TEL AVIV UNIVERSITY אוניברסיטת תל-אביב

Raymond and Beverly Sackler Faculty of Exact Sciences
The Blavatnik School of Computer Science

On the problem of approximating eigenvalues of undirected graphs in probabilistic logspace

Dean Doron

TEL AVIV UNIVERSITY אוניברסיטת תל-אביב

Raymond and Beverly Sackler Faculty of Exact Sciences
The Blavatnik School of Computer Science

On the problem of approximating eigenvalues of undirected graphs in probabilistic logspace

Research Thesis

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Computer Science

by
Dean Doron

This work has been conducted
under the supervision of
Prof. Amnon Ta-Shma

July 2014

Acknowledgements

First and foremost, I would like to express my sincere gratitude to my supervisor, Prof. Amnon Ta-Shma, whom with his patience, enthusiasm and immense insight made this work possible. I attribute my emerging ability to think clearly and search for underlying ideas to his encouragement and effort.

I would like to offer my special thanks to my family for their continuous support. Also, a deep appreciation goes to a small yet remarkable group of friends, that stood by me and still do.

The generous financial help of Tel-Aviv University is gratefully acknowledged.

Contents

Abstract

Many works on probabilistic space bounded algorithms try to show that probabilistic logspace computation is weak, i.e., can be derandomized and be done in deterministic small space. In this work we take a step aside, and go in another direction. Instead of asking *"How can we derandomize probabilistic space-bounded algorithms?"* we ask *"What good can we make out of the probabilistic power we have?"*.

We focus on the problem of approximating eigenvalues of stochastic Hermitian operators. Solving the problem with polynomially-small accuracy, would essentially imply that one can approximately compute linear algebra in full for such operators already in BPL. In this paper we achieve approximations only for constant accuracy. Our technique is new (at least as far as we know). We also show that going beyond constant accuracy requires a new idea.

We do not have a conjecture as to whether such a better approximation is possible in BPL or not, and we believe solving this problem is important, and may shed new light on the strengths and weaknesses of the probabilistic space bounded model. We hope the paper would stimulate this line of research.

Chapter 1

Introduction

One of the most basic questions in complexity theory is whether randomness buys extra computational power or not. Two famous instantiations of the problem are the time and space versions of it, namely the $\mathsf{BPP} \stackrel{?}{=} \mathsf{P}$ and $\mathsf{BPL} \stackrel{?}{=} \mathsf{L}$ problems.

The time bounded model. Impagliazzo and Wigderson [IW97] showed that if there exists a language in E that requires sub-exponential size circuits then $\mathsf{P} = \mathsf{BPP}$. Thus, if non-uniformity does not help solving natural uniform languages, then randomness does not help either. Many people see that as conclusive evidence that randomness does not help. A weak converse (that derandomization implies lower bounds on non-uniform computation) was shown by Kabanets and Impagliazzo [KI04].

Yet, there are problems for which we know an efficient probabilistic algorithm, but not a deterministic one. Polynomial identity testing (PIT) is a prominent example (perhaps, in a sense, a canonical one – see [KI04]). Another example is approximating the permanent. Exactly computing the permanent of a matrix with $0, 1$ entries is $\#\mathsf{P}$ complete. Jerrum, Sinclair and Vigoda [JSV04] showed a randomized FPRAS (fully polynomial randomized approximation scheme) approximating the permanent of such matrices. If $\mathsf{PromiseBPP} = \mathsf{P}$ there exists a *deterministic* polynomial time approximation scheme (FPTAS) to the problem [DTS14], but no such algorithm is known to date.

The space-bounded model. Nisan [Nis92] constructed a pseudo-random generator (PRG) against non-uniform algorithms with logarithmic space that uses seed of 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. Saks and Zhou [SZ99] showed BPL is contained in $\mathsf{DSPACE}(\log^{1.5} n)$. Reingold [Rei08] showed undirected st-connectivity (which was shown to be in RL by [AKL$^+$79]) already belongs to L.

Thus, in the space bounded model, there are *unconditional* derandomization results. Yet, we currently do not know a PRG with seed length $o(\log^2 n)$, nor

a general derandomization result that simultaneously uses $o(\log^2 n)$ space and polynomial time.

There is a lot of work trying to extend the PRG and the derandomization results in the space-bounded model, often by supplying a better result for specific, more restricted models ([BRRY14, BPW11, De11, IMZ12, RSV13, SVW14] to cite a few). In this work we take a step aside, and go in another direction. Instead of asking *"How can we derandomize probabilistic space-bounded algorithms"* we ask *"What good can we make out of the probabilistic power we have?"*.

Thus, instead of looking "down" below RL we look "up". We know that

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

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 [Coo85] for a definition of DET (and also of NC^k). As it turns out, many important problems in linear algebra (like inverting a matrix, or equivalently, solving a set of linear equations) is in DET, and often complete for it, see again [Coo85]. The fact that $\mathsf{NL} \subseteq \mathsf{DET}$ is due to [Coo85] who showed that the directed connectivity problem, STCON is reducible to *intdet*. $\mathsf{DET} \subseteq \mathsf{NC}^2$ follows from Csansky's Algorithm [Csa76] for the parallel computation of the determinant. $\mathsf{BPL} \subseteq \mathsf{DET}$ is a corollary of the fact that matrix powering is DET complete.

Recently, it was shown [TS13] how to approximate the singular value decomposition (SVD) of a given linear operator in BQL, which is the quantum space bounded class of computation using only $O(\log n)$ qubits.[1] With that, one can *approximately* invert a matrix in BQL and approximate all the singular values of a linear operator and all the eigenvalues of a Hermitian operator in BQL. In a sense, what we see is an analogue of what is known regarding the permanent. Computing the permanent exactly is $\#\mathsf{P}$ complete but approximating it is in BPP. Similarly here computing the SVD exactly is DET complete while approximating it is in BQL.

The question we tackle is whether one can do the same already in the classical world, namely, we ask whether one can approximate the spectrum in BPL. Thus, unlike previous works we are not trying to show that probabilistic logspace computation is weak, but rather we would like to use its strength to show we can solve problems we currently cannot solve deterministically.

We remark that recently there was another "de-quantumization" attempt on the algorithm of [HHL09, TS13] by Ben-Or and Eldar [BOE13] who developed new classical matrix inversion algorithms based on the above quantum algorithm. However, Ben-Or and Eldar are concerned with the time complexity of the algorithm and their algorithm

[1] It is not difficult to see that $\mathsf{BQL} \subseteq \mathsf{DET}$ [DTS14].

requires polynomial space.

1.1 Approximating eigenvalues of a matrix

One thing we can clearly do in BPL is simulate stochastic processes over n states. A random walk on a graph is such a process. In fact one may convert a BPL computation to a stochastic operator A, such that the probability the machine moves from s to t in k steps is $A^k[s,t]$, see, e.g., [Nis92]. We know we can output an approximation of its spectrum in BQL. We ask: suppose we are given a *stochastic* operator A. Can we approximate its spectrum in BPL?

We do not know how to answer this question, and we further relax it in several ways. First, we work with stochastic processes corresponding to walks over undirected graphs. For such graphs Reingold showed that the connectivity problem can be solved in L [Rei08]. A simple observation shows we can actually work with Hermitian operators (see the first paragraph in Chapter 4). In such a case eigenvalues and singular values essentially coincide (i.e., they are the same except for, possibly, their sign). Second, instead of asking to approximate the whole spectrum, we further relax the problem and ask about a single eigenvalue. I.e., we would like to solve the following promise problem:

Definition 1.1. ($EV_{\alpha,\beta}(A)$) We are given as input a stochastic, Hermitian matrix A of norm 1, $\lambda \in [-1,1]$ and $\alpha < \beta$. The promise problem has the following Yes and No instances:

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.

The BQL algorithm solves the above problem for any Hermitian operator A whose eigenvalues are well separated, say $\tau = n^{-c}$, $\alpha = \frac{\tau}{4}$ and $\beta = 2\alpha$. With such polynomial accuracy one can turn the solution of the promise problem to a procedure approximating the whole spectrum.

We, however, do not achieve polynomially small accuracies. Instead, we prove that for any two *constants* $\alpha < \beta$ the above promise problem belongs to BPL. Before going on we encourage the reader to think about the promise problem with constant parameters $\alpha < \beta$. In what follows we explain our technique and its limitations and give a short description of an attempt to bypass this limitation.

1.2 Proof idea

The starting point of this work is the observation that for many operators A we can approximate high powers of A in small space, i.e., we can approximate $A^k[s,t]$ in BPL for any k polynomial in n with polynomially good accuracy. We call such an operator

simulatable (see Definition 3.1). As explained before, BPL naturally corresponds to random walks on stochastic operators, and it is not hard to see that this implies that if A is the transition matrix of a directed graph then A is simulatable (see Lemma 3.2). We extend this to any stochastic matrix (see Section 3.1). We also extend this to non-stochastic operators, even with negative or complex entries, as long as the matrix has bounded infinity norm, namely, those matrices A for which all rows $i \in [n]$ have bounded ℓ_1 norm, $\sum_j |A[i,j]| \leq c$ for some constant c, see Section 3.2. We also apply this method in order to approximate matrix functions. Specifically, the matrix exponential, sine and cosine, which appear in the quantum logspace algorithm. However, here we can only handle limited parameters, see Section 3.3.

Our main result, solving $EV_{\alpha,\beta}(A)$ for constants $\alpha < \beta$, is obtained by considering the action of certain polynomials on a matrix. If A is simulatable and p is a univariate polynomial with not too large coefficients (i.e., all coefficients have magnitude polynomial in n) then we can approximate the entries of the *matrix* $p(A)$ (that is obtained by evaluating p over the ring of matrices) by using the BPL matrix power approximations. An important observation is that if A is Hermitian (or normal) we can view the action of p on A as the action of p on the eigenvalues of A, see the discussion after Theorem 4.1.

