Sublinear Time Spectral Density Estimation

Vladimir Braverman
Johns Hopkins University
vova@cs.jhu.edu

Aditya Krishnan
Johns Hopkins University
akrish23@jhu.edu

Christopher Musco
New York University
cmusco@nyu.edu

April 18, 2022

Abstract

We present a new sublinear time algorithm for approximating the spectral density (eigenvalue distribution) of an $n \times n$ normalized graph adjacency or Laplacian matrix. The algorithm recovers the spectrum up to ϵ accuracy in the Wasserstein-1 distance in $O(n \cdot \text{poly}(1/\epsilon))$ time given sample access to the graph. This result compliments recent work, which obtains a solution with runtime independent of n, but exponential in $1/\epsilon$ [CKSV18]. We conjecture that the trade-off between dimension dependence and accuracy is inherent.

Our method is simple and works well experimentally. It is based on a Chebyshev polynomial moment matching method that employees randomized estimators for the matrix trace. We prove that, for any Hermitian A, this moment matching method returns an ϵ approximation to the spectral density using just $O(1/\epsilon)$ matrix-vector products with A. By leveraging stability properties of the Chebyshev polynomial three-term recurrence, we then prove that the method is amenable to the use of coarse approximate matrix-vector products. Our sublinear time algorithm follows from combining this result with a novel sampling algorithm for approximating matrix-vector products with a normalized graph adjacency matrix.

Of independent interest, we show a similar result for the widely used *kernel polynomial method* (KPM), proving that this practical algorithm nearly matches the theoretical guarantees of our moment matching method. Our analysis uses tools from Jackson's seminal work on approximation with positive polynomial kernels [Jac12].

1 Introduction

A ubiquitous task in computational science, engineering, and data science is to extract information about the eigenvalue spectrum of a matrix $A \in \mathbb{R}^{n \times n}$. A full eigendecomposition takes at least $O(n^\omega)$ time[1], which is prohibitively expensive for large matrices [Par98, BVKS19]. So, we are typically interested in extracting *partial information* about the spectrum. This can be done using iterative methods like the power or Lanczos methods, which access A via a small number of matrix-vector multiplications. Each multiplication takes at most $O(n^2)$ time to compute, and can be accelerated when A is sparse or structured, leading to fast algorithms.

However, the partial spectral information computed by most iterative methods is limited. Algorithms typically only obtain accurate approximations to the outlying, or *largest magnitude* eigenvalues of A, missing information about the *interior* of A's spectrum that may be critical in applications.

[1] Here $\omega < 2.373$ is the fast matrix multiplication exponent.

For example, in network science, clusters of interior eigenvalues can indicate graph structures like repeated motifs [DBB19]. In deep learning, information on how the spectrum of a weight matrix differs from its random initialization can give hints about model convergence and generalization [PSG18, MM19], and Hessian eigenvalues are useful in optimization [GKX19]. Coarse information about interior eigenvalues is also used to initialize parallel GPU based methods for full eigendecomposition [AKS17, LXES19].

To address these needs and many other applications, there has been substantial interest in methods for estimating the full *spectral density* of a matrix A [WWAF06]. Concretely, assume that A is Hermitian with real eigenvalues $\lambda_1, \ldots, \lambda_n$. We view its spectrum as a probability density s:

$$\text{Spectral density:} \qquad s(x) = \frac{1}{n}\sum_{i=1}^{n} \delta(x - \lambda_i). \tag{1}$$

Here δ is the Dirac delta function. The goal is to find a probability density q that approximates s in some natural metric, like the Wasserstein distance. The density q can either be continuous (represented in some closed form) or discrete (represented as a list of approximate eigenvalues $\tilde{\lambda}_1, \ldots, \tilde{\lambda}_n$). See Figure 1 for an illustration. Both sorts of approximation are useful in applications.

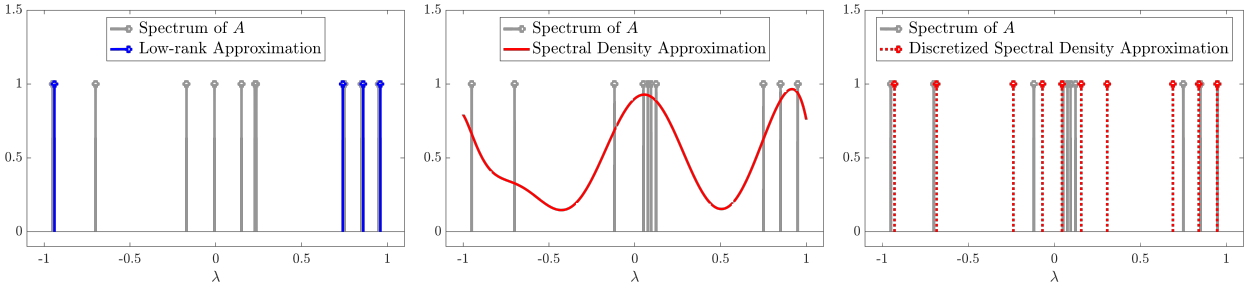

Figure 1: Different approximations for the spectrum of a matrix A with eigenvalues in $[-1, 1]$. A typical approximation computed using an iterative eigenvalue algorithm mostly preserves information about the largest magnitude eigenvalues. In contrast, the spectral density estimates in the two right figures coarsely approximate the entire distribution of A's eigenvalues, the first with a low-degree polynomial, and the second with a discrete distribution.

Methods for spectral density estimation that run in $o(n^\omega)$ time were first introduced for applications in condensed matter physics and quantum chemistry [Ski89, SR94, Wan94]. Many are based on the combination of two important tools: 1) moment matching, and 2) stochastic trace estimation. Specifically, if we had access to moments of the distribution s, i.e. $\frac{1}{n}\sum_{i=1}^n \lambda_i$, $\frac{1}{n}\sum_{i=1}^n \lambda_i^2$, $\frac{1}{n}\sum_{i=1}^n \lambda_i^3$, etc., then we could find a good approximation q by finding a distribution that agrees with s on these moments. Moreover, these *spectral moments* can be computed via the matrix trace: note that $\text{tr}(A) = \sum_{i=1}^n \lambda_i$, $\text{tr}(A^2) = \sum_{i=1}^n \lambda_i^2$, $\text{tr}(A^3) = \sum_{i=1}^n \lambda_i^3$, etc. While we cannot hope to compute $\text{tr}(A^k)$ exactly in $o(n^\omega)$ time, thanks to stochastic trace estimators like Hutchinson's method, this trace can be approximated much more quickly [Hut90, AT11]. Such estimators are based on the observation that, for any matrix $B \in \mathbb{R}^{n\times n}$, $\text{tr}(B)$ can be well approximated by $\text{tr}(G^T BG)$ where $G \in \mathbb{R}^{n\times m}$ contains random sub-Gaussian entries and $m \ll n$. For any k degree polynomial g, $G^T g(A)G$ can be computed with just $O(km)$ matrix-vector multiplications, so we can quickly approximate any low-degree moment of A's spectral density.

While this high-level approach and related techniques have been applied successfully to estimating the spectra of a wide variety of matrices [WWAF06, LSY16], theoretical guarantees have only

appeared relatively recently. Perhaps surprisingly, it can be shown that many common methods provably run in *linear time* for any Hermitian matrix A. For instance, in work concurrent to ours, Chen, Trogdan, and Ubaru [CTU21] show that for any $n \times n$ Hermitian matrix A with spectral density s, the popular Stochastic Lanczos Quadrature (SLQ) method provably computes an approximate spectral density q satisfying:

$$W_1(s, q) \leq \epsilon \tag{2}$$

using just $\text{poly}(1/\epsilon)$ matrix-vector multiplications with A. Above W_1 denotes the Wasserstein-1 distance, aka the "earth-movers distance".[2] We defer a formal definition of W_1 to Section 2. The measure is convenient because, unlike many other measures of statistical distance, it allows a discrete distribution like the spectral density to be meaningfully compared to a possibly continuous approximation. For discrete approximations, the Wasserstein distance is related to a simple ℓ_1 metric. If we let $\Lambda = [\lambda_1, \ldots, \lambda_n]$ be a vector of A's eigenvalues and $\tilde{\Lambda} = [\tilde{\lambda}_1, \ldots, \tilde{\lambda}_n]$ be a vector of approximate eigenvalues, then $\|\Lambda - \tilde{\Lambda}\|_1 \leq n\epsilon$ if and only if $W_1(s, q) \leq \epsilon$ for the discrete spectral density q with eigenvalues in $\tilde{\Lambda}$.

As a step towards our main sublinear time result, in this work we show that similar bounds to [CTU21] can also be proven for the popular kernel polynomial method (KPM) [WWAF06] and for a natural moment matching algorithm based on Chebyshev polynomials.

1.1 Our contributions

With linear time spectral density estimation algorithms in hand for all Hermitian matrices, a natural question is if we can go faster for specific classes of matrices. In particular, there has been growing interest in SDE algorithms for graph structured matrices like adjacency matrices and Laplacians [DBB19]. A remarkable recent result by Cohen et al. [CKSV18] shows that, for normalized graph adajeceny matrices, it is possible to achieve guarantee (2) in $2^{O(1/\epsilon)}$ time, given appropriate query access to the target graph. Importantly, this runtime *does not depend on* n. However, given the exponential dependence on ϵ, the algorithm is impractical even for coarse spectral approximations.

Our main contribution is a method that obtains a *polynomial* dependence on ϵ, at the cost of a linear dependence on the matrix dimension n. Since A can have n^2 non-zero entries, the runtime is still *sublinear* in the problem size, but with a much more acceptable dependence on accuracy.

Theorem 1.1 (Sublinear time spectral density estimation for graphs.)**.** *Let $G = (V, E)$ be an unweighted, undirected n-vertex graph and let $A \in \mathbb{R}^{n \times n}$ be the normalized adjacency of G with spectral density s. Let $\epsilon, \delta \in (0, 1)$ be fixed values. Assume that we can 1) uniformly sample a random vertex in constant time, 2) uniformly sample a random neighbor of any vertex $i \in V$ in constant time, and 3) for a vertex i with degree d_i, read off all neighbors in $O(d_i)$ time.*[3] *Then there is a randomized algorithm with expected running time $O(n \, \text{poly}(\log(1/\delta)/\epsilon))$ which outputs a density function $q : [-1, 1] \to \mathbb{R}^+$ such that $W_1(q, s) \leq \epsilon$ with probability at least $1 - \delta$.*

Note that the normalized graph Laplacian $L = I - A$ has the same eigenvalues as A up to a shift

[2] We assume $\|A\|_2 \leq 1$ for simplicity of stating errror guarantees, noting that Wasserstein distance is not scale invariant. This assumption is without loss of generality since $\|A\|_2$ can always be scaled after computing the top eigenvector up to constant fact accuracy, which takes just $O(\log n)$ matrix-vector multiplications [MM15].

[3] A standard adjacency list representation of the graph would support these operations. As discussed in Section 5, assumption (3) can be eliminated at the cost of an extra $\log n$ in the runtime as long as we know vertex degrees.

and reflection, so Theorem 1.1 also yields a sublinear time result for normalized Laplacians, whose spectral densities are of interest in network science [DBB19].

Robust spectral density estimation

Theorem 1.1 is proven in Section 5. A key component of the result is a sublinear time routine for computing coarse approximate matrix-vector products with any normalized graph adjacency matrix. To make use of such a routine, we need to develop an SDE algorithm that is *robust* to the use of an approximate matrix-vector oracle. This is one of the main contributions of our work, as previous methods assume exact matrix-vector products. Formally, we assume access to the oracle:

Definition 1.2. An ϵ_{MV}*-approximate matrix-vector multiplication oracle* for $A \in \mathbb{R}^{n\times n}$ and error parameter $\epsilon_{\mathsf{MV}} \in (0,1)$ is an algorithm that, given any vector $y \in \mathbb{R}^n$, outputs a vector z such that $\|z - Ay\|_2 \leq \epsilon_{\mathsf{MV}} \|A\|_2 \|y\|_2$. We will denote a call to such an oracle for by $\mathrm{AMV}(A, y, \epsilon_{\mathsf{MV}})$.

