

# Decremental $(1 + \epsilon)$ -Approximate Maximum Eigenvector: Dynamic Power Method

Deeksha Adil\*  
Institute for Theoretical Studies  
ETH Zürich  
deeksha.adil@eth-its.ethz.ch

Thatchaphol Saranurak†  
Department of Computer Science  
University of Michigan  
thsa@umich.edu

September 4, 2025

## Abstract

We present a dynamic algorithm for maintaining  $(1 + \epsilon)$ -approximate maximum eigenvector and eigenvalue of a positive semi-definite matrix  $A$  undergoing *decreasing* updates, i.e., updates which may only decrease eigenvalues. Given a vector  $v$  updating  $A \leftarrow A - vv^\top$ , our algorithm takes  $\tilde{O}(\text{nnz}(v))$  amortized update time, i.e., polylogarithmic per non-zeros in the update vector.

Our technique is based on a novel analysis of the influential power method in the dynamic setting. The two previous sets of techniques have the following drawbacks (1) algebraic techniques can maintain exact solutions but their update time is at least polynomial per non-zeros, and (2) sketching techniques admit polylogarithmic update time but suffer from a crude additive approximation.

Our algorithm exploits an oblivious adversary. Interestingly, we show that any algorithm with polylogarithmic update time per non-zeros that works against an adaptive adversary and satisfies an additional natural property would imply a breakthrough for checking psd-ness of matrices in  $\tilde{O}(n^2)$  time, instead of  $O(n^\omega)$  time.

---

\*Supported by Dr. Max Rössler, the Walter Haefner Foundation and the ETH Zürich Foundation

†Supported by NSF grant CCF-2238138.

# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Our Results . . . . .	2
1.2	Towards Separation between Oblivious and Adaptive adversaries . . . . .	2
<b>2</b>	<b>Preliminaries</b>	<b>4</b>
<b>3</b>	<b>Algorithms against an Oblivious Adversary</b>	<b>6</b>
3.1	Proof Overview . . . . .	8
3.2	Proof of the Main Theorem 1.2 assuming the Key Lemma 3.5 . . . . .	11
3.3	Proof of the Key Lemma 3.5: Few Executions of Line 4 . . . . .	13
3.4	Proof of the Progress Lemma 3.8 . . . . .	15
<b>4</b>	<b>Conditional Lower Bounds for an Adaptive Adversary</b>	<b>23</b>
<b>5</b>	<b>Conclusion and Open Problems</b>	<b>26</b>
<b>A</b>	<b>Connections to Dynamic Positive Semi-definite Programs</b>	<b>31</b>
<b>B</b>	<b>Omitted Proofs</b>	<b>33</b>

# 1 Introduction

Computing eigenvalues and eigenvectors of a matrix is predominant in several applications, including principle component analysis, clustering in high-dimensional data, semidefinite programming, spectral graph partitioning, and algorithms such as Google’s PageRank algorithm. In the era of massive and dynamic datasets, developing algorithms capable of efficiently updating spectral information with changing inputs has become indispensable.

The study of the change in the spectrum of a matrix undergoing updates spans over five decades, with Golub [Gol73] providing the first algebraic characterization of the change for positive semi-definite matrices via the *secular equation*. This result offered an explicit formula for exactly computing all new eigenvalues when a matrix undergoes a single rank-one update, assuming knowledge of the entire eigen-decomposition of the initial matrix. Subsequently, Bunch, Nielsen, and Sorensen [BNS78] showed how to additionally compute eigenvectors explicitly, and handle the case of repeated eigenvalues of the initial matrix. A decade later, Arbenz, Gander, and Golub [AGG88] extended the works of [Gol73; BNS78] to compute all new eigenvalues and eigenvectors of a positive semi-definite matrix undergoing an update of the form of a small-rank matrix, or a single batch update. Further works extend these techniques to computing singular values [Sta08], as well as computing new eigenvalues and eigenvectors when only the partial eigen-decomposition of the initial matrix is known [MSS19]. However, all these works require a full eigen-decomposition of the initial matrix and only handle a single update to the matrix.

Another independent line of work aims at finding a small and sparse matrix that has eigenvalues close to the original matrix. A recent work in this direction by Bhattacharjee et al. [Bha+23] provides a universal sparsifier  $\mathbf{S}$  for positive semi-definite matrices  $\mathbf{A}$ , such that  $\mathbf{S}$  has at most  $n/\epsilon^4$  non-zero entries and  $\|\mathbf{A} - \mathbf{A} \circ \mathbf{S}\| \leq \epsilon n$ . Swartworth and Woodruff [SW23] give an alternate technique by using *gaussian sketches* with  $O(1/\epsilon^2)$  rows to approximate all eigenvalues of  $\mathbf{A}$  to an additive error of  $\epsilon\|\mathbf{A}\|_F$ . The algorithms that find sparse matrices approximating the eigenvalues in these works may be extended to handle updates in the initial matrix quickly. However, the approximation of the eigenvalues achieved is quite crude, i.e., at least  $\epsilon n$  additive factor. This approximation is also shown to be tight for such techniques.

Now consider the simpler task of only maintaining the maximum eigenvalue and eigenvector of a matrix  $\mathbf{A}$  as it undergoes updates. As we have seen above, known methods based on algebraic techniques require full spectral information before computing the new eigenvalues, and maintaining the entire eigen-decomposition can be slow. Sparsification-based algorithms are fast but only achieve large additive approximations of at least  $\epsilon n$ , which is not quite satisfactory. Works on streaming PCA, which have a similar goal of maintaining large eigenvectors focus on optimizing the space complexity instead of runtimes [AL17]. Previous works based on dynamically maintaining the matrix inverse can maintain  $(1 + \epsilon)$ -multiplicative approximations to the maximum eigenvalues of an  $n \times n$  symmetric matrix undergoing single-entry updates with an update time of  $O(n^{1.447}/\text{poly}(\epsilon))$  [San04]. This was further improved to  $O(n^{1.407}/\text{poly}(\epsilon))$  [BNS19]. The update time is even slower for row, column, or rank-one updates. In any case, it takes polynomial time per non-zeros of the updates. This leads to the following natural question:

*Is there a dynamic algorithm that can maintain a multiplicative approximation of the maximum eigenvalue of a matrix using polylogarithmic update time per non-zeros in the update?*

In this paper, we study the problem of maintaining a  $(1 + \epsilon)$ -multiplicative approximation to the maximum eigenvalue and eigenvector of a positive semi-definite matrix as it undergoes a sequence

of rank-one updates that may only decrease the eigenvalues. We note that this is equivalent to finding the minimum eigenvalue and eigenvector of a positive semi-definite matrix undergoing rank-one updates that may only increase the eigenvalues. We now formally state our problem.

**Problem 1.1** (Decremental Approximate Maximum Eigenvalue and Eigenvector). *We are given a size parameter  $n$ , an accuracy parameter  $\epsilon \in (0, 1)$ , a psd matrix  $\mathbf{A}_0 \succeq 0$  of size  $n \times n$ , and an online sequence of vectors  $\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_T$  that update  $\mathbf{A}_t \leftarrow \mathbf{A}_{t-1} - \mathbf{v}_t \mathbf{v}_t^\top$  with a promise that  $\mathbf{A}_t \succeq 0$  for all  $t$ . The goal is to maintain an  $\epsilon$ -approximate eigenvalue  $\lambda_t \in \mathbb{R}$  and  $\epsilon$ -approximate eigenvector  $\mathbf{w}_t \in \mathbb{R}^n$  of  $\mathbf{A}_t$  for all time  $t$ . That is  $\lambda_t$  and unit vector  $\mathbf{w}_t$  satisfying,*

$$\max_{\mathbf{u}: \text{unit}} \mathbf{u}^\top \mathbf{A}_t \mathbf{u} \geq \lambda_t \geq (1 - \epsilon) \max_{\mathbf{u}: \text{unit}} \mathbf{u}^\top \mathbf{A}_t \mathbf{u}, \quad (1)$$

and

$$\mathbf{w}_t^\top \mathbf{A}_t \mathbf{w}_t \geq (1 - \epsilon) \max_{\mathbf{u}: \text{unit}} \mathbf{u}^\top \mathbf{A}_t \mathbf{u}. \quad (2)$$

## 1.1 Our Results

We give an algorithm for the Decremental Approximate Maximum Eigenvalue and Eigenvector Problem that has an amortized update time of  $\approx \tilde{O}(\text{nnz}(\mathbf{v}_t)) \leq \tilde{O}(n)$ <sup>1</sup>. In other words, the total time required by our algorithm over  $T$  updates is at most  $\tilde{O}(\text{nnz}(\mathbf{A}_0) + \sum_{i=1}^T \text{nnz}(\mathbf{v}_i))$ . Observe that our algorithm only requires time that is  $\tilde{O}(1) \times$  (time required to read the input) and can handle any number of decremental updates. Our algorithm works against an *oblivious* adversary, i.e., the update sequence is fixed from the beginning. This is the first algorithm that can handle a sequence of updates while providing a multiplicative approximation to the eigenvalues and eigenvectors in a total runtime of  $\leq \tilde{O}(n^2 + n \cdot T)$  and an amortized update time faster than previous algorithms by a factor of  $n^{\Omega(1)}$ . Formally, we prove the following:

**Theorem 1.2.** *There is an algorithm for Problem 1.1 under a sequence of  $T$  decreasing updates, that given parameters  $n$ ,  $\mathbf{A}_0$ , and  $\epsilon > 1/n$  as input, with probability at least  $1 - 1/n$  works against an oblivious adversary in total time,*

$$O \left( \frac{\log^3 n \log^6 \frac{n}{\epsilon} \log \frac{\lambda_{\max}(\mathbf{A}_0)}{\lambda_{\max}(\mathbf{A}_T)}}{\epsilon^4} \left( \text{nnz}(\mathbf{A}_0) + \sum_{i=1}^T \text{nnz}(\mathbf{v}_i) \right) \right).$$

Our algorithm is a novel adaptation of the classical *power method* (see, e.g., [TB22]) to the dynamic setting, along with a new analysis that may be of independent interest.

Our work can also be viewed as the first step towards generalizing the dynamic algorithms for solving positive linear programs of [BKS23] to solving dynamic positive semi-definite programs. We discuss this connection in detail in Appendix A.

## 1.2 Towards Separation between Oblivious and Adaptive adversaries

We also explore the possibility of removing the assumption of an oblivious adversary and working against *adaptive adversaries*. Recall that update sequences are given by adaptive adversaries when the update sequence can depend on the solution of the algorithm.

---

<sup>1</sup> $\tilde{O}$  hides polynomials in  $\log n$ .

We show that if there is an algorithm for Problem 1.1 with a total running time of at most  $\tilde{O}(n^2)$  such that the output  $\mathbf{w}_t$ 's satisfy an additional natural property (which is satisfied by the output of the power method, but *not* by our algorithm), then it contradicts the hardness of a well-known barrier in numerical linear algebra, therefore ruling out such algorithms.

We first state the barrier formally, and then state our result for adaptive adversaries. Recall that, given a matrix  $\mathbf{A}$ , the condition number of  $\mathbf{A}$  is  $\frac{\max_{\mathbf{x}: \|\mathbf{x}\|=1} \|\mathbf{A}\mathbf{x}\|}{\min_{\mathbf{x}: \|\mathbf{x}\|=1} \|\mathbf{A}\mathbf{x}\|}$ .

**Problem 1.3** (Checking psdness with certificate). *Given  $\delta \in (0, 1)$ , parameter  $\kappa$ , and a symmetric matrix  $\mathbf{A}$  of size  $n \times n$  with condition number at most  $\kappa$ , either*

- *Compute a matrix  $\mathbf{X}$  where  $\|\mathbf{A} - \mathbf{X}\mathbf{X}^T\| \leq \delta \min_{\|\mathbf{x}\|=1} \|\mathbf{A}\mathbf{x}\|$ , certifying that  $\mathbf{A}$  is a psd matrix, or*
- *Report that  $\mathbf{A}$  is not a psd matrix.*

Recall that  $\mathbf{A}$  is a psd matrix iff there exists  $\mathbf{X}$  such that  $\mathbf{A} = \mathbf{X}\mathbf{X}^T$ . The matrix of  $\mathbf{X}$  is called a *vector realization* of  $\mathbf{A}$ , and note that  $\mathbf{X}$  is not unique. The problem above asks to compute a (highly accurate) vector realization of  $\mathbf{A}$ . Both eigendecomposition and Cholesky decomposition of  $\mathbf{A}$  give a solution for Problem 1.3.<sup>2</sup> Banks et al [Ban+22] showed how to compute with high accuracy an eigendecomposition of a psd matrix in  $O(n^{\omega+\eta})$  time for any constant  $\eta > 0$ , where  $\omega > 2.37$  is the matrix multiplication exponent. Hence, a vector realization of  $\mathbf{A}$  can be found in the same time. Observe that, when  $\delta < \kappa$ , Problem 1.3 is *at least as hard as certifying that a given matrix  $\mathbf{A}$  is a psd matrix*. It is a notorious open problem whether certifying the psd-ness of a matrix can be done faster than  $o(n^\omega)$  even when the condition number is polynomial in  $n$ . Therefore, we view Problem 1.3 as a significant barrier and formalize it as follows.

**Hypothesis 1.4** (PSDness Checking Barrier). *There is a constant  $\eta > 0$  such that, every randomized algorithm for solving Problem 1.3 for instances with  $\frac{\kappa}{\delta} \leq \text{poly}(n)$ , correctly with probability at least  $2/3$  requires  $n^{2+\eta}$  time.*

Our negative result states that, assuming Hypothesis 1.4, there is no algorithm for Problem 1.1 against adaptive adversaries with sub-polynomial update time that maintains  $\mathbf{w}_t$  satisfying an additional property stated below.

**Property 1.5.** *For every  $t$  let  $\mathbf{u}_i(\mathbf{A}_t)$  denote the eigenvectors of  $\mathbf{A}_t$  and  $\lambda_i(\mathbf{A}_t)$  denote the eigenvalues. For all  $i$  such that  $\lambda_i(\mathbf{A}_t) \leq \lambda_1(\mathbf{A}_t)/2$ ,*

$$\left(\mathbf{w}_t^\top \mathbf{u}_i(\mathbf{A}_t)\right)^2 \leq \frac{1}{n^2} \cdot \frac{\lambda_i(\mathbf{A}_t)}{\lambda_1(\mathbf{A}_t)}.$$

**Theorem 1.6.** *Assuming Hypothesis 1.4, there is no algorithm for Problem 1.1 that maintains  $\mathbf{w}_t$ 's additionally satisfying Property 1.5, and given parameters  $n$  and  $\epsilon = \min\left\{1 - \frac{1}{n^{o(1)}}, \frac{1-\delta}{1+\delta}\right\}$  as input, works against an adaptive adversary in time  $n^{o(1)} \cdot \left(\text{nnz}(\mathbf{A}_0) + \sum_{t=1}^T \text{nnz}(\mathbf{v}_i)\right)$ .*

---

<sup>2</sup>Given an eigendecomposition  $\mathbf{A} = \mathbf{Q}\mathbf{\Lambda}\mathbf{Q}^\top$  where  $\mathbf{\Lambda}$  is a non-negative diagonal matrix and  $\mathbf{Q}$  is an orthogonal matrix, we set  $\mathbf{X} = \mathbf{Q}\mathbf{\Lambda}^{1/2}$ . Given a Cholesky decomposition  $\mathbf{A} = \mathbf{L}\mathbf{L}^\top$  where  $\mathbf{L}$  is a lower triangular matrix, we set  $\mathbf{X} = \mathbf{L}$ .

Let us motivate Property 1.5 from several aspects below. First, in the static setting, this property can be easily satisfied since the static power method strongly satisfies it (see Lemma 3.3). Second, the statement of the property itself is a natural property that we might expect from an algorithm. Consider a “bad” eigenvector  $\mathbf{u}_i$  whose corresponding eigenvalue is very small, i.e., less than half of the maximum one. It states that the projection of the output  $\mathbf{w}_t$  along  $\mathbf{u}_i$  should be very small, i.e., a polynomial factor smaller than the projection along the maximum eigenvector. This is intuitively useful because we do not want the output  $\mathbf{w}_t$  to direct to the “bad” direction. It should mainly direct along the approximately maximum eigenvector. Third, we formalize this intuition and crucially exploit Property 1.5 to prove a reduction in Theorem 1.6. Specifically, Property 1.5 allows us to decrease eigenvalues of a psd matrix while maintaining the psd-ness, which is crucial for us. See Section 4 for more details. Lastly, our current algorithm from Theorem 1.2, a dynamic version of the power method, actually maintains  $\mathbf{w}_t$ ’s that satisfy this property for certain snapshots, but not at every step. Unfortunately, we do not see how to strengthen our algorithm to satisfy Property 1.5 nor how to remove it from the requirement of Theorem 1.6. We leave both possibilities as open problems.

Understanding the power and limitations of dynamic algorithms against an oblivious adversary vs. an adaptive adversary has become one of the main research programs in the area of dynamic algorithms. However, finding a natural dynamic problem that separates oblivious and adaptive adversaries is still a wide-open problem.<sup>3</sup> In this paper, we suggest the problem of maintaining approximate maximum eigenvectors as a natural candidate for the separation and give some preliminary evidence for this.

**Organization.** In the following sections, we begin with some preliminaries required to prove our results in Section 2, followed by our algorithm and its analysis in Section 3, and our conditional lower bound in Section 4. In Section A we show connections with positive semi-definite programs, and finally, in Section 5, we present some open problems.

## 2 Preliminaries

Let  $\mathbf{A}_0$  denote the initial matrix. Let  $\lambda_0 = \lambda_{\max}(\mathbf{A}_0) = \|\mathbf{A}_0\|$  denote the maximum eigenvalue of the initial matrix  $\mathbf{A}_0$ . The following are the key definitions we use in our analysis.

**Definition 2.1** ( $\epsilon$ -max span and dimension). *We define  $\text{span}(\epsilon, \mathbf{A})$  to denote the space spanned by all eigenvectors of  $\mathbf{A}$  corresponding to eigenvalues  $\lambda$  satisfying  $\lambda \geq (1 - \epsilon)\lambda_0$ . Let  $\text{dim}(\epsilon, \mathbf{A})$  to denote the dimension of the space  $\text{span}(\epsilon, \mathbf{A})$ .*

We emphasize that  $\lambda_0$  depends only on  $\mathbf{A}_0$ . So, it is a static value that does not change through time. We will use the following linear algebraic notations.

**Definition 2.2.** *Let  $S_1$  and  $S_2$  be two subspaces of a vector space  $S$ . The sum  $S_1 + S_2$  is the space,*

$$S_1 + S_2 = \{s = s_1 + s_2 : s_1 \in S_1, s_2 \in S_2\}.$$

---

<sup>3</sup>Beimel et al. [Bei+22] gave the first separations for artificial problems assuming a strong cryptographic assumption. Bateni et al. [Bat+23] shows a separation between the adversaries for the  $k$ -center clustering problem, however, their results are based on the existence of a black box which the adaptive adversary can control.

The complement  $\bar{S}$  of  $S$  is the vector space such that  $S + \bar{S} = \mathbb{R}^n$ , and  $S \cap \bar{S} = \{0\}$ . The difference,  $S_1 - S_2$  is defined as,

$$S_1 - S_2 = S_1 \cap \bar{S}_2.$$

Next, we list standard facts about high-dimensional probability needed in our analysis.

**Lemma 2.3** (Chernoff Bound). *Let  $x_1, \dots, x_m$  be independent random variables such that  $a \leq x_i \leq b$  for all  $i$ . Let  $x = \sum_i x_i$  and let  $\mu = \mathbb{E}[x]$ . Then for all  $\delta > 0$ ,*

$$\Pr[x \geq (1 + \delta)\mu] \leq \exp\left(-\frac{2\delta^2\mu^2}{m(b-a)^2}\right)$$

$$\Pr[x \leq (1 - \delta)\mu] \leq \exp\left(-\frac{\delta^2\mu^2}{m(b-a)^2}\right)$$

**Lemma 2.4** (Norm of Gaussian Vector). *A random vector  $\mathbf{v} \in \mathbb{R}^n$  with every coordinate chosen from a normal distribution,  $N(0, 1)$  satisfies,*

$$\Pr[|\|\mathbf{v}\|^2 - n| \leq 2(1 + \delta)\delta \cdot n] \geq 1 - e^{-\delta^2 n}.$$

*Proof.* The vector  $\mathbf{v}$  has entries that are from  $N(0, 1)$ . Now, every  $\mathbf{v}_i^2$  follows a  $\chi^2$  distribution. From Lemma 1 of [LM00] we have the following tail bound for a sum of  $\chi^2$  random variables,

$$\Pr\left[\left|\sum_i \mathbf{v}_i^2 - n\right| > 2\sqrt{nx} + 2x\right] \leq e^{-x}.$$

Choosing  $x = n\delta^2$  gives,

$$\Pr[|\|\mathbf{v}\|^2 - n| \leq 2\delta(1 + \delta)n] \geq 1 - e^{-\delta^2 n},$$

as required. □

**Lemma 2.5** (Distribution of  $\chi^2$  Variable). *Let  $x \sim N(0, 1)$  be a gaussian random variable. Then,*

$$\Pr\left[x^2 \geq \frac{1}{n^4}\right] \geq 1 - \frac{1}{n^2}.$$

*Proof.* The probability distribution function for  $y = x^2$  is given by,

$$f(y) = \frac{1}{\sqrt{2}\Gamma(\frac{1}{2})} y^{-\frac{1}{2}} e^{-\frac{y}{2}}.$$

It is known that  $\Gamma(\frac{1}{2}) = \sqrt{\pi}$ . Now,

$$\Pr\left[x^2 \leq \frac{1}{n^4}\right] = \int_0^{1/n^4} \frac{1}{\sqrt{2}\Gamma(\frac{1}{2})} y^{-\frac{1}{2}} e^{-\frac{y}{2}} dy \leq \frac{e^0}{\sqrt{2\pi}} \int_0^{1/n^4} y^{-\frac{1}{2}} dy = \sqrt{\frac{2}{\pi}} \cdot \frac{1}{n^2} \leq \frac{1}{n^2}.$$

□

### 3 Algorithms against an Oblivious Adversary

To prove Theorem 1.2, we first reduce Problem 1.1 to solving a normalized threshold version of the problem where we assume that initially, the maximum eigenvalue is not much bigger than one. Then we want to maintain a certificate that the maximum eigenvalue is not much less than one until no such certificate exists. This is formalized below.