In this work we choose p to be a polynomial that has a peak around λ and decays fast otherwise. Applying p on A amplifies the eigenvalues α–close to λ to a value close to 1, and damps eigenvalues β–far from λ close to 0. Thus, Tr $(p(A))$ approximately counts the number of eigenvalues close to λ. The quantum algorithm, although stated in a completely different way, can also be interpreted that way. The function p we work with is essentially the majority (or a general threshold function). We work with logarithmic-degree polynomials, as the majority function for higher degrees has super-polynomial large coefficients. With such parameters we can obtain arbitrarily small constant accuracy. See Chapter 4.

There are many other candidate functions for such a polynomial p. However, we prove in Theorem 5.1 that no polynomial can do significantly better than the above. I.e., if we insist on not too large coefficients, no polynomial p can achieve better than constant accuracy.

We then consider a multi-stage approach where we first take A and approximate some $p_1(A)$ from it. We then run the above amplification step, via p, on $p_1(A)$. As logspace reductions compose we can do constant number of p_1 compositions, followed by a single composition of p. This essentially amounts to composting two (or a constant number of) polynomials of degree $\log n$, and results in a polynomial of degree poly$(\log n)$. However, for the above to work we need to find p_1 that not only expands gap, but also does not increase the ℓ_∞ norm of A too much. In Section 6.1 we study this approach with the Féjer kernel, which is essentially the function underlying the quantum protocol. We find that the function amplifies well gaps, has small coefficients, and also by numeric experiments, often does not increase the ℓ_∞ norm beyond a small fixed

constant. However, we showed numerically (and also gave theoretical evidence) that there are graphs where the ℓ_∞ norm becomes super-constant. We therefore cannot employ the above approach to achieve super-constant accuracy for general undirected graphs.

We believe it is highly natural to explore what probabilistic logspace can achieve and not only what can be derandomized. We also believe approximating eigenvalues is a natural problem. Solving the problem in full (i.e., with polynomial accuracy) would essentially imply that one can approximately compute linear algebra in full (for stochastic, Hermitian operators) already in BPL. In this paper we achieve approximations only for constant accuracy. Our technique is new (at least as far as we know). We also show going beyond constant accuracy requires a new idea. We do not have a conjecture as to whether such a better approximation is possible in BPL or not, and we believe solving this problem is important, and may shed new light on the strengths and weaknesses of the probabilistic space bounded model.

Chapter 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 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 $s(n)$. $\mathsf{BPL} = \cup_c \mathsf{BPSPACE}(c \log n)$.

Often we are interested in computing a *value* (e.g., an entry in a matrix with integer values or the whole matrix) and are only able to approximate it 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 $M(x,y)$ (δ, ε)–approximates $u(x)$ if:

$$\forall_{x \in \{0,1\}^n} \qquad \Pr_y \left[|M(x,y) - u(x)| \geq \delta \right] \leq \varepsilon \tag{2.1}$$

The values $M(x,y)$ are almost always close to the correct value $u(x)$, but highly depend on y, and thus change from one execution of M to another. In fact, we can do better:

Lemma 2.2. *(based on [SZ99]) Suppose $M(x,y)$ (δ,ε)–approximates $u(x)$ using $O(\log \frac{n}{\delta\varepsilon})$ space. Let $\zeta > 0$ be arbitrary. Then, there exists another probabilistic TM $M'(x,y;s)$ with $|y| = O(\log \frac{n}{\delta\varepsilon\zeta})$ and $|s| = O(\log \zeta^{-1})$, and there exists a function $u'(x,s)$ such that:*

- *For all s, $|u'(x,s) - u(x)| \leq \delta$.*
- $\Pr_s\left[\Pr_y[M'(x,y;s) = u'(x,s)] < 1-\varepsilon\right] \leq \zeta$.

In words, throughout the computation we have two kinds of random bits: few random bits s that we toss and keep (*offline* random bits) and many random bits y that we toss and forget (as usual in space bounded computation). The main thing to notice is that M' almost always outputs a single value $u'(x,s)$ that does not depend on the random coins y, and this value is almost always close to the desired value $u(x)$.

2.2 Matrices and random walks

For a positive integer n, we denote $[n] = \{1,\ldots,n\}$. $A^\dagger$ is the (conjugate) transpose of a matrix A. For a vector $v \in \mathbb{C}^n$ and $p > 0$, $\|v\|_p$ is the ℓ_p norm, $\|v\|_p = (\sum_i |v_i|^p)^{1/p}$. As p approaches infinity, the ℓ_p norm approaches the infinity norm $\|v\|_\infty = \max_i |v_i|$. $\|v\|$ denotes the ℓ_2 norm of v.

For a matrix $A \in \mathbb{C}_{n\times n}$, the operator norm corresponding to the ℓ_p norm is $\|A\|_p = \max_{\|v\|_p=1} \|Av\|_p$. The special case of $p = 2$, the spectral norm, is also the largest singular value of the operator A. Also, $\|A\|_\infty$ is

$$\|A\|_\infty \;=\; \max_r \sum_j |A_{r,j}|.$$

For every p (including infinity), $\|Av\|_p \leq \|A\|_p\|v\|_p$. When p is not stated explicitly, we denote $\|A\|$ as the induced ℓ_2 norm of v. Note that the following well-known inequality always holds: $\|A\|_\infty \leq \sqrt{n}\|A\|$. Also, for any $U, V \in \mathbb{R}^{n\times n}$ such that $\|U\|_\infty \leq 1$ and $\|V\|_\infty \leq 1$ and for any nonnegative integer p, $\|U^p - V^p\|_\infty \leq p\|U - V\|_\infty$.

We can view a directed or undirected graph $G = (V,E)$ over n vertices, as a linear operator that describes the transition probabilities of a random walk on G. Specifically, let $\tilde{A}$ be the adjacency matrix of the graph. Let D be a diagonal matrix D with $D(i,i) = d_{\text{out}}(v_i)$ for every $i \in [n]$. Then the *transition matrix* of G is the linear operator $A = D^{-1}\tilde{A}$. Notice that A is stochastic and corresponds to a random walk on G.

Claim 2.3. *Let A be a stochastic matrix. Then, $\mathbf{1}_n$ is an eigenvector of A with eigenvalue 1. All other eigenvalues have absolute value at most 1.*

Proof. As the sum of every row of A is exactly 1, $A\mathbf{1}_n = \mathbf{1}_n$ so $\mathbf{1}_n$ is an eigenvector with eigenvalue 1. Now assume there exists an eigenvector v with eigenvalue λ. Then, $\|Av\|_\infty = |\lambda|\|v\|_\infty$. Also, as A is stochastic $\|A\|_\infty = 1$ so $\|Av\|_\infty \leq \|A\|_\infty\|v\|_\infty \leq \|v\|_\infty$ and $|\lambda| \leq 1$. □

Claim 2.3 shows that the largest eigenvalue of any (directed or undirected) graph is 1. However, the largest singular value of a graph might be larger than 1. For example, the undirected graph $\{\{1,3\},\{2,3\}\}$ on $V = \{1,2,3\}$ has the transition matrix

$$A = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 0 & 1 \\ \frac{1}{2} & \frac{1}{2} & 0 \end{pmatrix}$$

and SVD decomposition

$$A = \begin{pmatrix} \frac{\sqrt{2}}{2} & 0 & -\frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} & 0 & \frac{\sqrt{2}}{2} \\ 0 & -1 & 0 \end{pmatrix} \begin{pmatrix} \sqrt{2} & 0 & 0 \\ 0 & \frac{\sqrt{2}}{2} & 0 \\ 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} 0 & -\frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} \\ 0 & -\frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \\ 1 & 0 & 0 \end{pmatrix}^{\dagger},$$

so its largest singular value is $\sqrt{2} > 1$.

Chapter 3

Simulatable matrices

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 well powers of A. 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 3.1. A matrix A of dimension n is *simulatable* if $\|A\| \leq \text{poly}(n)$ and there exists a probabilistic algorithm that on input A, $k \in \mathbb{N}$, $s, t \in [n]$, runs in space $O(\log \frac{nk}{\delta\varepsilon})$ and (δ, ε)–approximates $A^k[s,t]$.

As expected, the transition matrix of a (directed or undirected) graph is simulatable. Namely,

Lemma 3.2. *Let $G = (V, E)$ be a directed or undirected graph with n vertices and let A be its transition matrix. Then, A is simulatable.*

Proof. Let $k \in \mathbb{N}$, $s, t \in [n]$ and $\varepsilon, \delta > 0$. 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,

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

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

Intuitively, any stochastic matrix corresponds to a walk on some directed graph. A technical issue is that the entries of the matrix might have high precision beyond our small space capabilities. In Section 3.1 we deal with this problem and show:

Lemma 3.3. *Let A be a stochastic matrix of dimension n. Then, A is simulatable.*

In fact, this can be further generalized to any real matrix A (with possibly negative entries) with bounded infinity norm, i.e., for every row r the sum of the absolute values of the entries in row r is at most 1. In Section 3.2 we show:

Lemma 3.4. *Let A be a real matrix of dimension n such that $\|A\|_\infty \leq 1$. Then, A is simulatable.*

However, for matrices A that have infinity norm slightly larger than 1 we give a disappointing answer. For such matrices we prove (in the above section):

Lemma 3.5. *Let A be a real matrix of dimension n such that $\|A\|_\infty \leq 1+\gamma$ for $\gamma > 0$. Then, there exists a probabilistic algorithm that on input A, $k \in \mathbb{N}$, $s,t \in [n]$, runs in space $O(\lceil k \log(1+\gamma)\rceil + \log \frac{nk}{\delta\varepsilon})$ and (δ,ε)–approximates $A^k[s,t]$.*

Notice that when the infinity norm is bounded by 1 or less, we can simulate in probabilistic logspace high powers k, namely k can be polynomial in n, while for infinity norm above 1 we can only support k that is logarithmic in n. We prove both lemmas in the next section. This result is tight in general, as, for e.g., $\|(2I_{n\times n})^k\|_\infty = 2^k$.