In Section 4.2 we prove the following for any Hermitian matrix A (e.g., real symmetric) under the assumption that $\|A\|_2 \leq 1$, i.e., that A's eigenvalues lie in $[-1,1]$:

Theorem 1.3 (Robust spectral density estimation)**.** *Let $A \in \mathbb{R}^{n\times n}$ be a Hermitian matrix with spectral density s and $\|A\|_2 \leq 1$. Let C, C', C'' be fixed positive constants. For any $\epsilon, \delta \in (0,1)$ and $\epsilon_{MV} = C''\epsilon^{-3}\ln(1/\epsilon)$, there is an algorithm (Algorithm 1, with Algorithm 3 used as a subroutine to approximate moments) which makes $T = C\ell/\epsilon$ calls to an ϵ_{MV}-approximate matrix-vector oracle for A, where $\ell = \max\left(1,\ \frac{C'}{n}\epsilon^{-2}\log^2(\frac{1}{\epsilon\delta})\log^2(\frac{1}{\epsilon})\right)$, and in $\mathrm{poly}(1/\epsilon)$ additional runtime, outputs a probability density function $q : [-1,1] \to \mathbb{R}^{\geq 0}$ such that $W_1(s,q) \leq \epsilon$ with probability $1-\delta$.*

The requirement for the approximate matrix-vector oracle in Theorem 1.3 is relatively weak: we only need accuracy ϵ_{MV} that is polynomial in the final accuracy ϵ. Importantly, there is no dependence on $1/n$, which allows for the theorem to be combined with coarse AMV methods, including the one developed in Section 5 for normalized adjacency matrices. Based on random sampling, that method returns an ϵ-approximate matrix-vector multiply in $O(n/\epsilon^2)$ time. This immediately yields our result for graphs given by Theorem 1.1. We hope that Theorem 1.3 will find broader applications, since spectral density estimation is often applied to matrices where we only have inexact access to A. For example, A might be a Hessian matrix that we can multiply by approximately using stochastic approximation [Pea94, YGKM20], or the inverse of some other matrix, which we can multiply by approximately using an iterative solver.

We note that the result in Theorem 1.3 actually *improves* as n increases. Intuitively, when A is larger, each matrix-vector product returns more information about the spectral density s, so we can estimate it more easily. We also remark that the density function q returned by Algorithm 1 is in the form of an $O(1/\epsilon^3)$ dimensional vector, with the i-th entry corresponding to probability mass placed on the i-th point of an evenly spaced grid on $[-1,1]$. Alternatively, a simple rounding scheme that runs in $O(n + \mathrm{poly}(1/\epsilon))$ time can extract from q a vector of approximate eigenvalues $\tilde{\Lambda} = [\tilde{\lambda}_1, \ldots, \tilde{\lambda}_n]$ satisfying $\|\Lambda - \tilde{\Lambda}\|_1 \leq n\epsilon$, which, as discussed, is ϵ close to the spectral density s in Wasserstein distance (see Theorem B.1).

Our approach for density estimation is based on a moment matching method that approximates *Chebyshev polynomial* moments instead of the standard moments. I.e. we approximate $\mathrm{tr}(T_0(A))$, …, $\mathrm{tr}(T_N(A))$ where $T_0, \ldots, T_N$ are the Chebyshev polynomials of the first kind and then return a

distribution whose Chebyshev moments closely match our approximations. By leveraging Jackson's theorem on polynomial approximation of Lipschitz functions [Jac30], we show how to bound the Wasserstein distance between two distributions in terms of the magnitude of the differences between their first $N = O(1/\epsilon)$ Chebyshev moments (see Lemma 3.1). Unlike results for standard moments [KV17], the bound shows a near-linear relationship between Wasserstein distance and difference in the Chebyshev moments. Ultimately this allows us to obtain a polynomial dependence on ϵ in the number of approximate matrix-vector multiplications needed in Theorem 1.3.

Along the way to proving that theorem, in Section 4.1 we first establish the follow result that is compatible with exact matrix-vector multiplications:

Theorem 1.4 (Linear time spectral density estimation)**.** *Let $A \in \mathbb{R}^{n\times n}$ be a Hermitian matrix with spectral density s and $\|A\|_2 \leq 1$. Let C, C' be fixed positive constants. For any $\epsilon, \delta \in (0,1)$, there is an algorithm (Algorithm 1, with Algorithm 2 used as a subroutine to approximate moments) which computes $T = C\ell/\epsilon$ matrix-vector multiplications with A where $\ell = \max\left(1,\ \frac{C'}{n}\epsilon^{-2}\log^2(\frac{1}{\epsilon\delta})\log^2(\frac{1}{\epsilon})\right)$, and in $\mathrm{poly}(1/\epsilon)$ additional runtime, outputs a probability density function $q : [-1,1] \to \mathbb{R}^{\geq 0}$ such that $W_1(s,q) \leq \epsilon$ with probability $1-\delta$.*

As in Theorem 1.3, the theorem improves as n increases, requiring just $T = O(1/\epsilon)$ matrix vector multiplies when $n = \Omega(1/\epsilon^2)$. The runtime of Theorem 1.4 is dominated by the cost of the matrix-vector multiplications, which take $O(T \cdot n^2)$ time to compute for a dense matrix, and $O(T \cdot \mathrm{nnz}(A))$ time for a sparse matrix with $\mathrm{nnz}(A)$ non-zero entries, so the algorithm runs in linear time when ϵ, δ are considered constant.

Given Theorem 1.4, we prove Theorem 1.3 by showing that the error introduced by approximate matrix-vector multiplications does not hinder our ability to estimate the Chebyshev polynomial moments. We do so by drawing on stability results for the three-term recurrence relation defining these polynomials [Cle55, MMS18].

Remark. The number of matrix-vector multiplies $N\ell = N \cdot \max(1, \frac{C'}{n}\epsilon^{-2}\log^2(\frac{1}{\epsilon\delta})\log^2(\frac{1}{\epsilon}))$ in Theorems 1.3 and 1.4 can be improved by up to a $\log^2(1/\epsilon)$ factor in the regime when n is small, specifically $n \leq C'\epsilon^{-2}\log^2(1/(\epsilon\delta))$. This is discussed further in Section 4.

Spectral density estimation via the kernel polynomial method

In addition to the Chebyshev moment matching method used to give Theorem 1.4 and Theorem 1.3, we prove that a version of the popular kernel polynomial method (KPM) can be used to obtain a spectral density estimate with similar running times, albeit with slightly worse dependence on the accuracy parameter ϵ.[4] Along with the Stochastic Lanczos Quadrature method, the kernel polynomial method is one of two dominant spectrum estimation algorithms used in practice.

Given sufficiently accurate approximations to the Chebyshev polynomial moments, the KPM method outputs a density function q in the form of a $O(1/\epsilon)$ degree polynomial multiplied by a simple closed form function. This is described in Algorithm 6 in Section A.2 and should be thought of as analagous to Algorithm 1. Specifically, we can obtain Theorem 1.4 and Theorem 1.3

[4] We believe that the extra $O(\epsilon^{-2})$ factor in the number of matrix-vector multiplications (or calls to an approximate matrix-vector oracle in the robust setting) may be an artifact of our analysis and can be further improved to match the approximate Chebyshev moment matching bounds.

with $\ell = \max(1, \frac{C'}{n}\epsilon^{-4}\log^2(\frac{1}{\epsilon\delta}))$ and $\epsilon_{\mathsf{MV}} = C''\epsilon^{-4}$ (in the robust setting), by using Algorithm 6 instead of Algorithm 1. Our proof in the KPM case is again based on Jackson's work on polynomial approximations for Lipschitz functions: we take advantage of the fact that Jackson constructs approximations that are both *linear* and *preserve positivity* [Jac12].

1.2 Related work

As mentioned, most closely related to our sublinear time result on graphs is the result of Cohen et al. [CKSV18]. They prove a result which matches the guarantee of Theorem 1.1, but with runtime of $2^{O(1/\epsilon)}$ – i.e., with *no dependence* on n. In comparison, our result depends linearly on n, but only polynomially on $1/\epsilon$. An interesting open question is if a $\text{poly}(1/\epsilon)$ time algorithm is possible but we conjecture that the trade-off between the dependence on n and the accuracy ϵ is inherent. Our bound in Lemma 3.1 on the Wasserstein-1 distance between two distributions can be seen as analagous to Proposition 1 from [KV17], which is the basis of the result in [CKSV18]. They bound the Wasserstein-1 distance between two distributions in terms of the differences in the standard moments of the distributions. The bound requires an exponentially small dependence on $1/\epsilon$, i.e. $2^{-O(1/\epsilon)}$, in the difference between the standard moments while the bound from Lemma 3.1 only requires an $O(\epsilon/\ln(1/\epsilon))$ difference in the Chebyshev moments.

As discussed, algorithms for spectral density estimation have been studied since the early 90s [Ski89, SR94, Wan94] but only analyzed recently. In addition to the work of Chen, Trogdon, and Ubaru that was discussed [CTU21], [MNS+18] provides an algorithm for computing an approximate histogram for the spectrum of matrix. That result can be shown to yield an ϵ error approximation to the spectral density in the Wasserstein-1 distance with roughly $O(1/\epsilon^5)$ matrix-vector multiplications. This compares to the improved $O(1/\epsilon)$ matrix-vector multiplications required by our Theorem 1.4.

Matrix-vector query algorithms. Our work fits into a broader line of work on proving upper and lower bounds on the *matrix-vector query complexity* of linear algebraic problems, from top eigenvector, to matrix inversion, to rank estimation [SWYZ19, SEAR18, BHSW20, MMMW20, DM21]. The goal in this model is to minimize the total number of matrix-vector multiplications with A, recognizing that such multiplications either 1) dominate runtime cost or 2) are the only way to access A when it is an implicit matrix. The matrix-vector query model generalizes both classical Krylov subspace methods, as well as randomized sketching methods [Woo14]. Studying other basic linear algebra problem when matrix-vector multiplication queries are only assumed to be approximate (as in Definition 1.2) is an interesting future direction.

1.3 Paper Roadmap

We describe notation and preliminaries on polynomial approximation in Section 2. We use these tools in Section 3 to prove that a good approximation to the first $O(1/\epsilon)$ Chebyshev polynomial moments of the spectral density can be used to extract a good approximation in Wasserstein-1 distance. This result is the basis for our result on robust spectral density estimation stated in Theorem 1.4 and linear time spectral density estimation stated in Theorem 1.3, which are proven in Section 4. Finally, we give a randomized algorithm to implement an approximate matrix-vector multplication oracle for adjacency matrices in Section 5 and prove our main result, Theorem 1.1. In Section A we describe and analyze the kernel polynomial method, showing that it too can be used to obtain a spectral density estimate given approximations to the first $O(1/\epsilon)$ Chebyshev polynomial

moments. In Section 6, we empirically investigate the potential of combining approximate matrix-vector multiplications with our moment matching method, the kernel polynomial method, and the stochastic Lanczos quadrature method studied in [CTU21]. We show that all three can achieve accurate SDE estimates in sublinear time for a variety of graph Laplacians.

2 Preliminaries

Throughout we assume that $A \in \mathbb{R}^{n\times n}$ is Hermitian with eigendecomposition $A = U\Lambda U^*$, where $UU^* = U^*U = I_{n\times n}$. We assume that A's eigenvalues satisfy $-1 \leq \lambda_n \leq \cdots \leq \lambda_1 \leq 1$. In many applications A is real symmetric. We denote A's spectral density by s, which is defined in (1). Our goal is to approximate s in the Wasserstein-1 metric with another distribution q supported on $[-1, 1]$. Specifically, as per the dual formulation given by the Kantorovich-Rubinstein theorem [KR57], for s, q supported on $[-1, 1]$ the metric is equal to:

$$W_1(s, q) = \sup_{\substack{f:\mathbb{R}\to\mathbb{R} \\ |f(x)-f(y)|\leq|x-y| \ \forall x,y}} \left\{ \int_{-1}^{1} f(x)\left(s(x) - q(x)\right) dx \right\}. \tag{3}$$

In words, s and q are close in Wasserstein-1 distance if their difference has small inner product with all 1-Lipschitz functions f. Alternatively, $W_1(s, q)$ is equal to the cost of "changing" one distribution to another, where the cost of moving one unit of mass from x to y is $|x - y|$: this is the "earthmover's" formulation common in computer science. Note that (3) can be applied to arbitrary functions s, q, even if they are *not distributions*, and we will occasionally do so.

Functions and inner products. We introduce notation for functions used throughout the paper. Let $\mathcal{F}([-1, 1], \mathbb{R})$ denote the space of real-valued functions on $[-1, 1]$. For $g, h \in \mathcal{F}([-1, 1], \mathbb{R})$, let $\langle g, h\rangle$ denote $\langle g, h\rangle \coloneqq \int_{-1}^{1} g(x)h(x)dx$. For $f \in \mathcal{F}([-1, 1], \mathbb{R})$, we define $\|f\|_2 \coloneqq \sqrt{\langle f, f\rangle}$ and let $\|f\|_\infty$ denote the max-norm $\|f\|_\infty = \max_{x\in[-1,1]} |f(x)|$. We let $\|f\|_1$ denote $\|f\|_1 = \int_{-1}^{1} |f(x)|dx$.

Let $\mathcal{F}(\mathbb{Z}, \mathbb{R})$ be the space of real-valued functions on the integers, $\mathbb{Z}$. For $f, g \in \mathcal{F}(\mathbb{Z}, \mathbb{R})$ let $(f * g)$ denote the discrete convolution: $(f * g)[n] = \sum_{m=-\infty}^{\infty} f[m]g[n - m]$. Let $\mathcal{F}(\mathbb{N}, \mathbb{R})$ be the space of real-valued functions on the natural numbers, $\mathbb{N}$. For functions in $\mathcal{F}(\mathbb{Z}, \mathbb{R})$ or $\mathcal{F}(\mathbb{N}, \mathbb{R})$ we typically used square brackets instead of parentheses.

