On approximating the eigenvalues of stochastic matrices in probabilistic logspace

Dean Doron, Amir Sarid, and Amnon Ta-Shma

The Blavatnik School of Computer Science
Tel-Aviv University
Israel 69978

September 6, 2016

Abstract

We show that approximating the second eigenvalue of stochastic operators is BPL–complete, thus giving a *natural* problem complete for this class. We also show that approximating any eigenvalue of a stochastic and *Hermitian* operator with constant accuracy can be done in BPL.

This work together with related work on the subject reveals a picture where the various space-bounded classes (e.g., probabilistic logspace, quantum logspace and the class DET) can be characterized by algebraic problems (such as approximating the spectral gap) where, roughly speaking, the difference between the classes lies in the kind of operators they can handle (e.g., stochastic, Hermitian or arbitrary).

1 Introduction

Derandomization is a major challenge of theoretical computer science. In the time-bounded model there are conditional results showing that $\mathsf{BPP} = \mathsf{P}$ under certain plausible assumptions ([22, 13] to cite a few). Showing $\mathsf{BPP} = \mathsf{P}$ would immediately give deterministic polynomial-time algorithms to many central problems which currently are not known to be in P such as polynomial identity testing and polynomial factorization over arbitrary finite fields (see, e.g., [16]).

In the space-bounded model there are also conditional results showing that $\mathsf{BPL} = \mathsf{L}$ under certain plausible assumptions [15]. However, in the space-bounded model there are also *unconditional* results, showing partial derandomization of the probabilistic classes. Nisan [20] constructed a pseudorandom generator (PRG) against logarithmic space-bounded non-uniform algorithms that use seed length $O(\log^2 n)$. Using that he showed BPL is contained in the class having simultaneously polynomial time and $O(\log^2 n)$ space [21]. Saks and Zhou [27] showed BPL is contained in $\mathsf{DSPACE}(\log^{1.5} n)$. Many conjecture that, in fact, $\mathsf{BPL} = \mathsf{L}$.

There are not too many natural problems in BPL that are not known to be in L. Aleliunas et al. [1] showed that undirected st-connectivity is in RL, and many natural problems reduce (via logspace reductions) to undirected connectivity [18, 2], but in a seminal result Reingold [23] showed undirected st-connectivity already belongs to L. To our knowledge, there are no other natural languages in BPL that are not known to be in L.

In this paper we show that the problem of approximating the spectral gap of a stochastic matrix is BPL–complete. Estimating the spectral gap is a natural problem, interesting in its own right and captured a lot of attention in graph theory, numerical linear algebra and quantum computation. We also believe the fact it is BPL–complete contributes to the related work done on the problem, which we discuss later on.

The eigenvalues of a linear operator A are the roots of its characteristic polynomial $p(\lambda) = \det(\lambda I - A)$. Any linear operator has a full set of eigenvalues, even though it may not have a full set of eigenvectors. The eigenvalues of a *stochastic* matrix are always bounded by 1, and the all-one vector is a 1-eigenvector. Also, the Perron Frobenius theorem tells us that if a graph is strongly connected and aperiodic then the 1-eigenvector of its transition matrix is unique. If the graph is periodic with a period h then its transition matrix has h complex eigenvalues corresponding to the h different h-roots of unity. We now present the problem more formally:

Definition 1.1 (The promise problem $SecondEV_{\alpha,\beta}$)**.** *The input is $0 \leq \alpha < \beta < 1$ and a stochastic matrix A of dimension n with possibly complex eigenvalues $\lambda_1, \ldots, \lambda_n$. We have the promise that $\lambda_1 = 1$ and $\lambda_2 \in \mathbb{R}$. Also, $|\lambda_n| \leq \ldots \leq |\lambda_3| \leq 1 - \beta$.*

No instances *: $\lambda_2 \geq 1 - \alpha$.*

Yes instances *: $\lambda_2 \leq 1 - \beta$.*

We prove:

Theorem 1.1. *For every non-negligible ζ, $SecondEV_{\zeta,2\zeta}$ is in BPL and $SecondEV_{\frac{1}{n},\frac{2}{n}}$ is BPL–complete.*

We also consider the problem of estimating specific eigenvalues and not only the spectral gap. We define:

Definition 1.2 ($EV_{\alpha,\beta}$)**.** *The input is a stochastic, Hermitian matrix A, $\lambda \in [-1, 1]$ and $\alpha < \beta$.*

Yes instances *: There is an eigenvalue λ_i of A such that $|\lambda_i - \lambda| \leq \alpha$.*

No instances *: All eigenvalues of A are β–far from λ.*

We prove:

Theorem 1.2. *For every constants $\alpha < \beta$, the promise problem $EV_{\alpha,\beta}$ belongs to* BPL.

1.1 Our technique

We wish to approximate eigenvalues of a given linear operator A whose spectrum is contained in $[0, 1]$. We manipulate the eigenvalues of the input matrix A, without knowing the decomposition of A to eigenvectors and eigenvalues, by using the simple fact that if $\lambda_1, \ldots, \lambda_n$ are the roots of the characteristic polynomial of A, and if p is an arbitrary univariate polynomial, then $p(\lambda_1), \ldots, p(\lambda_n)$ are the roots of the characteristic polynomial of the matrix $p(A)$. The core idea is then to take a polynomial p with a sharp peak around $\lambda \in [0, 1]$ (namely, a polynomial for which $p(x)$ is large around λ and small for values far from λ) and estimate Tr $(p(A))$. Applying p on A amplifies the eigenvalues that are close to λ and damps eigenvalues that are far from λ and so approximating Tr $(p(A))$ approximately counts the number of eigenvalues close to λ. We are therefore left with the problem of approximating the entries of $p(A)$ in BPL.

Approximating matrix powering of *stochastic* matrices is in BPL. To see that, assume A is a stochastic matrix. Then, one can approximate $A^k[s, t]$ by estimating the probability a random walk over A starting at s reaches t after k steps. We say a matrix A is *simulatable* if a probabilistic logspace algorithm can approximate $A^k[s, t]$ for any k polynomial in n and with polynomially-small accuracy (see Definition 2.3 for the exact details). We show that even non-stochastic matrices A with negative or complex entries are simulatable as long as A has infinity norm at most 1, namely, those matrices A for which all rows $i \in [n]$ have ℓ_1 norm at most 1, $\sum_j |A[i, j]| \leq 1$.

If A is simulatable and the coefficients of $p(x) = \sum_i c_i x^i$ are not too large (i.e., only polynomially large in n), then we can approximate in BPL the matrix $p(A) = \sum_i c_i A^i$. In particular, we can also approximate Tr $(p(A))$. By taking an appropriate polynomial p (see Section 4.1) we can solve $EV_{\alpha,\beta}$ for constants $\alpha < \beta$ (see Section 4). We remark that there are many possible candidate polynomials p with a peak around λ. However, in Theorem 4.4 we prove that no polynomial can do significantly better than the one we use.

While the algorithm is simple, we believe it features a new component that has not been used before by probabilistic space-bounded algorithms. An algorithm that takes a random walk on a graph and takes a decision based on the walk length and connectivity properties of the graph (as, e.g., [1]) works with some power of the input matrix A. More generally, such an algorithm can work with a convex combination of powers of the input matrix (by probabilistically choosing which power to take). The algorithm we present in this paper utilizes *arbitrary* (positive or *negative*) combinations of matrix powers and we believe it is a crucial feature of the solution. We are not aware of previous BPL algorithms using such a feature.

The approach outlined above does *not* work for approximating the eigenvalues of *directed* graphs. It is still true that the resulting operator A is stochastic and therefore simulatable. Also, it remains true that if λ is an eigenvalue of A (i.e., a root of the characteristic polynomial) then $p(\lambda)$ is a root of $p(A)$. However, the eigenvalues λ of A may be complex and we do not know how to control $p(\lambda)$ when p has both negative and positive coefficients.[1]

[1]The reason why we manage to prove Theorem 1.1 for arbitrary stochastic operators, is that in the special case of the second eigenvalue (and when the first eigenvalue is known) we can amplify with a polynomial having only positive coefficients, and then we can control the complex eigenvalues.