As explained in the introduction, one motivation to our research is exploring whether the quantum algorithm for approximating the SVD decomposition of a linear operator, can be made classical for stochastic matrices. The quantum algorithm proceeds by viewing an undirected graph G with transition matrix A as a Hamiltonian describing the unitary transformation $U = e^{iA}$. We show that if A is simulatable then small powers of $U = e^{iA}$ can also be approximated in BPL. Specifically,

Lemma 3.6. *Let A be a simulatable matrix of dimension n, k an integer. Let $U = e^{iA}$. Then, there exists a probabilistic algorithm that on input A, k and $s,t \in [n]$, runs in space $O(|k|(1+\|A\|_\infty) + \log \frac{n}{\delta\varepsilon})$ and (δ,ε)–approximates $U^k[s,t]$.*

The proof of this lemma appears in Section 3.3. Apart from matrix exponentials the above lemma implies the approximation of the matrix sine and cosine, as for every $A \in \mathbb{C}^{n\times n}$, $\sin(A) = \frac{1}{2i}\left(e^{iA} - e^{-iA}\right)$ and $\cos(A) = \frac{1}{2}\left(e^{iA} + e^{-iA}\right)$.

We conclude with a simple observation:

Corollary 3.1. *There exists a probabilistic algorithm* TrPower$(A,n,k,\delta,\varepsilon)$ *that gets as input a simulatable matrix A of dimension n and $k \in \mathbb{N}$, and (δ,ε)–approximates* Tr (A^k) *using $O(\log \frac{nk}{\delta\varepsilon})$ space.*

3.1 Stochastic matrices are simulatable

Intuitively, any stochastic matrix corresponds to a walk on some directed graph. A technical issue is that the entries of the matrix might have high precision beyond our

small space capabilities. To deal with that we truncate the matrix to the desired precision,

Following [SZ99], we define the *truncation operator* $\lfloor\ \rfloor_\zeta$ as a function mapping any real number $z \in [0,1]$ and integer ζ to $\lfloor z \rfloor_\zeta$ obtained by truncating the binary expansion of z after ζ binary digits. Thus, $\lfloor z \rfloor_\zeta = 2^{-\zeta}\lfloor 2^\zeta z \rfloor$. The truncation operator is extended to matrices by simply applying it entry by entry. It is obvious that this operator maps stochastic matrices to *substochastic* matrices, namely, those matrices with all entries nonnegative and all row sums at most 1. Also, clearly $\|M - \lfloor M \rfloor_\zeta\|_\infty \leq n2^{-\zeta}$.

We utilize the truncation operator to describe a space-bounded random walk on a graph originating from the stochastic matrix. We define:

Definition 3.7. (Truncated transition graph) Given a stochastic matrix A of dimension n and an integer ζ, denote $\bar{A} = \lfloor A \rfloor_\zeta$. Then, the truncated transition (directed) graph $G(A,\zeta) = (V,E)$ is defined as follows:

- $V = \{0,1,\ldots,n\}$.
- For every $i,j \in [n]$ there are $2^\zeta \bar{A}(i,j)$ edges going from vertex i to j.
- For every $i \in [n]$ there are $2^\zeta \left(1 - \sum_{j=1}^n \bar{A}(i,j)\right)$ edges going from vertex i to 0.
- There is one edge going from 0 to itself.

We are now ready to prove that stochastic matrices are simulatable.

Proof. (of Lemma 3.3) Let $k \in \mathbb{N}$, $s,t \in [n]$ and $\varepsilon, \delta > 0$. Denote $\zeta = \lceil \log(kn\delta^{-1}) + 1 \rceil$ and $G(A,\zeta) = (V,E)$ the truncated transition graph of A. Note that the number of edges in G is $n \cdot 2^\zeta + 1$. The transition matrix of G is of the form

$$\tilde{A} = \begin{pmatrix} 1 & 0_n \\ e & \bar{A} \end{pmatrix}$$

where $\bar{A} = \lfloor A \rfloor_\zeta$, 0_n is a row vector of zeros and e is such that $e(i) = 1 - \sum_{j=1}^n \bar{A}(i,j)$ for every $i \in [n]$. Then,

$$\tilde{A}^k = \begin{pmatrix} 1 & 0_n \\ \sum_{i=0}^{k-1} \bar{A}^i e & \bar{A}^k \end{pmatrix}.$$

Let the labeling of rows and columns of $\tilde{A}$ start from 0. Now, consider running the algorithm described in Lemma 3.2 on G to obtain a $(\delta/2,\varepsilon)$–approximation of $\bar{A}^k(s,t)$. Denote the algorithm's outcome by $M(k,s,t)$. Then, we obtain with probability at least $1-\varepsilon$,

$$\begin{aligned} |M(k,s,t) - A^k(s,t)| &\leq |M(k,s,t) - \bar{A}^k(s,t)| + |\bar{A}^k(s,t) - A^k(s,t)| \\ &\leq \delta/2 + \|\bar{A}^k - A^k\|_\infty \\ &\leq \delta/2 + k \cdot n \cdot 2^{-\zeta} \ \leq\ \delta. \end{aligned}$$

The algorithm runs in space $O(\log nk\delta^{-1}|E| + \log\log\varepsilon^{-1}) = O(\log nk\delta^{-1} + \log\log\varepsilon^{-1})$. As the sum of every row of A is exactly 1, $\|A\|_\infty = 1$, implying $\|A\| \leq \sqrt{n}$, so A is simulatable. □

3.2 Matrices with bounded infinity norm are simulatable

In the previous section we were dealing with matrices with non-negative entries. Now, we would like to extend the above results to arbitrary real matrices, with positive or negative entries, as long as they have bounded infinity norm. Given a real[1] matrix A of dimension n and a constant c, recall that $\|A\|_\infty \leq c$ if for every $i \in [n]$, $\sum_j |A[i,j]| \leq c$. We show that if $c \leq 1$, A is simulatable. For $c > 1$, however, we obtain a much weaker result.

Given a bounded matrix A, let $d_i(A) = \sum_j |A[i,j]|$. Define a mapping φ such that $\varphi(A)[i,j] = \frac{1}{d_i(A)}|A[i,j]|$. It is immediate that $\varphi(A)$ is stochastic. We prove Lemma 3.4, which states that if $\|A\|_\infty \leq 1$ then A is simulatable.

Proof. (of Lemma 3.4) Let $k \in \mathbb{N}$, $s,t \in [n]$ and $\varepsilon,\delta > 0$. 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

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

Fix, as before, $\zeta = \lceil\log(kn\delta^{-1})+1\rceil$. Let $G = G(\varphi(A),\zeta)$ be the truncated transition graph of $\varphi(A)$ and let $\tilde{A}$ be its transition matrix. First, assume that in the appropriate indices, $\tilde{A} = \varphi(A)$ (i.e., the entries of $\varphi(A)$ can be accurately represented using ζ bits).

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,

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

[1] In what follows, by generalizing the sign of an entry to its *phase*, the result easily applies to complex matrices as well.

Denote the algorithm's outcome by $M(k,s,t)$. As in Lemma 3.2, and using the fact that $|p(A,i)| \leq 1$, the algorithm can (δ,ε)–approximates $\mathbb{E}[Y_i]$ by choosing T which is $\mathrm{poly}(\delta^{-1}, \log \varepsilon^{-1})$. Following the same analysis as of Lemma 3.2, the algorithm runs in $O(\log nk\delta^{-1} + \log\log \varepsilon^{-1})$ space.

We now consider the general case, where $\tilde{A}$ is only an *approximation* to $\varphi(A)$. Let $V \in \mathbb{R}^{n\times n}$ be such that $\mathbb{E}[Y_i] = V^k[s,t]$. Then $\|V - A\|_\infty \leq n \cdot 2^{-\zeta}$ and $|V^k[s,t] - A^k[s,t]| \leq k \cdot n \cdot 2^{-\zeta} \leq \delta$. Overall, the algorithm $(2\delta,\varepsilon)$–approximates $A^k[s,t]$ in this case as well, with the same space requirements. As $\|A\| \leq \sqrt{n}$, we conclude that A is simulatable. □

The case where $c > 1$ now also follows easily:

Proof. (of Lemma 3.5) Consider the matrix $A' = (1+\gamma)^{-1}A$. As $\|A'\|_\infty \leq 1$, A' can be simulated. Since $A^k[s,t] = (1+\gamma)^k A'^k[s,t]$, $(1+\gamma)^{-k}\delta$ accuracy is sufficient in order to (δ,ε)–approximate $A^k[s,t]$. The result follows directly from Lemma 3.4. □

3.3 Matrix exponentials are simulatable to small powers

Proof. (Lemma 3.6) Consider the Taylor expansion of U^k,

$$U^k \;=\; e^{ikA} \;=\; \sum_{\ell=0}^{\infty}\left(\frac{k^{4\ell}A^{4\ell}}{(4\ell)!} - \frac{k^{4\ell+2}A^{4\ell+2}}{(4\ell+2)!}\right) + i\sum_{\ell=0}^{\infty}\left(\frac{k^{4\ell+1}A^{4\ell+1}}{(4\ell+1)!} - \frac{k^{4\ell+3}A^{4\ell+3}}{(4\ell+3)!}\right).$$

Let $\tilde{U}$ be the approximation obtained by taking the first overall T terms for $T+1 = \lceil e^2|k|(1+\|A\|_\infty) + \ln(2n\delta^{-1})\rceil$. Then, the truncation error can be bounded as follows [GVL96, Theorem 11.2.4][2].

$$\|U^k - \tilde{U}\|_\infty \leq \frac{n\|ikA\|_\infty^{T+1}}{(T+1)!}\max_{0\leq s\leq k}\|e^{isA}\|_\infty$$

Playing with the parameters:

$$\|U^k - \tilde{U}\|_\infty \leq \frac{n\|ikA\|_\infty^{T+1}}{(T+1)!}e^{\|ikA\|_\infty} \leq ne^{|k|\|A\|_\infty}\left(\frac{e|k|\|A\|_\infty}{T+1}\right)^{T+1} \leq \frac{\delta}{2}.$$

We now approximate the truncated series by (δ',ε')–approximating powers of A, for $\delta' = \frac{\delta}{4T\cdot 4^T}$ and $\varepsilon' = \varepsilon/T$. We approximate the real part and imaginary part separately. For the real part, the accumulated accuracy error is $\sum_{\ell=0,\ell \text{ even}}^{T}\frac{|k|^\ell}{\ell!}\delta'$. However, $\frac{|k|^\ell}{\ell!}$ maximizes at $\ell = |k|$ and using Stirling formula it is bounded by $4^{|k|} \leq 4^T$. Thus, the accumulated error is at most $T4^T\delta' \leq \delta/4$ and the same holds for the imaginary part.