**Problem 3.1** (DecMaxEV( $\epsilon, \mathbf{A}_0, \mathbf{v}_1, \dots, \mathbf{v}_T$ )). *Let  $\mathbf{A}_0$  be an  $n \times n$  symmetric PSD matrix such that  $\lambda_{\max}(\mathbf{A}_0) \leq 1 + \frac{\epsilon}{\log n}$ . The DECMAxEV( $\epsilon, \mathbf{A}_0, \mathbf{v}_1, \dots, \mathbf{v}_T$ ) problem asks to find for every  $t$ , a vector  $\mathbf{w}_t$  such that*

$$\|\mathbf{w}_t\| = 1 \quad \text{and} \quad \mathbf{w}_t^\top \mathbf{A}_t \mathbf{w}_t \geq 1 - 40\epsilon,$$

*or return FALSE indicating that  $\lambda_{\max}(\mathbf{A}_t) \leq 1 - \frac{\epsilon}{\log n}$ .*

We defer the proof of the standard reduction stated below to the appendix.

**Lemma 3.2.** *Given an algorithm  $\mathcal{A}$  that solves the decision problem DECMAxEV( $\epsilon, \mathbf{A}_0, \mathbf{v}_1, \dots, \mathbf{v}_T$ ) (Definition 3.1) for any  $\epsilon > 0$ ,  $\mathbf{A}_0 \succeq 0$  and vectors  $\mathbf{v}_1, \dots, \mathbf{v}_T$  in time  $\mathcal{T}$ , we can solve Problem 1.1 in total time  $O\left(\frac{\log^2 n \log \frac{n}{\epsilon}}{\epsilon} \cdot \text{nnz}(\mathbf{A}_0) + \frac{\log n}{\epsilon} \log \frac{\lambda_{\max}(\mathbf{A}_0)}{\lambda_{\max}(\mathbf{A}_T)} \mathcal{T}\right)$ .*

Next, we describe Algorithm 1 which can be viewed as an algorithm for Problem 3.1 when there are no updates. Our algorithm essentially applies the *power iteration*, which is a standard algorithm used to find an approximate maximum eigenvalue and eigenvector of a matrix. In the algorithm, we make  $R = O(\log n)$  copies to boost the probability. Below, we state the guarantees

---

**Algorithm 1** DECMAxEV with no update

---

```

1: procedure POWERMETHOD( $\epsilon, \mathbf{A}$ )
2:    $R \leftarrow 10 \log n, r_0 \leftarrow 1$ 
3:    $K \leftarrow \frac{4 \log \frac{n}{\epsilon}}{\epsilon}$ 
4:   for  $r = 1, \dots, R$  do
5:      $\mathbf{v}^{(0,r)} \leftarrow$  random vector with coordinates chosen from  $N(0, 1)$ 
6:     for  $k = 1 : K$  do
7:        $\mathbf{v}^{(k,r)} \leftarrow \mathbf{A} \mathbf{v}^{(k-1,r)}$ 
8:        $\mathbf{w}^{(r)} \leftarrow \frac{\mathbf{v}^{(K,r)}}{\|\mathbf{v}^{(K,r)}\|}$ 
9:    $\mathbf{W} = [\mathbf{w}^{(1)}, \dots, \mathbf{w}^{(R)}]$ 
10:  if  $(\mathbf{w}^{(r)})^\top \mathbf{A} \mathbf{w}^{(r)} < 1 - \epsilon$  for all  $r \leq R$  then
11:    return FALSE
12:  else
13:     $r_0 \leftarrow$  smallest  $r$  such that  $(\mathbf{w}^{(r)})^\top \mathbf{A} \mathbf{w}^{(r)} \geq 1 - 5\epsilon$ 
14:    return  $[r_0, \mathbf{W}]$ 
```

---

of the power method.

**Lemma 3.3.** *Let  $\epsilon > 0$  and  $\mathbf{A} \succeq 0$ . Let  $\mathbf{W}$  be as defined in Line 9 in the execution of POWERMETHOD( $\epsilon, \mathbf{A}$ ). With probability at least  $1 - 1/n^{10}$ , for some  $\mathbf{w} \in \mathbf{W}$ , it holds that  $\mathbf{w}^\top \mathbf{A} \mathbf{w} \geq (1 - \frac{\epsilon}{2}) \lambda_{\max}(\mathbf{A})$ . The total time taken by the algorithm is at most  $O\left(\frac{\text{nnz}(\mathbf{A}) \log n \log \frac{n}{\epsilon}}{\epsilon}\right)$ .*



Furthermore, let  $\lambda_i$  and  $\mathbf{u}_i$  denote the eigenvalues and eigenvectors of  $\mathbf{A}$ . For all  $i$  such that  $\lambda_i(\mathbf{A}) \leq \frac{\lambda_{\max}(\mathbf{A})}{2}$ , with probability at least  $1 - 2/n^{10}$ ,  $\left[\mathbf{w}^\top \mathbf{u}_i\right]^2 \leq \frac{1}{n^8} \cdot \frac{\lambda_i}{\lambda_1}$ .

We note that the last line of the above lemma is saying that the vectors returned by the power method satisfy Property 1.5, which we state for completeness but is not required by our algorithm. The following result is a direct consequence of Lemma 3.3.

**Corollary 3.4.** *Let  $\epsilon > 0$ ,  $\mathbf{A} \succeq 0$ . Let  $\mathbf{W}$  be as defined in Line 9 in the execution of `POWERMETHOD`( $\epsilon, \mathbf{A}$ ). If  $\lambda_{\max}(\mathbf{A}) \geq 1 - \epsilon$ , then with probability at least  $1 - 1/n^{10}$ ,  $\mathbf{w}^\top \mathbf{A} \mathbf{w} \geq 1 - 5\epsilon$  for some  $\mathbf{w} \in \mathbf{W}$ . Furthermore, if  $\lambda_{\max}(\mathbf{A}) \geq 1 - \epsilon/\log n$ , then with probability at least  $1 - 1/n^{10}$ ,  $\mathbf{w}^\top \mathbf{A} \mathbf{w} \geq 1 - \epsilon$  for some  $\mathbf{w} \in \mathbf{W}$ . The total time taken by the algorithm is at most  $O\left(\frac{\text{nnz}(\mathbf{A}) \log n \log \frac{n}{\epsilon}}{\epsilon}\right)$ .*

Observe that, if the algorithm returns  $[r_0, \mathbf{W}]$ , then  $(\mathbf{w}^{(r)})^\top \mathbf{A} \mathbf{w}^{(r)} \geq 1 - 5\epsilon$  for  $r = r_0$ , and  $\mathbf{w}^{(r_0)}$  is therefore a solution to Problem 3.1 when there is no update. The power method and its analysis are standard, and we thus defer the proof of Lemma 3.3 to the appendix.

Next, in Algorithms 2 and 3 we describe an algorithm for Problem 3.1 when we have an online sequence of updates  $\mathbf{v}_1, \dots, \mathbf{v}_T$ . The algorithm starts by initializing  $R = O(\log n)$  copies of the approximate maximum eigenvectors from the power method. Given a sequence of updates, as long as one of the copies is the witness that the current matrix  $\mathbf{A}_t$  still has a large eigenvalue, i.e., there exists  $r$  where  $(\mathbf{w}^{(r)})^\top \mathbf{A}_t \mathbf{w}_t^{(r)} \geq 1 - 40\epsilon$ , we can just return  $\mathbf{w}^{(r)}$  as the solution to Problem 3.1. Otherwise,  $(\mathbf{w}^{(r)})^\top \mathbf{A}_t \mathbf{w}_t^{(r)} < 1 - 40\epsilon$  for all  $r \leq R$  and none of the vectors from the previous call to the power method are a witness of large eigenvalues anymore. In this case, we simply recompute these vectors by calling the power method again. If the power method returns that there is no large eigenvector, then we return FALSE from now. Otherwise, we continue in the same manner. Note that our algorithm is very simple, but as we will see, the analysis is not straightforward.

---

**Algorithm 2** Initialization

---

```

1: procedure INIT( $\epsilon, \mathbf{A}_0$ )
2:    $\mathbf{W} \leftarrow \text{POWERMETHOD}(\epsilon, \mathbf{A}_0)$ 
3:   return  $\mathbf{W}$ 

```

---



---

**Algorithm 3** Update algorithm at time  $t$  ( $\mathbf{A}_{t-1}, r_t, \mathbf{W}_{t-1} = [\mathbf{w}_{t-1}^{(r)} : r = 1, \dots, R], \epsilon$  are maintained)

---

```

1: procedure UPDATE( $\mathbf{v}_t$ )
2:    $\mathbf{A}_t \leftarrow \mathbf{A}_{t-1} - \mathbf{v}_t \mathbf{v}_t^\top$ 
3:   if  $(\mathbf{w}_{t-1}^{(r)})^\top \mathbf{A}_t \mathbf{w}_{t-1}^{(r)} < 1 - 40\epsilon$  for all  $r \leq R$  then
4:      $[r_t, \mathbf{W}_t] \leftarrow \text{POWERMETHOD}(\epsilon, \mathbf{A}_t)$ 
5:     if POWERMETHOD( $\epsilon, \mathbf{A}_t$ ) returns FALSE then
6:       return FALSE for all further updates
7:   else
8:      $r_t \leftarrow$  smallest  $r$  such that  $(\mathbf{w}_{t-1}^{(r)})^\top \mathbf{A}_t \mathbf{w}_{t-1}^{(r)} \geq 1 - 40\epsilon$ 
9:      $\mathbf{W}_t \leftarrow \mathbf{W}_{t-1}$ 
10:  return  $[r_t, \mathbf{W}_t]$ 

```

---

### 3.1 Proof Overview

The overall proof of Theorem 1.2, including the proof of correctness and the runtime depends on the number of executions in Line 4 in Algorithm 3. If the number of executions of Line 4 is bounded by  $\text{poly}(\log n/\epsilon)$ , then the remaining analysis is straightforward. Therefore, the majority of our analysis is dedicated to proving this key lemma, i.e.,  $\text{poly}(\log n/\epsilon)$  bound on the number of calls to the power method:

**Lemma 3.5** (Key Lemma). *The number of executions of Line 4 over all updates is bounded by  $O(\log n \log^5 \frac{n}{\epsilon}/\epsilon^2)$  with probability at least  $1 - \frac{1}{n}$ .*

Given the key lemma, the correctness and runtime analyses are quite straightforward and are presented in Section 3.2. We now give an overview of the proof of Lemma 3.5.

Let us consider what happens between two consecutive calls to Line 4, say at  $\mathbf{A}$  and  $\tilde{\mathbf{A}} = \mathbf{A} - \sum_{i=1}^k \mathbf{v}_i \mathbf{v}_i^\top$ . We first define the following subspaces of  $\mathbf{A}$  and  $\tilde{\mathbf{A}}$ . Recall Definition 2.2, which we use to define the following subspaces.

**Definition 3.6** (Subspaces of  $\mathbf{A}$  and  $\tilde{\mathbf{A}}$ ). *Given  $\epsilon > 0$ ,  $\mathbf{A}$ , and  $\tilde{\mathbf{A}}$  define for  $\nu = 0, 1, \dots, 15 \log \frac{n}{\epsilon} - 1$ :*

$$T_\nu = \text{span} \left( \frac{(\nu+1)\epsilon}{5 \log \frac{n}{\epsilon}}, \mathbf{A} \right) - \text{span} \left( \frac{\nu\epsilon}{5 \log \frac{n}{\epsilon}}, \mathbf{A} \right),$$

and,

$$\tilde{T}_\nu = \text{span} \left( \frac{(\nu+1)\epsilon}{5 \log \frac{n}{\epsilon}}, \tilde{\mathbf{A}} \right) - \text{span} \left( \frac{\nu\epsilon}{5 \log \frac{n}{\epsilon}}, \tilde{\mathbf{A}} \right).$$

That is, the space  $T_\nu$  and  $\tilde{T}_\nu$  are spanned by eigenvectors of  $\mathbf{A}$  and  $\tilde{\mathbf{A}}$ , respectively, corresponding to eigenvalues between  $\left(1 - (\nu+1) \frac{\epsilon}{5 \log \frac{n}{\epsilon}}\right) \lambda_0$  and  $\left(1 - \nu \frac{\epsilon}{5 \log \frac{n}{\epsilon}}\right) \lambda_0$ .

Let  $d_\nu = \dim(T_\nu)$  and  $\tilde{d}_\nu = \dim(\tilde{T}_\nu)$ . Also define,

$$\tilde{T} = \text{span}(3\epsilon, \tilde{\mathbf{A}}), \quad T = \text{span}(3\epsilon, \mathbf{A}),$$

and let  $d = \dim(T)$ ,  $\tilde{d} = \dim(\tilde{T})$ .

Observe that  $T = \sum_{\nu=0}^{15 \log \frac{n}{\epsilon} - 1} T_\nu$  and similarly  $\tilde{T} = \sum_{\nu=0}^{15 \log \frac{n}{\epsilon} - 1} \tilde{T}_\nu$ . We next define some indices/levels corresponding to large subspaces, which we call “important levels”.

**Definition 3.7** (Important  $\nu$ ). *We say a level  $\nu$  is important if,*

$$d_\nu \geq \frac{\epsilon}{600 \log^3 \frac{n}{\epsilon}} \sum_{\nu' < \nu} d_{\nu'}.$$

We will use  $\mathcal{I}$  to denote the set of  $\nu$  that are important.

The main technical lemma that implies Lemma 3.5 is the following:

**Lemma 3.8** (Measure of Progress). *Let  $\epsilon > 0$  and let  $\mathbf{W} = [\mathbf{w}^{(1)}, \dots, \mathbf{w}^{(R)}]$  be as defined in Line 9 in the execution of POWERMETHOD( $\epsilon, \mathbf{A}$ ). Let  $\mathbf{v}_1, \dots, \mathbf{v}_k$  be a sequence of updates generated by an oblivious adversary and define  $\tilde{\mathbf{A}} = \mathbf{A} - \sum_{i=1}^k \mathbf{v}_i \mathbf{v}_i^\top$ .*

*Suppose that  $\lambda_{\max}(\mathbf{A}) \geq 1 - \epsilon$  and  $\mathbf{w}^\top \tilde{\mathbf{A}} \mathbf{w} < 1 - 40\epsilon$  for all  $\mathbf{w} \in \mathbf{W}$ . Then, with probability at least  $1 - \frac{50 \log \frac{n}{\epsilon}}{n^2}$ , for some  $\nu \in \mathcal{I}$ ,*

- $\dim(T_\nu - \tilde{T}) \geq \frac{\epsilon}{300 \log \frac{n}{\epsilon}} d_\nu$  if  $d_\nu \geq \frac{3000 \log n \log \frac{n}{\epsilon}}{\epsilon}$ , or
- $\dim(T_\nu - \tilde{T}) \geq 1$  if  $d_\nu < \frac{3000 \log n \log \frac{n}{\epsilon}}{\epsilon}$ .

We prove this lemma in Section 3.4. Intuitively speaking, it means that, whenever Line 4 of Algorithm 3 is executed, there is some important level  $\nu$  such that an  $\Omega(\epsilon/\text{poly} \log(n/\epsilon))$ -fraction of eigenvalues of  $\mathbf{A}$  at level  $\nu$  have decreased in value. This is the crucial place where we exploit an oblivious adversary.

Given Lemma 3.8, the remaining proof of Lemma 3.5 follows a potential function analysis which is presented in detail in Section 3.3. We consider potentials  $\Phi_j = \sum_{\nu=0}^j d_\nu$  for  $j = 0, \dots, 15 \log \frac{n}{\epsilon} - 1$ . The main observation is that  $\Phi_j$  is non-increasing over time for all  $j$ , and whenever there exists  $\nu_0 \in \mathcal{I}$  that satisfies the condition of Lemma 3.8,  $\Phi_{\nu_0}$  decreases by  $\dim(T_{\nu_0} - \tilde{T})$ . Since  $\dim(T_{\nu_0} - \tilde{T}) \geq \Omega(\epsilon/\text{poly} \log(n/\epsilon)) d_{\nu_0}$  and  $\nu_0 \in \mathcal{I}$ , i.e.,  $\Phi_{\nu_0} = \sum_{\nu < \nu_0} d_\nu + d_{\nu_0} \leq d_{\nu_0} \left( \frac{O(\log^3 \frac{n}{\epsilon})}{\epsilon} + 1 \right)$ , we can prove that  $\Phi_{\nu_0}$  decreases by a multiplicative factor of  $\Omega(1 - \epsilon^2/\text{poly} \log(n/\epsilon))$ . As a result, every time our algorithm executes Line 4,  $\Phi_j$  decreases by a multiplicative factor for some  $j$ , and since we have at most  $15 \log \frac{n}{\epsilon}$  values of  $j$ , we can only have  $\text{poly}(\log n/\epsilon)$  executions of Line 4.

It remains to describe how we prove Lemma 3.8 at a high level. We can write  $\mathbf{w}^\top \tilde{\mathbf{A}} \mathbf{w}$  for any  $\mathbf{w} \in \mathbf{W}$  as

$$\mathbf{w}^\top \tilde{\mathbf{A}} \mathbf{w} = \mathbf{w}^\top \mathbf{A} \mathbf{w} - \mathbf{w}^\top \mathbf{V} \mathbf{w},$$

for  $\mathbf{V} = \sum_{i=1}^k \mathbf{v}_i \mathbf{v}_i^\top$ . Our strategy is to show that:

$$\begin{aligned} &\text{If } \dim(T_\nu - \tilde{T}) \text{ does not satisfies the inequalities in Lemma 3.8 for all } \nu \in \mathcal{I}, \\ &\text{then } \mathbf{w}^\top \mathbf{V} \mathbf{w} \leq 35\epsilon \text{ for all } \mathbf{w} \in \mathbf{W}. \end{aligned} \quad (\star)$$

Given  $(\star)$  as formalized later in Claim 3.16, we can conclude Lemma 3.8 because, from the definition of  $\mathbf{A}$  and  $\tilde{\mathbf{A}}$ , we have that for some  $\mathbf{w} \in \mathbf{W}$ ,  $\mathbf{w}^\top \mathbf{A} \mathbf{w} \geq 1 - 5\epsilon$  by Corollary 3.4 and  $\mathbf{w}^\top \tilde{\mathbf{A}} \mathbf{w} < 1 - 40\epsilon$ . As a result for this  $\mathbf{w}$ ,  $\mathbf{w}^\top \mathbf{V} \mathbf{w} > 35\epsilon$ . Now, by contra-position of  $(\star)$ , we have that  $\dim(T_\nu - \tilde{T})$  is large for some  $\nu \in \mathcal{I}$ .

To prove  $(\star)$ , we further decompose  $\mathbf{w}^\top \mathbf{V} \mathbf{w}$  as

$$\mathbf{w}^\top \mathbf{V} \mathbf{w} = \mathbf{w}^\top \mathbf{V}_{\tilde{T}} \mathbf{w} + \sum_{\nu=0}^{15 \log \frac{n}{\epsilon} - 1} \mathbf{w}^\top \mathbf{V}_{T_\nu - \tilde{T}} \mathbf{w} + \mathbf{w}^\top \mathbf{V}_{\bar{T}} \mathbf{w}.$$

In the above equation,  $\mathbf{V}_{\tilde{T}} = \Pi_{\tilde{T}} \mathbf{V} \Pi_{\tilde{T}}$ ,  $\mathbf{V}_{T_\nu - \tilde{T}} = \Pi_\nu \mathbf{V} \Pi_\nu$ , and  $\mathbf{V}_{\bar{T}} = \Pi_{\bar{T}} \mathbf{V} \Pi_{\bar{T}}$  where  $\Pi_{\tilde{T}}$ ,  $\Pi_\nu$ ,  $\Pi_{\bar{T}}$  denote the projections matrices that project any vector onto the spaces  $\tilde{T}$ ,  $T_\nu - \tilde{T}$ , and  $\bar{T}$  respectively<sup>4</sup>. Refer to Section 3.4 for proof of why such a split is possible. Our proof of  $(\star)$  then bounds the terms on the right-hand side. Let us consider each term separately.