1.2 Subsequent and related work

One can also study the problem of approximating the spectral gap (or the whole spectrum) of Hermitian matrices. An even harder problem is that of approximating the singular values (or the singular value decomposition – SVD) of *arbitrary* linear operators.[2]

In 1999, Watrous [32] defined the model of quantum logspace computation, and proved several facts on it. The definition was modified several times, see, [31]. Roughly speaking, a language is in BQL if there exists an L–uniform family of quantum circuits solving the language with only $O(\log n)$ qubits. The quantum circuits are over some universal basis of gates (e.g., CNOT, HAD, T) plus intermediate measurements (that in particular may simulate a stream of random coins). For details we refer the reader to [31, 28]. In [28], building on an earlier work [11], it is shown that it is possible to approximate the SVD (and in particular the spectral gap) in BQL, with polynomially-small accuracy (for exact details see [28, Theorem 1.2]). This readily shows one can approximately invert a well-conditioned matrix in BQL.

Recently, Fefferman and Lin [9] showed (using Marriott-Watrous amplification) that the result also holds for the model without intermediate measurements. Furthermore, they proved that matrix inversion is in fact *complete* for BQL. They also addressed the problem of approximating the minimal eigenvalue of a positive semi-definite Hermitian matrix and proved that this problem is complete for BQL as well. In this view, roughly speaking, BPL is equivalent to approximating a specific eigenvalue of stochastic matrices, while BQL is equivalent to approximating a specific eigenvalue of Hermitian matrices.

There is a corresponding phenomenon regarding the whole spectrum of operators. In BQL one can approximate the singular values of any operator, and the eigenvalues of any Hermitian operator, with polynomially-small accuracy. In contrast, the results of this paper show that in BPL one can approximate the eigenvalues of any Hermitian *and stochastic* operator, but only with *constant* accuracy.

We remark that the usual way of describing the quantum algorithm for eigenvalue approximation is that it applies quantum phase estimation on the completely mixed state. The completely mixed state is a uniform mixture of the pure states that are formed from the eigenvectors of A, and on each such eigenvector, the quantum phase estimation estimates the corresponding eigenvalue. Thus, if the procedure can be run in (quantum) logspace, we essentially sample a random eigenvector/eigenvalue pair, and from that we can approximately get the SVD of A.

Another way of looking at the quantum algorithm (which becomes apparent when one computes the acceptance probability of the measurement done in the algorithm) is that the algorithm approximates Tr $(p(A))$ for p being a shift of the Fejér kernel (see, e.g., [12, Chapter 2]). The Fejér Kernel is a high-degree polynomial with a sharp peak around the shift. Thus, the quantum algorithm also uses a similar framework to the one used in this paper (in fact, the quantum algorithm preceded this paper and motivated the approach we take here). The reason the quantum algorithm achieves polynomially-small accuracy and bypasses the impossibility result of Section 4.2 is because it is able to take p up to some polynomial degree (rather than logarithmic degree), not worrying about the quite large size of the coefficients, thus leading to much better accuracy.

In another direction, Le Gall [17] recently showed that given the transition matrix A of a weighted, undirected graph, and a vector b, one can approximately solve the linear system $Ax = b$ in BPL with polynomially-small accuracy.[3] Le Gall's technique uses our framework. In fact, the polynomial used by Le Gall falls within the framework of the impossibility result proved in Section 4.2, and indeed he does not approximate specific eigenvalues but rather the (pseudo) inverse of the whole linear system.

[2] This problem is harder since for normal matrices eigenvalues are almost in one to one correspondence with singular values.

[3] Le Gall's result is a bit more general and applies to the pseudo-inverse of the Laplacian of any weighted, undirected graph.

It is also interesting to compare the BPL–complete promise problem $SecondEV$ that we get with the RL–complete promise problem given in [24]. First, following [19, 10] they define $\lambda_\pi(G)$ to be the second largest eigenvalue of $A\tilde{A}$, where $\tilde{A}[i,j] = \frac{\pi(j)A[j,i]}{\pi(i)}$ and π is the stationary distribution of a random walk on G. Then, they define the promise problem POLY-MIXING S-T CONNECTIVITY to be:

Input : $(G, s, t, 1^k)$ where $G = (V, E)$ is an out-regular directed graph, $s, t \in V$, and $k \in \mathbb{N}$.

Yes instances : $\lambda_\pi(G) \leq 1 - \frac{1}{k}$, and $\pi(s), \pi(t) \geq \frac{1}{k}$.

No instances : There is no path from s to t in G.

Notice that the problem is made RL–complete by forcing in the No instances the condition that there is no path from s to t. The main difference between POLY-MIXING S-T CONNECTIVITY and $SecondEV$ is that in $SecondEV$ the spectral gap is defined with respect to the eigenvalues of the transition operator A (i.e., the roots of its characteristic polynomial) whereas in [24] the spectral gap is defined with respect to a *normalized* inner product space, and the normalization is with respect to the *unknown* stationary distribution π, which is, by itself, a major component of the solution.

1.3 Open problems and the bigger picture

The following chain of containments is well known:

$$\mathsf{NC}^1 \subseteq \mathsf{L} \subseteq \mathsf{RL} \subseteq \mathsf{NL} \subseteq \mathsf{DET} \subseteq \mathsf{NC}^2 \subseteq \mathsf{DSPACE}(O(\log^2 n)),$$

where DET is the class of languages that are NC^1 Turing-reducible to the problem $intdet$ of computing the determinant of an integer matrix (see [3] for a definition of DET). As it turns out, many important problems in linear algebra, such as inverting a matrix, or equivalently, solving a set of linear equations are in DET, and often complete for it (see, e.g., [3]). The fact that $\mathsf{NL} \subseteq \mathsf{DET}$ is due to [3] who showed that the directed connectivity problem, STCON is reducible to $intdet$. $\mathsf{DET} \subseteq \mathsf{NC}^2$ follows from Csansky's algorithm [4] for the parallel computation of the determinant. In addition to the above we also know that $\mathsf{BPL} \subseteq \mathsf{DET}$ (e.g., using the fact that matrix powering is DET complete). The work of Watrous [32] shows that BQL is also contained in NC^2.

It is well known that matrix powering is complete for DET. We also saw that *approximating* matrix powering of *stochastic* matrices is in BPL. Conversely, it is possible to convert a BPL machine to a stochastic operator A such that the probability the machine moves from s to t in k steps is $A^k[s,t]$.[4] Thus, approximating matrix-powering of stochastic operators is *complete* for BPL. The essence of this paper is to find a corresponding decision problem that is BPL–complete. Roughly speaking, the results of this paper, and those described in the previous subsection, show that:

- The promise problem of approximating the spectral gap of stochastic operators (with a real second eigenvalue) is BPL–complete, and,
- The promise problem of approximating the spectral gap of Hermitian operators is BQL–complete.

[4]This reduction is standard and appears in many papers, e.g., already in [20]. We also employ such a reduction, see Subsection 3.1 in this paper.

Finally, note that exactly computing the coefficients of the characteristic polynomial is complete for DET [3]. Having these coefficients, one can approximate the roots of the characteristic polynomial to within an arbitrary accuracy.

Thus, there are several natural open problems that are still wide open:

- Is it sill BPL–hard to approximate the spectral gap of stochastic and Hermitian operators (i.e., undirected graphs)? Alternatively, can one find a deterministic logspace algorithm solving the problem? We remark that while Reingold's algorithm [23] uses the spectral gap to solve undirected connectivity, his algorithm *does not* approximate the spectral gap of the given undirected graph.
- Is it possible to approximate in BPL the second eigenvalue of a general (not necessarily stochastic or non-negative) operator?
- More generally, is it possible to approximate the SVD of an arbitrary linear operator already in BPL? A positive answer would imply BPL approximations to many problems in linear algebra that are currently only known to be in NC^2. A negative answer would imply a separation between BQL and BPL [6]. In fact, the problem is also open for Hermitian operators, where singular values and eigenvalues coincide (up to their sign).