[2]The theorem is stated with $\|\cdot\|$ replacing $\|\cdot\|_\infty$, but the proof also proves the claim as we state.

Adding it to it the truncation error (that we showed is at most $\delta/2$) the truncation error is at most δ. The error probability is at most $T\varepsilon' = \varepsilon$.

The space complexity is bounded by $O(\log T + \log nT\delta'^{-1}\varepsilon'^{-1}) = O(|k|(1 + \|A\|_\infty) + \log n\delta^{-1}\varepsilon^{-1})$. □

Chapter 4

Approximating eigenvalues with constant accuracy

In this chapter we try to approximate the eigenvalues of the transition matrix of an undirected graph. Consider an undirected graph $G = (V, E)$ with an adjacency matrix $\tilde{A}$ and degree matrix D and let $A = D^{-1}\tilde{A}$ be its transition matrix, as described in Chapter 2. Note that unless G is regular, A need not be Hermitian. However, consider the matrix [1] $L = D^{-1/2}\tilde{A}D^{-1/2}$. L is Hermitian and thus has an eigenvector basis. $A = D^{-1/2}LD^{1/2}$, so A is conjugate to L and A itself is also diagonalizable and has real eigenvalues. A is stochastic, so from Claim 2.3 we infer that its eigenvalues are in the range $[-1, 1]$. The matrix we work with is $B = \frac{1}{2}A + \frac{1}{2}I_{n\times n}$, which is stochastic, has eigenvalues in the range $[0, 1]$, and whose eigenvectors are in a natural one-to-one correspondence with A's eigenvalues.

So assume we have a matrix B, with a full eigenvector basis and real eigenvalues $0 \leq \lambda_n \leq \ldots \leq \lambda_1 \leq 1$. The ultimate goal is to output an approximation of its spectrum in small space. We do not know how to do that, nor whether this is possible or not. Instead, we achieve a much more modest goal: we show a probabilistic algorithm that approximates with constant accuracy whether there exists an eigenvalue close to a given value λ or not. Formally,

Theorem 4.1. *There exists a probabilistic algorithm that on input B as above, constants $\beta > \alpha > 0$ and $\lambda \in [0, 1]$ such that:*

- *There are d eigenvalues λ_i satisfying $|\lambda - \lambda_i| \leq \alpha$.*
- *All other eigenvalues λ_i satisfy $|\lambda - \lambda_i| \geq \beta$.*

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

The parameters α, β describe the accuracy of the algorithm. The accuracy we achieve is far from being satisfying. The matrix B has n eigenvalues in the range $[0, 1]$, so the

[1] The matrix $\mathcal{L} = I - L$ is usually referred to as the normalized Laplacian of a graph.

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 [TS13], thus getting a true classical analogue to the quantum algorithm in [TS13]. However, we do not know how to achieve subconstant accuracy. The question whether better accuracy is possible in BPL is the main problem raised by this work. Even proving Theorem 4.1 with its constant accuracy is non-trivial.

The main idea behind the algorithm is to manipulate the eigenvalues of B by computing a polynomial in the input matrix B, without knowing the decomposition of B to eigenvectors and eigenvalues. More precisely, assume B decomposes to $B = VDV^{-1}$ where V is invertible and D diagonal with B's eigenvalues on the main diagonal. Then, $B^k = VD^kV^{-1}$ and

$$D^k = \begin{pmatrix} \lambda_1^k & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & \lambda_n^k \end{pmatrix}$$

Thus, if $p : \mathbb{R} \to \mathbb{R}$ is a degree T univariate polynomial $p(x) = \sum_{i=0}^{T} c_i x^i$ then $p(B) = \sum_{i=0}^{T} c_i B^i$ is

$$p(B) = V \begin{pmatrix} p(\lambda_1) & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & p(\lambda_n) \end{pmatrix} V^{-1}$$

In particular, if $p : \mathbb{R} \to \mathbb{R}$ has a peak around λ and is close to 0 otherwise, then Tr $(p(B))$ approximately counts the number of eigenvalues B has around λ. Thus, if we can compute $p(B)$ in small probabilistic space, we can solve the problem.

More formally, we assume $p(x) = \sum_{i=0}^{M} c_i x^i$ can be efficiently computed in the sense that $M, |c_i| = 2^{O(s(n))}$ and for every i, c_i can be computed (exactly) by a deterministic Turing machine that uses $O(s(n))$ space. We further assume $p(x)$ 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 $x \in [0,1] \setminus (\lambda-\beta, \lambda+\beta)$. In the next section we show how to obtain such a polynomial p with $s(n) = O(\frac{\log n}{(\beta-\alpha)^2})$ and $\eta = \eta(n) = n^{-2}$.

Having that, we are ready to present the algorithm. The input to the algorithm is $n, A, \lambda, \alpha, \beta$. We set $M = s(n) = 32(\beta - \alpha)^{-2} \log n$, $\delta' = \delta \cdot 2^{-2s(n)}$ and $\varepsilon' = \varepsilon \cdot 2^{-s(n)}$. The algorithm evaluates

$$\sum_{i=0}^{M} c_i \cdot \textsc{TrPower}(A, n, i, \delta', \varepsilon'))$$

where TrPower is the probabilistic algorithm guaranteed by Corollary 3.1.

Lemma 4.1. *The algorithm runs in space* $O(s(n) + \log(n\delta^{-1}\varepsilon^{-1}))$.

Proof. The algorithm has a loop over i. For every i it stores c_i (that can be computed in $O(s(n))$ space), and $\textsc{TrPower}(A, n, i, \delta', \varepsilon')$ (that can be computed in $O(s(n) + \log(n\delta'^{-1}\varepsilon'^{-1}))$ space). As $\mathrm{Tr}\,(A^i) \leq n$, the number of bits required to store the output is also bounded by $O(s(n))$, obtaining the desired space complexity. □

Lemma 4.2. *The algorithm (δ, ε)–approximates* $\mathrm{Tr}\,(p(A))$.

Proof. As $\mathrm{Tr}\,(p(A)) = \sum_i c_i \mathrm{Tr}\,(A^i)$, applying Corollary 3.1 we obtain that with probability at least $1 - 2^{s(n)}\varepsilon' = 1 - \varepsilon$,

$$\left| \sum_{i=0}^{\deg(p)} c_i \left(\mathrm{Tr}\,(A^i) - \textsc{TrPower}(A, n, i, \delta', \varepsilon') \right) \right| \leq \delta' 2^{2s(n)} = \delta. \tag{4.1}$$

□

With that we can complete the proof of Theorem 4.1.

Proof. (of Theorem 4.1) Consider $R(B, \lambda)$, the above approximation of $\mathrm{Tr}\,(p(B))$ with $\delta = n^{-2}$ and $\varepsilon = 1/3$. Then, from Lemma 4.2, with probability at least $2/3$,

$$\begin{aligned} R(B,\lambda) &\leq \mathrm{Tr}\,(p(B)) + \delta \\ &\leq \sum_{i:|\lambda_i - \lambda| \leq \alpha} p(\lambda_i) + \sum_{i:|\lambda_i - \lambda| \geq \beta} p(\lambda_i) + \delta \\ &\leq d + (n-d)\eta + \delta \\ &\leq d + (n\eta + \delta). \end{aligned}$$

Similarly, $R(B, \lambda) \geq d - (n\eta + \delta)$. As for large enough n, $n\eta + \delta < 1/2$, d is obtained by taking the integer closest to $R(B, \lambda)$.