1.  $\mathbf{w}^\top \mathbf{V}_{\tilde{T}} \mathbf{w}$ : We prove that this is always at most  $10\epsilon(1 + \epsilon)$  (Equation (7)). From the definition of  $\mathbf{V}$ ,

$$\mathbf{w}^\top \mathbf{V}_{\tilde{T}} \mathbf{w} = \mathbf{w}^\top \Pi_{\tilde{T}} \mathbf{A} \Pi_{\tilde{T}} \mathbf{w} - \mathbf{w}^\top \Pi_{\tilde{T}} \tilde{\mathbf{A}} \Pi_{\tilde{T}} \mathbf{w}.$$

<sup>4</sup>Suppose a subspace  $S$  is spanned by vectors  $\mathbf{u}_1, \dots, \mathbf{u}_k$ . Let  $\mathbf{U} = [\mathbf{u}_1, \dots, \mathbf{u}_k]$ . Recall that the projection matrix onto  $S$  is  $\mathbf{U}(\mathbf{U}^\top \mathbf{U})^{-1} \mathbf{U}^\top$ .

Since  $\Pi_{\tilde{T}}\mathbf{w}$  is the projection of  $\mathbf{w}$  along the large eigenspace of  $\tilde{\mathbf{A}}$ , the second term on the right-hand side above is large, i.e.  $\geq (1 - 10\epsilon)\lambda_0\|\Pi_{\tilde{T}}\mathbf{w}\|^2$ . The first term on the right-hand side can be bounded as,  $\mathbf{w}^\top \Pi_{\tilde{T}}\mathbf{A}\Pi_{\tilde{T}}\mathbf{w} \leq \|\mathbf{A}\|\|\Pi_{\tilde{T}}\mathbf{w}\|^2 \leq \lambda_0\|\Pi_{\tilde{T}}\mathbf{w}\|^2$ . Therefore the difference on the right-hand side is at most  $10\epsilon\lambda_0\|\Pi_{\tilde{T}}\mathbf{w}\|^2 \leq 10\epsilon\lambda_0\|\mathbf{w}\|^2 = 10\epsilon\lambda_0 \leq 10\epsilon(1 + \epsilon)$ .

2.  $\mathbf{w}^\top \mathbf{V}_{\tilde{T}}\mathbf{w}$ : Observe that this term corresponds to the projection of  $\mathbf{w}$  along the space spanned by the eigenvalues of  $\mathbf{A}$  of size at most  $1 - 3\epsilon$ . Let  $\mathbf{u}_i$  and  $\lambda_i$  denote an eigenvector and eigenvalue pair with  $\lambda_i < 1 - 3\epsilon$ . Since the power method can guarantee that  $\mathbf{w}^\top \mathbf{u}_i \approx \lambda_i^{2K}$ , we have  $\lambda_i^{2K} \leq (1 - 3\epsilon)^{2K} \leq \text{poly}(\frac{\epsilon}{n})$  is tiny. So we have that  $\mathbf{w}^\top \mathbf{V}_{\tilde{T}}\mathbf{w} \leq \epsilon$  (Lemma 3.18).

Before we look at the final case, we define a basis for the space  $T_\nu$ .

**Definition 3.9** (Basis for  $T_\nu$ ). *Let  $T_\nu$  be as defined in Definition 3.6. Define indices  $a_\nu$  and  $b_\nu$  with  $b_\nu - a_\nu + 1 = d_\nu$  such that the basis of  $T_\nu$  is given by  $\mathbf{u}_{a_\nu}, \dots, \mathbf{u}_{b_\nu}$ , where  $\mathbf{u}_1, \mathbf{u}_2, \dots, \mathbf{u}_n$  are the eigenvectors of  $\mathbf{A}$  in decreasing order of eigenvalues.*

3.  $\mathbf{w}^\top \mathbf{V}_{T_\nu - \tilde{T}}\mathbf{w}$ : For this discussion, we will ignore the constant factors and assume that the high probability events hold. Let  $\Pi_\nu$  denote the projection matrix for the space  $T_\nu - \tilde{T}$ . Observe that  $\mathbf{w}^\top \mathbf{V}_{T_\nu - \tilde{T}}\mathbf{w} = \mathbf{w}^\top \Pi_\nu \mathbf{V} \Pi_\nu \mathbf{w} \leq \|\mathbf{V}\|\|\Pi_\nu \mathbf{w}\|^2 \leq (1 + \epsilon)\|\Pi_\nu \mathbf{w}\|^2$ , where the last inequality is because  $\tilde{\mathbf{A}} = \mathbf{A} - \mathbf{V} \succeq 0$ , and therefore,  $\|\mathbf{V}\| \leq \|\mathbf{A}\| \leq (1 + \epsilon)$ . Hence, it suffices to bound  $\|\Pi_\nu \mathbf{w}\|^2 = O(\epsilon)$ .

We can write  $\mathbf{w} = \frac{\sum_{i=1}^n \lambda_i^K \alpha_i \mathbf{u}_i}{\sqrt{\sum_{i=1}^n \lambda_i^{2K} \alpha_i^2}}$  where  $\lambda_i, \mathbf{u}_i$ 's are the eigenvalues and eigenvectors of  $\mathbf{A}$  and  $\alpha_i \sim N(0, 1)$ . Define  $\mathbf{z} = \sum_{i=1}^n z_i \mathbf{u}_i$  where  $z_i = \lambda_i^K \alpha_i$ . That is,  $\mathbf{w} = \frac{\mathbf{z}}{\|\mathbf{z}\|}$ . Since  $\|\Pi_\nu \mathbf{w}\| = \|\Pi_\nu \mathbf{z}\|/\|\mathbf{z}\|$ , it suffices to show that  $\|\Pi_\nu \mathbf{z}\|^2 \leq O(\epsilon)\|\mathbf{z}\|^2$ . We show this in two separate cases. In both cases, we start with the following bound

$$\|\Pi_\nu \mathbf{z}\|^2 \leq \lambda_{a_\nu}^{2K} \cdot \dim(T_\nu - \tilde{T}),$$

which holds with high probability. To see this, let  $\mathbf{g}_\nu \sim N(0, 1)$  be a gaussian vector in the space  $T_\nu - \tilde{T}$ . We can couple  $\mathbf{g}_\nu$  with  $\Pi_\nu \mathbf{z}$  so that  $\Pi_\nu \mathbf{z}$  is dominated by  $\lambda_{a_\nu}^K \cdot \mathbf{g}_\nu$ . So  $\|\Pi_\nu \mathbf{z}\|^2 \leq \lambda_{a_\nu}^{2K} \|\mathbf{g}_\nu\|^2$ . By Lemma 2.4, the norm square of gaussian vector is concentrated to its dimension so  $\|\mathbf{g}_\nu\|^2 \leq \dim(T_\nu - \tilde{T})$  with high probability, thus proving the inequality. Next, we will bound  $\dim(T_\nu - \tilde{T})$  in terms of  $\|\mathbf{z}\|$  in two cases.

**When  $\nu \notin \mathcal{I}$  (Lemma 3.21):** From the definition of the important levels, we have

$$\dim(T_\nu - \tilde{T}) \leq d_\nu \leq \frac{O(\epsilon)}{\log^3 \frac{n}{\epsilon}} \sum_{\nu' < \nu} d_{\nu'}.$$

Now, we have  $\sum_{\nu' < \nu} d_{\nu'} \approx \sum_{i=1}^{b_{\nu-1}} \alpha_i^2$  because  $\alpha_i \sim N(0, 1)$  is gaussian and the norm square of gaussian vector is concentrated to its dimension (Lemma 2.4). Since  $\alpha_i = z_i/\lambda_i^K$ , we have that

$$\sum_{\nu' < \nu} d_{\nu'} \approx \sum_{i=1}^{b_{\nu-1}} \alpha_i^2 = \sum_{i=1}^{b_{\nu-1}} \frac{z_i^2}{\lambda_i^{2K}} \leq \|\mathbf{z}\|^2 / \lambda_{b_{\nu-1}}^{2K}.$$

Therefore, we have

$$\|\Pi_\nu \mathbf{z}\|^2 \leq \lambda_{a_\nu}^{2K} \dim(T_\nu - \tilde{T}) \leq \left( \frac{\lambda_{a_\nu}}{\lambda_{b_{\nu-1}}} \right)^{2K} \frac{O(\epsilon)}{\log^3 \frac{n}{\epsilon}} \|\mathbf{z}\|^2 \leq O(\epsilon) \|\mathbf{z}\|^2$$

where the last inequality is trivial because  $\lambda_{b_{\nu-1}} \geq \lambda_{a_\nu}$  by definition.

**When  $\nu \in \mathcal{I}$  (Lemma 3.20):** In this case, according to  $(\star)$ , we can assume

$$\dim(T_\nu - \tilde{T}) \lesssim \epsilon d_\nu.$$

Again, by Lemma 2.4, we have that  $d_\nu \approx \sum_{i=a_\nu}^{b_\nu} \alpha_i^2$  because  $\alpha_i \sim N(0, 1)$  is gaussian. Since  $\alpha_i = z_i / \lambda_i^K$ , we have

$$d_\nu \approx \sum_{i=a_\nu}^{b_\nu} \alpha_i^2 = \sum_{i=a_\nu}^{b_\nu} \frac{z_i^2}{\lambda_i^{2K}} \leq \|\mathbf{z}\|^2 / \lambda_{b_\nu}^{2K}.$$

Therefore,

$$\|\Pi_\nu \mathbf{z}\|^2 \leq \lambda_{a_\nu}^{2K} \dim(T_\nu - \tilde{T}) \leq \left( \frac{\lambda_{a_\nu}}{\lambda_{b_\nu}} \right)^{2K} \epsilon \|\mathbf{z}\|^2 \leq O(\epsilon) \|\mathbf{z}\|^2$$

where the last inequality is because  $\left( \frac{\lambda_{a_\nu}}{\lambda_{b_\nu}} \right)^{2K} \leq \left( \frac{1 - \frac{\nu\epsilon}{5 \log \frac{n}{\epsilon}}}{1 - \frac{(\nu+1)\epsilon}{5 \log \frac{n}{\epsilon}}} \right)^{2K} \approx \left( 1 + \frac{\epsilon}{2 \log \frac{n}{\epsilon}} \right)^{2K} \approx O(1)$ .

From these three cases, we can conclude that if  $\dim(T_\nu - \tilde{T})$  is small for all  $\nu \in \mathcal{I}$ , then  $\mathbf{w}^\top \mathbf{V} \mathbf{w} \leq 35\epsilon$ , proving our claim.

In the remaining sections, we give formal proofs of the claims made in this section. In Section 3.2, we prove the main result, Theorem 1.2, assuming the key lemma. In Section 3.3, we prove the key lemma, Lemma 3.5, assuming the Lemma 3.8. Finally, we prove Lemma 3.8 in Section 3.4

### 3.2 Proof of the Main Theorem 1.2 assuming the Key Lemma 3.5

Here, we formally prove Theorem 1.2 assuming the key Lemma 3.5. We will first prove the correctness and then bound the total runtime.

**Correctness.** The following formalizes the correctness guarantee of Algorithm 3.

**Lemma 3.10.** *Let  $\epsilon > 1/n$ . With probability at least  $1 - 1/n$ , the following holds for all time step  $t \geq 1$ : if the maximum eigenvalue of  $\mathbf{A}_t$  is at least  $1 - \frac{\epsilon}{\log n}$ ,  $\text{UPDATE}(\mathbf{v}_t)$  returns  $[r_t, \mathbf{W}_t]$ .*

*Proof.* We upper bound the probability that Algorithm  $\text{UPDATE}(\mathbf{v}_t)$  returns FALSE. This can happen only when Line 4 is executed and  $\text{POWERMETHOD}(\epsilon, \mathbf{A}_t)$  returns FALSE, which happens with probability at most  $1/n^{10}$  by Corollary 3.4. By Lemma 3.5, we execute Line 4 at most  $O(\log n \log^5 \frac{n}{\epsilon} / \epsilon^2)$  times throughout the whole update sequence. Therefore, by union bounds,  $\text{UPDATE}(\mathbf{v}_t)$  may return FALSE with probability at most  $\text{poly}(\log(n)/\epsilon)/n^{10} \leq 1/n$ .  $\square$

**Runtime.** Next, we bound the runtime of the various lines of Algorithm 3.

**Lemma 3.11.** *For a fixed  $\mathbf{w}$  and any  $t$ , we can update  $\mathbf{w}^\top \mathbf{A}_{t-1} \mathbf{w}$  to  $\mathbf{w}^\top \mathbf{A}_t \mathbf{w}$  in time  $O(\text{nnz}(\mathbf{v}_t))$ .*

*Proof.* Note that,

$$\mathbf{w}^\top \mathbf{A}_t \mathbf{w} = \mathbf{w}^\top \mathbf{A}_{t-1} \mathbf{w} - (\mathbf{v}_t^\top \mathbf{w})^2.$$

The second term can be computed in time  $O(\text{nnz}(\mathbf{v}_t))$ .  $\square$

**Lemma 3.12.** *Fix time  $t$ . Given  $\mathbf{w}$  as input, we have that  $\mathbf{w}^\top \mathbf{A}_t \mathbf{w}$  and  $\mathbf{A}_t \mathbf{w}$  can be computed  $O(\text{nnz}(\mathbf{A}_0) + \sum_{i=1}^t \text{nnz}(\mathbf{v}_i))$  time.*

*Proof.* We can split  $\mathbf{A}_t$ , and get,

$$\mathbf{w}^\top \mathbf{A}_t \mathbf{w} = \mathbf{w}^\top \mathbf{A}_0 \mathbf{w} - \sum_{i=1}^t (\mathbf{v}_i^\top \mathbf{w})^2, \quad \text{and,} \quad \mathbf{A}_t \mathbf{w} = \mathbf{A}_0 \mathbf{w} - \sum_{i=1}^t \mathbf{v}_i (\mathbf{v}_i^\top \mathbf{w}).$$

The first term in both expressions can be computed in time  $O(\text{nnz}(\mathbf{A}_0))$  and for every  $i$  we can compute  $\mathbf{v}_i^\top \mathbf{w}$  in time  $O(\text{nnz}(\mathbf{v}_i))$ , thus concluding the proof.  $\square$

**Lemma 3.13.** *For any time  $t$ , we can implement  $\text{POWERMETHOD}(\epsilon, \mathbf{A}_t)$  in time at most*

$$O\left(\frac{\log n \log \frac{n}{\epsilon}}{\epsilon} \left(\text{nnz}(\mathbf{A}_0) + \sum_{i=1}^t \text{nnz}(\mathbf{v}_i)\right)\right).$$

*Proof.* When we call  $\text{POWERMETHOD}(\epsilon, \mathbf{A}_t)$  from Algorithm 1, we can to compute  $\mathbf{A}_t \mathbf{w}$  for some vector  $\mathbf{w}$  for  $R \cdot K$  times and, at the end, compute  $\mathbf{w}^\top \mathbf{A}_t \mathbf{w}$  for some vector  $\mathbf{w}$  for  $R$  times. By Lemma 3.12, this takes  $O((\text{nnz}(\mathbf{A}_0) + \sum_{i=1}^t \text{nnz}(\mathbf{v}_i)) \cdot \log n \log(\frac{n}{\epsilon})/\epsilon)$  because  $R = O(\log n)$  and  $K = O(\log(\frac{n}{\epsilon})/\epsilon)$ .  $\square$

Given the above results, we can now prove Theorem 1.2.

### Proof of Theorem 1.2

*Proof.* We will prove that Algorithms 2 and 3 solve Problem 3.1, which implies an algorithm for Problem 1.1 by Lemma 3.2.

From Corollary 3.4, Algorithm 2 solves Problem 3.1 for the initial matrix  $\mathbf{A}_0$  with probability at least  $1 - 1/n$ . From Lemma 3.10, Algorithm 3 returns  $[r_t, \mathbf{W}_t]$  for all  $t \geq 1$ , whenever the maximum eigenvalue of  $\mathbf{A}_t$  is at least  $1 - \epsilon/\log n$ , with probability at least  $1 - 1/n$ . Note that,  $\mathbf{w}^{(r_t)} \in \mathbf{W}_t$  is the required solution. Therefore, our algorithm returns the correct solution with a probability of at least  $1 - 2/n$ .

We now bound the total runtime of the algorithm over all time  $t$ . Define an indicator function  $I_t$  as  $I_t = 1$ , if we execute Line 4 in  $\text{UPDATE}(\mathbf{v}_t)$  and 0 otherwise. From Lemmas 3.11 and 3.12, executing Line 3 at  $t$  requires time  $O(\text{nnz}(\mathbf{v}_t) \log n)$  if  $I_{t-1} = 0$  and time  $O\left(\frac{\log n \log \frac{n}{\epsilon}}{\epsilon} \left(\text{nnz}(\mathbf{A}_0) + \sum_{i=1}^t \text{nnz}(\mathbf{v}_i)\right)\right)$

if  $I_{t-1} = 1$ . Summing these up, the total time  $\mathcal{T}$  required for solving Problem 3.1 is at most,

$$\begin{aligned}
\mathcal{T} &\leq O \left( \underbrace{\frac{\log n \log \frac{n}{\epsilon} \cdot \text{nnz}(\mathbf{A}_0)}{\epsilon}}_{\text{Time for INIT}(\epsilon, \mathbf{A}_0)} + \sum_t I_t \underbrace{\frac{\log n \log \frac{n}{\epsilon}}{\epsilon} \left( \text{nnz}(\mathbf{A}_0) + \sum_{i=1}^t \text{nnz}(\mathbf{v}_i) \right)}_{\text{Time required at } t \text{ when } I_t = 1} + \sum_t (1 - I_t) \underbrace{\log n \cdot \text{nnz}(\mathbf{v}_t)}_{\text{Time when } I_t = 0} \right) \\
&\leq O \left( \frac{\log n \log \frac{n}{\epsilon} \cdot \text{nnz}(\mathbf{A}_0)}{\epsilon} + \sum_t I_t \frac{\log n \log \frac{n}{\epsilon}}{\epsilon} \left( \text{nnz}(\mathbf{A}_0) + \sum_{i=1}^t \text{nnz}(\mathbf{v}_i) \right) + \sum_t \log n \cdot \text{nnz}(\mathbf{v}_t) \right) \\
&\stackrel{(a)}{\leq} O \left( \frac{\log n \log^2 \frac{n}{\epsilon} \cdot \text{nnz}(\mathbf{A}_0)}{\epsilon} + \frac{\log n \log^5 \frac{n}{\epsilon}}{\epsilon^2} \frac{\log n \log \frac{n}{\epsilon}}{\epsilon} \left( \text{nnz}(\mathbf{A}_0) + \sum_{i=1}^T \text{nnz}(\mathbf{v}_i) \right) + \sum_{t=1}^T \log n \cdot \text{nnz}(\mathbf{v}_t) \right) \\
&\stackrel{(b)}{\leq} O \left( \frac{\log^2 n \log^6 \frac{n}{\epsilon}}{\epsilon^3} \left( \text{nnz}(\mathbf{A}_0) + \sum_{i=1}^T \text{nnz}(\mathbf{v}_i) \right) \right),
\end{aligned}$$

with probability at least  $1 - \frac{1}{n}$ . In (a), we used Lemma 3.5 which states that the maximum number of  $t$  for which  $I_t = 1$  is at most  $O(\log n \log^5 \frac{n}{\epsilon} / \epsilon^2)$  with probability at least  $1 - 1/n$ . In (b), we combined all the terms. Therefore, by Lemma 3.2, our algorithms solve Problem 1.1 with probability at least  $1 - 3/n$  in total time

$$\begin{aligned}
O \left( \frac{\log^2 n \log \frac{n}{\epsilon}}{\epsilon} \cdot \text{nnz}(\mathbf{A}_0) + \frac{\log n}{\epsilon} \log \frac{\lambda_{\max}(\mathbf{A}_0)}{\lambda_{\max}(\mathbf{A}_T)} \mathcal{T} \right) &\leq \\
O \left( \frac{\log^3 n \log^6 \frac{n}{\epsilon} \log \frac{\lambda_{\max}(\mathbf{A}_0)}{\lambda_{\max}(\mathbf{A}_T)}}{\epsilon^4} \left( \text{nnz}(\mathbf{A}_0) + \sum_{i=1}^T \text{nnz}(\mathbf{v}_i) \right) \right).
\end{aligned}$$

□

### 3.3 Proof of the Key Lemma 3.5: Few Executions of Line 4

In this section, we prove Lemma 3.5 assuming the Progress Lemma 3.8. Let us first recall the precise definition of  $\mathbf{A}$  and  $\tilde{\mathbf{A}}$ . Suppose we execute Line 4 at update  $t_0$ . Now consider a sequence of updates,  $\mathbf{v}_{t_0+1}, \dots, \mathbf{v}_{t_0+k}$ , and let  $\mathbf{A}_{t_0+k} = \mathbf{A}_{t_0} - \sum_{i=1}^k \mathbf{v}_{t_0+i} \mathbf{v}_{t_0+i}^\top$ . Suppose the next execution of Line 4 happens at  $t_0 + k$ . For this to happen we must have that  $\mathbf{w}_{t_0}^\top \mathbf{A}_{t_0+k} \mathbf{w}_{t_0} < 1 - 40\epsilon$  for all  $\mathbf{w}_{t_0} \in \mathbf{W}_{t_0}$ . We let  $\mathbf{A} = \mathbf{A}_{t_0}$  and  $\tilde{\mathbf{A}} = \mathbf{A}_{t_0+k}$ .