We believe that the message emerging from this discussion is that, somewhat surprisingly, the deterministic/probabilistic/quantum space-bounded classes can be characterized by linear-algebraic promise problems that approximate the exact computation that can be done in DET. The different classes differ in the type of linear operators they can handle. As such, their relationship to the class DET is similar to the relationship between BPP, that can approximate the permanent function [14], and the class #P that solves it exactly [30].

We believe this view is not only important by itself (by giving algorithms approximating natural problems in linear algebra), but may also shed new light on the strengths and weaknesses of the deterministic, probabilistic and quantum models of space-bounded computation.

2 Preliminaries

2.1 Space bounded probabilistic computation

A deterministic space-bounded Turing machine has three semi-infinite tapes: an *input tape* (that is read-only); a *work tape* (that is read-write) and an *output tape* (that is write-only and uni-directional). The space complexity of the machine is the number of cells on the work tape. The running time of a space-bounded Turing machine with $s(n) \geq \log n$ space complexity is bounded by $2^{O(s(n))}$ time. A *probabilistic* space-bounded Turing machine is similar to the deterministic machine (and in particular we require it always halts within $2^{O(s(n))}$ time) except that it can also toss random coins. One convenient way to formulate this is by adding a fourth semi-infinite tape, the *random-coins tape*, that is read-only, uni-directional and is initialized with perfectly uniform bits. We are only concerned with bounded-error computation: We say a language is accepted by a probabilistic Turing machine if for every input in the language the acceptance probability is at least $2/3$, and for every input not in the language it is at most $1/3$. As usual, the acceptance probability can be amplified as long as there is some non-negligible gap between the acceptance probability of yes and no instances.

Definition 2.1. *A language is in* $\mathsf{BPSPACE}(s(n))$ *if it is accepted by a probabilistic space bounded TM with space complexity* $s(n)$. $\mathsf{BPL} = \cup_c \mathsf{BPSPACE}(c \log n)$.

We say a language is accepted by a probabilistic Turing machine with one-sided error if for every input in the language the acceptance probability is at least $1/2$, and for every input not in the language we always reject.

Definition 2.2. *A language is in* $\mathsf{RSPACE}(s(n))$ *if it is accepted by a probabilistic space bounded TM with one-sided error and space complexity* $s(n)$. $\mathsf{RL} = \cup_c \mathsf{RSPACE}(c \log n)$.

2.2 Simulatable families of matrices

Often we are interested in approximating a *value* (e.g., an entry in a matrix with integer values or the whole matrix) with a probabilistic machine. More precisely, assume there exists some value $u = u(x) \in \mathbb{R}$ that is determined by the input $x \in \{0,1\}^n$. We say a probabilistic TM $M(x,y)$ (ε, δ)–approximates $u(x)$ if:

$$\forall_{x \in \{0,1\}^n} \qquad \Pr_y \left[|M(x,y) - u(x)| \geq \varepsilon \right] \leq \delta \tag{1}$$

A random walk on a graph G (or its transition matrix A) can be simulated by a probabilistic logspace machine. As a consequence, a probabilistic logspace machine can approximate powers of A well. Here we try to extend this notion to arbitrary linear operators A, not necessarily stochastic. We say a matrix A is *simulatable* if any power of it can be approximated by a probabilistic algorithm running in small space. Formally:

Definition 2.3. *We say that a family of matrices* $\mathcal{A}$ *is* simulatable *if there exists a probabilistic algorithm that on input* $A \in \mathcal{A}$ *of dimension* n *with* $\|A\| \leq poly(n)$, $k \in \mathbb{N}$, $s, t \in [n]$, *runs in space* $O(\log \frac{nk}{\varepsilon\delta})$ *and* (ε, δ)*–approximates* $A^k[s,t]$.

In the appendix we give for completeness a proof that:

Lemma 2.1. *The family of transition matrices of (directed or undirected) graphs is simulatable.*

We say $\|A\|_\infty \leq c$ if for every $i \in [n]$, $\sum_j |A[i,j]| \leq c$. We also show:

Lemma 2.2. *The family of real matrices with infinity norm at most* 1 *is simulatable.*

3 A new canonical problem for probabilistic space-bounded classes

Recall our definition for $SecondEV_{\alpha,\beta}$:

Definition 3.1. *The input is* $0 \leq \alpha < \beta < 1$ *and a stochastic matrix* A *of dimension* n *with possibly complex eigenvalues* $\lambda_1, \ldots, \lambda_n$. *We have the promise that* $\lambda_1 = 1$ *and* $\lambda_2 \in \mathbb{R}$. *Also,* $|\lambda_n| \leq \ldots \leq |\lambda_3| \leq 1 - \beta$.

No instances *:* $\lambda_2 \geq 1 - \alpha$.

Yes instances *:* $\lambda_2 \leq 1 - \beta$.

In what follows, we shall see that this problem captures the hardness of probabilistic computation in logarithmic space.

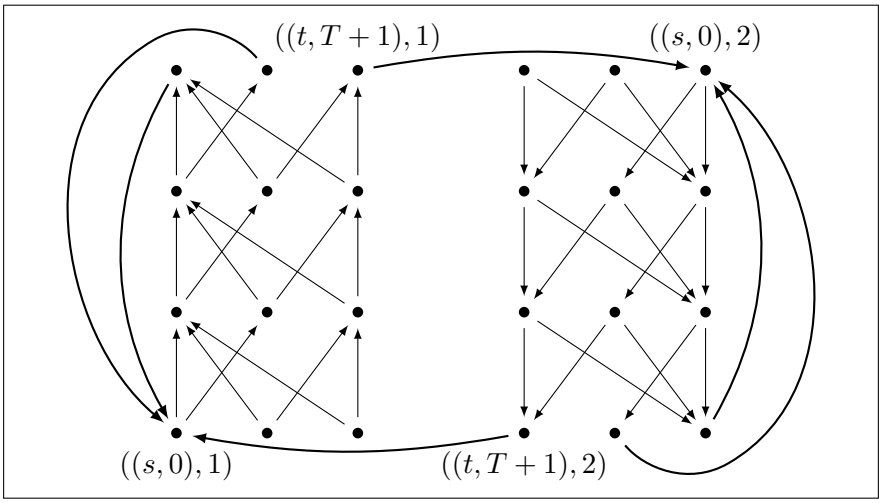

Figure 1: An example for G'' with $T = 3$.

3.1 Reducing Turing machines to graphs

Let M be a probabilistic space-bounded TM that accepts some language L with $c \log n_0$ space and $T = n_0^c$ time. Fix an input x. Let $G = (V, E)$ be the configuration graph of G on x, s the initial configuration and t the unique accepting configuration.

Define the *layered configuration graph* $G' = (V', E')$. $V' = V \times \{1, \ldots, T+1\}$, i.e., G' has $T+1$ layers, each layer containing $|V|$ vertices. There is an outgoing edge from vertex (i, ℓ) in layer $\ell \leq T$ to vertex $(j, \ell+1)$ in layer $\ell+1$ if and only if $(i, j) \in E$. Without loss of generality, any vertex (i, ℓ) in layer $\ell \leq T$ has at least one neighbor in layer $\ell+1$.