For two functions f, g let $h = fg$ (or $h = f \cdot g$) and $j = f/g$ denote the pointwise product and quotient respectively. I.e. $h(x) = f(x)g(x)$ and $j(x) = f(x)/g(x)$ for all x.

Chebyshev polynomials. Our approach is based on approximating Chebyshev polynomial moments of A's spectral density, and we will use basic properties of these polynomials, the k^{th} of which we denote T_k. The Chebyshev polynomial of the first kind can be defined via the recurrence:

$$\begin{aligned} T_0(x) &= 1 \qquad T_1(x) = x \\ T_k(x) &= 2x \cdot T_{k-1}(x) - T_{k-2}(x) \qquad \text{for } k \geq 2. \end{aligned}$$

We will use the well known fact that the Chebyshev polynomials of the first kind are bounded between $[-1, 1]$, i.e. $\max_{x\in[-1,1]} |T_k(x)| \leq 1$.

Let $w(x) \coloneqq \frac{1}{\sqrt{1-x^2}}$. It is well known that $\langle T_0, w \cdot T_0 \rangle = \pi$, $\langle T_k, w \cdot T_k \rangle = \pi/2$ for $k > 0$, and

$$\langle T_i, w \cdot T_j \rangle = 0 \qquad \text{for } i \neq j.$$

In other words, the Chebyshev polynomials are orthogonal on $[-1, 1]$ under the weight function w. The first k Chebyshev polynomials form an orthogonal basis for the degree k polynomials under this weight function. We let $\bar{T}_k$ denote the *normalized* Chebyshev polynomial $\bar{T}_k \coloneqq T_k / \sqrt{\langle T_k, w \cdot T_k \rangle}$.

Definition 2.1 (Chebyshev Series)**.** The *Chebyshev expansion or series* for a function $f \in \mathcal{F}([-1, 1], \mathbb{R})$ is given by

$$\sum_{k=0}^{\infty} \langle f, w \cdot \bar{T}_k \rangle \cdot \bar{T}_k.$$

We call $\sum_{k=0}^{N} \langle f, w \cdot \bar{T}_k \rangle \cdot \bar{T}_k$ the truncated Chebyshev expansion or series of degree N.

Other notation. Let $[n]$ denote $1, \ldots, n$. For a scalar function $f : \mathbb{R} \to \mathbb{R}$ and $n \times n$ matrix A with eigendecomposition $U \Lambda U^*$, we let $f(A)$ denote the matrix function $U f(\Lambda) U^*$. Here $f(\Lambda)$ is understood to mean f applied entrywise to the diagonal matrix Λ containing A's eigenvalues. Note that $\mathrm{tr}(f(A)) = \sum_{i=1}^{n} f(\lambda_i)$. When $f(x)$ is a degree q polynomial, $c_0 + c_1 x + \ldots, c_q x^q$, then we can check that $f(A)$ exactly equals $c_0 I + c_1 A + \ldots, c_q A^q$, where I is then $n \times n$ identity matrix. So $f(A)y$ can be computed for any vector y using q matrix-vector multiplications with A.

3 Approximate Chebyshev Moment Matching

In this section we show that the spectral density s of a Hermitian matrix A with eigenvalues in $[-1, 1]$ can be well approximated given access to *approximations* of the first $N = O(1/\epsilon)$ normalized Chebyshev polynomial moments of s, i.e., to approximations of $\mathrm{tr}(\bar{T}_1(A)), \ldots, \mathrm{tr}(\bar{T}_N(A))$. We state our result in Algorithm 1 and analyze it in Section 3.1. We show later, in Section 4, a method to approximate these moments using a stochastic trace estimator, implemented with either exact or approximate matrix vector multiplications with A.

Given approximations $\tilde{\tau}_1, \ldots, \tilde{\tau}_N$ to the first N normalized Chebyshev moments of A, a natural approach is to find a probability density $q : [-1, 1] \to \mathbb{R}^+$ such that the first N normalized Chebyshev moments of q, i.e., $\langle \bar{T}_1, q \rangle, \ldots, \langle \bar{T}_N, q \rangle$, closely approximate $\tilde{\tau}_1, \ldots, \tilde{\tau}_N$. In order for this approximate moment matching approach to return a good spectral density estimate, it requires that: *for any density function q, if the first N Chebyshev moments of q closely approximate those of s, then q must be close to s in Wasserstein distance.* To that end, we prove the following lemma:

Lemma 3.1. *Let $N \in 4\mathbb{N}^+$ be a degree parameter and p, q be distributions on $[-1, 1]$.*

$$W_1(p, q) \leq \frac{36}{N} + 2 \sum_{k=1}^{N} \frac{|\langle \bar{T}_k, p \rangle - \langle \bar{T}_k, q \rangle|}{k}.$$

Lemma 3.1 shows that if the first N normalized Chebyshev moments of two distributions are identical, then the Wasserstein distance between the distributions is at most $O(1/N)$. When the moments between the distributions differ, the contribution of the difference between the k-th moments to

the Wasserstein distance is scaled by $O(1/k)$. In particular, the lemma shows that deviation in the lower moments between distributions contributes more to the Wasserstein distance.

To prove Lemma 3.1, we will use two well-known results on approximating Lipschitz functions by polynomials. The first is proven in [Jac30]. and concerns uniform approximation of Lipschitz continuous functions by a Chebyshev series:

Fact 3.2. *Let $f \in \mathcal{F}([-1,1],\mathbb{R})$ be a Lipschitz continuous function with Lipschitz constant $\lambda > 0$. Then, for every $N \in 4\mathbb{N}^+$, there exists $N+1$ constants $\hat{b}_N[0] > \cdots > \hat{b}_N[N] \geq 0$ such that the polynomial $\bar{f}_N = \sum_{k=0}^{N} \frac{\hat{b}_N[k]}{\hat{b}_N[0]} \langle f, w \cdot \bar{T}_k \rangle \bar{T}_k$ has the property that $\max_{x\in[-1,1]} |f(x) - \bar{f}_N(x)| \leq 18\lambda/N$.*

The coefficients of the polynomial in Fact 3.2 are not explicitly stated since we only require the existence of such a polynomial in order to prove Lemma 3.1. We defer the reader to Appendix A.1 for an explicit construction of the polynomial[5] and Appendix C.6 for a proof of Fact 3.2.

Next, we state a well-known fact that the magnitude of the inner-product of a Lipschitz function f with the k-th Chebyshev polynomial (for $k \geq 1$) under the Chebyshev weight function $w = 1/\sqrt{1-x^2}$ is bounded by $O(1/k)$, i.e., $|\langle f, w \cdot \bar{T}_k \rangle| \leq O(1/k)$. Our proof is given in Appendix E and is a simple adaptation of the proof of Theorem 4.2 in [Tre08].

Fact 3.3. *Let $f \in \mathcal{F}([-1,1],\mathbb{R})$ be a Lipschitz continuous function with Lipschitz constant $\lambda > 0$. Then, for any $k \geq 1$, we have that $|\langle f, w \cdot \bar{T}_k \rangle| = |\int_{-1}^{1} f(x)\bar{T}_k(x)w(x)dx| \leq 2\lambda/k$.*

With Fact 3.2 and 3.3 in place, we are now ready to prove Lemma 3.1

Proof of Lemma 3.1. Recall that the dual formulation of the Wasserstein-1 distance due to Kantorovich-Rubinstein gives us that $W_1(p,q) = \sup_{f\in \text{lip}_1} \int_{-1}^{1} f(x)(p(x)-q(x))dx$ where lip_1 denotes the set of 1-Lipschitz functions on $[-1,1]$. Let $f \in \text{lip}_1$ be an arbitrary 1-Lipschitz function and let $\{\hat{b}_N[k]\}_{k=0}^{N}$ and $\bar{f}_N$ be the coefficients and polynomial respectively from Fact 3.2 for function f. We can then bound $W_1(p,q)$ using the triangle inquality as

$$W_1(p,q) \leq \underbrace{\int_{-1}^{1} |f(x) - \bar{f}_N(x)|(p(x)-q(x))dx}_{t_1} + \underbrace{\int_{-1}^{1} \bar{f}_N(p(x)-q(x))dx}_{t_2}.$$

Using the fact that f is Lipschitz and the bound from Fact 3.2, along with the fact that p and q are distributions, we have that $t_1 \leq 36/N$.

It is left to bound t_2. We expand t_2 using the Chebyshev series expansion of $\bar{f}_N$ and note that $\langle g/w, w \cdot \bar{T}_k \rangle = \langle g, \bar{T}_k \rangle$ for any function $g \in \mathcal{F}([-1,1],\mathbb{R})$, giving us

$$\begin{aligned} t_2 &= \int_{-1}^{1} \bar{f}_N(x)w(x) \cdot \frac{p(x)-q(x)}{w(x)}dx = \int_{-1}^{1} \bar{f}_N(x)w(x) \cdot \sum_{k=0}^{\infty} \langle p-q, \bar{T}_k \rangle \bar{T}_k(x)dx \\ &= \int_{-1}^{1} \left(w(x) \sum_{k=0}^{N} \frac{\hat{b}_N[k]}{\hat{b}_N[0]} \langle f, w \cdot \bar{T}_k \rangle \bar{T}_k(x) \right) \left(\sum_{k=0}^{\infty} \langle p-q, \bar{T}_k \rangle \bar{T}_k(x) \right) dx. \end{aligned}$$

[5]The construction of the polynomial $\bar{f}_N$ in Fact 3.2 and its uniform approximation to f forms the basis of our alternate approach, the Kernel Polynomial Method, which is discussed in-depth in Appendix A.1.

By the orthogonality of the Chebyshev polynomials under the weight function w and the fact that $\langle \bar{T}_k, \bar{T}_k \rangle = 1$ for all $k \in [N]$, we can bound the magnitude of t_2 as

$$|t_2| \leq \sum_{k=1}^{N} |\langle f, w \cdot \bar{T}_k \rangle| \cdot |\langle \bar{T}_k, p \rangle - \langle \bar{T}_k, q \rangle|$$

since we have that $0 \leq \hat{b}_N[k]/\hat{b}_N[0] \leq 1$ and $|\int_{-1}^{1} \bar{T}_k(p(x) - q(x))dx| = |\langle \bar{T}_k, p \rangle - \langle \bar{T}_k, q \rangle|$ for each $k \in [N]$. Additionally, since p and q are distributions we have that $\langle \bar{T}_0, s \rangle = \langle \bar{T}_0, z \rangle = 1/\sqrt{\pi}$. We then use the bound from Fact 3.3 on $|\langle f, w \cdot \bar{T}_k \rangle|$ for each $k \in [N]$. Putting this together gives us that $|t_2| \leq \sum_{k=1}^{N} 2|\langle \bar{T}_k, p \rangle - \langle \bar{T}_k, q \rangle|/k$.

Putting together the bound on t_1 and t_2 gives us the bound on $W_1(p, q)$. □

3.1 Moment Matching Algorithm

With Lemma 3.1 in place, our next step is develop a method to find a distribution q with Chebyshev moments closely matching a given set of target moments. In order to search for a distribution, we consider an evenly-spaced grid of the interval $[-1, 1]$. Specifically, let $d \in \mathbb{N}^+$ be a discretization parameter and let $X_d = [-1, -1 + \frac{2}{d}, \ldots, 1 - \frac{2}{d}, 1]$ be a $(d+1)$-length evenly-spaced grid of the interval $[-1, 1]$. Our goal is to output a distribution supported on X_d for an appropriately chosen value of d. Any such distribution can be described by a vector in $\mathbb{R}^d_{\geq 0}$ such that the i-th entry corresponds to the probability mass placed at point $-1 + 2i/d$ on the grid. Where it is clear from the context, we will denote the distribution and its probability mass vector interchangeably.

In order to compute the first N normalized Chebyshev moments of functions on the grid X_d, we define two matrices $\mathcal{T}_N^d, \widehat{\mathcal{T}}_N^d \in \mathbb{R}^{N \times d}$ such that for $k \in [N]$ and $i \in [d]$,

$$(\mathcal{T}_N^d)_{k,i} = \bar{T}_k(-1 + 2i/d) \qquad \text{and} \qquad (\widehat{\mathcal{T}}_N^d)_{k,i} = \frac{\bar{T}_k(-1 + 2id)}{k}.$$

The matrix $\mathcal{T}_N^d$ corresponds to a "discretization" of the continuous operator that computes the first N normalized Chebyshev moments of a continuous function on $[-1, 1]$. In particular, for a distribution q supported on X_d, we have that $\langle q, \bar{T}_k \rangle = \sum_{i=0}^{d} q_i \bar{T}_k(-1 + 2i/d) = (\mathcal{T}_N^d q)_k$. Notice that the matrix $\mathcal{T}_N^d$ does not contain the row for $\bar{T}_0$; since we are working with distributions we know that $\bar{T}_0(q) = 1/\sqrt{\pi} \cdot \int_{-1}^{1} q dx = 1/\sqrt{\pi}$ for any distribution q on $[-1, 1]$. The matrix $\widehat{\mathcal{T}}_N^d$ is the matrix $\mathcal{T}_N^d$ with the k-th row scaled by $1/k$. With this notation in place, we state the approximate moment matching algorithm in full in Algorithm 1.