From Lemma 4.1, the space complexity for computing d is given by $O(s(n) + \log(n\delta^{-1}\varepsilon^{-1})) = O(\log n)$. □

4.1 Using the symmetric threshold functions

There are several natural candidates for the function p above. In this section we use the threshold function to obtain such a function p. Assume λ is rational, $\lambda = \frac{k}{M}$ for some integers k and M. Define:

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

p 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,

Lemma 4.3. *Let $x \in [0,1]$. $p_\lambda(x)$ approximates $\mathbf{Th}_\lambda(x)$ over $[0,1]$ with accuracy $(\xi(\delta))^{Mx}$, where $\delta = \frac{\lambda - x}{x}$ and $\xi(\delta) = \frac{e^\delta}{(1+\delta)^{1+\delta}}$.*

Proof. $p_\lambda(x)$ represents the probability of getting at least k 1-s, in M independent trials X_i, each getting 1 with probability x. The lemma then follows by Chernoff. Set $\mu = Mx$. If $\delta > 0$, then $x < k/M$ so $\mathbf{Th}_\lambda(x) = 0$ and

$$p_\lambda(x) = \Pr\left[\sum_{i=1}^{M} X_i \geq (1+\delta)\mu\right] \quad \leq \quad \left(\frac{e^\delta}{(1+\delta)^{1+\delta}}\right)^\mu. \tag{4.2}$$

If $\delta \leq 0$, $\mathbf{Th}_\lambda(x) = 1$ and

$$p_\lambda(x) = 1 - \Pr\left[\sum_{i=1}^{M} X_i < (1+\delta)\mu\right] \quad \geq \quad 1 - \left(\frac{e^\delta}{(1+\delta)^{1+\delta}}\right)^\mu. \tag{4.3}$$

In both cases, $|p_\lambda(x) - \mathbf{Th}_\lambda(x)| \leq (\xi(\delta))^{Mx}$. □

Let us now express $p_\lambda(x)$ as a polynomial in x, $p_\lambda(x) = \sum_{i=0}^{M} c_i x^i$. We have,

$$p_\lambda(x) \;=\; \sum_{j=\lambda M}^{M} \binom{M}{j} x^j (1-x)^{M-j} \;=\; \sum_{j=\lambda M}^{M} \sum_{t=0}^{M-j} \binom{M}{j}\binom{M-j}{t} (-1)^t x^{j+t}.$$

Therefore, $c_i = (-1)^i \sum_{j=\lambda M}^{i} \binom{M}{j}\binom{M-j}{i-j}(-1)^j$ and $|c_i| \leq \sum_{j=\lambda M}^{i} \binom{M}{j}\binom{M-j}{i-j} \leq M\binom{M}{M/2}^2 = 2^{O(M)}$. 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 $(-1)^j\binom{M}{j}\binom{M-j}{i-j}$.

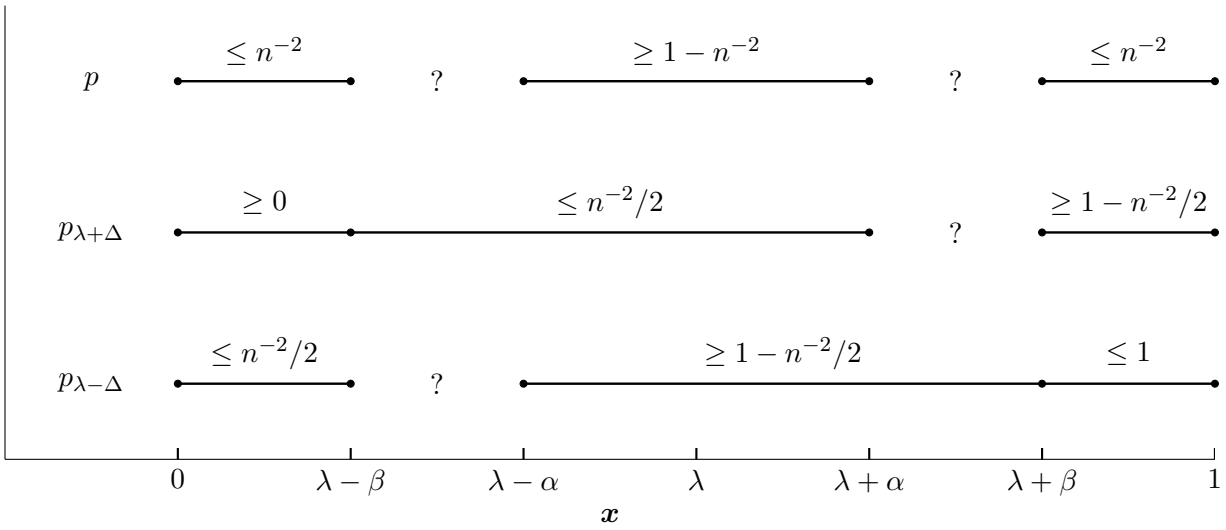
To obtain the polynomial p define:

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

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

Lemma 4.4. *$p(x) \geq 1 - n^{-2}$ 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:



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

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

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

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

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

Chapter 5

An approach for approximating eigenvalues with enhanced accuracy

We first formalize the above 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 $n \in \mathbb{N}$ there is a univariate polynomial $p_{\lambda,n}$ with coefficients in $\mathbb{R}$.

Definition 5.1. (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 5.2. (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 $\lambda \in [0,1]$,

- 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, $p_{\lambda,n}(x) \in [0,\eta]$.

As in the previous chapter, let B have a full set of real eigenvalues $0 \leq \lambda_n \leq \ldots \leq \lambda_1 \leq 1$. Also, assume there exist matrices V and D such that V is invertible, $B = VDV^{-1}$ and D is diagonal with B's eigenvalues. We further assume that $\|B\|_\infty \leq 1$.

We would like to try and relax the conditions of Theorem 4.1 and in particular for any given λ to reduce the promised interval in which there are no eigenvalues (i.e., to reduce the gap $\beta - \alpha$) and also to improve the accuracy of the result (i.e., to reduce α).

Unfortunately, finding threshold families of polynomials that are $O(\log n)$–small and have α and β that are $o(1)$ is impossible. For that we use:

Lemma 5.3. *[Tim63, Theorem 2.9.11] Let $T_n(x)$ be the Chebyshev polynomial (of the first kind) of degree n. Then, if the polynomial $P_n(x) = \sum_{i=0}^{n} c_i x^i$ satisfies the inequality $|P_n(x)| \leq L$ on the segment $[a, b]$ then at any point outside the segment we have*

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

For properties of the Chebyshev polynomials see [Riv74, Chapter 1.1]. We mention a few properties that we use. An explicit representation of $T_n(x)$ is given by

$$T_n(x) = \frac{\left(x - \sqrt{x^2 - 1}\right)^n + \left(x + \sqrt{x^2 - 1}\right)^n}{2}.$$

We can also see that $|T_n(-x)| = |T_n(x)|$ and that T_n is monotonically increasing for $x > 1$. Also,

$$|T_n(1 + \varepsilon)| \leq \left(1 + \varepsilon + \sqrt{(1 + \varepsilon)^2 - 1}\right)^n \leq \left(1 + 4\sqrt{\varepsilon}\right)^n \leq e^{4n\sqrt{\varepsilon}} \leq 2^{8n\sqrt{\varepsilon}} \qquad (5.1)$$

for $0 \leq \varepsilon \leq 1$. With that we prove:

Theorem 5.1. *Let $\alpha, \beta, \lambda, \eta$ be such that $\alpha \leq \beta$, $\lambda + \beta \leq \frac{1}{2}$, $\beta = o(1)$ and $\eta = o(n^{-1})$. 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 without loss of generality 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, w.l.o.g., 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 $x \in [\lambda + \beta, 1]$. Then, following Theorem 5.3 we have that:

$$\begin{aligned}
|p_{\lambda,n}(\lambda)| &\leq n^{-1} \cdot \left| T_{c\cdot\log n}\left(\tfrac{\lambda - \beta - 1}{-\lambda - \beta + 1}\right)\right| \\
&= n^{-1} \cdot \left| T_{c\cdot\log n}\left(1 + \tfrac{2\beta}{1 - \lambda - \beta}\right)\right| && \text{By } |T_n(x)| = |T_n(-x)| \\
&\leq n^{-1} \cdot |T_{c\cdot\log n}(1 + 4\beta)| && \text{By the monotonicity of } T_n(x) \text{ for } x > 1 \text{ and } \lambda + \beta \leq \tfrac{1}{2}
\end{aligned}$$

By Equation (5.1) $|p_{\lambda,n}(\lambda)| \leq n^{-1} 2^{32c\sqrt{\beta}\log n} \leq n^{-1+32c\sqrt{\beta}}$. As $\beta = o(1)$ for n large enough we have $|p_{\lambda,n}(\lambda)| \leq n^{-1/2}$, contradicting the fact that $|p_{\lambda,n}(\lambda)| \geq 1 - n^{-1}$. □

Theorem 5.1 leads us to consider instead a *multi-stage* approach. suppose B has the promise that all eigenvalues are either α–close to λ or β–far from λ, but $\beta - \alpha$ is too small to be handled using the techniques of Chapter 4. We say we can *expand* the gap (α, β) to a larger gap (α', β') if we can find a polynomial p_1 with the property that the number of eigenvalues of B in $[\lambda - \alpha, \lambda + \alpha]$ equals the number of eigenvalues of $p_1(B)$ in $[\lambda' - \alpha', \lambda' + \alpha']$ and similarly for β and β'. That is, p_1 *amplifies* the small gap (α, β) to a larger gap (α', β'). Then, we can first compute $p_1(B)$ using a small-space algorithm, and then run the algorithm of Theorem 4.1 on $p_1(B)$.

However, for this approach to work, it is crucial that $p_1(B)$ is *simulatable*. The only way we know to guarantee that $p_1(B)$ is simulatable is by bounding the infinity norm of $p_1(B)$ and using Lemma 3.4. In Chapter 6 we explore this framework with p_1 being the Féjer kernel, which is the function that appears in the quantum logspace algorithm for approximating the spectrum of a matrix. We find out the function is small and distinguishing, but unfortunately it does not always preserve the infinity norm. The bottom line is that currently we cannot get this framework to work.

Chapter 6

The Féjer kernel

Throughout, we extend the notion of a matrix *polynomial* to a general matrix *function*, whereas its computation is done via truncating the function's Taylor series. See [Hig08] for a comprehensive treatment of matrix functions. We use the same notation for a function of a real variable and a function of a matrix, as it is clear from context.

As noted, we need a distinguisher family of functions $\{f_{\lambda,T}\}_{\lambda\in[0,1],T\in\mathbb{N}}$ (T will be a function of n) that is computable in probabilistic small space, amplifies gap and preserves the infinity norm. All properties will soon be formally described. The function we consider as a candidate is the (normalized) Féjer kernel. It plays a central role in Fourier analysis (see [Hof07, Chapter 2]), and more interestingly, as the probability density function involved in the measurement of a quantum phase estimation circuit (see [KLM07, Chapter 7] or [NC00, Chapter 5] for more details).