Next, we define a graph $G'' = G''(M, x)$ that contains two copies of G' with some additional edges and has two strongly connected components if and only if s is not connected to t in G. Formally, the vertex set of G'' is $V'' = \{(v', k) \mid v' \in V', k \in \{1, 2\}\}$. The edges E'' of G'' are as follows:

- For every $(a, b) \in E'$ and $k \in \{1, 2\}$, $((a, k), (b, k)) \in E''$.
- For every $r \in V$, $r \neq t$ and $k \in \{1, 2\}$, $(((r, T+1), k), ((s, 1), k)) \in E''$.
- $(((t, T+1), 1), ((s, 1), 2)) \in E''$ and $(((t, T+1), 2), ((s, 1), 1)) \in E''$.

See Figure 1.

Let A be the transition matrix of G'' and note that it is stochastic. We claim:

Lemma 3.1. *Let $p = p(M, x) = A^T[s, t]$ denote the probability that M accepts x. Let ω be a primitive $(T+1)$-th root of unity. The eigenvalues of A are $\left\{0, \omega^k, (1-2p)^{\frac{1}{T+1}}\omega^k \mid 0 \leq k \leq T\right\}$.*

Proof. Denote $W = A^\dagger$. It is known that the eigenvalues of A and W coincide. Let v be any eigenvector of W with eigenvalue $\lambda \neq 0$.

Claim 3.1. *For every $s \neq j \in V$, $v[((j, 1), 1)] = v[((j, 1), 2)] = 0$*

Proof. The in-degree of $((j, 1), 1), (j, 1), 2)$ is zero and $\lambda \neq 0$. □

Denote $v[((s,1),1)] = x$ and $v[((s,1),2)] = y$.

Claim 3.2. *For every $j \in V$ and $\ell \geq 0$, $W^{\ell-1}[j,s]$ is the probability M reaches j after taking $\ell - 1$ random steps from s. Then,*

$$\begin{aligned} v[((j,\ell),1)] &= \frac{x}{\lambda^{\ell-1}} W^{\ell-1}[j,s] \\ v[((j,\ell),2)] &= \frac{y}{\lambda^{\ell-1}} W^{\ell-1}[j,s]. \end{aligned}$$

Proof. The proof is by induction. The case of $\ell = 1$ was already covered. Assume the claim holds for ℓ and fix a vertex $((j,\ell+1),1)$. Then:

$$\begin{aligned} (Wv)[((j,\ell+1),1)] &= \sum_{k:(k,j)\in E} v[((k,\ell),1)] \cdot \frac{1}{d_{\text{out}}(k)} \\ &= \sum_{k:(k,j)\in E} \frac{x}{\lambda^{\ell-1}} W^{\ell-1}[k,s] \cdot \frac{1}{d_{\text{out}}(k)} \quad \text{(Hypotesis)} \\ &= \frac{x}{\lambda^{\ell-1}} W^{\ell}[j,s]. \end{aligned}$$

As $Wv = \lambda v$, the claim for $((j,\ell+1),1)$ follows. The other claim is similar. □

Finally, the equation $Wv = \lambda v$ for the vertices $((s,1),1)$ and $((s,1),2)$ give us two equations. The equation for the vertex $((s,1),1)$ is

$$(Wv)[((s,1),1)] = \sum_{j \neq s} v[((j,T+1),1)] + v[((t,T+1),2)].$$

As $\sum_{j\in V} v[((j,T+1),1)] = \frac{(1-p)x}{\lambda^T}$ and $v[((t,T+1),2)] = \frac{py}{\lambda^T}$, we get the equation

$$\lambda x = \frac{(1-p)x}{\lambda^T} + \frac{py}{\lambda^T}. \tag{2}$$

Similarly, the equation for the vertex $((s,1),2)$ is

$$\lambda y = \frac{(1-p)y}{\lambda^T} + \frac{px}{\lambda^T}. \tag{3}$$

Adding the two equations we get:

$$\lambda(x+y) = \frac{1}{\lambda^T}(x+y).$$

If $x \neq -y$ we must have $\lambda^{T+1} = 1$. If $x = -y$ Equation (2) and Equation (3) are the same and give $\lambda^{T+1}x = (1-2p)x$. Since $x \neq 0$ (otherwise the whole vector v is zero) we have $\lambda^{T+1} = 1 - 2p$. Hence, all the eigenvalues belong to the declared set.

Finally, we notice that for $x = -y$ and λ such that $\lambda^{T+1} = 1 - 2p$ we can build a corresponding eigenvector, and similarly for $\neq -y$ and λ s.t. $\lambda^{T+1} = 1$ we may take $x = y$ and the equations work. Hence, the eigenvalues are exactly the given set. □

3.2 An RL–hard problem

In this subsection we show that $SecondEV_{\alpha,\beta}$ with $\alpha = 0$ and $\beta = \frac{1}{n}$, that is – distinguishing between the case that the eigenvalue 1 has algebraic multiplicity 2 and the case that the second eigenvalue is $\frac{1}{n}$–far from 1, is RL–hard. We stress that we only prove RL-hardness and we do not know how to solve $SecondEV_{0,\frac{1}{n}}$ in RL. We prove:

Theorem 3.2. *$SecondEV_{0,\frac{1}{n}}$ is RL–hard.*

Proof. Let $L \in \mathsf{RL}$ and let M be the probabilistic space-bounded TM that accepts it. Given an input x of length n_0, let G, G', G'', ω and p be as above. If $x \in L$ then without loss of generality $\frac{2\pi^2}{T} \leq p < \frac{1}{2}$ and otherwise $p = 0$. G'' has $n = 2(T+1)|V| = \text{poly}(n_0)$ vertices. Note that $|V| \geq T$.

Let A be the transition matrix of G'' of dimension n and consider $B = \frac{1}{2}A + \frac{1}{2}I_{n\times n}$. By Lemma 3.1, the eigenvalues of B are $\left\{\frac{1}{2}, \frac{1}{2} + \omega^k, \frac{1}{2} + \frac{1}{2}(1-2p)^{\frac{1}{T+1}}\omega^k \mid 0 \leq k \leq T\right\}$. For $0 \leq k \leq T$, denote

$$\begin{aligned} \lambda_k &= \frac{1}{2} + \frac{1}{2}(1-2p)^{\frac{1}{T+1}}\omega^k \\ \mu_k &= \frac{1}{2} + \frac{1}{2}\omega^k. \end{aligned}$$

First, note that

$$|\mu_k|^2 = \frac{1}{2} + \frac{1}{2}\cos\left(\frac{2\pi k}{T+1}\right).$$

Similarly, $|\lambda_k|^2$ can be expressed as some constant independent of k plus $\frac{(1-2p)^{\frac{1}{T+1}}}{2}\cos\left(\frac{2\pi k}{T+1}\right)$. For $k > 0$, $\cos\left(\frac{2\pi k}{T+1}\right)$ is maximized at the lowest angle, i.e., at $k = 1$. Thus, for $k \neq 0$, $|\mu_k| \leq |\mu_1|$ and $|\lambda_k| \leq |\lambda_1|$. By inspection, $\mu_0 = 1$, $\lambda_0 = \frac{1}{2} + \frac{1}{2}(1-2p)^{\frac{1}{T+1}}$ and $|\lambda_1| \leq |\mu_1| \leq 1 - \frac{\pi^2}{2T^2}$. One can verify that:

- If $p \geq \frac{2\pi^2}{T}$ then $\lambda_0 \leq 1 - \frac{\pi^2}{2T^2} \leq 1 - \frac{1}{n}$, so all eigenvalues but μ_0 are at most $1 - \frac{1}{n}$ in magnitude.
- If $p = 0$ then $\lambda_0 = 1$, so all eigenvalues but μ_0 *and* λ_0 are at most $1 - \frac{1}{n}$ in magnitude.