Next, recall Definition 3.6 and define  $T_{\leq i} = \sum_{\nu=0}^i T_\nu$ ,  $\tilde{T}_{\leq i} = \sum_{\nu=0}^i \tilde{T}_\nu$ . The following observation will motivate our potential function analysis.

**Lemma 3.14.** *For all  $i \leq 15 \log \frac{n}{\epsilon}$ ,*

$$\dim(T_{\leq i}) \geq \dim(\tilde{T}_{\leq i}).$$

Let  $\nu_0$  be such that  $\dim(T_{\nu_0} - \tilde{T}) > 0$ . For any  $i \geq \nu_0$ , we have

$$\dim(T_{\leq i}) \geq \dim(\tilde{T}_{\leq i}) + \dim(T_{\nu_0} - \tilde{T}).$$

*Proof.* We claim that  $\tilde{T}_{\leq i} \subseteq T_{\leq i}$ , which implies the first claim. This is because the updates are decreasing. So, if  $\mathbf{v}^\top \tilde{\mathbf{A}} \mathbf{v} \geq 1 - \frac{(i+1)\epsilon}{5 \log \frac{n}{\epsilon}}$  then  $\mathbf{v}^\top \mathbf{A} \mathbf{v} \geq 1 - \frac{(i+1)\epsilon}{5 \log \frac{n}{\epsilon}}$ . That is, if  $\mathbf{v} \in \tilde{T}_{\leq i}$ , then  $\mathbf{v} \in T_{\leq i}$ . For the second part, we have  $T_{\leq i} = T_{\leq i} \cap \tilde{T}_{\leq i} + (T_{\leq i} - \tilde{T}_{\leq i}) \supseteq \tilde{T}_{\leq i} + (T_{\nu_0} - \tilde{T})$  because  $T_{\nu_0} \subseteq T_{\leq i}$  and  $\tilde{T}_{\leq i} \subseteq \tilde{T}$ . Since  $\tilde{T}_{\leq i} \cap (T_{\nu_0} - \tilde{T}) = \emptyset$ , we can conclude the second part.  $\square$

**The Potentials.** The above lemma and Lemma 3.8 motivate the following potentials. For every  $j \leq 15 \log \frac{n}{\epsilon}$ , we define the potentials  $\Phi_j = \dim(T_{\leq j})$  and  $\tilde{\Phi}_j = \dim(\tilde{T}_{\leq j})$ . For all  $j$ , the potential may only decrease, i.e.,  $\tilde{\Phi}_j \leq \Phi_j$  by Lemma 3.14. Also, clearly,  $\Phi_j \leq n$ .

We will show that for each execution of Line 4,  $\tilde{\Phi}_j$  decreases from  $\Phi_j$  by a significant factor with high probability for some  $j$ . This will bound the number of executions.

Consider any important level  $\nu_0 \in \mathcal{I}$ . We have two observations:

1.  $\tilde{\Phi}_{\nu_0} \leq \Phi_{\nu_0} - \dim(T_{\nu_0} - \tilde{T})$ , and
2.  $d_{\nu_0} \geq \Omega(1) \frac{\epsilon}{\log^3 \frac{n}{\epsilon}} \Phi_{\nu_0}$ .

The first point follows directly from the second part of Lemma 3.14. Moreover, since  $\nu_0 \in \mathcal{I}$  is important, we have  $d_{\nu_0} \geq \frac{\epsilon}{600 \log^3 \frac{n}{\epsilon}} \sum_{\nu' < \nu} d_{\nu'}$ . Therefore,

$$\Phi_{\nu_0} = \sum_{\nu=1}^{\nu_0} d_{\nu} = \sum_{\nu' < \nu_0} d_{\nu'} + d_{\nu_0} \leq \left( \frac{600 \log^3 \frac{n}{\epsilon}}{\epsilon} + 1 \right) d_{\nu_0},$$

implying the second point. Combining these observations with the Progress Lemma 3.8, we have:

**Lemma 3.15.** *Suppose that  $\lambda_{\max}(\mathbf{A}) \geq 1 - \epsilon$  and  $\mathbf{w}^\top \tilde{\mathbf{A}} \mathbf{w} < 1 - 40\epsilon$  for all  $\mathbf{w} \in \mathbf{W}$ . With probability at least  $1 - 50 \log \frac{n}{\epsilon} / n^2$ , there is a level  $\nu_0$  such that either*

- $\tilde{\Phi}_{\nu_0} \leq \left( 1 - \frac{\Omega(\epsilon^2)}{\log^4 \frac{n}{\epsilon}} \right) \Phi_{\nu_0}$ , or
- $\tilde{\Phi}_{\nu_0} \leq \Phi_{\nu_0} - 1$  and  $\Phi_{\nu_0} \leq O(1) \frac{\log^4 \frac{n}{\epsilon} \log n}{\epsilon^2}$ .

*Proof.* Given the assumption, it follows from Lemma 3.8 that with probability at least  $1 - 50 \log \frac{n}{\epsilon} / n^2$ , there is an important level  $\nu_0 \in \mathcal{I}$  be such that either  $d_{\nu_0} \geq \frac{3000 \log n \log \frac{n}{\epsilon}}{\epsilon}$  and  $\dim(T_{\nu_0} - \tilde{T}) \geq \frac{\epsilon}{300 \log \frac{n}{\epsilon}} d_{\nu_0}$  or  $d_{\nu_0} < \frac{3000 \log n \log \frac{n}{\epsilon}}{\epsilon}$  and  $\dim(T_{\nu_0} - \tilde{T}) \geq 1$ . So, by calculation, we have the following.

- If  $d_{\nu_0} \geq \frac{3000 \log n \log \frac{n}{\epsilon}}{\epsilon}$ , we have  $\tilde{\Phi}_{\nu_0} \leq \Phi_{\nu_0} - \frac{\epsilon}{300 \log \frac{n}{\epsilon}} d_{\nu_0} \leq \Phi_{\nu_0} (1 - \frac{\Omega(\epsilon^2)}{\log^4 \frac{n}{\epsilon}})$  where the first inequality is by (1) and Lemma 3.8. The second is by (2).
- If  $d_{\nu_0} < \frac{3000 \log n \log \frac{n}{\epsilon}}{\epsilon}$ , we have  $\tilde{\Phi}_{\nu_0} \leq \Phi_{\nu_0} - 1$  by (1) and Lemma 3.8. In this case, we also have  $\Phi_{\nu_0} \leq O(1) \frac{\log^4 \frac{n}{\epsilon} \log n}{\epsilon^2}$  by (2).

$\square$

We are now ready to prove Lemma 3.5.



### Proof of Lemma 3.5.

First, observe that whenever  $\lambda_{\max}(\mathbf{A}) < 1 - \epsilon$ , by Line 10 of Algorithm 1, we will always return FALSE at the next execution of Line 4 of Algorithm 3.

Therefore, it suffices to bound the number of executions while  $\lambda_{\max}(\mathbf{A}) \geq 1 - \epsilon$ . We execute Line 4 only if  $\mathbf{w}^\top \tilde{\mathbf{A}} \mathbf{w} < 1 - 40\epsilon$  for all  $\mathbf{w} \in \mathbf{W}$ . When this happens, there exists a level  $j$  where the potential  $\Phi_j$  significantly decreases according to Lemma 3.15 with probability at least  $1 - 50 \log \frac{n}{\epsilon} / n^2$ . For each level  $j$ , this can happen at most  $L = O(\frac{\log n \log^4 \frac{n}{\epsilon}}{\epsilon^2})$  times because for every  $j$ ,  $\Phi_j$  is an integer that may only decrease and is bounded by  $n$ . Suppose for contradiction that there are more than  $L \times 15 \log \frac{n}{\epsilon}$  executions of Line 4. So, with probability at least  $1 - L \cdot 50 \log \frac{n}{\epsilon} / n^2 \geq 1/n$ , there exists a level  $j$  where  $\Phi_j$  decreases according to Lemma 3.15 strictly more than  $L$  times. This is a contradiction.

### 3.4 Proof of the Progress Lemma 3.8

It remains to prove the Progress Lemma. We first restate the lemma here.

**Lemma 3.8** (Measure of Progress). *Let  $\epsilon > 0$  and let  $\mathbf{W} = [\mathbf{w}^{(1)}, \dots, \mathbf{w}^{(R)}]$  be as defined in Line 9 in the execution of POWERMETHOD( $\epsilon, \mathbf{A}$ ). Let  $\mathbf{v}_1, \dots, \mathbf{v}_k$  be a sequence of updates generated by an oblivious adversary and define  $\tilde{\mathbf{A}} = \mathbf{A} - \sum_{i=1}^k \mathbf{v}_i \mathbf{v}_i^\top$ .*

*Suppose that  $\lambda_{\max}(\mathbf{A}) \geq 1 - \epsilon$  and  $\mathbf{w}^\top \tilde{\mathbf{A}} \mathbf{w} < 1 - 40\epsilon$  for all  $\mathbf{w} \in \mathbf{W}$ . Then, with probability at least  $1 - \frac{50 \log \frac{n}{\epsilon}}{n^2}$ , for some  $\nu \in \mathcal{I}$ ,*

- $\dim(T_\nu - \tilde{T}) \geq \frac{\epsilon}{300 \log \frac{n}{\epsilon}} d_\nu$  if  $d_\nu \geq \frac{3000 \log n \log \frac{n}{\epsilon}}{\epsilon}$ , or
- $\dim(T_\nu - \tilde{T}) \geq 1$  if  $d_\nu < \frac{3000 \log n \log \frac{n}{\epsilon}}{\epsilon}$ .

Recall the definitions of subspaces  $T, T_\nu, \tilde{T}$  and  $\bar{T}$  in Definition 3.6. We will first state the following claim and show that this is sufficient to prove Lemma 3.8. After concluding the proof of Lemma 3.8, we would prove the claim.

**Claim 3.16.** *Suppose that  $\lambda_{\max}(\mathbf{A}) \geq 1 - \epsilon$ . If for every  $\nu \in \mathcal{I}$ ,*

- $\dim(T_\nu - \tilde{T}) < \frac{\epsilon}{300 \log \frac{n}{\epsilon}} d_\nu$  if  $d_\nu \geq \frac{3000 \log n \log \frac{n}{\epsilon}}{\epsilon}$ , and
- $\dim(T_\nu - \tilde{T}) < 1$  if  $d_\nu < \frac{3000 \log n \log \frac{n}{\epsilon}}{\epsilon}$ .

*Then, with probability at least  $1 - \frac{40 \log \frac{n}{\epsilon}}{n^2}$ ,  $\mathbf{w}^\top \mathbf{V} \mathbf{w} \leq 35\epsilon$  for all  $\mathbf{w} \in \mathbf{W}$ .*

### Proof of Lemma 3.8 using Claim 3.16

Suppose for contradiction that Lemma 3.8 does not hold, i.e., the conditions on  $\dim(T_\nu - \tilde{T})$  of Claim 3.16 hold for all  $\nu \in \mathcal{I}$ . On one hand, since  $\lambda_{\max}(\mathbf{A}) \geq 1 - \epsilon$ , Claim 3.16 says that  $\mathbf{w}^\top \mathbf{V} \mathbf{w} \leq 35\epsilon$  for all  $\mathbf{w} \in \mathbf{W}$  with probability at least  $1 - \frac{40 \log \frac{n}{\epsilon}}{n^2}$ . From the assumption of Lemma 3.8, we have  $\mathbf{w}^\top \tilde{\mathbf{A}} \mathbf{w} < 1 - 40\epsilon$  for all  $\mathbf{w} \in \mathbf{W}$  as well, this implies that, for all  $\mathbf{w} \in \mathbf{W}$ ,

$$\mathbf{w}^\top \mathbf{A} \mathbf{w} < \mathbf{w}^\top \tilde{\mathbf{A}} \mathbf{w} + \mathbf{w}^\top \mathbf{V} \mathbf{w} < 1 - 5\epsilon.$$

On the other hand, since  $\lambda_{\max}(\mathbf{A}) \geq 1 - \epsilon$ , we must have  $\mathbf{w}^\top \mathbf{A} \mathbf{w} \geq 1 - 5\epsilon$  for some  $\mathbf{w} \in \mathbf{W}$  with probability at least  $1 - 1/n^2$  by Corollary 3.4. This gives a contradiction.

Therefore, we conclude that, with probability at least  $1 - \frac{40 \log \frac{n}{\epsilon} + 1}{n^2}$ , the conditions of Claim 3.16 must be false for some  $\nu \in \mathcal{I}$ . That is, we have

- $\dim(T_\nu - \tilde{T}) \geq \frac{\epsilon}{300 \log \frac{n}{\epsilon}} d_\nu$  if  $d_\nu \geq \frac{3000 \log n \log \frac{n}{\epsilon}}{\epsilon}$ , and
- $\dim(T_\nu - \tilde{T}) \geq 1$  if  $d_\nu < \frac{3000 \log n \log \frac{n}{\epsilon}}{\epsilon}$ .

This concludes the proof of Lemma 3.8.

### Setting Up for the Proof of Claim 3.16

Recall the definitions of subspaces  $T, T_\nu, \tilde{T}$  and  $\bar{T}$  in Definition 3.6.

**Proposition 3.17.** *We can cover the entire space with the following subspaces*

$$\mathbb{R}^n = T + \bar{T} = \tilde{T} + \sum_{\nu=0}^{15 \log \frac{n}{\epsilon} - 1} (T_\nu - \tilde{T}) + \bar{T} \quad (3)$$

where all subspaces in the sum are mutually orthogonal.

*Proof.* It suffices to show that  $T = \tilde{T} + \sum_{\nu=0}^{15 \log \frac{n}{\epsilon} - 1} (T_\nu - \tilde{T})$ . To see this, note that  $\tilde{T} \subseteq T$ . Therefore, we have  $T = (T - \tilde{T}) + T \cap \tilde{T} = (T - \tilde{T}) + \tilde{T}$ . We also know that  $T = \sum_{\nu=0}^{15 \log \frac{n}{\epsilon} - 1} T_\nu$  and this gives  $T - \tilde{T} = \sum_{\nu=0}^{15 \log \frac{n}{\epsilon} - 1} (T_\nu - \tilde{T})$ , which concludes the proof.  $\square$

**Notation.** The goal of Claim 3.16 is to bound  $\mathbf{w}^\top \mathbf{V} \mathbf{w} \leq 35\epsilon$  for all  $\mathbf{w} \in \mathbf{W}$ . We will use the following notations.

- Let  $\Pi_{\tilde{T}}, \Pi_\nu, \Pi_{\bar{T}}$  denote projection matrices to the subspaces  $\tilde{T}, T_\nu - \tilde{T}$ , and  $\bar{T}$  respectively.
- Define  $\mathbf{V}_{\tilde{T}} = \Pi_{\tilde{T}} \mathbf{V} \Pi_{\tilde{T}}$ ,  $\mathbf{V}_{T_\nu - \tilde{T}} = \Pi_\nu \mathbf{V} \Pi_\nu$ , and  $\mathbf{V}_{\bar{T}} = \Pi_{\bar{T}} \mathbf{V} \Pi_{\bar{T}}$ .

By Proposition 3.17, for any  $\mathbf{w} \in \mathbf{W}$ , we can decompose  $\mathbf{w}^\top \mathbf{V} \mathbf{w}$  as

$$\mathbf{w}^\top \mathbf{V} \mathbf{w} = \mathbf{w}^\top \mathbf{V}_{\tilde{T}} \mathbf{w} + \sum_{\nu=0}^{15 \log \frac{n}{\epsilon} - 1} \mathbf{w}^\top \mathbf{V}_{T_\nu - \tilde{T}} \mathbf{w} + \mathbf{w}^\top \mathbf{V}_{\bar{T}} \mathbf{w}.$$

Our strategy is to upper bound each term one by one. Bounding  $\mathbf{w}^\top \mathbf{V}_{\tilde{T}} \mathbf{w}$  is straightforward, but bounding other terms requires technical helper lemmas. Lemma 3.18 is needed for bounding  $\mathbf{w}^\top \mathbf{V}_{\bar{T}} \mathbf{w}$ . Lemmas 3.20 and 3.21 are helpful for bounding  $\mathbf{w}^\top \mathbf{V}_{T_\nu - \tilde{T}} \mathbf{w}$  when  $\nu \in \mathcal{I}$  and when  $\nu \notin \mathcal{I}$ , respectively.

### Helper Lemmas for Claim 3.16

In all the statements of the helper lemmas below. Let  $\mathbf{W}$  be as defined in Line 9 in the execution of  $\text{POWERMETHOD}(\epsilon, \mathbf{A})$ . Consider any fixed  $\mathbf{w} \in \mathbf{W}$ . Observe that we can write

$$\mathbf{w} = \sum_{i=1}^n \frac{\lambda_i^K \alpha_i \mathbf{u}_i}{\sqrt{\sum_j \lambda_j^{2K} \alpha_j^2}} \quad (4)$$

where  $K = \frac{4 \log \frac{n}{\epsilon}}{\epsilon}$ ,  $\alpha_i \sim N(0, 1)$  are gaussian random variables, and  $\lambda_i$  and  $\mathbf{u}_i$  are the  $i$ -th eigenvalue and eigenvector of  $\mathbf{A}$ , respectively.

The following lemma shows that the projection of  $\mathbf{w}$  on  $\bar{T}$  is always small. At a high level, since  $\bar{T}$  is spanned by the eigenvectors with the small eigenvalues, the power method guarantees with high probability that the direction of  $\mathbf{w}$  along these eigenvectors will be exponentially small in the number of iterations  $K$ . Recall that  $\Pi_{\bar{T}}$  is the projection matrix to the space  $\bar{T}$ .

**Lemma 3.18.** *If  $\lambda_{\max}(\mathbf{A}) \geq 1 - \epsilon$ , then*

$$\Pr \left[ \|\Pi_{\bar{T}} \mathbf{w}\|^2 \leq \frac{\epsilon^2}{4} \right] \geq 1 - \frac{3}{n^2}.$$

*Proof.* From the definition of  $\bar{T}$  and  $d$  defined in Definition 3.6, we have  $\Pi_{\bar{T}} \mathbf{w} = \frac{\sum_{i>d} \lambda_i^K \alpha_i \mathbf{u}_i}{\sqrt{\sum_{j=1}^n \lambda_j^{2K} \alpha_j^2}}$  by Equation (4). Hence,

$$\|\Pi_{\bar{T}} \mathbf{w}\|^2 = \frac{\sum_{i>d} \lambda_i^{2K} \alpha_i^2}{\sum_{i=1}^n \lambda_i^{2K} \alpha_i^2}.$$

First, we give a crude lower bound for the denominator. We have

$$\sum_{i=1}^n \lambda_i^{2K} \alpha_i^2 \geq \frac{\lambda_1^{2K}}{n^4}$$

with probability at least  $1 - 1/n^2$ . This is because  $\sum_{i=1}^n \lambda_i^{2K} \alpha_i^2 \geq \lambda_1^{2K} \alpha_1^2$  and, since  $\alpha_1^2 \sim \chi_1^2$ , we have  $\alpha_1^2 \geq 1/n^4$  with probability at least  $1 - 1/n^2$  by Lemma 2.5.

Next, we upper bound the numerator as

$$\sum_{i>d} \lambda_i^{2K} \alpha_i^2 \leq \lambda_{d+1}^{2K} \sum_{i=1}^n \alpha_i^2.$$

From Lemma 2.4,  $\sum_{i=1}^n \alpha_i^2 \leq 2n$  with probability at least  $1 - 1/n^2$ . Also note that, since  $\lambda_{\max}(\mathbf{A}) \geq 1 - \epsilon \geq \lambda_0(1 - 2\epsilon)$ . We now have with probability  $1 - 3/n^2$ ,

$$\|\Pi \mathbf{w}\|^2 \leq \left( \frac{\lambda_{d+1}}{\lambda_1} \right)^{2K} \cdot 2n^5 \leq 2n^5 \left( \frac{\lambda_{d+1}}{\lambda_0(1 - 2\epsilon)} \right)^{2K} \leq 2n^5 \left( \frac{1 - 3\epsilon}{1 - 2\epsilon} \right)^{2K} \leq 2n^5 \frac{\epsilon^6}{n^6} \leq \frac{\epsilon^2}{4}. \quad \square$$

The next two helper lemmas are to show that the projection of  $\mathbf{w}$  to  $T_\nu - \tilde{T}$  is small. To do this, we introduce some more notations and one proposition. For any level  $\nu$ , we will use  $q_\nu$  to denote,

$$q_\nu \stackrel{\text{def}}{=} \dim(T_\nu - \tilde{T}). \quad (5)$$

Let

$$\mathbf{z} = \sum_i z_i \mathbf{u}_i \text{ where } z_i = \lambda_i^K \alpha_i. \quad (6)$$

Therefore,  $\mathbf{w} = \mathbf{z} / \|\mathbf{z}\|$ .