Algorithm 1 Approximate Chebyshev Moment Matching

Input: Symmetric $A \in \mathbb{R}^{n \times n}$, degree parameter $N \in 4\mathbb{N}^+$, algorithm $\mathcal{M}(A)$ that computes moment approximations $\tilde{\tau}_1, \ldots, \tilde{\tau}_N$ with the guarantee that $|\tilde{\tau}_k - \frac{1}{n} \operatorname{tr}(\bar{T}_k(A))| \leq (N \ln(eN))^{-1}$ for all k.

Output: A vector q corresponding to a discrete density function on $[-1, 1]$.

1: For $k = 1, \ldots, N$ use $\mathcal{M}$ to compute $\tilde{\tau}_1, \ldots, \tilde{\tau}_N$ and set $z = [\tilde{\tau}_1/1, \tilde{\tau}_2/2, \ldots, \tilde{\tau}_N/N]$.

2: Set $d = \lceil N^3/2 \rceil$ and compute matrix $\widehat{\mathcal{T}}_N^d \in \mathbb{R}^{N \times d}$. ▷ $(\widehat{\mathcal{T}}_N^d)_{k,i} = \bar{T}_k(-1 + \frac{2i}{d})/k$.

3: Minimize $\|\widehat{\mathcal{T}}_N^d q - z\|_1$ subject to $q^\top \vec{1} = 1$ and $q \geq 0$.

4: Return q.

Note that the optimization problem in Line 3 of Algorithm 1 can easily be written as a linear program in $O(d+N)$ variables and constraints and hence can be solved efficiently in $\mathrm{poly}(N,d) = \mathrm{poly}(1/\epsilon)$ time[6]. Since this method is independent of the matrix dimension n, it is a lower order term in the running time stated in Theorems 1.4 and 1.3, as we will discuss in Section 4.

We show that when $N = O(1/\epsilon)$, Algorithm 1 returns a distribution satisfying $W(s,q) \leq \epsilon$.

Lemma 3.4. *Let $\epsilon \in [0,1]$ and let $N \geq 18/\epsilon$. Then the distribution $q : [-1,1] \to \mathbb{R}^+$ returned by Algorithm 1 satisfies $W_1(q,s) \leq 3\epsilon$.*

Proof. We start by giving some notation – for a distribution $y : [-1,1] \to \mathbb{R}^+$, we denote $\vec{\tau}_y \coloneqq [\langle \bar{T}_1, y\rangle, \ldots, \langle \bar{T}_N, y\rangle]$ to be the vector of the first N normalized Chebyshev moments of y. For an integer $k \in \mathbb{N}^+$, we denote $\vec{k}$ to be the vector in $\mathbb{R}^k$ given by $\vec{k} \coloneqq [1,\ldots,k]$ and for a vector $y \in \mathbb{R}^k$ write $y/\vec{k}$ to denote the vector $y/\vec{k} \coloneqq [y_1/1, \ldots, y_k/k]$. Notice then that we have $\vec{\tau}_q = \mathcal{T}_N^d q$ and $\vec{\tau}_q/\vec{N} = \widehat{\mathcal{T}}_N^d q$.

We start by bounding the scaled differences in the first N normalized Chebyshev moments of q and s in order to use Lemma 3.1 on q and s.

$$\|\vec{\tau}_q/\vec{N} - \vec{\tau}_s/\vec{N}\|_1 \leq \|\vec{\tau}_q/\vec{N} - z\|_1 + \|z - \vec{\tau}_s/\vec{N}\|_1 \leq \|\vec{\tau}_q/\vec{N} - z\|_1 + \frac{1}{N}. \tag{4}$$

The first inequality follows by applying the triangle inequality and in the second inequality we used the fact that $\|z - \vec{\tau}_s/\vec{N}\|_1 = \sum_{k=1}^N |\tilde{\tau}_k - (\vec{\tau}_s)_k|/k \leq H_n \cdot (N\ln(eN))^{-1} \leq 1/N$.

Next we show that there exists a distribution q' supported on X_d such that $\|\vec{\tau}_{q'}/\vec{N} - z\| \leq 1/N$. To this end, consider the following distribution q^* on X_d:

$$q^*(x) = \frac{1}{n}\sum_{i=1}^n \delta(x - \operatorname*{argmin}_{p \in X_d} |p - \lambda_i|).$$

In words, q^* is the distribution corresponding to moving the mass from each λ_i to its nearest point on the grid X_d. Notice that we have $W_1(s,q^*) \leq 1/d$ due to the earthmover distance interpretation of the Wasserstein-1 distance.

Applying the triangle inequality and the guarantee from the moment approximations, we get that $\|\vec{\tau}_{q^*}/\vec{N} - z\|_1 \leq 1/N + \|\vec{\tau}_{q^*}/\vec{N} - \vec{\tau}_s/\vec{N}\|_1$. It is left then to bound $\|\vec{\tau}_{q^*}/\vec{N} - \vec{\tau}_s/\vec{N}\|_1$. To this end, we state the following well-known fact about the derivatives of Chebyshev polynomials.

Fact 3.5. *For $k \geq 1$, $\frac{dT_k(x)}{dx} = kU_{k-1}(x)$.*

We then have using the definition of q^* that, for any $1 \leq k \leq N$,

$$|\langle \bar{T}_k, s\rangle - \langle \bar{T}_k, q^*\rangle| = \left|\frac{1}{n}\sum_{i=1}^n \bar{T}_k(\lambda_i) - \bar{T}_k(\operatorname*{argmin}_{p\in X_d}|p-\lambda_i|)\right| \leq \frac{1}{n}\sum_{i=1}^n \left|\bar{T}_k(\lambda_i) - \bar{T}_k(\operatorname*{argmin}_{p\in X_d}|p-\lambda_i|)\right|$$

[6] Additionally, note that the optimization problem has a convex objective and constraints – in particular, the set of distributions supported on X_d is a convex set. The objective function $\|\widehat{\mathcal{T}}_N^d q - z\|_1$ is not differentiable, but has subgradients. Hence, this program can be solved efficiently in $\mathrm{poly}(1/\epsilon)$ time using a projected subgradient method. This requires an oracle that projects onto the the probability simplex supported on the grid X_d – an algorithm that runs in $O(d\log d)$ time has been given in multiple papers, see [WCP13] for more details.

$$\leq \frac{\sqrt{2}}{n\sqrt{\pi}} \sum_{i=1}^{n} \max_{x\in[-1,1]} \left|\frac{dT_k(x)}{dx}\right| \cdot |\lambda_i - \operatorname*{argmin}_{p\in X_d} |p - \lambda_i|| \leq \frac{\sqrt{2}k^2}{d\sqrt{\pi}}$$

where in the last inequality we used the fact that $\max_{x\in[-1,1]} |U_{k-1}(x)| \leq k$. It follows then that

$$\|\vec{\tau}_{q^*}/\vec{N} - \vec{\tau}_s/\vec{N}\|_1 = \sum_{k=1}^{N} \frac{|(\vec{\tau}_{q^*})_k - (\vec{\tau}_s)_k|}{k} \leq \frac{N(N+1)}{d\sqrt{2\pi}} \leq \frac{1}{N}$$

by taking the sum over all k and noting that $d \geq N^3/2$. Putting these bounds together gives us that $\|\vec{\tau}_{q^*}/\vec{N} - z\|_1 \leq 2/N$.

Since $\|\vec{\tau}_q/\vec{N} - z\|_1 \leq \|\vec{\tau}_{q^*}/\vec{N} - z\|_1$ from Line 3 of Algorithm 1, we plug this into (4) to get that $\|\vec{\tau}_q/\vec{N} - \vec{\tau}_s/\vec{N}\|_1 \leq 3/N$. We can then use Lemma 3.1 with distributions q and s along with the fact that $\|\vec{\tau}_q/\vec{N} - \vec{\tau}_s/\vec{N}\|_1 = \sum_{k=1}^{N} |(\vec{\tau}_s)_k - (\vec{\tau}_q)_k|/k \leq 3/N$ to give us the result since $N > 18/\epsilon$. □

Remark. Note that Algorithm 1 can easily be adapted when the minimization problem in Line 3 is solved approximately – as is the case if projected subgradient descent methods are used. In particular, a constant factor approximation to the minimal loss increases the Wasserstein distance bound in Lemma 3.4 by an $O(1)$ factor.

4 Efficient Chebyshev Moment Approximation

With Lemma 3.4 in place, we are ready to prove our main results. To do so, we need to show how to efficiently approximate the first N Chebyshev moments of a matrix A's spectral density s, as required by Algorithm 1. Recall that the k^{th} normalized Chebyshev moment of s is equal to $\langle s, \bar{T}_k\rangle = \frac{1}{n}\operatorname{tr}(\bar{T}_k(A))$. We will prove that this trace can be approximated using Hutchinson's stochastic trace estimator, implemented with either exact or approximate matrix-vector multiplications with A.

This estimator requires repeatedly computing $\bar{T}_k(A)g$ for a random vector g, which is done using the standard three-term (forward) recurrence for the Chebyshev polynomials and requires a total of k matrix-vector multiplications with A. We analyze the basic approach in Section 4.1, which yields Theorem 1.4. Then in Section 4.2, we argue that the approach is stable even when implemented with approximate matrix-vector multiplication, which yields Theorem 1.3.

4.1 Exact Matrix-Vector Multiplications

Hutchinson's estimator is a widely used estimator to efficiently compute accurate estimates of $\operatorname{tr}(R)$ for any square matrix $R \in \mathbb{R}^{n\times n}$. Each instance of the estimator computes the quadratic form $g^\top Rg$ for a random vector $g \in \{-1, 1\}^n$ whose entries are Rademacher random variables. This an unbiased estimator for $\operatorname{tr}(R)$ with variance $\leq 2\|R\|_F^2$, and its error has been analyzed in several earlier results [AT11, RA15]. We apply a standard high-probability bound from [MMMW20, RV13]:

Lemma 4.1 (Lemma 2, [MMMW20])**.**[7] *Let $R \in \mathbb{R}^{n\times n}$, $\delta \in (0, 1/2]$, $l \in \mathbb{N}$. Let $g^{(1)}, \ldots, g^{(\ell)} \in \{-1, 1\}^{n\times n}$ be ℓ random vectors with i.i.d $\{-1, +1\}$ random entries. For a fixed constant C, with*

[7] In [MMMW20] the lemma is stated with an assumption that $\ell > O(1/\delta)$. However, it is easy to see that the same claim holds without this assumption, albeit with a quadratically worse $\log(1/\delta)$ dependence. The proof follows from same application of the Hanson-Wright inequality used in that work.

probability at least $1-\delta$,

$$\left|\operatorname{tr}(R) - \frac{1}{\ell}\sum_{i=1}^{l}(g^{(i)})^\top R g^{(i)}\right| \le \frac{C\log(1/\delta)}{\sqrt{\ell}}\|R\|_F.$$

For a polynomial $p \in \mathcal{F}([-1,1],\mathbb{R})$ with degree k, applying Hutchinson's estimator to $R = p(A)$ requires computing $p(A)g$, which can always be done with k matrix-vector multiplies with A. If $p(x)$ admits a recursive construction, like the Chebyshev polynomials, then this recurrence can be used. Specifically, for the Chebyshev polynomials, we have:

$$\begin{aligned} &T_0(A)g = g \qquad T_1(A)g = Ag \\ &T_k(A)g = 2A\cdot T_{k-1}(A)g - T_{k-2}(A)g \qquad \text{for } k \ge 2. \end{aligned} \tag{5}$$

A moment estimation algorithm based on Hutchinson's estimator is stated as Algorithm 2.

Algorithm 2 Hutchinson Moment Estimator

Input: Symmetric $A \in \mathbb{R}^{n\times n}$ with $\|A\|_2 \le 1$, degree $N \in 4\mathbb{N}^+$, number of repetitions $\ell \in \mathbb{N}^+$.
Output: Approximation $\tilde{\tau}_k$ to moment $\frac{1}{n}\operatorname{tr}(\bar{T}_k(A))$ for all $k \in 1,\dots,N$.