B then satisfies the promise of $SecondEV_{0,\frac{1}{n}}$. It is well-known that A can be computed using logarithmic space and so can B. Hence, the reduction is a logspace reduction. □

3.3 A BPL–complete problem

While we do not know how to solve $SecondEV_{0,\frac{1}{n}}$ in RL, we *do* know how to solve it in BPL. In fact we can do it for even stricter parameters. Specifically, we show that $SecondEV_{\zeta,2\zeta}$, for every non-negligible ζ, is in BPL, and that $SecondEV_{\frac{1}{n},\frac{2}{n}}$ is BPL–complete. We begin with:

Lemma 3.3. *Let $0 < \zeta \leq \frac{1}{2}$. Then, $SecondEV_{\zeta,2\zeta}$ can be solved using $O(\log n + \log\frac{1}{\zeta})$ space.*

Proof. Let A be the input to the promise problem and set m such that $(1-\zeta)^m = \frac{1}{n^2}$. Note that $m \leq \frac{2\ln n}{\zeta}$. We then have:

Claim 3.3. *If A is a Yes instance then* $\text{Tr}\,(A^m) \leq 1 + \frac{1}{n^3}$.

Proof. $\text{Tr}\,(A^m) = \sum_{k=1}^{n} \lambda_k^m$.[5] Then:

$$|\text{Tr}\,(A^m)| \;=\; \left|\sum_{k=1}^{n} \lambda_k^m\right| \;\le\; 1 + \sum_{k=2}^{n} |\lambda_k|^m \;\le\; 1 + n\,(1-2\zeta)^m\,.$$

Using the fact that $(1-2\zeta)^m \le (1-\zeta)^{2m}$ and $(1-\zeta)^{2m} = \frac{1}{n^4}$, the claim thus follows. □

Claim 3.4. *If A is a No instance then* $\text{Tr}\,(A^m) \ge 1 + \frac{1}{n^2} - \frac{1}{n^3}$.

Proof. By the reversed triangle inequality:

$$|\text{Tr}\,(A^m)| \;\ge\; 1 + \lambda_2^m - \left|\sum_{k=3}^{n} \lambda_k^m\right| \;\ge\; 1 + (1-\zeta)^m - n\,(1-2\zeta)^m\,.$$

Again, by $(1-2\zeta)^m \le (1-\zeta)^{2m}$ we have that $(1-\zeta)^m - n(1-2\zeta)^m \ge (1-\zeta)^m - n(1-\zeta)^{2m}$. Now, since $(1-\zeta)^m = \frac{1}{n^2}$ we overall conclude that $|\text{Tr}\,(A^m)| \ge 1 + \frac{1}{n^2} - \frac{1}{n^3}$. □

Approximating $\text{Tr}\,(A^m)$ well with high probability can be done in logarithmic space. By Lemma 2.2 and simple composition of logspace reductions, we can $(\frac{1}{n^4}, \frac{1}{3})$–approximate $\text{Tr}\,(A^m)$ and decide the promise problem using $O(\log nm)$ space. □

The main result is the converse direction that the problem for $\zeta = \frac{1}{n}$ is BPL–complete.

Lemma 3.4. *Let $L \in$ BPL. Then, L is logspace reducible to $SecondEV_{\frac{1}{n},\frac{2}{n}}$.*

Proof. Let $L \in$ BPL and fix an input x of length n_0. Let $M, G, G', G'', \omega, p, n, A, B$ be as as in the proof for Theorem 3.2. One can verify that:

- If $p \ge \frac{2\pi^2}{T}$ then $\lambda_0 \le 1 - \frac{\pi^2}{2T^2} \le 1 - \frac{2}{n}$, so all eigenvalues but μ_0 are at most $1 - \frac{2}{n}$ in magnitude.
- If $p \le \frac{1}{4T}$ then $\lambda_0 \ge \frac{1}{2} + \frac{1}{2}\left(1 - \frac{1}{2T}\right)^{\frac{1}{T}} \ge 1 - \frac{1}{3T^2} \ge 1 - \frac{1}{n}$, so all eigenvalues but μ_0 *and* λ_0 are at most $1 - \frac{2}{n}$ in magnitude. The second inequality is derived from the fact that $\left(1 - \frac{x}{2}\right)^x \ge 1 - \frac{2x^2}{3}$ for small $x > 0$.

Without loss of generality, p can be taken to be either larger than $\frac{2\pi^2}{T}$ or smaller than $\frac{1}{4T}$ (by simple BPL amplification arguments), so B satisfies the promise of $SecondEV_{\frac{1}{n},\frac{1}{n}}$. □

4 Approximating eigenvalues with constant accuracy

In this section we prove:

Theorem 4.1. *There exists a probabilistic algorithm that gets as an input a stochastic matrix B with real eigenvalues in $[0,1]$, constants $\beta > \alpha > 0$ and $\lambda \in [0,1]$ such that:*

- *There are d eigenvalues λ_i satisfying $|\lambda - \lambda_i| \le \alpha$,*

[5] This equation is true even when A is not diagonalizable. To see that, write A in its Jordan Normal Form, $A = VJV^{-1}$ where the eigenvalues of A, $\lambda_1, \ldots, \lambda_n$, lie on the diagonal of J. $A^m = VJ^mV^{-1}$ so $\text{Tr}\,(A^m) = \text{Tr}\,(J^m)$ and since J is triangular, the diagonal of J^m is $\lambda_1^m, \ldots, \lambda_n^m$.

- *All other eigenvalues λ_i satisfy $|\lambda - \lambda_i| \geq \beta$,*

and outputs d with probability at least $2/3$. Furthermore the algorithm runs in probabilistic space $O(\log n)$.

We remark that Theorem 4.1 covers the case of transition matrices of undirected graphs, whose spectrum is contained in $[-1, 1]$.[6] Taking $B = \frac{1}{2}A + \frac{1}{2}I_{n\times n}$ we get a stochastic matrix with eigenvalues in the range $[0, 1]$, and whose eigenvectors are in a natural one-to-one correspondence with A's eigenvalues.

Proof. (Of Theorem 4.1) The input to the algorithm is $n, B, \lambda, \alpha, \beta$. We assume the existence of a univariate polynomial $p(x) = \sum_{i=0}^{M} c_i x^i$ with the following properties:

- p has a sharp peak around λ, i.e., $p(x) \geq 1 - \eta$ for $x \in [\lambda - \alpha, \lambda + \alpha]$ and $p(x) \leq \eta$ for $x \in [0, 1] \setminus (\lambda - \beta, \lambda + \beta)$, where $\eta = \eta(n) = n^{-2}$.
- p can be computed in L. Formally, $M = \deg(p)$ and $|c_i|$ are at most $\text{poly}(n)$ and for every i, c_i can be computed (exactly) by a deterministic Turing machine that uses $O(\log n)$ space.

In the next subsection we show how to obtain such a polynomial p with $M = 32(\beta - \alpha)^{-2} \log n$ and $|c_i| \leq 2^{cM}$ for some constant $c \geq 1$.

Choose $\varepsilon = \frac{1}{n}$ and $\delta = \frac{1}{3}$. Set $\varepsilon' = \varepsilon \cdot 2^{-2cM}$ and $\delta' = \delta \cdot 2^{-cM}$. The output of the algorithm is the integer closest to

$$R = \sum_{i=0}^{M} c_i \cdot \text{TP}(B, n, i, \varepsilon', \delta'),$$

where TP is the probabilistic algorithm guaranteed by Lemma 2.2 that (ε', δ')–approximates Tr (B^i).

It is easy to check that:

Claim 4.1. $\Pr[|R - \text{Tr}\,(p(B))| \geq \varepsilon] \leq \delta$.