We now bound the norm of the projection of  $\mathbf{z}$  to  $T_\nu - \tilde{T}$ . The proof is based on the fact that  $\mathbf{z}$  is a *scaled* gaussian random vector, and the projection of a gaussian on a  $q_\nu$ -dimensional subspace should have norm proportional to  $q_\nu$ .

**Proposition 3.19.** *For  $\mathbf{z}$  as defined in (6), we have*

$$\|\Pi_\nu \mathbf{z}\|^2 \leq \lambda_{a_\nu}^{2K} \cdot \sum_{j=a_\nu}^{b_\nu} \alpha_j^2.$$

Furthermore, if  $q_\nu \geq 10 \log n$ , then with probability at least  $1 - 1/n^2$ ,

$$\|\Pi_\nu \mathbf{z}\|^2 \leq 2q_\nu \cdot \lambda_{a_\nu}^{2K}.$$

*Proof.* Let  $\Pi_\nu^{\text{full}}$  be a projection matrix to the subspace  $T_\nu$ . Recall from Definition 3.9 that  $\mathbf{u}_{a_\nu}, \dots, \mathbf{u}_{b_\nu}$  form an orthonormal basis of  $T_\nu$  and so we have  $\Pi_\nu^{\text{full}} = \sum_{j=a_\nu}^{b_\nu} \mathbf{u}_j \mathbf{u}_j^\top$  and so

$$\Pi_\nu^{\text{full}} \mathbf{z} = \sum_{j=a_\nu}^{b_\nu} (\lambda_j^K \alpha_j) \mathbf{u}_j.$$

Since  $T_\nu - \tilde{T} \subseteq T_\nu$ , we have

$$\|\Pi_\nu \mathbf{z}\|^2 \leq \|\Pi_\nu^{\text{full}} \mathbf{z}\|^2 = \lambda_{a_\nu}^{2K} \sum_{j=a_\nu}^{b_\nu} \alpha_j^2,$$

which proves the first part of the lemma.

Before proving the second part, we consider the vector  $\mathbf{y} = \sum_i \alpha_i \mathbf{u}_i$ . Since  $\alpha_i \sim N(0, 1)$  for all  $i$ , we also have  $\Pi_\nu \mathbf{y} \sim N(0, 1)$  is a gaussian in a  $q_\nu$ -dimensional space. So by Lemma 2.4, we have

$$\|\Pi_\nu \mathbf{y}\|^2 \leq 2q_\nu$$

with probability at least  $1 - e^{-q_\nu/4} \geq 1 - e^{-2 \log n} = 1 - 1/n^2$ .

To prove the second part, observe that  $(\lambda_{a_\nu}^K \cdot \mathbf{y})$  “dominates”  $\Pi_\nu^{\text{full}} \mathbf{z}$  in every coordinate, i.e., the coefficient of each  $\mathbf{u}_i$  in  $(\lambda_{a_\nu}^K \cdot \mathbf{y})$  is at least that of  $\Pi_\nu^{\text{full}} \mathbf{z}$  for every  $i$ . Therefore,  $\|\mathbf{P}(\Pi_\nu^{\text{full}} \mathbf{z})\| \leq \|\mathbf{P}(\lambda_{a_\nu}^K \cdot \mathbf{y})\|$  for any projection matrix  $\mathbf{P}$ . Since  $T_\nu - \tilde{T} \subseteq T_\nu$ , we have  $\Pi_\nu \mathbf{z} = \Pi_\nu \Pi_\nu^{\text{full}} \mathbf{z}$ . Therefore, we can conclude that

$$\|\Pi_\nu \mathbf{z}\|^2 = \|\Pi_\nu \Pi_\nu^{\text{full}} \mathbf{z}\|^2 \leq \|\Pi_\nu (\lambda_{a_\nu}^K \cdot \mathbf{y})\|^2$$

which is at most  $\lambda_{a_\nu}^{2K} \cdot 2q_\nu$  with probability at least  $1 - 1/n^2$ .  $\square$

The following lemma shows that the projection of  $\mathbf{w}$  on  $T_\nu - \tilde{T}$  is small when  $\dim(T_\nu - \tilde{T}) := q_\nu$  is roughly at most an  $\epsilon$ -factor of  $\dim(T_\nu) := d_\nu$ , and  $q_\nu$  is still at least logarithmic. We will use this lemma to characterize the projection of  $\mathbf{w}$  on  $T_\nu$  for  $\nu \in \mathcal{I}$ . Recall that  $\Pi_\nu$  is a projection matrix that projects any vector to the space  $T_\nu - \tilde{T}$ .

**Lemma 3.20.** *If  $10 \log n \leq q_\nu \leq \frac{\epsilon}{300 \log \frac{n}{\epsilon}} d_\nu$ , then*

$$\Pr \left[ \|\Pi_\nu \mathbf{w}\|^2 \leq \frac{\epsilon}{\log \frac{n}{\epsilon}} \right] \geq 1 - \frac{2}{n^2}.$$

*Proof.* Since  $\mathbf{w} = \mathbf{z}/\|\mathbf{z}\|$ , it is equivalent to show that, with probability  $1 - \frac{2}{n^2}$ ,

$$\|\Pi_\nu \mathbf{z}\|^2 \leq \frac{\epsilon}{\log \frac{n}{\epsilon}} \|\mathbf{z}\|^2.$$

We first bound  $\|\Pi_\nu \mathbf{z}\|$  in terms of  $d_\nu$ . As  $q_\nu \geq 10 \log n$ , by Proposition 3.19, we have with probability at least  $1 - 1/n^2$ ,

$$\|\Pi_\nu \mathbf{z}\|^2 \leq \lambda_{a_\nu}^{2K} \cdot 2q_\nu \leq \lambda_{a_\nu}^{2K} \cdot \frac{\epsilon}{150 \log \frac{n}{\epsilon}} d_\nu.$$

where the second inequality follows from the assumption  $q_\nu \leq \frac{\epsilon}{300 \log \frac{n}{\epsilon}} d_\nu$ .

Next, we bound  $d_\nu$  in terms of  $\|\mathbf{z}\|$ . Consider the  $d_\nu$ -dimensional gaussian vector with coordinate  $\alpha_i$  for  $i = a_\nu, \dots, b_\nu$ . Applying Lemma 2.4 to this vector with  $\delta = 1/10$ , we have  $\Pr[\sum_{i=a_\nu}^{b_\nu} \alpha_i^2 \geq (1 - \frac{1}{2})d_\nu] \geq 1 - e^{-d_\nu/100} \geq 1 - \frac{1}{n^2}$  where the last inequality used that  $d_\nu \geq 3000 \log n$ . With probability  $1 - 1/n^2$ , we now have

$$d_\nu \leq 2 \sum_{i=a_\nu}^{b_\nu} \alpha_i^2 = 2 \sum_{i=a_\nu}^{b_\nu} \frac{z_i}{\lambda_i^{2K}} \leq \frac{2}{\lambda_{b_\nu}^{2K}} \|\mathbf{z}\|^2$$

Combining the two inequalities, we can conclude that, with probability at least  $1 - 2/n^2$ ,

$$\|\Pi_\nu \mathbf{z}\|^2 \leq \left( \frac{\lambda_{a_\nu}}{\lambda_{b_\nu}} \right)^{2K} \frac{\epsilon}{75 \log \frac{n}{\epsilon}} \|\mathbf{z}\|^2 \leq \frac{\epsilon}{\log \frac{n}{\epsilon}} \|\mathbf{z}\|^2$$

as desired. To see the last inequality, recall from Definition 3.9 that  $\lambda_{a_\nu} \leq \left(1 - \frac{\nu\epsilon}{5 \log \frac{n}{\epsilon}}\right) \lambda_0$  and  $\lambda_{b_\nu} \geq \left(1 - \frac{(\nu+1)\epsilon}{5 \log \frac{n}{\epsilon}}\right) \lambda_0$ . So  $\frac{\lambda_{a_\nu}}{\lambda_{b_\nu}} \leq 1 + \frac{\epsilon}{2 \log \frac{n}{\epsilon}}$  and, hence,

$$\left( \frac{\lambda_{a_\nu}}{\lambda_{b_\nu}} \right)^{2K} \leq \left( 1 + \frac{\epsilon}{2 \log \frac{n}{\epsilon}} \right)^{2K} \leq e^4 \approx 54.6. \quad \square$$

We next prove that for all  $\nu \notin \mathcal{I}$ , arbitrary projections of  $\mathbf{w}$  on  $T_\nu - \tilde{T}$  are always small. This proof uses a similar idea as that of the previous lemma and the main difference is that we can use the fact that  $d_\nu$  is small for  $\nu \notin \mathcal{I}$  to additionally show that the projection of  $\mathbf{w}$  is small even for small dimensional arbitrary subspaces of  $T_\nu$ . Recall that  $\Pi_\nu$  is a projection matrix to the space  $T_\nu - \tilde{T}$ .

**Lemma 3.21.** *For any non-important level,  $\nu \notin \mathcal{I}$ ,*

$$\Pr \left[ \|\Pi_\nu \mathbf{w}\|^2 \leq \frac{\epsilon}{\log \frac{n}{\epsilon}} \right] \geq 1 - \frac{1}{n^2}.$$

*Proof.* Again, it is sufficient to prove for  $\mathbf{z}$  as defined in Equation (6),

$$\Pr \left[ \|\Pi_\nu \mathbf{z}\|^2 \leq \frac{\epsilon}{\log \frac{n}{\epsilon}} \|\mathbf{z}\|^2 \right] \geq 1 - \frac{1}{n^2}.$$

In this proof, we consider the case of  $q_\nu \geq 20 \log n$  and  $q_\nu < 20 \log n$  separately. Let us first look at  $q_\nu \geq 20 \log n$ .

**Case  $q_\nu \geq 20 \log n$ :** Our strategy will be to first bound  $\|\Pi_\nu \mathbf{z}\|^2$  by  $d_\nu$ , which can be further bounded by  $\sum_{\nu' < \nu} d_{\nu'}$ . From Proposition 3.19, with probability at least  $1 - 1/n^2$ ,

$$\|\Pi_\nu \mathbf{z}\|^2 \leq 2q_\nu \lambda_{a_\nu}^{2K}.$$

Since  $T_\nu - \tilde{T} \subseteq T_\nu$ ,  $q_\nu \leq d_\nu$ . Furthermore, since  $\nu \notin \mathcal{I}$ ,  $d_\nu \leq \frac{\epsilon}{600 \log^3 \frac{n}{\epsilon}} \sum_{\nu' < \nu} d_{\nu'}$ . Using these bounds, we then have with probability  $1 - 1/n^2$ ,

$$\|\Pi_\nu \mathbf{z}\|^2 \leq \frac{\epsilon}{300 \log^3 \frac{n}{\epsilon}} \lambda_{a_\nu}^{2K} \sum_{\nu' < \nu} d_{\nu'}.$$

We next bound  $\sum_{\nu' < \nu} d_{\nu'}$  in terms of  $\|\mathbf{z}\|$ . Consider the  $\sum_{\nu' < \nu} d_{\nu'}$  dimensional gaussian vector with coordinates  $\alpha_i$ , for  $i = 1, \dots, b_{\nu-1}$ . Applying Lemma 2.4 to this vector with  $\delta = 1/3$  gives,

$$\Pr \left[ \sum_{i=1}^{b_{\nu-1}} \alpha_i^2 \geq \left(1 - \frac{8}{9}\right) \sum_{\nu' < \nu} d_{\nu'} \right] \geq 1 - e^{-\frac{\sum_{\nu' < \nu} d_{\nu'}}{9}} \geq 1 - \frac{1}{n^2}.$$

In the last inequality we used that  $\sum_{\nu' < \nu} d_{\nu'} \geq d_\nu$  and  $d_\nu \geq q_\nu \geq 20 \log n$ . We now know that with probability at least  $1 - 1/n^2$ ,

$$\sum_{\nu' < \nu} d_{\nu'} \leq 9 \sum_{i=1}^{b_{\nu-1}} \alpha_i^2 = 9 \sum_{i=1}^{b_{\nu-1}} \frac{z_i^2}{\lambda_i^{2K}} \leq 9 \frac{\|\mathbf{z}\|^2}{\lambda_{b_{\nu-1}}^{2K}}.$$

Therefore, with a probability of at least  $1 - 2/n^2$ ,

$$\|\Pi_\nu \mathbf{z}\|^2 \leq \frac{\epsilon}{15 \log^3 \frac{n}{\epsilon}} \left( \frac{\lambda_{a_\nu}}{\lambda_{b_{\nu-1}}} \right)^{2K} \|\mathbf{z}\|^2 \leq \frac{\epsilon}{15 \log^3 \frac{n}{\epsilon}} \|\mathbf{z}\|^2.$$

The last inequality follows from the fact  $\lambda_{a_\nu} \leq \lambda_{b_{\nu-1}}$ .

**Case  $q_\nu < 20 \log n$ :** In this case, since  $q_\nu < 20 \log n$ , we apply the first part of Proposition 3.19 to get

$$\|\Pi_\nu \mathbf{z}\|^2 \leq \lambda_{a_\nu}^{2K} \sum_{j=a_\nu}^{b_\nu} \alpha_j^2.$$

Observe that in this case  $d_\nu$  can be less than  $20 \log n$ . We also know that since  $\nu \notin \mathcal{I}$ , we can bound  $d_\nu$  by  $\sum_{\nu' < \nu} d_{\nu'}$ . In our analysis, we consider the value of  $\sum_{\nu' < \nu} d_{\nu'}$  and further split it into two parts based on whether  $\sum_{\nu' < \nu} d_{\nu'}$  is large or small.

- $\sum_{\nu' < \nu} d_{\nu'} \geq 20 \log n$ : Our strategy would be to first bound  $\|\Pi_\nu \mathbf{z}\|^2$  by  $\sum_{\nu' < \nu} d_{\nu'}$ , and then bound  $\sum_{\nu' < \nu} d_{\nu'}$  by  $\|\mathbf{z}\|^2$ . Note that since  $\alpha_i$ 's are gaussian random variables,  $\sum_{j=a_\nu}^{b_\nu} \alpha_j^2$  follows a  $\chi_k^2$  distribution with  $k = d_\nu$ . Therefore,  $\sum_{j=a_\nu}^{b_\nu} \alpha_j^2 \leq d_\nu \log n$  with probability at least  $1 - 1/n^2$ . Further since  $d_\nu \leq \frac{\epsilon}{600 \log^3 \frac{n}{\epsilon}} \sum_{\nu' < \nu} d_{\nu'}$ , we get with probability at least  $1 - 2/n^2$ ,

$$\|\Pi_\nu \mathbf{z}\|^2 \leq \lambda_{a_\nu}^{2K} \cdot \log n \cdot d_\nu \leq \lambda_{a_\nu}^{2K} \frac{\epsilon}{600 \log \frac{n}{\epsilon}} \sum_{\nu' < \nu} d_{\nu'}.$$

Now, since  $\sum_{\nu' < \nu} d_{\nu'} \geq 20 \log n$  we use Lemma 2.4 again with  $\delta = 1/3$ , on a vector with coordinates  $\alpha_i$ 's for  $i = 1, \dots, b_{\nu-1}$  to get with probability at least  $1 - 1/n^2$ ,

$$\sum_{\nu' < \nu} d_{\nu'} \leq 9 \sum_{i=1}^{b_{\nu-1}} \alpha_i^2 = 9 \sum_{i=1}^{b_{\nu-1}} \frac{z_i^2}{\lambda_i^{2K}} \leq 9 \frac{\|\mathbf{z}\|^2}{\lambda_{b_{\nu-1}}^{2K}}.$$

Plugging this back, we get with probability at least  $1 - 3/n^2$ ,

$$\|\Pi_\nu \mathbf{z}\|^2 \leq \frac{\epsilon}{60 \log \frac{n}{\epsilon}} \left( \frac{\lambda_{a_\nu}}{\lambda_{b_{\nu-1}}} \right)^{2K} \|\mathbf{z}\|^2 \leq \frac{\epsilon}{60 \log \frac{n}{\epsilon}} \|\mathbf{z}\|^2.$$

Last inequality follows from  $\lambda_{a_\nu} \leq \lambda_{b_{\nu-1}}$ .

- $\sum_{\nu' < \nu} d_{\nu'} < 20 \log n$ : Since  $\nu \notin \mathcal{I}$ , we know that

$$d_\nu \leq \frac{\epsilon}{600 \log^3 \frac{n}{\epsilon}} \sum_{\nu' < \nu} d_{\nu'}.$$

Since  $\sum_{\nu' < \nu} d_{\nu'} < 20 \log n$ , we must then have that,

$$d_\nu \leq \frac{\epsilon}{30 \log^2 \frac{n}{\epsilon}} < 1.$$

Since the dimension  $d_\nu$  must be an integer, it must be the case that  $d_\nu = 0$ , and therefore,  $\|\Pi_\nu \mathbf{z}\| = 0$ .

□

**Proof of Claim 3.16.** We are now ready to finally prove Claim 3.16.

*Proof.* We want to show that if  $\lambda_{\max}(\mathbf{A}) \geq 1 - \epsilon$  and  $\dim(T_\nu - \tilde{T})$  is small for all  $\nu \in \mathcal{I}$  as stated in Claim 3.16, then  $\mathbf{w}^\top \mathbf{V} \mathbf{w} \leq 35\epsilon$  for all  $\mathbf{w} \in \mathbf{W}$  with high probability. Recall that

$$\mathbf{w}^\top \mathbf{V} \mathbf{w} = \mathbf{w}^\top \mathbf{V}_{\tilde{T}} \mathbf{w} + \sum_{\nu=0}^{15 \log \frac{n}{\epsilon} - 1} \mathbf{w}^\top \mathbf{V}_{T_\nu - \tilde{T}} \mathbf{w} + \mathbf{w}^\top \mathbf{V}_{\bar{T}} \mathbf{w}.$$

Let us upper bound each term in the sum below.

1.  $\mathbf{w}^\top \mathbf{V}_{\tilde{T}} \mathbf{w}$ : We have

$$\mathbf{w}^\top \mathbf{V}_{\tilde{T}} \mathbf{w} = (\Pi_{\tilde{T}} \mathbf{w})^\top \mathbf{V} \Pi_{\tilde{T}} \mathbf{w} = (\Pi_{\tilde{T}} \mathbf{w})^\top \mathbf{A} \Pi_{\tilde{T}} \mathbf{w} - (\Pi_{\tilde{T}} \mathbf{w})^\top \tilde{\mathbf{A}} \Pi_{\tilde{T}} \mathbf{w}.$$

From the definition of  $\tilde{T}$  (see Definition 3.6), we know that  $(\Pi_{\tilde{T}} \mathbf{w})^\top \tilde{\mathbf{A}} \Pi_{\tilde{T}} \mathbf{w} \geq (1 - 10\epsilon) \lambda_0 \|\Pi_{\tilde{T}} \mathbf{w}\|^2$  because  $\Pi_{\tilde{T}} \mathbf{w} \in \tilde{T} = \text{span}(3\epsilon, \tilde{\mathbf{A}})$ . We also know that  $(\Pi_{\tilde{T}} \mathbf{w})^\top \mathbf{A} \Pi_{\tilde{T}} \mathbf{w} \leq \lambda_0 \|\Pi_{\tilde{T}} \mathbf{w}\|^2$ . So

$$\mathbf{w}^\top \mathbf{V}_{\tilde{T}} \mathbf{w} \leq 10\epsilon \lambda_0 \|\Pi_{\tilde{T}} \mathbf{w}\|^2 \leq 10\epsilon(1 + \epsilon) \|\mathbf{w}\|^2 = 10\epsilon(1 + \epsilon). \quad (7)$$

2.  $\mathbf{w}^\top \mathbf{V}_{\bar{T}} \mathbf{w}$ : Since  $\mathbf{w} \in \mathbf{W}$ , from Lemma 3.18, with probability at least  $1 - \frac{3}{n^2}$ ,  $\|\Pi_{\bar{T}} \mathbf{w}\| \leq \epsilon/2$ . Now,

$$\mathbf{w}^\top \mathbf{V}_{\bar{T}} \mathbf{w} = (\Pi_{\bar{T}} \mathbf{w})^\top \mathbf{V} (\Pi_{\bar{T}} \mathbf{w}) \leq \|\Pi_{\bar{T}} \mathbf{w}\|^2 \|\mathbf{V}\| \leq \frac{\epsilon^2}{4} \lambda_0 \leq \frac{\epsilon^2}{2}, \quad (8)$$

where we used that  $\|\mathbf{V}\| \leq \lambda_0$  since  $\tilde{\mathbf{A}} = \mathbf{A} - \mathbf{V} \succeq 0$  and  $\lambda_0 \leq 1 + \epsilon/\log n$ .