1: Draw $g^{(1)},\dots,g^{(l)} \sim \text{Uniform}(\{-1,1\}^n)$.
2: For $k = 1,\dots,N$, $\tilde{\tau}_k \leftarrow \frac{\sqrt{2/\pi}}{\ell n}\sum_{i=1}^{l}(g^{(i)})^\top T_k(A)g^{(i)}$. ▷ Computed using recurrence in (5)
3: Return $\tilde{\tau}_1,\dots,\tilde{\tau}_N$.

Remark. In total, Algorithm 2 requires $N\cdot\ell$ matrix multiplications with A since for each i $T_1(A)g^{(i)},\dots,T_N(A)g^{(i)}$ can but computed using the same N steps of the (5) recurrence. It requires $O(n\ell N)$ additional runtime to compute and sum all inner products of the form $(g^{(i)})^T T_k(A)g^{(i)}$.

Our main bound on the accuracy of Algorithm 2 follows:

Lemma 4.2. *If Algorithm 2 is run with* $\ell = \max\left(1, C\cdot\log^2(N/\delta)/(n\Delta^2)\right)$*, where* C *is a fixed positive constant, then with probability* $1-\delta$ *the approximate moments returned satisfy* $|\tilde{\tau}_k - \frac{1}{n}\operatorname{tr}(\bar{T}_k(A))| \le \Delta$ *for all* $k = 1,\dots,N$.

Proof. Fix $k \in \{1,\dots,N\}$. Note that $\frac{1}{n}\operatorname{tr}(\bar{T}_k(A)) = \frac{\sqrt{2/\pi}}{n}\operatorname{tr}(T_k(A))$. Let C be the constant from Lemma 4.1. If $\ell = \max\left(1, C^2\cdot\log^2(N/\delta)/(n\Delta^2)\right)$, then by that lemma we have that with probability at least $1-\delta/N$:

$$\left|\tilde{\tau}_k - \frac{\sqrt{2/\pi}}{n}\operatorname{tr}(T_k(A))\right| \le \frac{1}{n}\frac{C\log(N/\delta)}{\sqrt{\ell}}\|T_k(A)\|_F \le \frac{C\sqrt{2/\pi}}{\sqrt{n}}\sqrt{\frac{\log(N/\delta)}{\ell}} \le \Delta.$$

The second to last inequality follows from the fact that $\|T_k(A)\|_2 \le 1$ and thus $\|T_k(A)\|_F \le \sqrt{n}$. Applying a union bound over all $k \in 1,\dots,N$ gives the claim. □

Theorem 1.4 immediately follows as a corollary of Lemma 4.2 and Lemma 3.4.

Proof of Theorem 1.4. We implement Algorithm 1 with Algorithm 2 used as a subroutine to approximate the Chebyshev polynomial moments, which requires setting $\Delta = \frac{1}{N\ln(eN)}$. By Lemma 4.2, we conclude that we need to set $\ell = \max\left(1, CN^2\log^2(N/\delta)\log^2(eN)/n\right)$. Then, by Lemma 3.4, setting $N = O(1/\epsilon)$ ensures that Algorithm 1 returns a distribution q which is ϵ close to A's spectral density s in Wasserstein distance. □

4.2 Approximate Matrix-Vector Multiplications

Algorithm 2 assumes access to an oracle for computing exact matrix-vector multiplies with A. In this section, we show that the method continues to work well even when each term in Hutchinson's estimator, $g^\top T_k(A)g$, is computed using an approximate matrix-vector multiplication oracle for A (see Definition 1.2). As discussed in Section 1.1, the robustness of the estimator allows the approximate moment matching method to be applied in many settings where A can only be access implicitly. It also forms the basis of our sublinear time algorithm for computing the spectral density of a normalized graph adjacency or Laplacian matrix, which are presented in the Section 5.

To show that approximate matrix-vector multiplications suffice, we leverage well understood stability properties of the three-term forward recurrence for Chebyshev polynomials of the first kind [Cle55, MMS18]. These properties allows us to analyze the cumulative error when $T_k(A)g$ is computed via this recurrence. Specifically, we analyze the following algorithm:

Algorithm 3 Hutchinson Moment Estimator w/ Approximate Multiplications

Input: Symmetric $A \in \mathbb{R}^{n\times n}$ with $\|A\|_2 \le 1$, degree $N \in 4\mathbb{N}^+$, number of repetitions $\ell \in \mathbb{N}^+$, ϵ_{MV}-approximate matrix vector multiplication oracle AMV for A (see Definition 1.2).

Output: Approximation $\tilde{\tau}_k$ to moment $\frac{1}{n}\operatorname{tr}(\bar{T}_k(A))$ for all $k \in 1,\ldots,N$.

for $i = 1,\ldots,\ell$ iterations **do**
  Draw $g \sim \text{Uniform}(\{-1,1\}^n)$.
  $\tilde{v}_0 \leftarrow g$, $\tilde{v}_1 \leftarrow \text{AMV}(A, g, \epsilon_{\mathsf{MV}})$.
  $\tilde{\tau}_{1,i} \leftarrow g^T\tilde{v}_1$
  for $k = 2$ to N **do**
    $\tilde{v}_k \leftarrow 2\cdot\text{AMV}(A, \tilde{v}_{k-1}, \epsilon_{\mathsf{MV}}) - \tilde{v}_{k-2}$.
    $\tilde{\tau}_{k,i} \leftarrow g^T\tilde{v}_k$
For $k = 1,\ldots,N$, $\tilde{\tau}_k \leftarrow \frac{1}{\ell}\sum_{i=1}^{\ell}\tilde{\tau}_{k,i}$.
Return $\tilde{\tau}_1,\ldots,\tilde{\tau}_N$.

Algorithm 3 assumes access to an approximate matrix-vector multiplication oracle for A with error ϵ_{MV} (recall Definition 1.2). Since $\|A\|_2 \le 1$, for any vector y, we have that:

$$\|\text{AMV}(A, y, \epsilon_{\mathsf{MV}}) - Ay\|_2 \le \epsilon_{\mathsf{MV}}\|y\|_2. \tag{6}$$

The algorithm uses this oracle to apply the recurrence from (5), approximately computing each $T_k(A)g$ for $k = 1,\ldots,N$, which in turn allows us to approximately compute $g^\top T_k(A)g$. Note that when $\epsilon_{\mathsf{MV}} = 0$, Algorithm 3 is exactly equivalent to Algorithm 2.

Notation. Analyzing this approach requires accounting for error accumulates across iterations. To do so, we introduce some basic notation. Let v_k denote the true value of $T_k(A)g$, and let $\tilde{v}_k$ denote our computed approximation. We initialize the recurrence with $\tilde{v}_{-1} = \vec{0}$ and $\tilde{v}_0 = v_0 = g$. For

$k = 0, \ldots, N-1$, let $w_k = \text{AMV}(A, \tilde{v}_k, \epsilon_{\mathsf{MV}})$ and note that $\|w_k - A\tilde{v}_k\|_2 \leq \epsilon_{\mathsf{MV}} \|\tilde{v}_k\|_2$. In iteration k of the recurrence, we compute $\tilde{v}_{k+1}$ by applying the recurrence:

$$\tilde{v}_{k+1} \coloneqq 2w_k - \tilde{v}_{k-1}.$$

For each $i \in 0, \ldots, N$ we denote:

- $\delta_k \coloneqq v_k - \tilde{v}_k$, with $\delta_0 = \vec{0}$. This is the *accumulated error* up to iteration k.
- $\xi_{k+1} \coloneqq A\tilde{v}_k - w_k$, with $\xi_0 = 0$. $2\xi_{k+1}$ is the *new error* introduced in iteration k due to approximate matrix-vector multiplication.

As in Clenshaw's classic work [Cle55], it can be shown that δ_k *itself evolves according to a simple recurrence*, which ultimately lets us show that it can be expressed as a summation involving Chebyshev polynomials of the *second* kind, which are easily bounded. Specifically, we have:

Fact 4.3. *$\delta_1 = \xi_1$ and for $2 \leq k \leq N$, $\delta_k = 2A\delta_{k-1} - \delta_{k-2} + 2\xi_k$.*

Proof. The claim for δ_1 is direct since $v_0 = \tilde{v}_0$: we have $\delta_1 = v_1 - \tilde{v}_1 = Av_0 - w_0$. For $2 \leq k \leq N$, we prove the claim by writing the difference $\delta_k = v_k - \tilde{v}_k = v_k - 2(A\tilde{v}_{k-1} + \xi_k) + \tilde{v}_{k-2}$. We can then replace $v_k = 2Av_{k-1} - v_{k-2}$ and substitute in $(v_{k-1} - \tilde{v}_{k-1}) = \delta_{k-1}$ and $(v_{k-2} - \tilde{v}_{k-2}) = \delta_{k-2}$. □

The Chebyshev polynomials of the second kind are defined via the following recurrence:

Definition 4.4 (Chebyshev Polynomials of the Second Kind)**.** For $k \in \mathbb{N}^{\geq 0}$ the k-th Chebyshev polynomial of the second kind $U_k(x)$ is given by

$$\begin{aligned} U_0(x) &= 1 \qquad U_1(x) = 2x \\ U_k(x) &= 2x \cdot U_{k-1}(x) - U_{k-2}(x) \qquad \text{for } k \geq 2. \end{aligned}$$

We also define $U_{-1}(x) = 0$, which is consistent with the recurrence.

Using these polynomials, we can characterize the accumulated error δ_k in terms of the error introduced in each of the prior iterations.

Lemma 4.5. *For $k = 1, \ldots, N$, we have*

$$\delta_k = U_{k-1}(A)\xi_1 + 2\sum_{i=2}^{k} U_{k-i}(A)\xi_i. \tag{7}$$

Proof. We prove the lemma by induction on $j \leq k$. For $j = 0$, the lemma is trivial since $\delta_0 = 0$ by definition and $U_{-1}(A) = 0$. For $j = 1$, $\delta_1 = \xi_1 = U_0(A)\xi_1$. By Fact 4.3, for $2 \leq j < k$, we have:

$$\delta_j = 2\xi_j + \underbrace{2A\delta_{j-1} - \delta_{j-2}}_{z_1}. \tag{8}$$

We can apply the inductive hypothesis on z_1 and recombine terms using Definition 4.4 to get:

$$z_1 = 2A \cdot \left(U_{j-2}(A)\xi_1 + 2\sum_{i=2}^{j-1} U_{j-1-i}(A)\xi_i \right) - U_{j-3}(A)\xi_1 - 2\sum_{i=2}^{j-2} U_{j-2-i}(A)\xi_i$$

$$= U_{j-1}(A)\xi_1 + U_1(A)\cdot 2\xi_{j-1} + \sum_{i=2}^{j-2}\left(2AU_{j-1-i}(A) - U_{j-2-i}(A)\right)\cdot 2\xi_i$$
$$= U_{j-1}(A)\xi_1 + \sum_{i=2}^{j-1} U_{j-i}(A)\cdot 2\xi_i$$

Noting that plugging into (8) and noting that $2\xi_j = 2U_0(A)\xi_j$ completes the proof. □

Our goal is to use Lemma 4.5 to establish that δ_k is small because each ξ_i is small. It is well known that the Chebyshev polynomials of the second kind satisfy the following bounds for any $k \in \mathbb{N}$:

$$|U_k(x)| \le k+1 \qquad \text{for} \qquad x \in [-1,1]. \tag{9}$$

This is the upper bound we need to proceed. Specifically, we will show that each estimator using Algorithm 3, $g^\top \tilde{v}_k$, well approximates Hutchinson's estimator $g^\top T_k(A)g = g^\top v_k$.