The Féjer kernel has the following form:

$$\begin{aligned} f_T(x) &= \frac{1}{T^2}\sum_{k=0}^{T-1}\left(1+2\sum_{j=1}^{k}\cos(jx)\right) \\ &= \frac{1}{T}+\frac{2}{T^2}\sum_{k=1}^{T-1}(T-k)\cos(kx). \end{aligned} \tag{6.1}$$

Up to removable discontinuities, it can also be written as:

$$f_T(x) = \frac{1}{T^2}\frac{\sin^2\frac{Tx}{2}}{\sin^2\frac{x}{2}}, \tag{6.2}$$

so it is immediate that the function is even, and for every x and $T>0$, $f_T(x)\in[0,1]$. It is then natural to set $f_{T,\lambda}(x)=f_T(x-\lambda)$. Equation (6.1), together with Lemma 3.6, shows:

Claim 6.1. *Let A be a simulatable matrix of dimension n, an integer $T>0$, $\lambda\in[0,1]$ and let $U=f_T(A-\lambda I_{n\times n})$. Then, there exists a probabilistic algorithm that on input A, T, λ and $s,t\in[n]$, runs in space $O(T+\log\frac{n}{\delta\varepsilon})$ and (δ,ε)–approximates $U[s,t]$.*

Also, f_T is a good distinguisher:

Lemma 6.2. *The family* $\{f_{T,\lambda}\}_{T\in\mathbb{N},\lambda\in[0,1]}$ *is a* $\left(\frac{\pi}{2T}, \frac{2\pi}{T}, 1/4\right)$*–distinguisher. Also, for every* $T \in \mathbb{N}$*,* $f_T(x)$ *is monotonically decreasing in* $\left[0, \frac{\pi}{2T}\right]$*.*

Proof. Fix $T \in \mathbb{N}$. By differentiation of Equation (6.1) we infer that:

$$f'_T(x) = -\frac{2}{T^2}\sum_{k=1}^{T} k(T-k)\sin(kx).$$

Hence, $f_T(0) = 1$ is a maxima. Also, for $x \in (0, \frac{\pi}{T}]$, $\sin(kx) > 0$ for every $0 < k \leq T-1$ so $f'_T(x) < 0$ and thus monotonically decreasing there.

The fact that for $x \geq 2\pi/T$, $f_T(x) \leq 1/4$ follows from [HS65, Section 18]. Now, for $x \leq \pi/(2T)$, we have $f_T(x) \geq f_T(\pi/(2T))$. And indeed,

$$f_T\left(\frac{\pi}{2T}\right) = \frac{1}{T^2}\frac{1}{2\sin^2\left(\frac{\pi}{4T}\right)} \geq \frac{1}{T^2}\frac{16T^2}{2\pi^2} \geq \frac{3}{4}. \qquad \square$$

The last lemma concerns amplification.

Lemma 6.3. *Let* $T > 0$*,* $0 < c_1 < c_2 \leq 2$ *and* $c_2 \geq 7c_1$*. Then, the following hold, for every integer* $m \geq 0$ *and large enough* T*.*

1. $f_T\left(c_1 T^{-2+2^{-m}}\right) \geq 1 - \frac{c_1^2}{12}T^{-2+2^{-m+1}}$.
2. $f_T\left(c_2 T^{-2+2^{-m}}\right) \leq 1 - \left(\frac{c_1^2}{12} + \frac{c_2^2-c_1^2}{96}\right)T^{-2+2^{-m+1}}$.

Proof. First, by definition, and using $\cos(x) \geq 1 - \frac{x^2}{2}$ and $\sum_{k=0}^{T-1}\sum_{j=1}^{k} j^2 \leq \frac{T^4}{12}$,

$$f_T(c_1 T^{-2+2^{-m}}) \;\geq\; 1 - \frac{2}{T^2}\sum_{k=0}^{T-1}\sum_{j=1}^{k}\frac{\left(c_1 j T^{-2+2^{-m}}\right)^2}{2} \;\geq\; 1 - \frac{c_1^2}{12}T^{-2+2^{-m+1}}.$$

For the second part, by definition,

$$f_T(c_1 T^{-2+2^{-m}}) - f_T(c_2 T^{-2+2^{-m}}) = \frac{2}{T^2}\sum_{k=0}^{T-1}\sum_{j=1}^{k}\cos\left(c_1 j T^{-2+2^{-m}}\right) - \cos\left(c_2 j T^{-2+2^{-m}}\right).$$

As $\cos(c_1 x) - \cos(c_2 x) \geq \frac{1}{4}(c_2^2 - c_1^2)x^2$ for $c_2 \leq 2$ and $\sum_{k=0}^{T-1}\sum_{j=1}^{k} j^2 \geq \frac{T^4}{24}$,

$$\begin{aligned} f_T(c_1 T^{-2+2^{-m}}) - f_T(c_2 T^{-2+2^{-m}}) \;&\geq\; \frac{2}{T^2}\sum_{k=0}^{T-1}\sum_{j=1}^{k}\frac{1}{4}\left(c_2^2 - c_1^2\right)\left(jT^{-2+2^{-m}}\right)^2 \\ &\geq\; \frac{c_2^2 - c_1^2}{48}T^{-2+2^{-m+1}}. \end{aligned} \tag{6.3}$$

From the Taylor expansion of f_T, we infer that $f_T(x) \leq 1 + \frac{1}{12}(1-T^2)x^2 + \frac{T^4}{360}x^4$, so

$$f_T(c_1 T^{-2+2^{-m}}) - \left(1 - \frac{c_1^2}{12}T^{-2+2^{-m+1}}\right) \;\leq\; \frac{c_1^2}{12}\left(T^{-2+2^{-m}}\right)^2 + \frac{c_1^4}{360}\left(T^{-1+2^{-m}}\right)^4 \quad (6.4)$$

For $m \geq 1$, the right hand side of (6.4) is $o(T^{-2+2^{-m+1}})$ (w.r.t. T). Thus, for large enough T,

$$f_T(c_1 T^{-2+2^{-m}}) - \left(1 - \frac{c_1^2}{12}T^{-2+2^{-m+1}}\right) \;\leq\; \frac{1}{2}\cdot\frac{c_2^2 - c_1^2}{48}T^{-2+2^{-m+1}},$$

which considering (6.3), implies that

$$f_T(c_2 T^{-2+2^{-m}}) \;\leq\; 1 - \left(\frac{c_1^2}{12} + \frac{c_2^2 - c_1^2}{96}\right)T^{-2+2^{-m+1}}.$$

For $m = 0$, (6.4) becomes:

$$f_T(c_1) - \left(1 - \frac{c_1^2}{12}\right) \;\leq\; \frac{c_1^2 T^{-2}}{12} + \frac{c_1^4}{360} \;\leq\; \frac{c_1^2 T^{-2}}{12} + \frac{c_1^2}{90},$$

and for large enough T, $f_T(c_1) \leq 1 - \frac{5c_1^2}{72}$. Therefore, following (6.3) and the fact that $c_2^2 \geq 3c_1^2$, we obtain:

$$\begin{aligned} f_T(c_2) \;&\leq\; f_T(c_1) - \frac{c_2^2 - c_1^2}{48} \;\leq\; 1 - \left(\frac{c_1^2}{12} + \frac{3c_2^2 - 5c_1^2}{144}\right) \\ &\leq\; 1 - \left(\frac{c_1^2}{12} + \frac{c_2^2 - c_1^2}{72} + \frac{c_2^2 - 3c_1^2}{144}\right) \;\leq\; 1 - \left(\frac{c_1^2}{12} + \frac{c_2^2 - c_1^2}{96}\right). \end{aligned}$$ □

The last property we required is norm preservation. That is, given that A satisfies our mentioned properties and has a bounded infinity norm (let us say by 1), there exists a universal constant C so that $\|f_{T,\lambda}(A)\|_\infty \leq C$ for every T and $\lambda \in [0,1]$. The Féjer kernel does not satisfy the norm preservation, at least not without imposing further restrictions on A, as we show in Section 6.2.

6.1 The multistage framework with the Féjer kernel

For the role of p_1, the gap expanding polynomial, consider the family of functions $\{f_{\lambda,T}\}_{\lambda\in[0,1],T\in\mathbb{N}}$ such that $\lambda \in [0,1]$, $f_{T,\lambda}(B) = f_T(B - \lambda I_{n\times n})$ for every $\lambda \in [0,1]$, $T = O(\log n)$ and f_T is the Féjer kernel. Recall that $\|B\|_\infty \leq 1$. We further restrict B to satisfy the following assumption.

Assumption 6.4. There exists an explicit universal constant C such that for every $\lambda \in [0,1]$, $\|f_{T,\lambda}(B)\|_\infty \leq C$.

We begin with some intuition. Assume we are given a promise on the eigenvalues' separation around λ. Specifically:

- There are d eigenvalues λ_i of B satisfying $|\lambda - \lambda_i| \leq \frac{1}{4}T^{-1}$.
- All other eigenvalues λ_i satisfy $|\lambda - \lambda_i| \geq 2T^{-1}$.

Following Lemmas 6.2 and 6.3, $f_{T,\lambda}$ has a peak around λ and it increases the gap in such a way that there exist positive constants $c_1 > c_2$ so that:

- There are d eigenvalues λ_i of $f_{T,\lambda}(B)$ satisfying $\lambda_i \geq c_1$.
- All other eigenvalues λ_i satisfy $\lambda_i \leq c_2$.