As Tr $(p(B)) = \sum_{i=1}^{n} p(\lambda_i)$, $\Pr[|R - \sum_{i=1}^{n} p(\lambda_i)| \geq \varepsilon] \leq \delta$. However, $p(\lambda_i)$ is large when λ_i is α–close to λ and small when it is β–far from λ, and we are promised that *all* eigenvalues λ_i are either α–close or β–far from λ. Thus,

$$|\text{Tr}\,(p(B)) - d| \leq n\eta.$$

Altogether, except for probability δ, $|R - d| \leq \varepsilon + n\eta \leq \frac{1}{3}$, and the nearest integer closest to R is d. The correctness follows. It is also straightforward to check that the space complexity is $O(\log(n\varepsilon^{-1}\delta^{-1})) = O(\log n)$. □

The constant accuracy we achieve is far from being satisfying. The matrix B has n eigenvalues in the range $[0, 1]$, so the average distance between two neighboring eigenvalues is $1/n$. Thus, the assumption that there is an interval of length $\beta - \alpha$ with no eigenvalue is often not true. The desired accuracy we would like to get is $o(1/n)$. Having such accuracy would enable outputting an approximation of the whole spectrum of B, using methods similar to those in [28], thus getting a true classical analogue to the quantum algorithm in [28]. However, we do not know how to achieve subconstant accuracy. The question whether better accuracy is possible in BPL is one of the main questions raised by this work.

[6] If G is undirected and irregular, then the adjacency matrix $\tilde{A}$ is symmetric (because the graph is undirected) but the transition matrix $A = D^{-1}\tilde{A}$, where D is the diagonal degree matrix, is not symmetric. Yet, consider the matrix $L = D^{-1/2}\tilde{A}D^{-1/2}$. L is symmetric and thus has an eigenvector basis with real eigenvalues. $A = D^{-1/2}LD^{1/2}$ is conjugate to L and is thus diagonalizable and has the same eigenvalues. As A is stochastic its eigenvalues are in the range $[-1, 1]$.

4.1 Using the symmetric threshold functions

There are several natural candidates for the function p above. In this subsection we use the threshold function to obtain such a function p. For $\lambda = \frac{k}{M}$ for some integers k and M, define:

$$p_\lambda(x) = \sum_{i=k}^{M} \binom{M}{i} x^i (1-x)^{M-i}.$$

p_λ approximates well the threshold function $\mathbf{Th}_\lambda(x) : [0,1] \to \{0,1\}$ that is one for $x \geq \lambda$ and zero otherwise. Specifically, using the Chernoff bound, we obtain:

Lemma 4.2. *Let $x \in [0,1]$. $p_\lambda(x)$ approximates $\mathbf{Th}_\lambda(x)$ over $[0,1]$ with accuracy $(\xi(\varepsilon))^{Mx}$, where $\varepsilon = \frac{\lambda - x}{x}$ and $\xi(\varepsilon) = \frac{e^\varepsilon}{(1+\varepsilon)^{1+\varepsilon}}$.*

As a polynomial in x, $p_\lambda(x) = \sum_{i=0}^{M} c_i x^i$ with $c_i = (-1)^i \sum_{j=\lambda M}^{i} \binom{M}{j}\binom{M-j}{i-j}(-1)^j$ and therefore $|c_i| \leq \sum_{j=\lambda M}^{i} \binom{M}{j}\binom{M-j}{i-j} \leq M\binom{M}{M/2}^2 = 2^{O(M)}$. Furthermore, c_i can be computed (exactly) by a deterministic Turing machine that uses $O(M)$ space by simply running through the loop over j, each time updating the current result by $(-1)^j\binom{M}{j}\binom{M-j}{i-j}$.

To obtain our polynomial p, define p as the difference between the threshold polynomial around $\lambda + \Delta$ and the threshold polynomial around $\lambda - \Delta$,

$$p(x) = p_{\lambda-\Delta}(x) - p_{\lambda+\Delta}(x)$$

where $M = 32(\beta - \alpha)^{-2} \log n$ and $\Delta = (\alpha + \beta)/2$. It indeed holds that:

Lemma 4.3. *$p(x) \geq 1 - n^{-2}$ for every x that is α–close to λ (i.e., $|x - \lambda| < \alpha$) and $p(x) \leq n^{-2}$ for every x that is β–far from λ (i.e., $|x - \lambda| \geq \beta$).*

Proof. We show that $p_{\lambda+\Delta}$ approximates $\mathbf{Th}_{\lambda+\Delta}$ for $x \geq \lambda + \beta$ and $x \leq \lambda + \alpha$, and $p_{\lambda-\Delta}$ approximates $\mathbf{Th}_{\lambda-\Delta}$ for $x \geq \lambda - \alpha$ and $x \leq \lambda - \beta$. Specifically, the following bounds hold:

	$p_{\lambda-\Delta}(x)$	$p_{\lambda+\Delta}(x)$	$p(x)$
$0 \leq x \leq \lambda - \beta$	$\leq n^{-2}/2$	≥ 0	$\leq n^{-2}$
$\lambda - \beta < x < \lambda - \alpha$	**?**	$\leq n^{-2}/2$	**?**
$\lambda - \alpha \leq x \leq \lambda + \alpha$	$\geq 1 - n^{-2}/2$	$\leq n^{-2}/2$	$\geq 1 - n^{-2}$
$\lambda + \alpha < x < \lambda + \beta$	$\geq 1 - n^{-2}/2$	**?**	**?**
$\lambda + \beta \leq x \leq 1$	≤ 1	$\geq 1 - n^{-2}/2$	$\leq n^{-2}$

Let us show that the second column is correct. For $|x - (\lambda + \Delta)| \geq (\beta - \alpha)/2$ and $\varepsilon = \frac{\lambda + \Delta - x}{x}$, the approximation error is $(\xi(\varepsilon))^{Mx}$, and if $\varepsilon > 0$,

$$\begin{aligned}(\xi(\varepsilon))^{Mx} &\leq e^{-\frac{\varepsilon^2}{\varepsilon+2}Mx} \leq e^{-\frac{(x-(\lambda+\Delta))^2}{\lambda+\Delta+x}M} \\ &\leq e^{-(x-(\lambda+\Delta))^2 M/2} \leq n^{-2}/2,\end{aligned}$$

and if $-1 < \varepsilon \leq 0$,

$$(\xi(\varepsilon))^{Mx} \leq e^{-\varepsilon^2 Mx/2} \leq e^{-(x-(\lambda+\Delta))^2 M/2} \leq n^{-2}/2.$$

Similar calculations also holds for the other entries. Hence, $p(x) = p_{\lambda-\Delta}(x) - p_{\lambda+\Delta}(x)$ satisfies the required conditions. □

4.2 The limitation of the technique

In this subsection, we prove the accuracy of the above technique cannot be enhanced merely by choosing a different polynomial p. Approximating threshold functions by a polynomial is well-studied and well understood (see, for example, [26, 8, 5] and references therein). However, we need to adapt this work to our needs because we have an additional requirement that the magnitude of the polynomial's coefficients is small.

We start by formalizing the properties of p that were useful to us. We say that $\mathcal{P} = \{p_{\lambda,n}\}_{\lambda\in[0,1],n\in\mathbb{N}}$ is a family of polynomials if for every $\lambda \in [0,1]$ and $n \in \mathbb{N}$, $p_{\lambda,n}$ is a univariate polynomial with coefficients in $\mathbb{R}$.