3.  $\mathbf{w}^\top \mathbf{V}_{T_\nu - \tilde{T}} \mathbf{w}$  when  $\nu \in \mathcal{I}$ : Note that the dimension of the space  $T_\nu - \tilde{T}$  is small. We now have,

$$\mathbf{w}^\top \mathbf{V}_{T_\nu - \tilde{T}} \mathbf{w} = \mathbf{w}^\top \Pi_\nu \mathbf{V} \Pi_\nu \mathbf{w}.$$

We will now consider the large  $d_\nu$  and small  $d_\nu$  cases separately.

**Large dimension:**  $d_\nu \geq \frac{3000 \log n \log \frac{n}{\epsilon}}{\epsilon}$ . In this case,  $\dim(T_\nu - \tilde{T}) \leq \frac{\epsilon}{300 \log \frac{n}{\epsilon}} d_\nu$ . We can now apply Lemma 3.20, which gives with probability at least  $1 - \frac{2}{n^2}$ ,

$$\|\Pi_\nu \mathbf{w}\|^2 \leq \frac{\epsilon}{\log \frac{n}{\epsilon}}.$$

Now, using this value,

$$\mathbf{w}^\top \mathbf{V}_{T_\nu - \tilde{T}} \mathbf{w} \leq \|\Pi_\nu \mathbf{w}\|^2 \|\mathbf{V}\| \leq \frac{\epsilon}{\log \frac{n}{\epsilon}} \|\mathbf{V}\| \leq \frac{\epsilon}{\log \frac{n}{\epsilon}} \lambda_0 \leq \frac{\epsilon(1 + \epsilon)}{\log \frac{n}{\epsilon}}. \quad (9)$$

As in case 2, we again used the fact that  $\|\mathbf{V}\| \leq \lambda_0 \leq (1 + \epsilon)$ .

**Small dimension:**  $d_\nu < \frac{3000 \log n \log \frac{n}{\epsilon}}{\epsilon}$ . In this case,  $\dim(T_\nu - \tilde{T}) < 1$ . Therefore, the space  $T_\nu - \tilde{T}$  is empty and as a result,

$$\mathbf{w}^\top \mathbf{V}_{T_\nu - \tilde{T}} \mathbf{w} = 0. \quad (10)$$

4.  $\mathbf{w}^\top \mathbf{V}_{T_\nu - \tilde{T}} \mathbf{w}$  when  $\nu \notin \mathcal{I}$ : From Lemma 3.21,  $\|\Pi_\nu \mathbf{w}\|^2 \leq \frac{\epsilon}{\log \frac{n}{\epsilon}}$  with probability at least  $1 - 1/n^2$ . Since  $\|\mathbf{V}\| \leq \lambda_0 \leq 1 + \epsilon$ , we get,

$$\mathbf{w}^\top \mathbf{V}_{T_\nu - \tilde{T}} \mathbf{w} \leq \|\Pi_\nu \mathbf{w}\|^2 \|\mathbf{V}\| \leq \frac{\epsilon(1 + \epsilon)}{\log \frac{n}{\epsilon}}. \quad (11)$$

We now combine all the cases. We have for both large and small  $d_\nu$  from Equations (7),(8),(9),(10) and (11), with probability at least  $1 - \frac{40 \log \frac{n}{\epsilon}}{n^2}$  for any  $\mathbf{w} \in \mathbf{W}$ ,

$$\mathbf{w}^\top \mathbf{V} \mathbf{w} \leq \mathbf{w}^\top \mathbf{V}_{\bar{T}} \mathbf{w} + \mathbf{w}^\top \mathbf{V}_{\tilde{T}} \mathbf{w} + \sum_{\nu=0}^{15 \log \frac{n}{\epsilon} - 1} \mathbf{w}^\top \mathbf{V}_{T_\nu - \tilde{T}} \mathbf{w} \leq \frac{\epsilon^2}{2} + 10\epsilon(1 + \epsilon) + 10\epsilon(1 + \epsilon) \leq 35\epsilon.$$

□



## 4 Conditional Lower Bounds for an Adaptive Adversary

In this section, we will prove a conditional hardness result for algorithms against adaptive adversaries. In particular, we will prove Theorem 1.6. Consider Algorithm 4 for solving Problem 1.3. The only step in Algorithm 4 whose implementation is not specified is Line 8. We will implement this step using an algorithm for Problem 1.1.

---

### Algorithm 4 Algorithm for Checking PSDness

---

```

1: procedure CHECKPSD( $\delta, \kappa, \mathbf{A}$ )
2:    $\epsilon \leftarrow \min\{1 - n^{-o(1)}, (1 - \delta)/(1 + \delta)\}$ 
3:    $T \leftarrow \frac{2n}{\epsilon(1-\epsilon)^2} \log \frac{\kappa}{\delta}$ 
4:    $\mathbf{A}_0 \leftarrow \mathbf{A}$ 
5:    $\mu_0 = 0, \mathbf{w}_0 = 0$ 
6:   for  $t = 1, 2, \dots, T$  do
7:      $\mathbf{A}_t = \mathbf{A}_{t-1} - \frac{\mu_{t-1}}{10} \mathbf{w}_{t-1} \mathbf{w}_{t-1}^\top$ 
8:      $(\mu_t, \mathbf{w}_t) \leftarrow \epsilon$ -approximate maximum eigenvalue and eigenvector of  $\mathbf{A}_t$  (Equations (1),(2))
9:     if  $\mu_t < 0$  then
10:      return FALSE:  $\mathbf{A}$  is not PSD
11:    $\sigma^2 \leftarrow \text{PowerMethod}(\epsilon, \mathbf{A}_T^\top \mathbf{A}_T)$ 
12:   if  $0 \leq \sigma \leq \frac{(1+\epsilon)\mu_1\delta}{\kappa}$  then
13:     return  $\mathbf{X} = \frac{1}{\sqrt{10}} \begin{bmatrix} \sqrt{\mu_1} \mathbf{w}_1 & \sqrt{\mu_2} \mathbf{w}_2 & \cdots & \sqrt{\mu_T} \mathbf{w}_T \end{bmatrix}$ 
14:   else
15:     return FALSE:  $\mathbf{A}$  is not PSD

```

---

**High-level idea.** Overall for our hardness result, we use the idea that an adaptive adversary can use the maximum eigenvectors returned to perform an update. This can happen  $n$  times and in the process, we would recover the entire eigen-decomposition of the matrix, which is hard. Now consider Algorithm 4. We claim that Algorithm 4 solves Problem 1.3. At the first glance, this claim looks suspicious because the input matrix for Problem 1.3 might not be PSD, but the dynamic algorithm for Problem 1.1 at Line 8 has any guarantees only when the matrices remain PSD. However, the reduction does work by crucially exploiting Property 1.5. The high-level idea is as follows.

- If the input matrix  $\mathbf{A}$  is initially PSD, then we can show that  $\mathbf{A}_t$  remains PSD for all  $t$  by exploiting Property 1.5, (see Lemma 4.1). So, the approximation guarantee of the algorithm at Line 8 is valid at all steps. From this guarantee,  $\|\mathbf{A}_T\|$  must be tiny since we keep decreasing the approximately maximum eigenvalues (see Lemma 4.2). At the end, the reduction will return  $\mathbf{X}$ .
- If the input matrix  $\mathbf{A}$  is initially *not* PSD, there must exist a direction  $\mathbf{v}$  such that  $\mathbf{v}^\top \mathbf{A} \mathbf{v} < 0$ . Since in the reduction, we update  $\mathbf{A}_T = \mathbf{A} - \mathbf{W}$  for some  $\mathbf{W} \succeq 0$ , we must have that  $\mathbf{v}^\top \mathbf{A}_T \mathbf{v} < \mathbf{v}^\top \mathbf{A} \mathbf{v}$ . That is, this negative direction remains negative or gets even more negative. It does not matter at all what guarantees the algorithm at Line 8 has. We still have that  $\|\mathbf{A}_T\|$  cannot be tiny. We can distinguish whether  $\|\mathbf{A}_T\|$  is tiny or not using the static power method at Line 11, and, hence, we will return FALSE in this case (see Lemma 4.2).

**Analysis.** We prove the guarantees of the output of Algorithm 4 when  $\mathbf{w}_t$ 's satisfy Property 1.5 for all  $t$ .

**Lemma 4.1.** *In Algorithm 4, let  $\mathbf{w}_t$ 's,  $t = 1, \dots, T$  be generated such that they additionally satisfy Property 1.5. If  $\mathbf{A}_0 \succeq 0$ , then  $\mathbf{A}_t \succeq 0$  for all  $t$ .*

We would like to point out that our parameter  $\epsilon$  is quite large. This just implies that our reduction can work even if we find crude approximations to the maximum eigenvector as long as this is along the directions with large eigenvalue, since  $\mathbf{w}$  also has to satisfy Property 1.5.

*Proof.* We will prove this by induction. This is true in the beginning because  $\mathbf{A}_0 \succeq 0$  by assumption. Let us assume  $\mathbf{A}_t \succeq 0$ . We now look at  $\mathbf{A}_{t+1} = \mathbf{A}_t - \frac{\mu}{10} \mathbf{w}_t \mathbf{w}_t^\top$ . For simplicity, we use  $\lambda_i, \mathbf{u}_i$  to denote the  $i^{\text{th}}$  eigenvalue and eigenvector of  $\mathbf{A}_t$ , and  $\mu = \mu_t, \mathbf{w} = \mathbf{w}_t$ . By Equation (1),  $\lambda_1 \geq \mu \geq (1 - \epsilon)\lambda_1$ . Now, for any  $\mathbf{y}$

$$\mathbf{y}^\top \mathbf{A}_{t+1} \mathbf{y} = \mathbf{y}^\top \mathbf{A}_t \mathbf{y} - \frac{\mu}{10} (\mathbf{w}^\top \mathbf{y})^2 = \sum_{i=1}^n \lambda_i (\mathbf{u}_i^\top \mathbf{y})^2 - \frac{\mu}{10} \left( \sum_{i=1}^n (\mathbf{w}^\top \mathbf{u}_i) (\mathbf{y}^\top \mathbf{u}_i) \right)^2. \quad (12)$$

We will upper bound the second term. We know that  $\mathbf{w}$  satisfies Property 1.5. Let  $d$  be such that  $\lambda_i \leq \lambda_1/2$  for all  $i \geq d$ .

$$\begin{aligned} \left( \sum_{i=1}^n (\mathbf{w}^\top \mathbf{u}_i) (\mathbf{y}^\top \mathbf{u}_i) \right)^2 &\leq 2 \left( \sum_{i=1}^d (\mathbf{w}^\top \mathbf{u}_i) (\mathbf{y}^\top \mathbf{u}_i) \right)^2 + 2 \left( \sum_{i=d}^n (\mathbf{w}^\top \mathbf{u}_i) (\mathbf{y}^\top \mathbf{u}_i) \right)^2 \\ &\stackrel{(a)}{\leq} 2 \sum_{i=1}^d (\mathbf{w}^\top \mathbf{u}_i)^2 \sum_{i=1}^d (\mathbf{y}^\top \mathbf{u}_i)^2 + 2(n-d) \sum_{i=d}^n (\mathbf{w}^\top \mathbf{u}_i)^2 (\mathbf{y}^\top \mathbf{u}_i)^2 \\ &\stackrel{(b)}{\leq} 2 \sum_{i=1}^d (\mathbf{y}^\top \mathbf{u}_i)^2 + 2(n-d) \sum_{i=d}^n \frac{\lambda_i (\mathbf{y}^\top \mathbf{u}_i)^2}{\lambda_1 n^2} \\ &\stackrel{(c)}{\leq} \frac{4}{\lambda_1} \sum_{i=1}^d \lambda_i (\mathbf{y}^\top \mathbf{u}_i)^2 + \sum_{i=d}^n \frac{\lambda_i (\mathbf{y}^\top \mathbf{u}_i)^2}{\lambda_1 n}. \end{aligned}$$

In the above, (a) follows by applying Cauchy Schwarz to both terms, (b) follows by using  $\|\mathbf{w}\| = 1$  in the first term and applying Property 1.5 on the second term, and (c) follows by using the fact that for all  $i \leq d$ ,  $\lambda_i \geq \lambda_1/2$  in the first term.

Using this in (12),

$$\begin{aligned} \mathbf{y}^\top \mathbf{A}_{t+1} \mathbf{y} &\geq \sum_{i=1}^n \lambda_i (\mathbf{u}_i^\top \mathbf{y})^2 - \left( \frac{4\mu}{10\lambda_1} \sum_{i=1}^d \lambda_i (\mathbf{u}_i^\top \mathbf{y})^2 + \frac{\mu}{10} \sum_{i=d}^n \frac{\lambda_i (\mathbf{y}^\top \mathbf{u}_i)^2}{\lambda_1 n} \right) \\ &= \sum_{i=1}^d \left( 1 - \frac{2\mu}{5\lambda_1} \right) \lambda_i (\mathbf{u}_i^\top \mathbf{y})^2 + \sum_{i=d}^n \left( 1 - \frac{\mu}{10n\lambda_1} \right) \lambda_i (\mathbf{u}_i^\top \mathbf{y})^2 \\ &\geq 0. \end{aligned}$$

We have thus shown that  $\mathbf{A}_{t+1}$  is psd, as required.  $\square$

**Lemma 4.2.** *In Algorithm 4, let  $\mathbf{w}_t$ 's,  $t = 1, \dots, T$  be generated such that they additionally satisfy Property 1.5.*

- If  $\mathbf{A} \succeq 0$ , then Algorithm 4 returns  $\mathbf{X}$  such that  $\|\mathbf{A} - \mathbf{X}\mathbf{X}^\top\| \leq \delta \min_{\|x\|=1} \|\mathbf{A}x\|$ .
- If  $\mathbf{A}$  is not psd, then Algorithm 4 returns FALSE.

*Proof.* We prove the first part first. Let  $\mathbf{A}_0 \succeq 0$  with maximum eigenvalue  $\lambda_1$ , and  $t$  be the iterate such that the maximum eigenvalue of  $\mathbf{A}_{t-1}$  is at least  $(1-\epsilon)\lambda_1$  and the maximum eigenvalue of  $\mathbf{A}_t$  is at most  $(1-\epsilon)\lambda_1$ . We also know that  $\mathbf{A}_t \succeq 0$  for all  $t$  from Lemma 4.1.

In this case,  $\mu_i \geq (1-\epsilon)^2\lambda_1$  for all  $i \leq t-1$  (since  $\mu_i$  is an  $\epsilon$ -max eigenvalue and the max eigenvalue is at least  $(1-\epsilon)\lambda_1$ ) and we must have that  $\text{Tr}[\mathbf{A}_t] \leq \text{Tr}[\mathbf{A}] - (1-\epsilon)^2\lambda_1(t-1)$ . Also,

$$\lambda_{\max}(\mathbf{A}_t) \leq \text{Tr}[\mathbf{A}_t] \leq \text{Tr}[\mathbf{A}] - (1-\epsilon)^2\lambda_1(t-1) \leq n\lambda_1 - (1-\epsilon)^2\lambda_1(t-1).$$

Thus we require at most  $t = 2n/(1-\epsilon)^2$  iterates after which  $\mathbf{A}_t$  will have a maximum eigenvalue at most  $(1-\epsilon)\lambda_1$ .

Our algorithm runs for  $T = \frac{2n}{\epsilon(1-\epsilon)^2} \log \frac{\kappa}{\delta}$  and every  $2n/(1-\epsilon)^2$  iterations we decrease the maximum eigenvalue by a factor of at least  $1-\epsilon$ . Finally, the maximum eigenvalue of  $\mathbf{A}_T$  is therefore at most,

$$\lambda_1 \cdot (1-\epsilon)^{\frac{T(1-\epsilon)^2}{2n}} = \lambda_1 \cdot (1-\epsilon)^{\frac{1}{\epsilon} \log \frac{\kappa}{\delta}} \leq \lambda_1 e^{-\log \frac{\kappa}{\delta}} \leq \lambda_1 \frac{\delta}{\kappa} = \delta \cdot \min_{\|x\|=1} \|\mathbf{A}x\|,$$

as required. We have proved that, if  $\mathbf{A} \succeq 0$ , then the maximum eigenvalue of  $\mathbf{A} - \mathbf{X}\mathbf{X}^\top$  is at most  $\lambda_1\delta/\kappa \leq \delta \min_{\|x\|=1} \|\mathbf{A}x\|$ .

Next, if  $\mathbf{A}$  is not psd, then we will prove that  $\sigma \geq \frac{(1-\epsilon)\mu_1}{\kappa}$ . This implies that  $\sigma > \frac{(1+\epsilon)\mu_1\delta}{\kappa}$  from the value of  $\epsilon$ . and so the algorithm will return FALSE by the condition at Line 12. Since  $\mathbf{A}$  is not psd, there exists a unit vector  $\mathbf{v}$  such that  $\mathbf{v}^\top \mathbf{A} \mathbf{v} < 0$ . We can lower bound  $\sigma$  as follows.

$$\sigma \geq (1-\epsilon)\|\mathbf{A}_T\| \geq (1-\epsilon)|\mathbf{v}^\top \mathbf{A}_T \mathbf{v}|.$$

On the other hand, we have

$$|\mathbf{v}^\top \mathbf{A}_T \mathbf{v}| \geq |\mathbf{v}^\top \mathbf{A} \mathbf{v}| \geq \min_{\|x\|=1} x^\top \mathbf{A} x \geq \frac{\mu_1}{\kappa}$$

where the first inequality is because  $\mathbf{v}^\top \mathbf{A}_T \mathbf{v} = \mathbf{v}^\top \mathbf{A} \mathbf{v} - \|\mathbf{X}^\top \mathbf{v}\|^2 < \mathbf{v}^\top \mathbf{A} \mathbf{v} < 0$ . Combining the two inequalities, we get  $\sigma \geq \frac{(1-\epsilon)\mu_1}{\kappa}$ , which concludes the proof.  $\square$

**Proof of Theorem 1.6.** We are now ready to prove our conditional lower bound.

*Proof.* Let  $\mathcal{M}(\epsilon, \mathbf{A}_0, \mathbf{v}_1, \dots, \mathbf{v}_T)$  denote an algorithm for Problem 1.1 that maintains an  $\epsilon$ -max eigenvalue (1),  $\mu_t$ , and eigenvector (2),  $\mathbf{w}_t$ , for matrices  $\mathbf{A}_t = \mathbf{A}_{t-1} - \mathbf{v}_t \mathbf{v}_t^\top$  such that  $\mathbf{w}_t$ 's satisfy Property 1.5. We will show that if the total update time of  $\mathcal{M}$  is  $n^{o(1)} \cdot \left( nnz(\mathbf{A}_0) + \sum_{t=1}^T nnz(\mathbf{v}_t) \right)$ , then there is an  $n^{2+o(1)}$ -time algorithm for Problem 1.3 which contradicts Hypothesis 1.4.

Given an instance  $(\delta, \kappa, \mathbf{A})$  of Problem 1.3, we will run Algorithm 4 where Line 8 is implemented using  $\mathcal{M}$ . We will generate the input and the update sequence for  $\mathcal{M}$  as follows. Set  $\mathbf{A}_0 \leftarrow \mathbf{A}$ .

Set  $\epsilon$  and  $T$  according to Algorithm 4. For  $1 \leq t \leq T$ , we set  $\mathbf{v}_t = \frac{1}{\sqrt{10}} \cdot \sqrt{\mu_{t-1}} \mathbf{w}_{t-1}$  according to Line 7 of Algorithm 4. We note that this is a valid update sequence for Problem 1.1 when  $\mathbf{A} \succeq 0$  since from Lemma 4.1, if  $\mathbf{A} \succeq 0$  then,  $\mathbf{A}_t = \mathbf{A} - \sum_{i=0}^{t-1} \frac{\mu_i}{10} \mathbf{w}_i \mathbf{w}_i^\top \succeq 0$ .

Now, we describe what we return as an answer for Problem 1.3. From Lemma 4.2 if  $\mathbf{A}$  is not PSD, then Algorithm 4 returns FALSE and reports the matrix is not PSD. Additionally if  $\mathbf{A} \succeq 0$ , the algorithm returns  $\mathbf{X} = \frac{1}{\sqrt{10}} \begin{bmatrix} \sqrt{\mu_1} \mathbf{w}_1 & \sqrt{\mu_2} \mathbf{w}_2 & \cdots & \sqrt{\mu_T} \mathbf{w}_T \end{bmatrix}$  as a certificate that  $\mathbf{A}$  is PSD. This completes the reduction from Problem 1.3 to Problem 1.1.

The total time required by the reduction is

$$O\left(n^{o(1)} \cdot (nnz(\mathbf{A}) + \sum_{t=1}^T nnz(\mathbf{w}_t))\right) \leq O\left(n^{o(1)} \cdot (n^2 + T \cdot n)\right) \leq O(n^{2+o(1)} \log \frac{\kappa}{\delta}),$$

which is at most  $n^{2+o(1)}$  when  $\frac{\kappa}{\delta} \leq \text{poly}(n)$ .

To conclude, we indeed obtain an  $n^{2+o(1)}$ -time algorithm for Problem 1.3. Assuming Hypothesis 1.4, the algorithm  $\mathcal{M}$  cannot have  $n^{o(1)} \cdot (nnz(\mathbf{A}_0) + \sum_{t=1}^T nnz(\mathbf{v}_t))$  total update time.  $\square$

## 5 Conclusion and Open Problems

**Upper Bounds.** We have presented a novel extension of the power method to the dynamic setting. Our algorithm from Theorem 1.2 maintains a multiplicative  $(1 + \epsilon)$ -approximate maximum eigenvalue and eigenvector of a positive semi-definite matrix that undergoes decremental updates from an oblivious adversary. The algorithm has polylogarithmic amortized update time per non-zeros in the updates.