Indeed, the problem of finding the number of eigenvalues of B around λ reduces to finding the number of eigenvalues of $f_{T,\lambda}(B)$ above a certain threshold. Moreover, we expanded the gap from $O(T^{-1})$ to a constant. Note that the tradeoff is that we cannot necessarily approximate $f_{T,\lambda}(B)$ to high powers. Therefore, we continue working with $C^{-1}f_{T,\lambda}(B)$ instead. This matrix *is* simulatable, so the above promise problem can be solved by the technique presented in Chapter 4.

Remark. In fact, given a constant $m \geq 1$, constant number of compositions can expand a gap of $O(T^{-2+2^{-m}})$ to a constant gap by considering the following matrix:

$$\mathcal{G}_{\lambda,T,m}(B) \;=\; C^{-1}f_{T,C^{-1}}\left(C^{-1}f_{T,C^{-1}}\left(\cdots\; C^{-1}f_{T,C^{-1}}\left(C^{-1}f_{T,\lambda}(B)\right)\cdots\right)\right)$$

where $f_{T,C^{-1}}$ is iterated m times. By slightly strengthening Assumption 6.4, $\mathcal{G}_{\lambda,T,m}(B)$ is simulatable and its eigenvalues around 1 are in one-to-one correspondence with the eigenvalues of B around λ. However, for simplicity, we only concern the case of $m = 1$.

Lemma 6.5. *Given $\lambda \in [0,1]$, $T > 0$ and constants $0 < \alpha < \beta \leq 2$ satisfying[1] $\beta \geq 7\alpha$, assume the following holds:*

- *There are d eigenvalues λ_i of B satisfying $|\lambda - \lambda_i| \leq \alpha T^{-1}$.*
- *All other eigenvalues satisfy $|\lambda - \lambda_i| \geq \beta T^{-1}$.*

Then, for some positive constants $\gamma_1 > \gamma_2$, there are d eigenvalue λ_i of $f_{T,\lambda}(B)$ satisfying $\lambda_i \geq \gamma_1$ and all other eigenvalues satisfy $\lambda_i \leq \gamma_2$.

Proof. From Lemma 6.2 and Lemma 6.3, there are d eigenvalues λ_i of $f_{T,\lambda}(B)$ satisfying $\lambda_i \geq f_T(\alpha T^{-1}) \geq 1 - \frac{\alpha^2}{12}$. Also, following Lemma 6.3, there are no eigenvalues satisfying

$$1 - \left(\frac{\alpha^2}{12} + \frac{\beta^2 - \alpha^2}{96}\right) \leq \lambda_i < 1 - \frac{\alpha^2}{12},$$

as desired. [2] □

[1] For the simplicity of the proofs, we do not attempt to optimize the constants' constraints.

[2] Also, note that $\frac{\alpha^2}{12} + \frac{\beta^2-\alpha^2}{96} \Big/ \frac{\alpha^2}{12} = \frac{7\alpha^2+\beta^2}{8\alpha^2} \geq 7$, so multiple compositions can be made, expanding smaller gaps.

We shall now state that the above transformation can be approximated probabilistically in small space. Its proof follows directly from Claim 6.1, Assumption 6.4 and Lemma 3.4.

Lemma 6.6. *Let B be a matrix of dimension n that satisfies Assumption 6.4, $T = O(\log n)$ and $\lambda \in [0,1]$. Then, $C^{-1} f_{\lambda,T}(B)$ is simulatable.*

The analogue of the above lemma for constant number of compositions (see the above remark) uses Lemma 2.2, that assures a truncated fixed approximation for every computation level, with high probability. Following the above discussion, we can now prove our main theorem for this chapter.

Theorem 6.1. *There exists a probabilistic algorithm that on input B and $\lambda \in [0,1]$ such that B is as above (specifically, satisfying Assumption 6.4), and*

- *There are d eigenvalues λ_i satisfying $|\lambda - \lambda_i| \leq \frac{1}{4}\log^{-1} n$.*
- *All other eigenvalues λ_i satisfy $|\lambda - \lambda_i| \geq 2\log^{-1} n$.*

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

Proof. The eigenvector basis of $X = C^{-1} f_{T,\lambda}(B)$ is the same as this of B, as eigenvectors are preserved under matrix polynomials (and, due to Cayley-Hamilton, every matrix function representable as an infinite series corresponds to a finite degree polynomial). Thus, following the properties of the Féjer kernel, X satisfies the conditions of Theorem 4.1. Also, it is easy to see that the proof of Lemma 6.5 determines the exact computation (in logarithmic space) of the pre-assumed α and β.

In what follows, we use the notations of Chapter 4. As the algorithm of Theorem 4.1 requires reading the entries of X multiple times, we use Lemma 2.2, that implies the existence of a matrix X' whose entries are computable in $O(\log n)$ space, such that with probability at least $5/6$, $\|X' - X\|_\infty \leq n^{-2} 2^{-3s(n)}$, regardless of the random coin tosses. We can also safely assume that $\|X'\|_\infty \leq 1$ (otherwise, simply truncate the entries, resulting in a controllable loss of accuracy).

Consider running the algorithm guaranteed from Chapter 4 with $\delta = n^{-2}/2$ and $\varepsilon = 1/6$ on X', obtaining $R(X', \lambda)$. We infer that

$$\begin{aligned} |R(X',\lambda) - R(X,\lambda)| &\leq |\mathrm{Tr}\,(p(X') - p(X))| + 2n\delta \\ &\leq n \cdot 2^{3s(n)} \|X - X'\|_\infty + 2n\delta \leq 2n^{-1}, \end{aligned}$$

so running the algorithm on X' returns, with high probability, the same result as running the algorithm with X itself, hence returning the number of eigenvalues in the given range with probability at least $2/3$, as desired. Also, the space complexity needed to compute X,X' and $R(X', \lambda)$ is bounded by $O(\log n)$. □

6.2 The Féjer kernel blows up the ℓ_∞ norm

We argue that the transition matrix of the hypercube graph does not satisfy Assumption 6.4. However this result, amongst other numerical trials that are not brought here, does not dismiss the possibility that more restricted class of matrices, such as transition matrices of undirected regular graphs of constant degree, does satisfy the assumption.

In what follows, we use definitions and notations from the theory of Cayley graphs and discrete Fourier analysis. See [Tre11] and [dW08] for the exact definitions. Also, we denote e_k be a vector with 1 in the k-th entry and 0 elsewhere (its dimension will be clear from the context).

For every integer $d \geq 0$, let G be the group $\{0,1\}^d$ with the bitwise XOR operations. Let $Q_d = Cay(G, \{e_1, \ldots, e_d\})$ be the corresponding Cayley graph, also what is known as the hypercube graph. Namely, the graph whose vertices correspond to elements of $\{0,1\}^d$ and two vertices are adjacent if and only if they differ in exactly one coordinate. Set $n = 2^d$ and define A to be the $n \times n$ transition matrix of Q_d.

It is well known that A has a full set of eigenvectors, corresponding to the n characters of G. That is, for every $r, x \in \{0,1\}^d$,

$$v_r[x] = \frac{1}{\sqrt{n}}\chi_r(x) = \frac{1}{\sqrt{n}}(-1)^{\sum_{i=1}^d r_i x_i},$$

where we map elements between $\{0,1\}^d$ and $[n]$ in the obvious way. Correspondingly, the eigenvalues of A satisfy

$$\lambda_r = \frac{1}{d}\sum_{i=1}^d \chi_r(e_i) = 1 - 2\frac{|r|}{d}.$$

Therefore, $A = \sum_{r\in\{0,1\}^d} \lambda_r v_r v_r^\dagger$ so $f_T(A)$, where f_T is the Féfer kernel[3], is given by

$$f_T(A) = \sum_{r\in\{0,1\}^d} f_T(\lambda_r) v_r v_r^\dagger.$$

For every $r \in \{0,1\}^d$, let $g(r) = f_T(1 - 2|r|/d)$ (the dependency on T becomes implicit). Then, we can consider g to be a function from $\{0,1\}^d$ to $[0,1]$ and it is also immediate that g is *symmetric*, i.e. its value does not change under permutation of indices. The infinity norm of $f_T(A)$ is given by:

$$\|f_T(A)\|_\infty = \max_{b\in[n]} \sum_{a\in[n]} \left| \sum_{r\in\{0,1\}^d} g(r) e_b^\dagger v_r v_r^\dagger e_a \right| = \max_{b\in[n]} \sum_{a\in[n]} \left| \sum_{r\in\{0,1\}^d} g(r) v_r[b] v_r[a] \right|.$$

However, note that $\sum_a |\sum_r g(r) v_r[a+b]|$ is the same for every $b \in [n]$ (as different

[3] We consider f_T rather than $f_{T,\lambda}$ for simplicity. The results are the same.

values of b correspond to permutations of the first row), so it is sufficient to take

$$\|f_T(A)\|_\infty = \sum_{a\in\{0,1\}^d} \left| \frac{1}{2^d} \sum_{r\in\{0,1\}^d} g(r)\chi_r(a) \right| = \sum_{a\in\{0,1\}^d} |\hat{g}(a)|,$$

where $\hat{g}(a)$ is the Fourier coefficient. Hence, if we define a vector of all coefficients $\hat{g}$, $\|f_T(A)\|_\infty = \|\hat{g}\|_1$, also known as the *spectral norm* of the function g.

Although no results concerning the spectral norm of symmetric functions over *continuous* ranges are known, [4] numerical experiments suggests that in our case, $\|\hat{g}\|_1 = \Omega(d)$, implying that the hypercube graph is a counterexample for our assumption.