Definition 4.1 (Small family)**.** *Let $\mathcal{P}$ be a family of polynomials and fix $\lambda \in [0,1]$. For every $n \in \mathbb{N}$, write $p_{\lambda,n}(x) = \sum_{i=0}^{\deg(p_{\lambda,n})} c_{\lambda,n,i}x^i$. We say the family is $s(n)$–small if,*

- $\deg(p_{\lambda,n}) \leq 2^{s(n)}$,
- *For every $0 \leq i \leq \deg(p_{\lambda,n})$, $|c_{\lambda,n,i}| \leq 2^{s(n)}$, and*
- *There exists a deterministic Turing machine running in space $s(n)$ that outputs $c_{\lambda,n,0}, \ldots, c_{\lambda,n,\deg(p_{\lambda,n})}$.*

Definition 4.2 (Distinguisher family)**.** *Let $\mathcal{P}$ be a family of polynomials and fix $n \in \mathbb{N}$. Given $\alpha < \beta$ in $(0,1)$ and $\eta < 1/2$, we say the family is (α,β,η)–distinguisher for $\lambda \in [0,1]$ if,*

- *For every $x \in [0,1]$ that is α–close to λ, $p_{\lambda,n}(x) \in [1-\eta, 1]$, and*
- *For every $x \in [0,1]$ that is β–far from λ, $p_{\lambda,n}(x) \in [0,\eta]$.*

Theorem 4.4. *Let $\alpha,\beta,\lambda,\eta$ be such that $\alpha \leq \beta$, $\beta = o(1)$, $\eta = o(n^{-1})$ and $\lambda + \beta \leq \frac{1}{2}$. Then there is no (α,β,η)–distinguisher family for λ that is $O(\log n)$–small.*

Proof. Assume there exists such a family $\{p_{\lambda,n}\}_{\lambda\in[0,1],n\in\mathbb{N}}$ with $s(n) = c'\log n$. We first show that we can assume p has logarithmic degree. Let $r_{\lambda,n}(x)$ be the residual error of truncating $p_{\lambda,n}(x)$ after $c\log n$ terms, for c that will soon be determined. Also, without loss of generality, assume $x \in [0,1)$ is bounded away from 1. Then:

$$r_{\lambda,n}(x) \leq \sum_{i=c\log n+1}^{\deg(p_{\lambda,n})} |c_{\lambda,n,i}| \cdot x^i \leq n^{c'} \cdot \frac{x^{c\log n}}{1-x} \leq \frac{1}{1-x} n^{c'-c\log(1/x)}.$$

So, by taking $c = \lceil \frac{c'+2-\log(1-x)}{\log(1/x)} \rceil$ we obtain $r_{\lambda,n}(x) \leq n^{-2}$.

We now show that $O(\log n)$–degree polynomials cannot decay around λ fast enough. Assume to the contrary that there exists such a distinguisher family, so $|p_{\lambda,n}(x)| < n^{-1}$ for $x \in [\lambda+\beta, 1]$. The following lemma states that if a function has a small value on an interval, then it cannot be too large outside it. Namely,

Lemma 4.5. *[29, Theorem 2.9.11] Let $T_d(x)$ be the Chebyshev polynomial (of the first kind) of degree d. Then, if the polynomial $P_d(x) = \sum_{i=0}^{d} c_i x^i$ satisfies the inequality $|P_d(x)| \leq L$ on the segment $[a,b]$ then at any point outside the segment we have*

$$|P_d(x)| \leq L \cdot \left| T_d\left(\frac{2x-a-b}{b-a}\right)\right|.$$

For properties of the Chebyshev polynomials see [25, Chapter 1.1]. We mention a few properties that we use. An explicit representation of $T_d(x)$ is given by $T_d(x) = \frac{\left(x-\sqrt{x^2-1}\right)^d+\left(x+\sqrt{x^2-1}\right)^d}{2}$. $|T_d(-x)| = |T_d(x)|$ and T_d is monotonically increasing for $x > 1$. Also,

$$|T_d(1+\delta)| \leq \left(1+\delta+\sqrt{(1+\delta)^2-1}\right)^d \leq \left(1+4\sqrt{\delta}\right)^d \leq e^{4d\sqrt{\delta}} \leq 2^{8d\sqrt{\delta}} \tag{4}$$

for $0 \leq \delta \leq 1$. Then:

$$\begin{array}{rcll} |p_{\lambda,n}(\lambda)| & \leq & n^{-1} \cdot \left|T_{c\cdot\log n}\left(\frac{\lambda-\beta-1}{-\lambda-\beta+1}\right)\right| & \text{By Lemma 4.5, considering the interval } [\lambda+\beta, 1]. \\ & = & n^{-1} \cdot \left|T_{c\cdot\log n}\left(1+\frac{2\beta}{1-\lambda-\beta}\right)\right| & \text{By } |T_d(x)| = |T_d(-x)|. \\ & \leq & n^{-1} \cdot |T_{c\cdot\log n}(1+4\beta)| & \text{By the monotonicity of } T_d(x) \text{ for } x > 1 \text{ and } \lambda+\beta \leq \frac{1}{2}. \end{array}$$

By Equation (4), $|p_{\lambda,n}(\lambda)| \leq n^{-1}2^{32c\sqrt{\beta}\log n} \leq n^{-1+32c\sqrt{\beta}}$. As $\beta = o(1)$ for n large enough we have $|p_{\lambda,n}(\lambda)| \leq n^{-1/2}$, contradicting the fact that $|p_{\lambda,n}(\lambda)| \geq 1 - n^{-1}$. □

We note that for values very close to 1, polynomials of higher degrees are useful, and indeed better approximations are possible. In particular, one can separate a 1 eigenvalue from $1 - \frac{1}{n}$ by using the polynomial x^{n^2}.

5 Acknowledgements

The first and third authors were supported by the Israel science Foundation grant no. 994/14 and by the United States – Israel Binational Science Foundation grant no. 2010120. The third author was also supported by the Blavatnik Fund.

A preliminary version of this paper appeared in [7] but without the BPL–completeness and RL–hardness of approximating the second eigenvalue of a stochastic operator.

We thank François Le Gall for interesting discussions.

References

[1] Romas Aleliunas, Richard M. Karp, R.J. Lipton, Laszlo Lovasz, and C. Rackoff. Random walks, universal traversal sequences, and the complexity of maze problems. In *Foundations of Computer Science, 1979., 20th Annual Symposium on*, pages 218–223, Oct 1979.

[2] Carme Alvarez and Raymond Greenlaw. A compendium of problems complete for symmetric logarithmic space. *Computational Complexity*, 9(2):123–145, 2000.

[3] Stephen A. Cook. A taxonomy of problems with fast parallel algorithms. *Information and Control*, 64(1–3), 1985. International Conference on Foundations of Computation Theory.

[4] L. Csansky. Fast parallel matrix inversion algorithms. *SIAM Journal of Computing*, 5(6):618–623, 1976.

[5] Ilias Diakonikolas, Parikshit Gopalan, Ragesh Jaiswal, Rocco A Servedio, and Emanuele Viola. Bounded independence fools halfspaces. *SIAM Journal on Computing*, 39(8):3441–3462, 2010.

[6] Dean Doron and Amnon Ta-Shma. On the de-randomization of space-bounded approximate counting problems. *Information Processing Letters*, 2015.

[7] Dean Doron and Amnon Ta-Shma. On the problem of approximating the eigenvalues of graphs in probabilistic logspace. In *Automata, Languages, and Programming*. Springer, 2015.

[8] Alexandre Eremenko and Peter Yuditskii. Uniform approximation of sgn(x) by polynomials and entire functions. *Journal d'Analyse Mathématique*, 101(1):313–324, 2007.

[9] Bill Fefferman and Cedric Yen-Yu Lin. A complete characterization of unitary quantum space. *arXiv preprint arXiv:1604.01384*, 2016.

[10] James Allen Fill. Eigenvalue bounds on convergence to stationarity for nonreversible markov chains, with an application to the exclusion process. *The annals of applied probability*, pages 62–87, 1991.

[11] Aram W. Harrow, Avinatan Hassidim, and Seth Lloyd. Quantum algorithm for linear systems of equations. *Phys. Rev. Lett.*, 103:150502, Oct 2009.

[12] K. Hoffman. *Banach Spaces of Analytic Functions*. Dover Books on Mathematics Series. Dover Publications, Incorporated, 2007.

[13] Russell Impagliazzo and Avi Wigderson. P = BPP if E requires exponential circuits: Derandomizing the XOR lemma. In *Proceedings of the Twenty-ninth Annual ACM Symposium on Theory of Computing*, STOC '97, pages 220–229, New York, NY, USA, 1997. ACM.

[14] Mark Jerrum, Alistair Sinclair, and Eric Vigoda. A polynomial-time approximation algorithm for the permanent of a matrix with nonnegative entries. *J. ACM*, 51(4):671–697, Jul 2004.