Our algorithm is simple, but our analysis is quite involved. While we believe a tighter analysis that improves our logarithmic factors is possible, it is an interesting open problem to give a simpler analysis for our algorithm. Other natural questions are whether we can get similar algorithms in incremental or fully dynamic settings and whether one can get a worst-case update time.

**Lower Bounds.** We have shown a conditional lower bound for a class of algorithms against an adaptive adversary in Theorem 1.6. It would also be very exciting to generalize our lower bound to hold for any algorithm against an adaptive adversary, as that would imply an oblivious-vs-adaptive separation for a natural dynamic problem.

**Incremental Updates.** We believe that the corresponding incremental updates problem, i.e., we update the matrix as  $\mathbf{A}_t \leftarrow \mathbf{A}_{t-1} + \mathbf{v}_t \mathbf{v}_t^\top$  cannot be solved in polylogarithmic amortized update time, even when the update sequence  $\mathbf{v}_t$ 's are from an oblivious adversary. At a high level, the incremental version of our problem seems significantly harder for the following reasons. When we perform decremental updates to a matrix, the new maximum eigenvector must be a part of the eigenspace spanned by the original maximum eigenvectors. Furthermore, it is easy to detect whether the maximum eigenvalue has gone down as we have shown in our paper. For the incremental setting, it is possible that after an update the maximum eigenvalue has gone up and the new maximum eigenvector is a direction that was not the previous one or the update direction and in such cases we cannot really detect this quickly with known information on previous eigenvectors and update directions. This can also happen  $n$  times and in every such case, we have to compute the eigenvalue

and eigenvector from scratch. Therefore, we leave lower bounds and algorithms for incremental setting as an open problem.

**Dynamic SDPs.** As discussed in Appendix A, Theorem 1.2 can be viewed as a starting point towards a dynamic algorithm for general positive semi-definite programs. Can we make further progress? The dynamic semi-definite program problem, even with just two matrix constraints, already seems to be challenging.

One promising approach to attack this problem is to dynamize the matrix multiplicative weight update (MMWU) method for solving a packing/covering SDP [PTZ12] since the corresponding approach was successful for linear programs – the near-optimal algorithms of [BKS23] are essentially dynamized multiplicative weight update methods for positive linear programs. However, in our preliminary study exploring this approach, we could only obtain an algorithm that solves Problem A.2, which has a single matrix constraint, and solves Problem 1.1 partially, i.e., maintains an approximate eigenvalue only. The main barrier in this approach is that the algorithm requires maintaining the exponential of the sum of the constraint matrices, and to do this fast, we require that for any two constraint matrices  $\mathbf{A}$  and  $\mathbf{B}$ ,  $e^{\mathbf{A}+\mathbf{B}} = e^{\mathbf{A}}e^{\mathbf{B}}$  which only holds when  $\mathbf{A}$  and  $\mathbf{B}$  commute i.e.,  $\mathbf{AB} = \mathbf{BA}$ . Note that when  $\mathbf{A}$  and  $\mathbf{B}$  are diagonal, this is true; therefore, we can obtain the required algorithms for positive LPs. Even when we have just two constraint matrices where one of them is a diagonal matrix, this remains an issue as the matrices still do not commute.

## References

- [AGG88] P. Arbenz, W. Gander, and G. H. Golub. “Restricted rank modification of the symmetric eigenvalue problem: Theoretical considerations”. In: *Linear Algebra and its Applications* 104 (1988), pp. 75–95 (cit. on p. 1).
- [AK07] S. Arora and S. Kale. “A combinatorial, primal-dual approach to semidefinite programs”. In: *Proceedings of the thirty-ninth annual ACM symposium on Theory of computing*. 2007, pp. 227–236 (cit. on p. 32).
- [AL17] Z. Allen-Zhu and Y. Li. “First efficient convergence for streaming  $k$ -pca: a global, gap-free, and near-optimal rate”. In: *2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE. 2017, pp. 487–492 (cit. on p. 1).
- [ALO16] Z. Allen-Zhu, Y. T. Lee, and L. Orecchia. “Using optimization to obtain a width-independent, parallel, simpler, and faster positive SDP solver”. In: *Proceedings of the twenty-seventh annual ACM-SIAM symposium on Discrete algorithms*. SIAM. 2016, pp. 1824–1831 (cit. on p. 31).
- [AO15] Z. Allen-Zhu and L. Orecchia. “Nearly-linear time positive LP solver with faster convergence rate”. In: *Proceedings of the forty-seventh annual ACM symposium on Theory of Computing*. 2015, pp. 229–236 (cit. on p. 31).
- [AO19] Z. Allen-Zhu and L. Orecchia. “Nearly linear-time packing and covering lp solvers: Achieving width-independence and-convergence”. In: *Mathematical Programming* 175 (2019), pp. 307–353 (cit. on p. 31).
- [Ban+22] J. Banks, J. Garza-Vargas, A. Kulkarni, and N. Srivastava. “Pseudospectral shattering, the sign function, and diagonalization in nearly matrix multiplication time”. In: *Foundations of Computational Mathematics* (2022), pp. 1–89 (cit. on p. 3).
- [Bat+23] M. Bateni, H. Esfandiari, H. Fichtenberger, M. Henzinger, R. Jayaram, V. Mirrokni, and A. Wiese. “Optimal Fully Dynamic  $k$ -Center Clustering for Adaptive and Oblivious Adversaries”. In: *Proceedings of the 2023 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*. SIAM. 2023, pp. 2677–2727 (cit. on p. 4).
- [Bei+22] A. Beimel, H. Kaplan, Y. Mansour, K. Nissim, T. Saranurak, and U. Stemmer. “Dynamic algorithms against an adaptive adversary: Generic constructions and lower bounds”. In: *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing*. 2022, pp. 1671–1684 (cit. on p. 4).
- [Bha+23] R. Bhattacharjee, G. Dexter, C. Musco, A. Ray, and D. P. Woodruff. “Universal Matrix Sparsifiers and Fast Deterministic Algorithms for Linear Algebra”. In: *arXiv preprint arXiv:2305.05826* (2023) (cit. on p. 1).
- [BKS23] S. Bhattacharya, P. Kiss, and T. Saranurak. “Dynamic algorithms for packing-covering LPs via multiplicative weight updates”. In: *Proceedings of the 2023 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*. SIAM. 2023, pp. 1–47 (cit. on pp. 2, 27, 31).
- [BNS19] J. van den Brand, D. Nanongkai, and T. Saranurak. “Dynamic matrix inverse: Improved algorithms and matching conditional lower bounds”. In: *2019 IEEE 60th Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE. 2019, pp. 456–480 (cit. on p. 1).

- [BNS78] J. R. Bunch, C. P. Nielsen, and D. C. Sorensen. “Rank-one modification of the symmetric eigenproblem”. In: *Numerische Mathematik* 31.1 (1978), pp. 31–48 (cit. on p. 1).
- [Gol73] G. H. Golub. “Some modified matrix eigenvalue problems”. In: *SIAM review* 15.2 (1973), pp. 318–334 (cit. on p. 1).
- [IPS10] G. Iyengar, D. J. Phillips, and C. Stein. “Feasible and accurate algorithms for covering semidefinite programs”. In: *Scandinavian Workshop on Algorithm Theory*. Springer. 2010, pp. 150–162 (cit. on p. 31).
- [IPS11] G. Iyengar, D. J. Phillips, and C. Stein. “Approximating semidefinite packing programs”. In: *SIAM Journal on Optimization* 21.1 (2011), pp. 231–268 (cit. on p. 31).
- [Jam+21] A. Jambulapati, Y. T. Lee, J. Li, S. Padmanabhan, and K. Tian. *Positive Semidefinite Programming: Mixed, Parallel, and Width-Independent*. 2021. arXiv: [2002.04830 \[cs.DS\]](#) (cit. on pp. 31, 32).
- [KL96] P. Klein and H.-I. Lu. “Efficient approximation algorithms for semidefinite programs arising from MAX CUT and COLORING”. In: *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*. 1996, pp. 338–347 (cit. on p. 31).
- [LM00] B. Laurent and P. Massart. “Adaptive estimation of a quadratic functional by model selection”. In: *Annals of statistics* (2000), pp. 1302–1338 (cit. on p. 5).
- [LN93] M. Luby and N. Nisan. “A parallel approximation algorithm for positive linear programming”. In: *Proceedings of the twenty-fifth annual ACM symposium on Theory of computing*. 1993, pp. 448–457 (cit. on p. 31).
- [LS17] Y. T. Lee and H. Sun. “An sdp-based algorithm for linear-sized spectral sparsification”. In: *Proceedings of the 49th annual acm sigact symposium on theory of computing*. 2017, pp. 678–687 (cit. on p. 32).
- [MSS19] R. Mitz, N. Sharon, and Y. Shkolnisky. “Symmetric rank-one updates from partial spectrum with an application to out-of-sample extension”. In: *SIAM Journal on Matrix Analysis and Applications* 40.3 (2019), pp. 973–997 (cit. on p. 1).
- [OSV12] L. Orecchia, S. Sachdeva, and N. K. Vishnoi. “Approximating the exponential, the Lanczos method and an  $O(m)$ -time spectral algorithm for balanced separator”. In: *Proceedings of the forty-fourth annual ACM symposium on Theory of computing*. 2012, pp. 1141–1160 (cit. on p. 31).
- [PST95] S. A. Plotkin, D. B. Shmoys, and É. Tardos. “Fast approximation algorithms for fractional packing and covering problems”. In: *Mathematics of Operations Research* 20.2 (1995), pp. 257–301 (cit. on p. 31).
- [PTZ12] R. Peng, K. Tangwongsan, and P. Zhang. “Faster and simpler width-independent parallel algorithms for positive semidefinite programming”. In: *Proceedings of the twenty-fourth annual ACM symposium on Parallelism in algorithms and architectures*. 2012, pp. 101–108 (cit. on pp. 27, 31).
- [Qua20] K. Quanrud. “Nearly linear time approximations for mixed packing and covering problems without data structures or randomization”. In: *Symposium on Simplicity in Algorithms*. SIAM. 2020, pp. 69–80 (cit. on p. 31).

- [San04] P. Sankowski. “Dynamic transitive closure via dynamic matrix inverse”. In: *45th Annual IEEE Symposium on Foundations of Computer Science*. IEEE. 2004, pp. 509–517 (cit. on p. 1).
- [Sta08] P. Stange. “On the efficient update of the singular value decomposition”. In: *PAMM: Proceedings in Applied Mathematics and Mechanics*. Vol. 8. 1. Wiley Online Library. 2008, pp. 10827–10828 (cit. on p. 1).
- [SW23] W. Swartworth and D. P. Woodruff. “Optimal Eigenvalue Approximation via Sketching”. In: *Proceedings of the 55th Annual ACM Symposium on Theory of Computing*. 2023, pp. 145–155 (cit. on p. 1).
- [TB22] L. N. Trefethen and D. Bau. *Numerical linear algebra*. Vol. 181. Siam, 2022 (cit. on p. 2).
- [Tre09] L. Trevisan. “Max cut and the smallest eigenvalue”. In: *Proceedings of the forty-first annual ACM symposium on Theory of computing*. 2009, pp. 263–272 (cit. on p. 32).
- [Tre98] L. Trevisan. “Parallel approximation algorithms by positive linear programming”. In: *Algorithmica* 21.1 (1998), pp. 72–88 (cit. on p. 31).
- [WRM16] D. Wang, S. Rao, and M. W. Mahoney. “Unified Acceleration Method for Packing and Covering Problems via Diameter Reduction”. In: *43rd International Colloquium on Automata, Languages, and Programming (ICALP 2016)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik. 2016 (cit. on p. 31).



## A Connections to Dynamic Positive Semi-definite Programs

This section discusses connections between our Problem 1.1 and the dynamic versions of positive semi-definite programs. Using this connection, we conclude that Theorem 1.2 implies a dynamic algorithm for a special case of the dynamic covering SDP problem.

We first define packing and covering semi-definite programs (SDPs).<sup>5</sup>

**Definition A.1** (Packing/Covering SDP). *Let  $\mathbf{C}, \mathbf{A}_i$ 's for  $i = 1, 2, \dots, m$  be  $n \times n$  symmetric PSD matrices and  $b_i$ 's denote positive real numbers. The packing SDP problem asks to find*

$$\max_{\mathbf{Y} \succeq 0} \text{Tr}[\mathbf{C} \mathbf{Y}] \quad \text{s.t.} \quad \text{Tr}[\mathbf{A}_i \mathbf{Y}] \leq b_i, \quad \forall i = 1, \dots, m.$$

*The covering SDP problem asks to find*

$$\min_{\mathbf{Y} \succeq 0} \text{Tr}[\mathbf{C} \mathbf{Y}] \quad \text{s.t.} \quad \text{Tr}[\mathbf{A}_i \mathbf{Y}] \geq b_i, \quad \forall i = 1, \dots, m.$$

Note that when the matrices  $\mathbf{C}$  and  $\mathbf{A}_i$ 's are all diagonal matrices, the packing and covering SDP problems above are precisely the well-studied packing and covering LP problems, respectively. Near-linear time algorithms for  $(1+\epsilon)$ -approximately solving packing and covering LPs are very well-studied in a long line of work, some of which are [AO15; AO19; WRM16; Qua20], and these problems have many applications, such as in graph embedding [PST95], approximation algorithms [LN93; Tre98], scheduling [PST95], to name a few.

**Dynamic LPs.** Near-optimal dynamic algorithms for packing and covering LPs were shown in [BKS23]. The paper studies two kinds of updates – *restricting* and *relaxing* updates. Restricting updates can only shrink the feasible region. In contrast, relaxing updates can only grow the feasible region. In [BKS23], the authors gave a deterministic algorithm that can maintain a  $(1+\epsilon)$ -approximate solution to either packing and covering LPs that undergo only restricting updates or only relaxing updates in total time  $\tilde{O}(N/\epsilon^3 + t/\epsilon)$ , where  $N$  is the total number of nonzeros in the initial input and the updates, and  $t$  is the number of updates. Hence, this is optimal up to logarithmic factors.

A natural question is whether one can generalize the near-optimal dynamic LP algorithms with polylogarithmic overhead by [BKS23] to work with SDPs since SDPs capture many further applications such as maximum cuts [IPS11; KL96], Sparse PCA [IPS11], sparsest cuts [IPS10], and balanced separators [OSV12], among many others.

**Static SDPs.** Unfortunately, the algorithms for solving packing and covering SDPs are much more limited, even in the static setting. Near-linear time algorithms are known only for *covering* SDPs when the cost matrix  $\mathbf{C} = \mathbf{I}$  is the identity [PTZ12; ALO16].

The fundamental barrier to working with general psd matrix  $\mathbf{C}$  in covering SDPs is that it is as hard as approximating the minimum eigenvalue of  $\mathbf{C}$  (consider the program  $\max_{\mathbf{Y} \succeq 0} \text{Tr}[\mathbf{C} \mathbf{Y}]$  such that  $\text{Tr}[\mathbf{Y}] \leq 1$ ). To the best of our knowledge, near-linear time algorithms for approximating the minimum eigenvalue assume a near-linear-time solver for  $\mathbf{C}$ , i.e., to compute  $\mathbf{C}^{-1}\mathbf{x}$  in the near-linear time given  $\mathbf{x}$ . This can be done, for example, by applying the power method to  $\mathbf{C}^{-1}$ .

---

<sup>5</sup>Some papers [Jam+21] flip our definition of packing and covering SDPs by considering their dual form.

When  $\mathbf{C}^{-1}$  admits a fast solver, sometimes one can approximately solve a covering SDP fast, such as for spectral sparsification of graphs [LS17], and the max-cut problem [AK07; Tre09].

For packing SDPs, there is simply no near-linear time algorithm known. An algorithm for approximately solving packing SDPs and even the generalization to mixed packing-covering SDPs was claimed in [Jam+21], but there is an issue in the convergence analysis even for pure packing SDPs. Fast algorithms for this problem, hence, remain open.

**Dynamic SDPs.** Since near-linear time static algorithms are prerequisites for dynamic algorithms with polylogarithmic overhead, we can only hope for a dynamic algorithm for covering SDPs when  $\mathbf{C}$  is an identity. Below, we will show that our algorithm for Theorem 1.2 implies a dynamic algorithm for maintaining the covering SDP solution when there is a single constraint and the updates are restricting. This follows because this problem is equivalence to Problem 1.1.

We first define the dynamic covering problem with a single constraint under restricting updates.

**Problem A.2** (Covering SDP with a Single Matrix Constraint under Restricting Updates). *Given  $\mathbf{A}_0 \succeq 0$ , an accuracy parameter  $\epsilon > 0$ , and an online sequence of vectors  $\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_T$  that update  $\mathbf{A}_t \leftarrow \mathbf{A}_t - \mathbf{v}_t \mathbf{v}_t^\top$  such that  $\mathbf{A}_t \succeq 0$ . The problem asks to explicitly maintain, for all  $t$ , an  $(1 + \epsilon)$ -approximate optimal value  $\nu_t$  of the SDP, i.e.,*

$$\nu_t \leq (1 + \epsilon) \text{OPT}_t \stackrel{\text{def}}{=} \min_{\mathbf{Y} \succeq 0, \text{Tr}[\mathbf{A}_t \mathbf{Y}] \geq 1} \text{Tr}[\mathbf{Y}].$$

Furthermore, given a query request, return a matrix  $\mathbf{Q}^{(t)}$  where  $\mathbf{Y}^{(t)} = \mathbf{Q}^{(t)} \mathbf{Q}^{(t)\top} \in \mathbb{R}^{n \times n}$  is a  $(1 + \epsilon)$ -approximate optimal solution, i.e.,

$$\text{Tr}[\mathbf{A}_t \mathbf{Y}^{(t)}] \geq 1 \text{ and } \text{Tr}[\mathbf{Y}^{(t)}] \leq (1 + \epsilon) \text{OPT}_t.$$

Problem A.2 is equivalent to Problem 1.1 in the following sense: given an algorithm for Problem 1.1, we can obtain an algorithm for Problem A.2 with the same total update time and optimal query time. Conversely, given an algorithm for Problem A.2, we can obtain an algorithm for Problem 1.1 in the *eigenvalue-only* version with the same total update time.

**Proposition A.3.** *The following holds,*

1. *Given an algorithm for Problem 1.1 with update time  $\mathcal{T}$ , there is an algorithm for Problem A.2 with update time  $\mathcal{T}$  and query time  $O(n)$ , i.e., the time required to query  $\mathbf{Q}^{(t)}$  at any time  $t$ .*
2. *Given an algorithm for Problem A.2 with update time  $\tilde{\mathcal{T}}$ , there is an algorithm for Problem 1.1 that only maintains the approximate eigenvalues with update time at most  $\tilde{\mathcal{T}}$ .*

*Proof.* Let us first characterize the solution to Problem A.2 at any  $t$  for  $\epsilon = 0$ . Since any feasible solution  $\mathbf{Y}$  is PSD it can be characterized as,  $\mathbf{Y} = \sum_{i=1}^n p_i \mathbf{y}_i \mathbf{y}_i^\top$  for some unit vectors  $\mathbf{y}_i$ 's. Now  $\text{Tr}[\mathbf{Y}] = \sum_i p_i$  and

$$1 \leq \text{Tr}[\mathbf{A}_t \mathbf{Y}] = \sum_{i=1}^n p_i \text{Tr}[\mathbf{A}_t \mathbf{y}_i \mathbf{y}_i^\top] = \sum_{i=1}^n p_i \mathbf{y}_i^\top \mathbf{A}_t \mathbf{y}_i \leq \lambda_{\max}(\mathbf{A}_t) \sum_{i=1}^n p_i.$$

The above implies that for any feasible solution  $\mathbf{Y}^{(t)}$ , it must hold that  $\sum_{i=1}^n p_i \geq 1/\lambda_{\max}(\mathbf{A}_t)$ , and equality holds iff  $\mathbf{y}_i$ 's are the maximum eigenvector of  $\mathbf{A}_t$  for all  $t$ . Since the problem is asking to

minimize  $\sum_{i=1}^n p_i$ , the solution must be  $\mathbf{Y}^{(t)} = \frac{1}{\lambda_{\max}(\mathbf{A}_t)} \mathbf{u} \mathbf{u}^\top$  where  $\mathbf{u}$  is the maximum eigenvector of  $\mathbf{A}_t$ . Note that here  $\mathbf{Q}^{(t)} = \frac{1}{\sqrt{\lambda_{\max}(\mathbf{A}_t)}} \mathbf{u}$ .