Claim 4.6. *For quantities $v_k, \tilde{v}_k$ and $0 \le \epsilon_{MV} \le 1/2k^2$, we have*

$$\left|g^\top T_k(A)g - g^\top \tilde{v}_k\right| \le 2\,\epsilon_{MV}\cdot(k+1)^2\|g\|_2^2.$$

Proof. By the definition of δ_k, we have $|g^\top T_k(A)g - g^\top \tilde{v}_k| = |g^\top \delta_k|$. By Cauchy-Schwarz we can bound $|g^\top \delta_k| \le \|g\|_2\|\delta_k\|_2$. We are left to bound $\|\delta_k\|_2$. Applying Lemma 4.5 and triangle inequality, we have

$$\|\delta_k\|_2 \le \|U_{k-1}(A)\|_2\|\xi_1\|_2 + \sum_{i=2}^{k} 2\|U_{k-i}(A)\|_2\|\xi_i\|_2$$

Then applying (9) and the fact that $\|A\|_2 \le 1$, we have $\|U_{k-i}(A)\|_2 \le (k-i+1)$. Hence,

$$\|\delta_k\|_2 \le k\|\xi_1\|_2 + \sum_{i=2}^{k} 2(k-i+1)\|\xi_i\|_2 \le \sum_{i=1}^{k} 2(k-i+1)\|\xi_i\|_2.$$

Using that $\xi_i \le \epsilon_{\mathsf{MV}}\|\tilde{v}_{i-1}\|_2$, and that $\|T_i(A)\|_2 \le 1$ for all i and thus $\|v_i\|_2 \le \|g\|_2$, we have:

$$\|\delta_k\|_2 \le \sum_{i=1}^{k} 2(k-i+1)\,\epsilon_{\mathsf{MV}}\|\tilde{v}_{i-1}\|_2 \le 2\,\epsilon_{\mathsf{MV}}\sum_{i=1}^{k}(k-i+1)(\|v_{i-1}\|_2 + \|\delta_{i-1}\|_2)$$
$$\le \epsilon_{\mathsf{MV}}\,k(k+1)\left(\|g\|_2 + \max_{i<k}\|\delta_i\|_2\right).$$

Inducting on δ_j for $j \le k$ gives us $\|\delta_k\|_2 \le 2\,\epsilon_{\mathsf{MV}}(k+1)^2\|g\|_2$, which completes the proof. □

Lemma 4.7. *If Algorithm 3 is run with $\ell = \max\left(1, C\cdot\log^2(N/\delta)/(n\Delta^2)\right)$ and $\epsilon_{MV} = \Delta/4N^2$, where C is a fixed positive constant, then with probability $1-\delta$ the approximate moments returned satisfy $|\tilde{\tau}_k - \frac{1}{n}\operatorname{tr}(\bar{T}_k(A))| \le \Delta$ for all $k = 1,\ldots,N$.*

Proof. Fix $k \in \{1, \ldots, N\}$. Let $g^{(1)}, \ldots, g^{(\ell)}$ be the random vectors drawn in the outer for-loop of Algorithm 3. Let $\{\tilde{v}_k^{(i)}\}_{i\in[\ell]}$ be the ℓ vectors computed by the inner for-loop and let $\{\delta_k^{(i)} := \tilde{v}_k^{(i)} - T_k(A)g^{(i)}\}_{i\in[\ell]}$ be the ℓ error vectors. Recalling that $\frac{1}{n}\operatorname{tr}(\bar{T}_k(A)) = \frac{\sqrt{2/\pi}}{n}\operatorname{tr}(T_k(A))$, we have:

$$\left|\tilde{\tau}_k - \frac{\sqrt{2/\pi}}{n}\operatorname{tr}(T_k(A))\right| \le \frac{\sqrt{2/\pi}}{n\ell}\sum_{i=1}^{\ell}\left|(g^{(i)})^\top \delta_k^{(i)}\right| + \left|\frac{\sqrt{2/\pi}}{n\ell}\sum_{i=1}^{\ell}(g^{(i)})^\top T_k(A)g^{(i)} - \frac{1}{n}\operatorname{tr}(T_k(A))\right|$$

Applying Claim 4.6 and Lemma 4.1, with probability at least $1 - \delta/N$, we thus have

$$|\tilde{\tau}_k - \frac{1}{n}\operatorname{tr}(\bar{T}_k(A))| \le 2(k+1)^2\,\epsilon_{\mathsf{MV}} \cdot \frac{\sqrt{2/\pi}}{n\ell}\sum_{i=1}^{\ell}\|g^{(i)}\|_2^2 + \Delta/2 \le \Delta/2 + \Delta/2.$$

The last inequality follows from the fact that $\|g^{(i)}\|_2^2 = n$ for all $i \in [\ell]$, and the choice of $\epsilon_{\mathsf{MV}} = \Delta/4N^2$. Applying a union bound over all $k = 1, \ldots, N$ gives the claim. □

Theorem 1.3 immediately follows.

Proof of Theorem 1.3. We implement Algorithm 1 with Algorithm 3 used as a subroutine to approximate the Chebyshev polynomial moments, which requires setting $\Delta = \frac{1}{N\ln(eN)}$. By Lemma 4.7, we conclude that we need to set $\ell = \max\left(1, CN^2\log^2(N/\delta)\log^2(eN)/n\right)$ and $\epsilon_{\mathsf{MV}} = 1/(4N^3\ln(eN))$. Then, by Lemma 3.4, setting $N = O(1/\epsilon)$ ensures that Algorithm 1 returns a distribution q which is ϵ close to A's spectral density s in Wasserstein distance. □

Improving the number of matrix-vector multiplications. We currently require the error bound in Algorithm 1 for estimating the Chebyshev moments to be the same for each of the N moments, i.e., parameter $\Delta = (N\ln(eN))^{-1}$. We note that the number of matrix-vector multiplications can be improved slightly in Theorems 1.3 and 1.4, potentially by a factor of $\log^2(1/\epsilon)$ for small n. This can be achieved by requiring a different error bound for estimating each moment. Specifically, we modify the requirement in Algorithm 1 for the estimate $\tilde{\tau}_k$ of the k-th normalized Chebyshev moment $\frac{1}{n}\operatorname{tr}(\bar{T}_k(A))$ to have error $|\tilde{\tau}_k - \frac{1}{n}\operatorname{tr}(\bar{T}_k(A))| \le (k/N^5)^{1/4}$. Plugging this into Lemma 4.2, we require at most $\sum_{k=1}^{N}\max(1, CN^{2.5}\log^2(N/\delta)/(n\sqrt{k}))$ matrix-vector multiplications to estimate the N moments, where C is a fixed constant. For comparison to the bounds in Theorems 1.4 and 1.3, the above bound decreases linearly in n until $n \ge CN^2\log^2(N/\delta)$ and for very large n is bounded by $O(1/N)$. In the regime where n is small, e.g., when $n \le CN^2\log^2(N/\delta)$, the bounds from the theorems give $O(N^3\log^2(N/\delta)\log^2(eN)/n)$ matrix-vector multiplications, whereas the above bound simplifies to at most $O(N^3\log^2(N/\delta)/n)$ multiplications, saving a $O(\log^2(N)) = O(\log^2(1/\epsilon))$ factor. Lemma 4.7 can be adapted identically to give the same bound in the approximate matrix-vector multiplication case. To give intuition for the Wasserstein error of the resulting density, if the density estimate q output by Algorithm 1 satisfied the requirement that $|\langle q, \bar{T}_k\rangle - \frac{1}{n}\operatorname{tr}(\bar{T}_k(A))| \le (k/N^5)^{1/4}$ for $k \in 1, \ldots, N$, then we have by Lemma 3.1 that $W_1(s,q) \le 36/N + (2/N^{5/4}) \cdot \sum_{k=1}^{N} k^{-3/4} \le 36/N + 8/N = O(1/N)$. This intuition can be used to adapt the proof of Lemma 3.4 to show that Algorithm 1 with moment guarantees as mentioned output a density q such that $W_1(s,q) \le O(1/N)$.

5 Sublinear Time Methods for Graphs

With the proof of Theorem 1.3 in place, we are now ready to state our sublinear time result for adjacency matrices of graphs. The significance of Theorem 1.3 is that it allows for the approximate Chebyshev moment matching method in Algorithm 1 to be combined with any randomized algorithm for approximating matrix-vector multiplications with A. In this section we prove Theorem 1.1 by showing that for the normalized adjacency matrix of any undirected, un-weighted graph, such an algorithm can actually be implemented in *sublinear time*, leading to a sublinear time spectral density estimation (SDE) algorithm for computing graph spectra from these matrices.

Computational Model. Let $A \in \mathbb{R}^{n\times n}$ be the adjacency matrix for an unweighted, n-vertex graph $G = (V, E)$ and let $\bar{A} = D^{-1/2}AD^{-1/2}$ be the symmetric normalized adjacency matrix, where D is an $n \times n$ diagonal matrix containing the degree of each vertex in V. For a node i, let $\mathcal{N}(i) = \{j : (j, i) \in E\}$ denote the set of i's neighboring vertices. We assume a computational model where we can 1) uniformly sample a random vertex in constant time, 2) uniformly sample a random neighbor of any vertex i in constant time, and 3) for a vertex i with degree d_i, read off all neighbors of i in $O(d_i)$ time. A standard adjacency list representation of the graph would allow us to perform these operations but weaker access models would also suffice.[8]

Using this model for accessing the adjacency matrix, we show that, for any $\epsilon_{\mathsf{MV}} \in (0, 1)$ and failure probability $\delta \in (0, 1)$, an ϵ_{MV}-approximate matrix-vector multiplication oracle for $\bar{A}$ can be implemented in $O(n\,{\epsilon_{\mathsf{MV}}}^{-2}\log(1/\delta))$ time. Via Theorem 1.3, this immediately yields an algorithm for computing an SDE that is ϵ close in Wasserstein-1 distance to $\bar{A}$'s spectral density in roughly $\tilde{O}(n/\epsilon^7)$ time for sufficiently large n, and at most $\tilde{O}(n/\epsilon^9)$ time, for fixed δ where the $\tilde{O}(\cdot)$ hides factors of $\text{poly}(\log(1/\epsilon))$. Our main result is stated as Theorem 1.1 in Section 1.1.

The same algorithm can be used to approximate the spectral density of the normalized Laplacian of G by a simple shift and scaling. Specifically, $\bar{A}$ can be obtained from the normalized Laplacian $\bar{L}$ via $\bar{A} = I - \bar{L}$, and the spectral density of $\bar{L}$, $s_{\bar{L}}(x)$ satisfies $s_{\bar{L}}(1 - x) = s_{\bar{A}}(x)$, where $s_{\bar{A}}$ is the spectral density of $\bar{A}$. So if we obtain an ϵ-approximate SDE q for $\bar{A}$ by Theorem 1.1, then the function p satisfying $p(1 - x) = q(x)$ is an ϵ-approximate SDE for $s_{\bar{L}}$. We thus have:

Corollary 5.1. *Given the the normalized adjacency matrix of G, there exists an algorithm that takes $O\left(n\,\text{poly}\left(\frac{\log(1/\delta)}{\epsilon}\right)\right)$ expected time and outputs a density function q that is ϵ close to the spectral density of the normalized Laplacian of G with probability at least $1 - \delta$.*

5.1 Approximate Matrix-Vector Multiplication for Adjacency Matrices

We implement an approximate matrix-vector multiplication oracle for $\bar{A}$ in Algorithm 4, which is inspired by a randomized matrix-multiplication method of [DKM06]. Throughout this section, let $\bar{A}^i$ denote the i^{th} column of $\bar{A}$. Given a sampling budget $t \in \mathbb{N}$, the algorithm samples t indices from $1, \ldots, n$ independently and with replacement – i.e., the same index might be sample multiple times. For each index it samples, the algorithm decides to accept or reject the column corresponding to

[8] E.g., random crawl access to a network [KLS11]. We also note that, if desired, assumption 3) can be removed entirely with a small logarithmic runtime overhead, as long as we know the degree of i. Specifically, 3) can be implemented with $O(d_i \log n)$ calls to 2): we simply randomly sample neighbors until all d_i are found. A standard analysis of the coupon collector problem [Section 3.6, MR95] shows that that the expected number of samples will be $O(d_i \log d_i) \leq O(d_i \log n)$.

that index with some probability. To approximate $\bar{A}y$, the algorithm outputs the multiplication of the accepted columns, rescaled appropriately, with the corresponding elements of y.

Algorithm 4 AMV Multiplication Oracle for Normalized Adjacency Matrices

Input: Normalized adjacency matrix $\bar{A} \in \mathbb{R}^{n \times n}$, degrees $[d_1, \ldots, d_n]$, $y \in \mathbb{R}^n$, and parameter $t \in \mathbb{N}$.
Output: A vector $z \in \mathbb{R}^n$ that approximates $\bar{A}y$.

1: Initialize $z \leftarrow \vec{0}$.
2: **for** t iterations **do**
3: Sample a node j uniformly at random from $\{1, \ldots, n\}$.
4: Sample a neighbor $i \in \mathcal{N}(j)$ uniformly at random.
5: Sample x uniformly at random from $[0, 1]$.
6: **if** $x \leq \frac{1}{d_i}$ **then**
7: $w \leftarrow \frac{1}{p_i} \cdot y_i \bar{A}^i$ where $p_i = \frac{1}{nd_i} \sum_{j \in \mathcal{N}(i)} \frac{1}{d_j}$.
8: **else**
9: $w \leftarrow \vec{0}$.
10: $z \leftarrow z + w$.
11: **return** $\frac{1}{t} z$

The following lemma bounds the expected squared error of Algorithm 4's:

Lemma 5.2. *Let $z \in \mathbb{R}^n$ be the output of Algorithm 4 with sampling budget t. We have:*

$$\mathbf{E}[\|\bar{A}y - z\|_2^2] = \frac{n}{t}\|y\|_2^2 - \frac{1}{t}\|\bar{A}y\|_2^2$$

Proof. Let b denote $b = \bar{A}y$. Consider a single iteration of the main loop in Algorithm 4, which generates a vector w that is added to z. Let X_i be an indicator random variable that is 1 if w is set to a scaling of $\bar{A}^i$ on that iteration, and 0 otherwise. $X_i = 1$ if and only if 1) a neighbor of i is sampled at Line 3 of the algorithm, 2) i is sampled at Line 4 of the algorithm, and 3) the uniform random variable x satisfies $x < 1/d_i$. So, we see that $\mathbf{Pr}[X_i = 1]$ is exactly equal to $p_i = \frac{1}{nd_i} \sum_{j \in \mathcal{N}(i)} \frac{1}{d_j}$. It follows that, by the time we reach Line 11, w is an unbiased estimator for b. I.e., $\mathbf{E}[w] = b$. Of course, this also implies that $\mathbf{E}[z] = b$.