[4] Useful bounds on the spectral norm are known when the range of g is *Boolean*. In [AFH12], the following Theorem is proven:

Claim 6.7. *[AFH12, Theorem 1.1] For a function $g : \{0,1\}^d \to \{-1,1\}$, let r_0 and r_1 be the minimum integers less than $d/2$ such that $f(x)$ or $f(x) \cdot \textsc{Parity}(x)$ is constant for x with $|x| \in [r_0, n - r_1]$. Define $r(g) = \max r_0, r_1$. Then, for any symmetric function $g : \{0,1\}^d \to \{-1,1\}$, we have*

$$\log \|\hat{g}\|_1 = \Theta\left(r(f) \log \frac{d}{r(f)} \right)$$

wherever $r(f) > 1$.

Chapter 7

Further directions

The results of Chapter 6 give rise to the following question.

Question 7.1. Given an Hermitian matrix A of dimension n with eigenvalues that are well-separated and in $[0,1]$, and an integer $T = \text{poly}(n)$, can the entries of e^{iTA} be approximated in $O(\log n)$ space, by a probabilistic algorithm, with high probability?

Or, more generally,

Question 7.2. Can any unitary matrix be simulated?

If the answer to either question is affirmative, then $f_{T,\lambda}(A)$ can be approximated for every $\lambda \in [0,1]$ and T that is $\text{poly}(n)$. Then we could use the following claim from [KLM07, Chapter 7].

- For every $x \in [-1,1]$ such that $|x| \leq \frac{\pi}{T}$, $f_T(x) \geq \frac{4}{\pi^2}$.
- For every $x \in [-1,1]$ such that $|x| \geq \frac{2\pi}{\sqrt{T}}$, $f_T(x) \leq \frac{1}{2(\sqrt{T}-1)}$.

Then, even for promise parameters α and β that are polynomially small we can use the algorithm of Chapter 4 in order to extract the eigenvalues of A using Tr $(f_{T,\lambda}(A))$ with $T = \text{poly}(n)$.

Bibliography

[AFH12] Anil Ada, Omar Fawzi, and Hamed Hatami. Spectral norm of symmetric functions. *CoRR*, abs/1205.5282, 2012.

[AKL+79] 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.

[BOE13] Michael Ben-Or and Lior Eldar. Optimal algorithms for linear algebra by quantum inspiration. *CoRR*, abs/1312.3717, 2013.

[BPW11] Andrej Bogdanov, Periklis A Papakonstaninou, and Andrew Wan. Pseudorandomness for read-once formulas. In *Foundations of Computer Science (FOCS), 2011 IEEE 52nd Annual Symposium on*, pages 240–246. IEEE, 2011.

[BRRY14] Mark Braverman, Anup Rao, Ran Raz, and Amir Yehudayoff. Pseudorandom generators for regular branching programs. *SIAM Journal on Computing*, 43(3):973–986, 2014.

[Coo85] Stephen A Cook. A taxonomy of problems with fast parallel algorithms. *Information and control*, 64(1):2–22, 1985.

[Csa76] L. Csansky. fast parallel matrix inversion algorithms. *SIAM Journal of Computing*, 5(6):618–623, 1976.

[De11] Anindya De. Pseudorandomness for permutation and regular branching programs. In *Computational Complexity (CCC), 2011 IEEE 26th Annual Conference on*, pages 221–231. IEEE, 2011.

[DTS14] Dean Doron and Amnon Ta-Shma. On determinstic approximation of the permanent, and the relationship between BQL, BPL and L. Manuscript, 2014.

[dW08] Ronald de Wolf. *A Brief Introduction to Fourier Analysis on the Boolean Cube*. Number 1 in Graduate Surveys. Theory of Computing Library, 2008.

[GVL96] Gene H. Golub and Charles F. Van Loan. *Matrix Computations*. Johns Hopkins University Press, Baltimore, MD, USA, 3rd edition, 1996.

[HHL09] Aram W. Harrow, Avinatan Hassidim, and Seth Lloyd. Quantum algorithm for linear systems of equations. *Phys. Rev. Lett.*, 103:150502, Oct 2009.

[Hig08] Nicholas J. Higham. *Functions of Matrices: Theory and Computation*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2008.

[Hof07] K. Hoffman. *Banach Spaces of Analytic Functions*. Dover Books on Mathematics Series. Dover Publications, Incorporated, 2007.

[HS65] Edwin Hewitt and Karl Stromberg. *Real and Abstract Analysis*. Springer Berlin Heidelberg, New York, NY, USA, 1st edition, 1965.

[IMZ12] Russell Impagliazzo, Raghu Meka, and David Zuckerman. Pseudorandomness from shrinkage. In *Foundations of Computer Science (FOCS), 2012 IEEE 53rd Annual Symposium on*, pages 111–119. IEEE, 2012.

[IW97] 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.

[JSV04] 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.

[KI04] Valentine Kabanets and Russell Impagliazzo. Derandomizing polynomial identity tests means proving circuit lower bounds. *Computational Complexity*, 13(1-2):1–46, 2004.

[KLM07] P. Kaye, R. Laflamme, and M. Mosca. *An Introduction to Quantum Computing*. Oxford University Press, 2007.

[NC00] M.A. Nielsen and I.L. Chuang. *Quantum Computation and Quantum Information*. Cambridge Series on Information and the Natural Sciences. Cambridge University Press, 2000.

[Nis92] Noam Nisan. Pseudorandom generators for space-bounded computation. *Combinatorica*, 12(4):449–461, 1992.

[Rei08] Omer Reingold. Undirected connectivity in log-space. *J. ACM*, 55(4), 2008.

[Riv74] T.J. Rivlin. *The Chebyshev polynomials*. Pure and applied mathematics. Wiley, 1974.

[RSV13] Omer Reingold, Thomas Steinke, and Salil Vadhan. Pseudorandomness for regular branching programs via fourier analysis. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 655–670. Springer, 2013.

[SVW14] Thomas Steinke, Salil Vadhan, and Andrew Wan. Pseudorandomness and fourier growth bounds for width 3 branching programs. *arXiv preprint arXiv:1405.7028*, 2014.

[SZ99] Michael E. Saks and Shiyu Zhou. $\mathrm{BP_HSPACE(S)} \subseteq \mathrm{DSPACE}(\mathrm{S}^{3/2})$. *J. Comput. Syst. Sci.*, 58(2):376–403, 1999.

[Tim63] A.F. Timan. *Theory of Approximation of Functions of a Real Variable*. Dover books on advanced mathematics. Pergamon Press, 1963.

[Tre11] Luca Trevisan. *Graph Partitioning and Expanders: Lecture Notes*. Stanford University, Jan 2011.
Available: https://resources.mpi-inf.mpg.de/conferences/adfocs-12/Trevisan.pdf.

[TS13] 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.

תקציר

מחקרים רבים העוסקים באלגוריתמים חסומי־זכרון מנסים להראות כי חישוב הסתברותי בזכרון לוגריתמי הוא חלש, במובן שניתן לבצע עבורו דרנדומיזציה מלאה ובכך להגיע לאלגוריתם דטרמינסטי שקול הרץ בזכרון לוגריתמי גם כן. בעבודה זו אנו לוקחים צעד בכיוון אחר. במקום לשאול "הכיצד ניתן לבצע דרנדומיזציה לאלגוריתמים הסתברותיים חסומי־זכרון?", אנו שואלים "הכיצד ניתן לנצל את הכוח ההסתברותי שברשותינו?".

אנו מתמקדים בבעיה של קירוב ערכים עצמיים של אופרטורים סטוכסטיים והרמיטיים. באם בעיה זו תפתר תוך דיוק פולינומי־קטן, נוכל לומר כי ניתן למעשה, עבור אופרטורים אלו, לבצע את כל קירובי האלגברה הלינארית ב־ BPL. במחקר זה אנו מגיעים לקירובים שהם עד כדי קבוע קטן כרצונינו. הטכניקה שמוצגת היא חדשה (למיטב ידיעתנו). אנו מראים גם כי לעבור את מחסום הדיוק הקבוע ידרוש בהכרח רעיון חדש.

אין לנו עדיין השערה לגבי התכנותו של קירוב טוב יותר, ואנו מאמינים כי פתרון הבעיה הזו הוא חשוב, ועשוי לשפוך אור חדש על החוזק והחולשה של המודל ההסתברותי חסום הזכרון. אנו מקווים כי עבודה זו תלבה כיוון זה של מחקר.

תודות

בראש ובראשונה ארצה להביע את הערכתי הרבה למנחה שלי, פרופ' אמנון תא־שמע שבסבלנותו, נלהבותו ותובנותיו הרבות איפשר את עבודה זו. אני זוקף לזכות השקעתו ועידודו את יכולתי המתהווה לחשוב בצלילות ולחפש אחר הרעיונות היסודיים.

ארצה להודות במיוחד למשפחתי על תמיכתם התמידית. בנוסף, הוקרה רבה מגיעה לקבוצה קטנה אך יוצאת דופן של חברים, אשר עמדו ועדיין עומדים לצידי.

הכרת תודה מסורה לאוניברסיטת תל־אביב על מימון מחקר זה.

אוניברסיטת תל-אביב TEL AVIV UNIVERSITY
הפקולטה למדעים מדויקים ע"ש ריימונד ובברלי סאקלר
בית הספר למדעי המחשב ע"ש בלבטניק

קירוב הסתברותי של ערכים עצמיים של גרפים לא מכוונים בזכרון לוגריתמי

חיבור על מחקר

לשם מילוי חלקי של הדרישות לקבלת התואר
מגיסטר למדעים במדעי המחשב

מאת
דין דורון

עבודה זו נכתבה תחת הנחייתו של
פרופ' אמנון תא־שמע

יולי 2014

אוניברסיטת תל-אביב TEL AVIV UNIVERSITY
הפקולטה למדעים מדויקים ע"ש ריימונד ובברלי סאקלר
בית הספר למדעי המחשב ע"ש בלבטניק

קירוב הסתברותי של ערכים עצמיים של גרפים לא מכוונים בזכרון לוגריתמי

דין דורון