We now show the first part, by proving that at any  $t$ , given  $\epsilon$ , the solution to Problem 1.1 gives a solution to Problem A.2. Let  $\lambda_t$  and  $\mathbf{w}_t$  denote an  $\epsilon/2$ -approximate solution to Problem 1.1 for some  $t$ . Consider the solution  $\mathbf{Q}^{(t)} = \frac{1}{\sqrt{(1-\epsilon/2)\lambda_t}} \mathbf{w}_t$  which gives  $\mathbf{Y}^{(t)} = \frac{1}{(1-\epsilon/2)\lambda_t} \mathbf{w}_t \mathbf{w}_t^\top$ . Then,

$$\text{Tr}[\mathbf{A}_t \mathbf{Y}^{(t)}] = \frac{1}{(1-\epsilon/2)\lambda_t} \mathbf{w}_t^\top \mathbf{A}_t \mathbf{w}_t \geq (1-\epsilon/2) \frac{\lambda_{\max}(\mathbf{A}_t)}{(1-\epsilon/2)\lambda_{\max}(\mathbf{A})} \geq 1.$$

Therefore,  $\mathbf{Y}^{(t)}$  satisfies the constraints of Problem A.2. Next we look at the objective value,  $\nu_t = \text{Tr}[\mathbf{Y}^{(t)}] = \frac{1}{(1-\epsilon/2)\lambda_t} \leq \frac{1}{(1-\epsilon/2)^2 \lambda_{\max}(\mathbf{A}_t)} \leq (1+\epsilon)OPT_t$ . We can maintain  $\mathbf{Q}^{(t)}$  by just maintaining  $\mathbf{w}_t, \lambda_t$  which requires no extra time. The time required to obtain  $\mathbf{Q}^{(t)}$ , which is the query time, from  $\mathbf{w}_t$  and  $\lambda_t$  is at most  $O(nnz(\mathbf{w}_t)) = O(n)$  and the value of  $\nu_t$  can be obtained in  $O(1)$  time.

To see the other direction, consider the solution  $\mathbf{Q}^{(t)}, \nu_t$  of Problem A.2. We can set  $\lambda_t = \frac{1}{\nu_t}$ . This implies that,

$$\lambda_t = \frac{1}{\nu_t} \geq \frac{\lambda_{\max}(\mathbf{A}_t)}{(1+\epsilon)} \geq (1-\epsilon)\lambda_{\max}(\mathbf{A}_t),$$

as required. In this case, we do not require any extra time.  $\square$

By plugging Theorem 1.2 into Proposition A.3, we conclude the following.

**Corollary A.4.** *There is a randomized algorithm for Problem A.2 under a sequence of  $T$  restricting updates, that given  $n, \mathbf{A}_0$  and  $\epsilon > 1/n$  as input, with probability at least  $1 - 1/n$  works against an oblivious adversary in total update time*

$$O\left(\frac{\log^3 n \log^6 \frac{n}{\epsilon} \log \frac{\lambda_{\max}(\mathbf{A}_0)}{\lambda_{\max}(\mathbf{A}_T)}}{\epsilon^4} \left(nnz(\mathbf{A}_0) + \sum_{i=1}^T nnz(\mathbf{v}_i)\right)\right),$$

and query time  $O(n)$ .

## B Omitted Proofs

**Lemma 3.2.** *Given an algorithm  $\mathcal{A}$  that solves the decision problem  $\text{DECMAXEV}(\epsilon, \mathbf{A}_0, \mathbf{v}_1, \dots, \mathbf{v}_T)$  (Definition 3.1) for any  $\epsilon > 0$ ,  $\mathbf{A}_0 \succeq 0$  and vectors  $\mathbf{v}_1, \dots, \mathbf{v}_T$  in time  $\mathcal{T}$ , we can solve Problem 1.1 in total time  $O\left(\frac{\log^2 n \log \frac{n}{\epsilon}}{\epsilon} \cdot nnz(\mathbf{A}_0) + \frac{\log n}{\epsilon} \log \frac{\lambda_{\max}(\mathbf{A}_0)}{\lambda_{\max}(\mathbf{A}_T)} \mathcal{T}\right)$ .*

*Proof.* We first estimate the maximum eigenvalue of  $\mathbf{A}_0$ . This can be done via the standard power method, or running  $\text{POWERMETHOD}(\epsilon/4 \log n, \mathbf{A}_0)$  until Line 9 and returning  $\mathbf{w} \in \mathbf{W}$  such that  $\mathbf{w} = \arg \max_{\mathbf{w} \in \mathbf{W}} \mathbf{w}^\top \mathbf{A}_0 \mathbf{w}$ . From Lemma 3.3, with probability at least  $1 - 1/n^{10}$ ,  $\mathbf{w}$  satisfies  $\nu = \mathbf{w}^\top \mathbf{A}_0 \mathbf{w} \geq (1 - \epsilon/\log n) \lambda_{\max}(\mathbf{A}_0)$ . Now,  $\mathbf{A}_0/\nu$  has maximum eigenvalue at most  $1 + \frac{\epsilon}{\log n}$ .

This procedure takes time at most  $O\left(\frac{\log^2 n \log \frac{n}{\epsilon}}{\epsilon} \cdot nnz(\mathbf{A}_0)\right)$ .

For the sequence of updates, let  $t_1, t_2, \dots, t_k$  denote the time steps where the maximum eigenvalue decreases by a factor of at least  $1 - \epsilon/\log n$ . We begin with solving  $\text{DECMAXEV}(\epsilon, \frac{\mathbf{A}_0}{\nu}, \frac{\mathbf{v}_1}{\sqrt{\nu}}, \dots, \frac{\mathbf{v}_T}{\sqrt{\nu}})$

using  $\mathcal{A}$ . The algorithm returns  $\mathbf{w}_1, \mathbf{w}_2, \dots, \mathbf{w}_{t_1-1}, \text{FALSE}, \dots$ . Now  $\nu$  is an  $\epsilon$ -approximate maximum eigenvalue for  $\mathbf{A}_1, \dots, \mathbf{A}_{t_1-1}$ , and the vectors  $\mathbf{w}_1, \dots, \mathbf{w}_{t_1-1}$  are the required eigenvectors for  $\mathbf{A}_1, \dots, \mathbf{A}_{t_1-1}$  respectively. We also know that,  $\lambda_{\max}(\mathbf{A}_{t_1}) \leq \nu \left(1 - \frac{\epsilon}{\log n}\right)$ . The total time taken so far is,  $O\left(\frac{\log^2 n \log \frac{n}{\epsilon}}{\epsilon} \cdot \text{nnz}(\mathbf{A}_0) + \mathcal{T}\right)$ .

Since,  $\lambda_{\max}\left(\frac{\mathbf{A}_{t_1}}{\nu(1-\frac{\epsilon}{\log n})}\right) \leq 1$ , again solve  $\text{DECMAXEV}(\epsilon, \frac{\mathbf{A}_{t_1}}{\nu(1-\frac{\epsilon}{\log n})}, \frac{\mathbf{v}_{t_1}}{\sqrt{\nu(1-\frac{\epsilon}{\log n})}}, \dots, \frac{\mathbf{v}_T}{\sqrt{\nu(1-\frac{\epsilon}{\log n})}})$  using  $\mathcal{A}$ . This time, the algorithm returns  $\mathbf{w}_{t_1+1}, \mathbf{w}_{t_1+2}, \dots, \mathbf{w}_{t_2-1}, \text{FALSE}, \dots$ . We can now repeat this process starting at  $t_2$  until  $t_k$ .

The total number of calls to  $\mathcal{A}$  can be bounded by the number of times the maximum eigenvalue decreases by a factor of  $1 - \frac{\epsilon}{\log n}$ . This can happen at most  $\frac{\log n}{\epsilon} \log \frac{\lambda_{\max}(\mathbf{A}_0)}{\lambda_{\max}(\mathbf{A}_T)}$  times. Therefore, the total time required is at most  $O\left(\frac{\log^2 n \log \frac{n}{\epsilon}}{\epsilon} \cdot \text{nnz}(\mathbf{A}_0) + \frac{\log n}{\epsilon} \log \frac{\lambda_{\max}(\mathbf{A}_0)}{\lambda_{\max}(\mathbf{A}_T)} \mathcal{T}\right)$ .  $\square$

## Power Method

The following lemma proves that at least one initial random vector has a component along the eigenspace of the maximum eigenvalue.

**Lemma B.1.** *Let  $\mathbf{u}_1, \dots, \mathbf{u}_l$  denote the maximum eigenvectors of  $\mathbf{A}$  and let  $\mathbf{v}^{(0)} \in \mathbb{R}^n$  denote the vector with entries sampled independently from  $N(0, 1)$ , i.e.,  $\mathbf{v}^{(0)} = \sum_{i=1}^n \alpha_i \mathbf{u}_i$ , where  $\alpha_i \sim N(0, 1)$ ,  $\mathbf{u}_i$ 's are eigenvectors of  $\mathbf{A}$ . Then, with probability at least  $3/4$ ,  $\sum_{i=1}^l \alpha_i^2 \geq \frac{1}{25}$ .*

*Proof.* Let  $\lambda_1, \lambda_2, \dots, \lambda_n$  denote the eigenvalues of  $\mathbf{A}$  in decreasing order corresponding to the eigenvectors  $\mathbf{u}_1, \dots, \mathbf{u}_n$ . Let  $\mathbf{u}$  be a vector in the span of  $\mathbf{u}_1, \dots, \mathbf{u}_l$ . We can write,  $\mathbf{u} = \sum_{i=1}^l \beta_i \mathbf{u}_i$  with  $\|\beta\| = 1$ . We note that the inner product  $\langle \mathbf{v}^{(0)}, \mathbf{u} \rangle = \sum_{i=1}^l \alpha_i \beta_i$  has mean 0 and variance  $\sum_i \text{Var}(\alpha_i) \beta_i^2 = 1$  (since  $\text{Var}(\alpha_i) = 1$  and  $\sum_i \beta_i^2 = 1$ ), and is also a gaussian variable, and thus follows the distribution  $N(0, 1)$ . Therefore from the standard gaussian tables,

$$\Pr\left[|\langle \mathbf{v}^{(0)}, \mathbf{u} \rangle| \geq \frac{1}{5}\right] \geq \frac{3}{4}.$$

Using Cauchy-Schwarz,  $|\langle \mathbf{v}^{(0)}, \mathbf{u} \rangle| \leq \left(\sum_{i=1}^l \alpha_i^2\right)^{1/2}$ . As a result, we also have,

$$\Pr\left[\left(\sum_{i=1}^l \alpha_i^2\right)^{1/2} \geq \frac{1}{5}\right] \geq \frac{3}{4}.$$

$\square$

**Lemma 3.3.** *Let  $\epsilon > 0$  and  $\mathbf{A} \succeq 0$ . Let  $\mathbf{W}$  be as defined in Line 9 in the execution of  $\text{POWERMETHOD}(\epsilon, \mathbf{A})$ . With probability at least  $1 - 1/n^{10}$ , for some  $\mathbf{w} \in \mathbf{W}$ , it holds that  $\mathbf{w}^\top \mathbf{A} \mathbf{w} \geq (1 - \frac{\epsilon}{2}) \lambda_{\max}(\mathbf{A})$ . The total time taken by the algorithm is at most  $O\left(\frac{\text{nnz}(\mathbf{A}) \log n \log \frac{n}{\epsilon}}{\epsilon}\right)$ .*

Furthermore, let  $\lambda_i$  and  $\mathbf{u}_i$  denote the eigenvalues and eigenvectors of  $\mathbf{A}$ . For all  $i$  such that  $\lambda_i(\mathbf{A}) \leq \frac{\lambda_{\max}(\mathbf{A})}{2}$ , with probability at least  $1 - 2/n^{10}$ ,  $\left[\mathbf{w}^\top \mathbf{u}_i\right]^2 \leq \frac{1}{n^8} \cdot \frac{\lambda_i}{\lambda_1}$ .

*Proof.* Algorithm 1 starts with a random vector  $\mathbf{v}^{(0,r)}$  (Line 5). Let  $l$  denote the dimension of the eigenspace corresponding to the maximum eigenvalue. We denote  $\mathbf{v}^{(0,r)} = \sum_i \alpha_i^{(r)} \mathbf{u}_i$  and from Lemma B.1, we have that for all  $r$ , initial vectors,

$$\Pr \left[ \sum_{i=1}^l (\alpha_i^{(r)})^2 \geq \frac{1}{25} \right] \geq 3/4.$$

We now analyze Algorithm 1. After  $K$  iterations, the algorithm computes,

$$\mathbf{A}^K \mathbf{v}^{(0,r)} = \sum_{i=1}^n \alpha_i^{(r)} \lambda_i^K \mathbf{u}_i.$$

Note that, since  $K = \frac{4 \log \frac{n}{\epsilon}}{\epsilon}$ , for every  $\lambda_i \leq (1 - \epsilon/4) \lambda_1$ , we have  $\lambda_i^{2K} \leq \frac{\epsilon^2}{n^2} \lambda_1^{2K}$ . Let  $\tilde{l}$  denote the dimension of the space spanned by all eigenvectors of  $\mathbf{A}$  with corresponding eigenvalues at least  $(1 - \epsilon/4) \lambda_1$ , we split the sum as,

$$\mathbf{v}^{(0,r)\top} \mathbf{A}^{2K+1} \mathbf{v}^{(0,r)} = \sum_{i=1}^{\tilde{l}} (\alpha_i^{(r)})^2 \lambda_i^{2K+1} + \sum_{i>\tilde{l}} (\alpha_i^{(r)})^2 \lambda_i^{2K+1}.$$

Now,

$$(\mathbf{w}^{(r)})^\top \mathbf{A} \mathbf{w}^{(r)} = \frac{(\mathbf{A}^K \mathbf{v}^{(0,r)})^\top \mathbf{A} (\mathbf{A}^K \mathbf{v}^{(0,r)})}{\|\mathbf{A}^K \mathbf{v}^{(0,r)}\|^2} = \frac{\mathbf{v}^{(0,r)\top} \mathbf{A}^{2K+1} \mathbf{v}^{(0,r)}}{\|\mathbf{A}^K \mathbf{v}^{(0,r)}\|^2} \geq \frac{\sum_{i=1}^{\tilde{l}} (\alpha_i^{(r)})^2 \lambda_i^{2K+1}}{\|\mathbf{A}^K \mathbf{v}^{(0,r)}\|^2}. \quad (13)$$

We now want to bound  $\|\mathbf{A}^K \mathbf{v}^{(0,r)}\|^2$ . We again expand this quantity and use the bounds on  $\lambda_i$ 's to get,

$$\|\mathbf{A}^K \mathbf{v}^{(0,r)}\|^2 = \sum_{i=1}^n (\alpha_i^{(r)})^2 \lambda_i^{2K} \leq \sum_{i=1}^{\tilde{l}} (\alpha_i^{(r)})^2 \lambda_i^{2K} + \left(\frac{\epsilon}{n}\right)^2 \lambda_1^{2K} \sum_{i>\tilde{l}} (\alpha_i^{(r)})^2 \leq \sum_{i=1}^{\tilde{l}} (\alpha_i^{(r)})^2 \lambda_i^{2K} + \lambda_1^{2K} \left(\frac{\epsilon}{n}\right)^2 \|\mathbf{v}^{(0,r)}\|^2.$$

Since  $\mathbf{v}^{(0,r)}$  has entries in  $N(0, 1)$ , from Lemma 2.4, with probability at least  $1 - 1/n^2$ ,  $\|\mathbf{v}^{(0,r)}\|^2 \leq 2n$  and as a result, with probability at least  $1 - 1/n^2$ ,  $\|\mathbf{A}^K \mathbf{v}^{(0,r)}\|^2 \leq \sum_{i=1}^{\tilde{l}} (\alpha_i^{(r)})^2 \lambda_i^{2K} + \left(\frac{\epsilon}{n}\right) \lambda_1^{2K}$  when  $n \geq 5$ . Using this bound in (13) and  $\epsilon < 1/20$ ,

$$(\mathbf{w}^{(r)})^\top \mathbf{A} \mathbf{w}^{(r)} \stackrel{(a)}{\geq} \frac{\lambda_{\tilde{l}} \sum_{i=1}^{\tilde{l}} (\alpha_i^{(r)})^2 \lambda_i^{2K}}{\sum_{i=1}^{\tilde{l}} (\alpha_i^{(r)})^2 \lambda_i^{2K} + \epsilon n^{-1} \lambda_1^{2K}} \stackrel{(b)}{\geq} \frac{\lambda_{\tilde{l}}}{1 + \frac{\epsilon n^{-1}}{\sum_{i=1}^{\tilde{l}} (\alpha_i^{(r)})^2}} \stackrel{(c)}{\geq} \frac{(1 - \epsilon/4) \lambda_1}{1 + \epsilon/4} \stackrel{(d)}{\geq} (1 - \epsilon/2) \lambda_1,$$

with probability at least  $3/5$ . To see this, in (a) we used that  $\|\mathbf{A}^K \mathbf{v}^{(0,r)}\|^2 \leq \sum_{i=1}^{\tilde{l}} (\alpha_i^{(r)})^2 \lambda_i^{2K} + \left(\frac{\epsilon}{n}\right) \lambda_1^{2K}$  with probability at least  $1 - 1/n^2$ , in (b) we used that  $\sum_{i=1}^{\tilde{l}} (\alpha_i^{(r)})^2 \lambda_i^{2K} \geq \lambda_1^{2K} \sum_{i=1}^{\tilde{l}} (\alpha_i^{(r)})^2$ , and in (c) we used that  $\sum_{i=1}^{\tilde{l}} (\alpha_i^{(r)})^2 \geq 1/25$  with probability at least  $3/4$ .

We next prove our concentration bound, i.e., for at least one  $r \in \{1, \dots, 10 \log n\}$ ,  $(\mathbf{w}^{(r)})^\top \mathbf{A} \mathbf{w}^{(r)} \geq (1 - \epsilon/2) \lambda_1$  with probability at least  $1 - 1/n$ . Observe that the probability that  $(\mathbf{w}^{(r)})^\top \mathbf{A} \mathbf{w}^{(r)} < (1 - \epsilon/2) \lambda_1$  for one  $r$  is at most  $2/5$ . Since we have  $R = 10 \log n$  values of  $r$ , the probability that

all of them are such that  $(\mathbf{w}^{(r)})^\top \mathbf{A} \mathbf{w}^{(r)} < (1 - \epsilon/2)\lambda_1$  is  $\left(\frac{2}{5}\right)^R \leq \frac{1}{n^{10}}$ . Therefore, we must have  $(\mathbf{w}^{(r)})^\top \mathbf{A} \mathbf{w}^{(r)} \geq (1 - \epsilon/2)\lambda_1$  for at least one  $r$  with probability at least  $1 - 1/n^{10}$ .

It remains to prove the runtime. In every iteration, we are multiplying  $\mathbf{A}$  with a vector. This takes time  $\text{nnz}(\mathbf{A})$ . The total number of iterations is  $O(\log \frac{n}{\epsilon})$  for every initial random vector  $\mathbf{v}^{(0,r)}$ . We run the entire method  $10 \log n$  times, giving us a total runtime of  $O(\text{nnz}(\mathbf{A}) \frac{\log^2 \frac{n}{\epsilon}}{\epsilon})$ . In the end we compute  $(\mathbf{w}^{(r)})^\top \mathbf{A} \mathbf{w}^{(r)}$  for  $r = 1, \dots, 10 \log n$ . This takes an additional time of  $O(\log n \cdot \text{nnz}(\mathbf{A}))$ .

We now prove that for  $i$  such that  $\lambda_i \leq \lambda_1/2$ , for some  $r$ , with probability at least  $1 - 1/n^{10}$ ,  $(\mathbf{w}^{(r)})^\top \mathbf{u}_i \leq \frac{1}{n^8} \cdot \frac{\lambda_i}{\lambda_1}$ . We use the same argument as above to get that for at least one  $r$ ,  $\sum_{i=1}^d (\alpha^{(r)})_i^2 \geq 1/25$  with probability at least  $1 - 1/n^{10}$ . For this  $r$ , and  $\epsilon < 1/20$ ,

$$(\mathbf{w}^{(r)})^\top \mathbf{u}_i = \frac{\alpha_i \lambda_i^K}{\sqrt{\sum_{i=1}^n \alpha_i^2 \lambda_i^{2K}}} \leq \frac{\alpha_i \lambda_i^{K-1/2} \lambda_i^{1/2}}{\lambda_1^K \sqrt{\sum_{i=1}^d \alpha_i^2}} \leq 5\alpha_i \left(\frac{\lambda_i}{\lambda_1}\right)^{1/2} \left(\frac{1}{2}\right)^{K-1/2} \leq 5\alpha_i \left(\frac{\lambda_i}{\lambda_1}\right)^{1/2} (1 - 10\epsilon)^{K-1/2}.$$

Now,  $\alpha_i \leq 10 \log n$  with probability at least  $1 - 1/n^{10}$ . Therefore, we have with probability at least  $1 - 2/n^{10}$ ,

$$\left[(\mathbf{w}^{(r)})^\top \mathbf{u}_i\right]^2 \leq 2500 \frac{\lambda_i}{\lambda_1} \log^2 n (1 - 10\epsilon)^{2K-1} \leq \frac{2500\epsilon^{10} \log n}{n^{10}} \frac{\lambda_i}{\lambda_1} \leq \frac{1}{n^8} \cdot \frac{\lambda_i}{\lambda_1}.$$

□