[15] Adam R Klivans and Dieter Van Melkebeek. Graph nonisomorphism has subexponential size proofs unless the polynomial-time hierarchy collapses. *SIAM Journal on Computing*, 31(5):1501–1526, 2002.

[16] Swastik Kopparty, Shubhangi Saraf, and Amir Shpilka. Equivalence of polynomial identity testing and deterministic multivariate polynomial factorization. In *2014 IEEE 29th Conference on Computational Complexity (CCC)*, pages 169–180. IEEE, 2014.

[17] François Le Gall. Solving Laplacian Systems in Logarithmic Space. *ArXiv e-prints*, August 2016.

[18] Harry R. Lewis and Christos H. Papadimitriou. Symmetric space-bounded computation. *Theoretical Computer Science*, 19(2):161–187, 1982.

[19] Milena Mihail. Conductance and convergence of markov chains-a combinatorial treatment of expanders. In *2013 IEEE 54th Annual Symposium on Foundations of Computer Science*, pages 526–531. IEEE Comput. Soc. Press, 1989.

[20] Noam Nisan. Pseudorandom generators for space-bounded computation. *Combinatorica*, 12(4):449–461, 1992.

[21] Noam Nisan. RL$\subseteq$SC. *Computational Complexity*, 4(1):1–11, 1994.

[22] Noam Nisan and Avi Wigderson. Hardness vs randomness. *Journal of computer and System Sciences*, 49(2):149–167, 1994.

[23] Omer Reingold. Undirected connectivity in log-space. *J. ACM*, 55(4), 2008.

[24] Omer Reingold, Luca Trevisan, and Salil Vadhan. Pseudorandom walks on regular digraphs and the RL vs. L problem. In *Proceedings of the thirty-eighth annual ACM symposium on Theory of computing*, pages 457–466. ACM, 2006.

[25] Theodore J. Rivlin. *The Chebyshev polynomials*. Pure and applied mathematics. Wiley, 1974.

[26] E. B. Saff and V. Totik. Polynomial approximation of piecewise analytic functions. *Journal of the London Mathematical Society*, s2-39(3):487–498, 1989.

[27] Michael E. Saks and Shiyu Zhou. $\mathrm{BP_HSPACE(S)} \subseteq \mathrm{DSPACE(S^{3/2})}$. *J. Comput. Syst. Sci.*, 58(2):376–403, 1999.

[28] Amnon Ta-Shma. Inverting well conditioned matrices in quantum logspace. In *Proceedings of the 45th annual ACM symposium on Symposium on theory of computing*, STOC '13, pages 881–890, New York, NY, USA, 2013. ACM.

[29] A. F. Timan. *Theory of Approximation of Functions of a Real Variable*. Dover books on advanced mathematics. Pergamon Press, 1963.

[30] Leslie G Valiant. The complexity of computing the permanent. *Theoretical computer science*, 8(2):189–201, 1979.

[31] Dieter van Melkebeek and Thomas Watson. Time-space efficient simulations of quantum computations. *Electronic Colloquium on Computational Complexity (ECCC)*, 17:147, 2010.

[32] John Watrous. Space-bounded quantum complexity. *Journal of Computer and System Sciences*, 59(2):281–326, 1999.

A Simulatable matrices

Lemma A.1. *The family of transition matrices of (directed or undirected) graphs is simulatable.*

Proof. (of Lemma 2.1) Let $G = (V, E)$ be a graph with n vertices and let A be its transition matrix. Let $k \in \mathbb{N}$, $s, t \in [n]$ and $\delta, \varepsilon > 0$. Consider the algorithm that on input k, s, t, takes T independent random walks of length k over G starting at vertex s. The algorithm outputs the ratio of walks that reach vertex t. Let Y_i be the random value that is 1 if the i-th trial reached t and 0 otherwise. Then, for every i, $\mathbb{E}[Y_i] = A^k[s,t]$. Also, $Y_1, \ldots, Y_T$ are independent. By Chernoff,

$$\Pr\left[\left|\frac{1}{T}\sum_{i=1}^{T} Y_i - A^k[s,t]\right| \geq \varepsilon\right] \quad \leq \quad 2e^{-2\varepsilon^2 T}$$

Taking $T = \mathrm{poly}(\varepsilon^{-1}, \log \delta^{-1})$, the error probability (i.e., getting an estimate that is ε far from the correct value) is at most δ. Altogether, the algorithm runs in space $O(\log(Tnk|E|)) = O(\log(nk\varepsilon^{-1}) + \log\log\delta^{-1})$, assuming $|E| = \mathrm{poly}(n, k)$. □

We say $\|A\|_\infty \leq c$ if for every $i \in [n]$, $\sum_j |A[i,j]| \leq c$. We show:

Lemma A.2. *The family of real matrices with infinity norm at most* 1 *is simulatable.*

Proof. (of Lemma 2.2) We prove the result to real matrices, with positive or negative entries, as long as they have bounded infinity norm. By generalizing the sign of an entry to its *phase*, the result easily applies to complex matrices as well.

Let A be a real matrix of dimension n such that $\|A\|_\infty \leq 1$. Let $d_i(A) = \sum_j |A[i,j]|$.Let $k \in \mathbb{N}$, $s,t \in [n]$ and $\delta, \varepsilon > 0$. Note that:

$$\begin{aligned} A^k[s,t] &= \sum_{i_1=1}^{n}\sum_{i_2=1}^{n}\cdots\sum_{i_{k-1}=1}^{n} A[s,i_1]\cdot A[i_1,i_2]\cdot\ldots\cdot A[i_{k-1},t] \\ &= \sum_{i_1=1}^{n}\sum_{i_2=1}^{n}\cdots\sum_{i_{k-1}=1}^{n} \frac{|A[s,i_1]|}{d_s(A)}\cdot\frac{|A[i_1,i_2]|}{d_{i_1}(A)}\cdot\ldots\cdot\frac{|A[i_{k-1},t]|}{d_{i_{k-1}}(A)}\cdot p\left(A,\langle s,i_1,i_2,\ldots,i_{k-1},t\rangle\right), \end{aligned}$$

where

$$p\left(A,\langle s,i_1,i_2,\ldots,i_{k-1},t\rangle\right) = \frac{d_s(A)\cdot d_{i_1}(A)\cdot\ldots\cdot d_{i_{k-1}}(A)}{\operatorname{sgn}\left(A[s,i_1]\cdot A[i_1,i_2]\cdot\ldots\cdot A[i_{k-1},i_t]\right)}.$$

Consider the algorithm that on input k,s,t, takes T independent random walks of length k over G starting from vertex s. Iterating over all random walks, the algorithm approximates $\frac{1}{T}\sum_i y(i)$, where $y(i) = p(A,i)$ if the walk i reached t, and 0 otherwise. Correspondingly, let Y_i be the random value that is $p(A,i)$ if the i-th walk reached t and 0 if it did not. Then,

$$\mathbb{E}[Y_i] = \sum_{i_1=1}^{n}\sum_{i_2=1}^{n}\cdots\sum_{i_{k-1}=1}^{n} A[s,i_1]\cdot A[i_1,i_2]\cdot\ldots\cdot A[i_{k-1},t]\cdot p(A,\langle s,i_1,\ldots,i_{k-1},t\rangle) = A^k[s,t].$$

Denote the algorithm's outcome by $M(k,s,t)$. As in Lemma 2.1, and using the fact that $|p(A,i)| \leq 1$, the algorithm can (ε,δ)–approximates $\mathbb{E}[Y_i]$ by choosing T which is poly$(\varepsilon^{-1}, \log\delta^{-1})$. Following the same analysis as of Lemma 2.1, the algorithm runs in $O(\log nk\varepsilon^{-1} + \log\log\delta^{-1})$ space. We conclude that A is simulatable. □