Our goal is to show that $\mathbf{E}[\|b - z\|^2] = \frac{n}{t}\|y\|_2^2 - \frac{1}{t}\|b\|_2^2$. Since the random vector $b - z$ has mean zero and is the average of t i.i.d. copies of the mean zero random vector $b - w$, it suffices that show:

$$\mathbf{E}[\|b - w\|_2^2] = n\|y\|_2^2 - \|b\|_2^2. \tag{10}$$

By linearity of expectation and the fact that $\mathbf{E}[w] = b$, we have

$$\mathbf{E}[\|b - w\|_2^2] = \|b\|_2^2 + \mathbf{E}[\|w\|_2^2] - 2\langle \mathbf{E}[w], b \rangle = \mathbf{E}[\|w\|_2^2] - \|b\|_2^2.$$

So to prove (10), we need to show that $\mathbf{E}[\|w\|_2^2] = n\|y\|_2^2$. We expand w in terms of the indicator random variables $X_1, \ldots, X_n$. Notice that since we only sample one column in each iteration, the random variable $X_i X_j = 0$ for all $i \neq j$. Thus, we have:

$$\mathbf{E}[\|w\|_2^2] = \sum_{k=1}^{n} \mathbf{E}\left[\sum_{i,j \in [n]} \frac{X_i X_j}{p_i p_j} (\bar{A}^i y_i)_k (\bar{A}^j y_j)_k\right] = \sum_{k=1}^{n} \mathbf{E}\left[\sum_{i=1}^{n} \frac{X_i^2}{p_i^2} \cdot (\bar{A}^i y_i)_k^2\right]$$

$$= \sum_{i=1}^{n} \frac{1}{p_i} \cdot \|\bar{A}^i y_i\|_2^2 = \sum_{i=1}^{n} n y_i^2 = n\|y\|_2^2$$

In the last equalities we used the fact that $\mathbf{E}[X_i^2] = p_i$ and that, for a normalized graph adjacency matrix, $\|\bar{A}^i\|_2^2 = \sum_{j \in \mathcal{N}(i)} \frac{1}{d_i d_j} = n p_i$. This proves (10), from which we conclude the lemma. □

Using Lemma 5.2, we show that there is an ϵ_{MV}-approximate matrix-vector oracle for $\bar{A}$ based on Algorithm 4 with success probability at least $1 - \delta$ that runs in $O(n\,{\epsilon_{\mathsf{MV}}}^{-2} \log^2(\frac{1}{\delta}))$ time.

Proposition 5.3. *Let $\bar{A} \in \mathbb{R}^{n \times n}$ be the symmetric normalized adjacency matrix of an n-vertex graph G and let $\epsilon_{MV}, \delta \in (0,1)$ be fixed constants. There is an algorithm that, given a vector $y \in \mathbb{R}^n$, and access to G as described above, takes $O(n\,{\epsilon_{MV}}^{-2} \log(\frac{1}{\delta}))$ expected time and outputs a vector $z \in \mathbb{R}^n$ such that $\|z - \bar{A}y\|_2 \leq \epsilon_{MV}\|y\|_2$ with probability at least $1 - \delta$.*

Proof. By Lemma 5.2, we have that $\mathbf{E}[\|\bar{A}y - z\|_2^2] \leq \frac{n}{t}\|y\|_2^2$. Fix $t = 48n\,{\epsilon_{\mathsf{MV}}}^{-2}$. Then, by Lemma 5.2 and Markov's inequality, we have that when Algorithm 4 is called on $\bar{A}$ with parameter t,

$$\mathbf{Pr}[\|\bar{A}y - z\|_2 > \frac{\epsilon_{\mathsf{MV}}}{4}\|y\|_2] \leq \frac{16n\|y\|_2^2}{t\,{\epsilon_{\mathsf{MV}}}^2\,\|y\|_2^2} \leq \frac{1}{4}. \tag{11}$$

In order improve our success probability from 3/4 to $1 - \delta$, we use the standard trick of repeating the above process $r = c\log(\frac{1}{\delta})$ times for a constant c to be fixed later. Let $z_1, \ldots, z_r \in \mathbb{R}^n$ be the output of running Algorithm 4 r times with parameter t. We can return as our estimate for $\bar{A}y$ the first z_i such that $\|z_i - z_j\|_2 \leq \frac{\epsilon_{\mathsf{MV}}}{2}\|y\|_2$ for at least $r/2 + 1$ vectors z_j from $z_1, \ldots, z_n$.

To see why this works, note that a Chernoff bound can be used to claim that with probability $> 1 - \delta$, at least $r/2 + 1$ vectors z_j from $z_1, \ldots, z_r$ have that $\|z_j - \bar{A}y\|_2 \leq \frac{\epsilon_{\mathsf{MV}}}{4}\|y\|_2$.

By a triangle inequality we have that for all such z_j and z_k,

$$\|z_j - z_k\|_2 \leq \|z_j - \bar{A}y\|_2 + \|z_k - \bar{A}y\|_2 \leq \frac{\epsilon_{\mathsf{MV}}}{2}\|y\|_2.$$

Thus, the z_i we picked must satisfy that $\|z_i - \bar{A}y\| \leq \frac{3\,\epsilon_{\mathsf{MV}}}{4}\|y\|_2$ by the triangle inequality.

All that remains is to bound the expected runtime of Algorithm 4, which we will run r separate times. To do so, note that all index sampling can be done in just $O(t)$ time, since sampling a random vertex and a random neighbor of the vertex are assumed to be $O(1)$ time operations. The costly part of the algorithm is computing the sampled column w at each iteration. In the case that $w = \vec{0}$, this cost is of course zero. However, when $w = \frac{1}{p_i}\bar{A}^i y_i$ for some i, computing the column and adding it to z takes $O(d_i)$ time, which can be large in the worst case. Nevertheless, we show that it is small in expectation. This may seem a bit surprising: while nodes with high degree are more likely to be sampled by Line 4 in Algorithm 4, they are rejected with higher probability in Line 6. Formally, let $\mathrm{nnz}(w)$ denote the number of non-zero entries in w. We have:

$$\mathbf{E}\left[\mathrm{nnz}(w)\right] = \sum_{i=1}^{n} \mathrm{nnz}(\bar{A}^i) \cdot p_i = \sum_{i=1}^{n} \sum_{j \in \mathcal{N}(i)} \frac{d_i}{n \cdot d_i d_j} = \frac{1}{n} \sum_{i=1}^{n} \sum_{j \in \mathcal{N}(i)} \frac{1}{d_j} = 1.$$

The final equality follows from expanding the double sum: since node j has exactly d_j neighbors, $\frac{1}{d_j}$ appears exactly d_j times in the sum. So $\sum_{i=1}^{n} \sum_{j \in \mathcal{N}(i)} \frac{1}{d_j} = n$.

We run Algorithm 4 with $t = O(n/\epsilon_{\mathsf{MV}}{}^2)$ iterations, so it follows that the expected total sparsity of all w's constructed equals $O(n/\epsilon_{\mathsf{MV}}{}^2)$, which dominates the expected running time of our method.

□

Proof of Theorem 1.1. The accuracy and running time claim follows from combining the ϵ_{MV}-approximate vector multiplication oracle described in Proposition 5.3 with Algorithm 1, which is analyzed in Theorem 1.3. □

As discussed in the introduction, Cohen et al. [CKSV18] prove a result which matches the guarantee of Theorem 1.1, but with runtime of $2^{O(1/\epsilon)}$ – i.e., with *no dependence* on n. In comparison, our result depends linearly on n, but only polynomially on $1/\epsilon$. In either case, the result is quite surprising, as the runtime is *sublinear* in the input size: A could have up to $O(n^2)$ non-zero entries.

6 Experiments

We support our theoretical results by implementing our Chebyshev moment matching method (Algorithm 1). When using exact matrix-vector multiplications, the kernel polynomial method (KPM) of Algorithm 6 and the stochastic Lanczos quadrature method (SLQ) studied in [CTU21] have both been confirmed to work well empirically. So, one set of experiments is aimed at comparing these methods to the moment matching method (MM) implemented with exact matrix-vector multiplications. A second set of experiments evaluates the performance of the MM and KPM methods when implemented with approximate matrix-vector multiplies. Specifically, we use our sublinear time randomized method for multiplication by graph adjacency matrices from Section 5.

We consider the normalized adjacency matrix of three graphs, two of which we construct and one which we obtain from a publicly available dataset for sparse matrices:

- `cliquePlusRandBipartite` is a graph with 10000 vertices, partitioned into two disconnected components. The first component is a clique with 5000 nodes and the second is a bipartite graph with 2500 vertices in each partition, constructed by sampling each of the 2500^2 possible edges independently with probability 0.05. This graph has a normalized adjacency matrix with ~ 5000 eigenvalues at 0, two eigenvalues at 1, one at -1 and the rest of its eigenvalues are roughly evenly spread out between -0.5 and 0.5.
- `hypercube` is a 16384 vertex boolean hypercube graph on 14 bit strings.[9] Its normalized adjacency matrix has eigenvalues at $-1, -\frac{6}{7}, \frac{-5}{7}, \ldots, 0, \ldots, \frac{6}{7}, 1$. The multiplicity of the 0 eigenvalue is largest, with eigenvalues closer to -1 and 1 having lower multiplicity.
- `Erdos992` is an undirected graph with 6100 vertices, containing 15030 edges from the sparse matrix suite of [DH11]. Its normalized adjacency matrix has ~ 5000 eigenvalues at 0, one at 1 and the rest evenly spread out between -0.5 and 0.5.

We consider three additional matrices to evaluate the performance of MM against KPM and SLQ when exact matrix-vector multiplies are used to estimate the Chebyshev moments:

[9] A boolean hypercube contains a vertex for each distinct b bit string, and an edge between two vertices if the corresponding strings differ on exactly 1 bit.

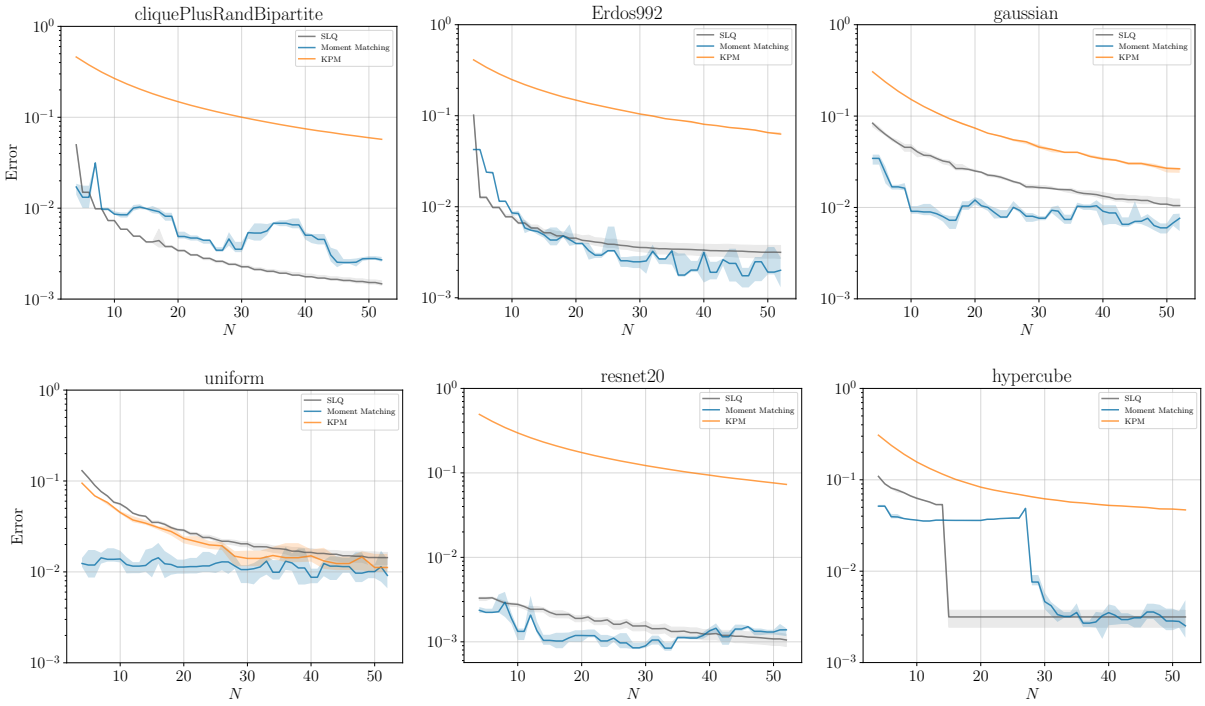

Figure 2: Wasserstein error of density estimate resulting from approximate Chebyshev moment matching method (MM), the Jackson damped kernel polynomial method (KPM) and Stochastic Lanczos Quadrature (SLQ) method. For MM and KPM, Hutchinson's estimator is used to estimate the Chebyshev moments. The x-axis corresponds to the number of moments computed for MM and KPM, and the number of Lanczos iterations used for SLQ. All methods use 5 (random) starting vectors except for resnet20 and hypercube that use 1 starting vector, so the x-axis is directly proportional to the number of matrix-vector multiplications used by each method. Each experiment is repeated 10 times; the solid line represents the median error of the 10 trials and the shaded regions represent the first and third quartiles.

