Combinatorial Search Algorithms. In J. F. Traub, editor, *Algorithms and Complexity: New Directions and Recent Results*, pages 1–20. Academic Press, New York, 1976.

[Kit95] A Yu Kitaev. Quantum measurements and the abelian stabilizer problem. 1995.

[KLL+17] Shelby Kimmel, Cedric Yen-Yu Lin, Guang Hao Low, Maris Ozols, and Theodore J Yoder. Hamiltonian simulation with optimal sample complexity. *npj Quantum Information*, 3(1):13, 2017.

[Kuc95] Ludek Kucera. Expected complexity of graph partitioning problems. *Discrete Applied Mathematics*, 57(2-3):193–212, 1995.

[LMR14] Seth Lloyd, Masoud Mohseni, and Patrick Rebentrost. Quantum principal component analysis. *Nature Physics*, 10(9):631–633, July 2014.

[MPW15] Raghu Meka, Aaron Potechin, and Avi Wigderson. Sum-of-squares lower bounds for planted clique. In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015, Portland, OR, USA, June 14-17, 2015*, pages 87–96, 2015.

[MW12] Dieter van Melkebeek and Thomas Watson. Time-space efficient simulations of quantum computations. *Theory of Computing*, 8(1):1–51, 2012.

[Rag08] Prasad Raghavendra. Optimal algorithms and inapproximability results for every csp? In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing, Victoria, British Columbia, Canada, May 17-20, 2008*, pages 245–254, 2008.

[Rou17] Tim Roughgarden. Cs 264 : Beyond worst-case analysis lectures 9 and 10 : Spectral algorithms for planted bisection and planted clique. 2017.

[RS15] Prasad Raghavendra and Tselil Schramm. Tight lower bounds for planted clique in the degree-4 SOS program. *CoRR*, abs/1507.05136, 2015.

[TS13] Amnon Ta-Shma. Inverting well conditioned matrices in quantum logspace. In *Proceedings of the Forty-fifth Annual ACM Symposium on Theory of Computing*, STOC '13, pages 881–890, 2013.

[TVW10] L. Tran, V. Vu, and K. Wang. Sparse random graphs: Eigenvalues and Eigenvectors. *ArXiv e-prints*, November 2010.

Appendix A: Grover's Algorithm for Planted Clique

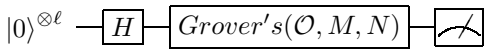
We sketch how to use Grover's Algorithm to obtain a quantum algorithm for planted clique which slightly improves upon the running time of the best known classical algorithm for k in the range $2\log n \ll k \ll \sqrt{n}$ using only $O(\log^2 n)$ qubits. It is well known that with extremely high probability, the largest clique in $G \sim G(n, 1/2)$ will be of size at most $(2 + o(1)) \log n$ [GM75, BE76]. Therefore, in order to determine whether there exists a clique of size $k > 2\log n$ in G, it is sufficient to decide whether there exists a clique of size larger than $2\log n$, say $t = (2+\varepsilon)\log n$ for some constant $\varepsilon \leq 1$. If such a t-clique exists then it is with extremely high probability part of a planted k-clique. On the other hand, because every size-t subset of a k-clique is a t-clique, if no t-clique exists in G, then G cannot contain a planted clique.

Claim 18. Let $t = (2+\varepsilon)\log n$. For any $k \geq t$, there exists a quantum algorithm that runs in time $O(en/k)^{t/2}$ that solves Planted k-Clique with high probability.

Our algorithm will use $\ell = \log_2 \binom{n}{(2+\varepsilon)\log n}$-qubits. Fix some canonical ordering to the possible cliques in a graph on n vertices. For each $i \in \binom{n}{t}$, we interpret the state $|i\rangle$ as being associated with the ith possible t-clique, we call it the ith- *clique indicator*. Let $\mathcal{O}$ be the phase oracle where

$$\mathcal{O}|i\rangle = (-1)^{clique_G(i)}|i\rangle,$$

where $clique_G(i)$ is the function that outputs 1 if i is a clique in the input graph G. The quantum circuit is as follows:



Here, $Grover's(\mathcal{O})$ is Grover's Algorithm for multiple solutions with oracle $\mathcal{O}$ and parameters number of solutions $M = \binom{k}{t}$ solutions and $N = \log_2 \binom{n}{t}$ number of qubits. Our choice of M follows from the fact that a k-clique contains exactly $\binom{k}{t}$ t-cliques, and $G \sim G(n, 1/2)$ contains no other t-cliques with high probability.

The algorithm behaves as follows: Applying H to $|0\rangle^{\otimes \ell}$ gives us a uniform super-position over clique indicators. Grover's Algorithm for multiple solutions applies the appropriate Grover iterate $O(\sqrt{M/N})$, which using the oracle $\mathcal{O}$ will separate the non-cliques from the cliques. Then, we measure and obtain some clique indicator $|i\rangle$. Finally, we check if the clique corresponding to $|i\rangle$ is in the graph or not; This can be done by looking at t bits of the input classically. If it is a clique, we return *planted*, otherwise we return *random*.

Proof. (Of Claim 18) The runtime of this algorithm is dominated by the runtime of Grover's Algorithm for multiple solutions, which is $O(\sqrt{N/M})$. By our setting of N and M this is

$$O\left(\sqrt{\frac{\binom{n}{t}}{\binom{k}{t}}}\right) \leq O\left(\sqrt{\frac{n^t/(t)!}{k^t/(t)^t}}\right) = O\left(\sqrt{(n/k)^t \left(1 + \frac{1}{t}\right)^t}\right) \leq O\left(\sqrt{(n/k)^t e^t}\right) \tag{3}$$

where we have used that $x!/x^x = (1 + 1/x)^x$.

Two factors can cause the algorithm to fail: First is if the graph contains a t-clique, which happens with extremely small probability. The second is if Grover's Algorithm fails. Boyer et al. [BBHT] show that the failure probability of Grover's Algorithm is at most $M/N \leq (ek/n)^t$ which is super-polynomially small for $k = o(n)$. Taking a union bound over both of these events concludes that this algorithm succeeds with high probability. □

Finally, we show that this improves quadratically upon the best-known classical algorithm for certain ranges of $k = n^\delta$. Plugging in $t = (2 + \varepsilon) \log n$ and $k = n^\varepsilon$ to Equation 3, the runtime of the quantum algorithm is bounded by

$$O\left(n^{(2+\varepsilon)\log(e)/2} n^{(1-\delta)(2+\varepsilon)\log n/2}\right) = O\left(n^{(2+\varepsilon)(1+(1-\delta)\log n)/2}\right). \tag{4}$$

In contrast, the classical algorithm runs in time

$$O\binom{n}{(2+\varepsilon)\log n} \geq O\left(\left(\frac{n}{(2+\varepsilon)\log n}\right)^{(2+\varepsilon)\log n}\right) = O\left(n^{(2+\varepsilon)(\log n - \log((2+\varepsilon)\log n))}\right).$$