- `gaussian` is a 1000×1000 matrix constructed by drawing $n = 1000$ Gaussian random variables $\lambda_1, \cdots, \lambda_n \sim \mathcal{N}(0, 1)$ and a random orthogonal matrix $U \in \mathbb{R}^{n \times n}$, and outputting $U \Lambda U^\top$ where Λ is a $n \times n$ diagonal matrix with entries $\frac{\lambda_1}{\max_i \lambda_i}, \ldots, \frac{\lambda_n}{\max_i \lambda_i}$.
- `uniform` is a 1000×1000 matrix constructed identically to `gaussian` except with $\lambda_1, \ldots, \lambda_n$ drawn independently and uniformly from the interval $[-1, 1]$.
- `resnet20` is a Hessian for the ResNet20 network [HZRS16] trained on the Cifar-10 dataset. The matrix is 3000×3000 and its eigenvalues have been normalized to lie between $[-1, 1]$ for our experiments.

For reference, the histogram of the eigenvalues for each matrix are shown in Figure 3 by breaking the range of the eigenvalues into 50 equally spaced intervals for each matrix.

In the first set of experiments, we compute the normalized Chebyshev moments $\tau_1, \ldots, \tau_N$ of each of the six aforementioned matrices using Hutchinson's moment estimator as in Algorithm 2, and, compute a spectral density estimate by passing these moments into Algorithm 1 for approximate

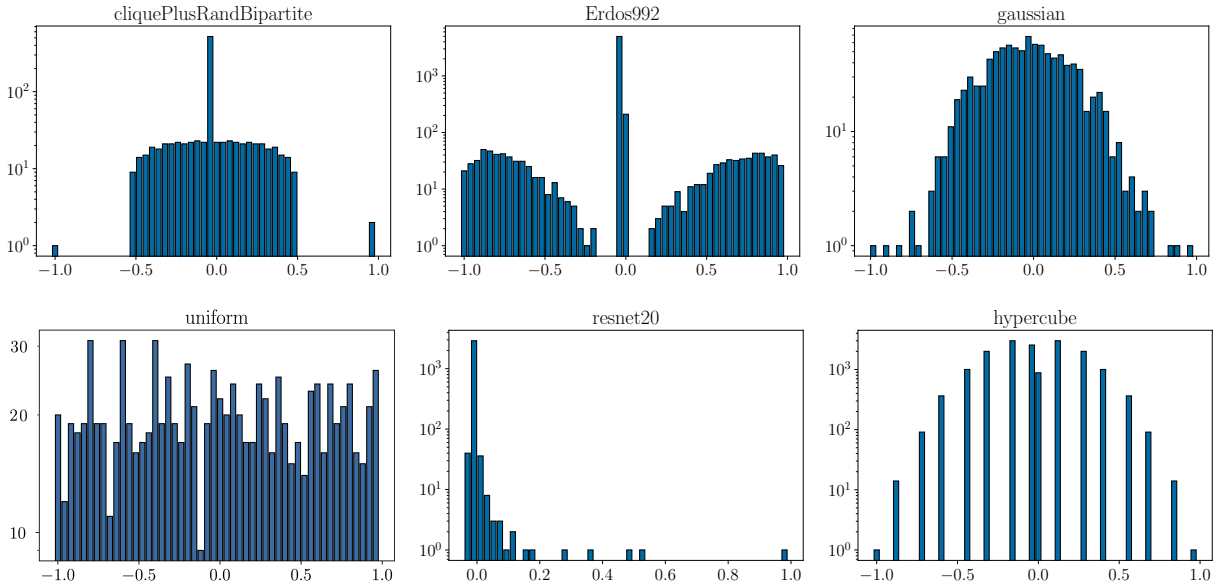

Figure 3: Histograms of the eigenvalues of `cliquePlusRandBipartite`, `Erdos992`, `gaussian`, `uniform`, `resnet20` and `hypercube` using 50 equally spaced buckets.

Chebyshev moment matching method (MM)[10] and into Algorithm 6 for the Jackson damped kernel polynomial method (KPM). For KPM we compute the density with $N = 4, 6, 8, 10, \ldots, 52$ and for MM we compute it with $N = 4, 5, 6, 7, \ldots, 52$. We also compute the density estimate resulting from the stochastic Lanczos quadrature (SLQ) method of [CTU21] with $N = 4, 5, 6, 7, \ldots, 52$ Lanczos iterations. We use $\ell = 5$ starting vectors (i.e., random vectors in Hutchinson's method, or random restarts of the SLQ method) for each method, except for the large `resnet20` and `hypercube` matrices, for which $\ell = 1$ random vector is used. Each experiment is repeated 10 times and the Wasserstein-error between the true density and the density estimate are shown in Figure 2. The results show that MM is more than 10x more accurate than KPM in almost all cases. The error of MM and SLQ are more comparable, except for `hypercube`, on which the errors are comparable for larger values of N. Both methods show an unusual convergence curve for this matrix, which we believe is related to the sparsify of its spectrum (a small number of distinct eigenvalues).

In our second set of experiments, we test the performance of our randomized sublinear time algorithm (Algorithm 4) for approximate matrix-vector multiplies with normalized graph adjacency matrices. This method is used to estimate Chebyshev moments in Algorithm 1 (MM) and in Algorithm 6 (KPM). We compute the normalized Chebyshev moments $\tau_1, \ldots, \tau_N$ for $N = 12$ using various values of the oversampling parameter t in the approximate matrix-vector multiplication method. We then compute, for each value of t, the average number of non-zero elements of A accessed by the method for each matrix-vector product, which reflects the runtime improvement over a full matrix-vector product. Figure 4 plots the Wasserstein error of the density estimate (y-axis) and the average fraction of non-zeros used in each matrix-vector multiplication (x-axis) to estimate the Chebyshev moments used by MM and KPM respectively.

The results show that the KPM method can achieve error nearly identical to that obtained when us-

[10] We solve the optimization problem from Line 3 by formulating it as a linear program and using an off-the-shelf solver from `scipy`.

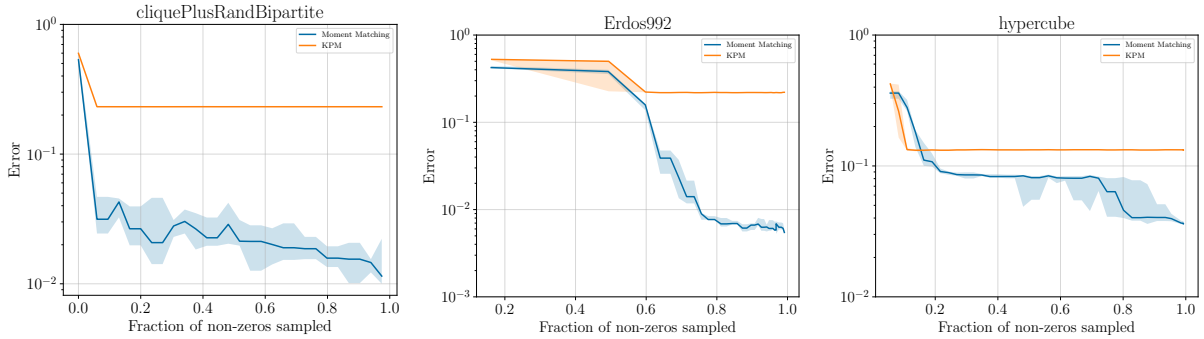

Figure 4: Wasserstein error of density estimate returned by MM and KPM on the `hypercube`, `cliquePlusRandBipartite` and `Erdos992` graphs using approximate matrix-vector multiplications (Algorithm 4) to estimate the Chebyshev moments. For both methods, $N = 32$ moments are computed using 5 random starting vectors for `cliquePlusRandBipartite` and `Erdos992` and 1 for `hypercube`. The x-axis corresponds to the average fraction of non-zeros sampled from the matrix and the y-axis is the Wasserstein error from the resulting density estimate. Each experiment is repeated for 10 trials: the solid line correponds to the median error of the 10 trials and the shaded region corresponds to the first and third quartiles.

ing exact matrix-vector multiplications, while only using a small fraction of non-zero entries for each approximate matrix-vector multiplication. Specifically, on the dense `cliquePlusRandBipartite` graph and even the relatively sparse `hypercube` graph, KPM uses less than 15% of the non-zero entries on average to achieve nearly the same error as when using exact multiplies. On `cliquePlusRandBipartite`, the MM method achieves error close to that of the exact method while using $\sim$ 20% of the non-zero entries on average. On the sparse `Erdos992` and `hypercube` graphs, the MM method requires $\sim$ 80% of the non-zero entries on average to achieve error comparable to exact matrix-vector multiplications. However, it still obtains a good approximation (consistently better than the KPM method) when coarse matrix-vector multiplications are used (i.e., fewer non-zeros are sampled).

7 Acknowledgements

We thank Cameron Musco, Raphael Meyer, and Tyler Chen for helpful discussions and suggestions. This research was supported in part by NSF CAREER grants 2045590 and 1652257, ONR Award N00014-18-1-2364, and the Lifelong Learning Machines program from DARPA/MTO.

References

[AKS17] Jared L. Aurentz, Vassilis Kalantzis, and Yousef Saad. Cucheb: A GPU implementation of the filtered Lanczos procedure. *Computer Physics Communications*, 220:332 – 340, 2017.

[AT11] Haim Avron and Sivan Toledo. Randomized algorithms for estimating the trace of an implicit symmetric positive semi-definite matrix. *J. ACM*, 58(2), 2011.

[BHSW20] Mark Braverman, Elad Hazan, Max Simchowitz, and Blake Woodworth. The gradient complexity of linear regression. In *Proceedings of the 33rd Annual Conference on Computational Learning Theory (COLT)*, volume 125, pages 627–647, 2020.

[BVKS19] Jess Banks, Jorge Vargas, Archit Kulkarni, and Nikhil Srivastava. Pseudospectral shattering, the sign function, and diagonalization in nearly matrix multiplication time. In *Proceedings of the 61st Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, 2019.

[CKSV18] David Cohen-Steiner, Weihao Kong, Christian Sohler, and Gregory Valiant. Approximating the spectrum of a graph. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 1263–1271, 2018.

[Cle55] C. W. Clenshaw. A note on the summation of chebyshev series. *Mathematics of Computation*, 9(51):118, 1955.

[CTU21] Tyler Chen, Thomas Trogdon, and Shashanka Ubaru. Analysis of stochastic lanczos quadrature for spectrum approximation. In *Proceedings of the International Congress of Mathematicians 2021 (ICM)*, 2021.

[DBB19] Kun Dong, Austin R Benson, and David Bindel. Network density of states. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 1152–1161, 2019.

[DH11] Timothy A Davis and Yifan Hu. The university of florida sparse matrix collection. *ACM Transactions on Mathematical Software (TOMS)*, 38(1):1–25, 2011.

[DKM06] Petros Drineas, Ravi Kannan, and Michael W Mahoney. Fast Monte Carlo algorithms for matrices I: Approximating matrix multiplication. *SIAM Journal on Computing*, 36(1):132–157, 2006.

[DM21] Prathamesh Dharangutte and Christopher Musco. Dynamic trace estimation. *Preprint*, 2021.

[GKX19] Behrooz Ghorbani, Shankar Krishnan, and Ying Xiao. An investigation into neural net optimization via hessian eigenvalue density. In *Proceedings of the 36th International Conference on Machine Learning (ICML)*, volume 97, pages 2232–2241, 2019.

[Hut90] Michael F. Hutchinson. A stochastic estimator of the trace of the influence matrix for Laplacian smoothing splines. *Communications in Statistics-Simulation and Computation*, 19(2):433–450, 1990.

[HZRS16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[Jac12] Dunham Jackson. On approximation by trigonometric sums and polynomials. *Transactions of the American Mathematical society*, 13(4):491–515, 1912.

[Jac30] Dunham Jackson. *The Theory of Approximation*, volume 11 of *Colloquium Publications*. American Mathematical Society, 1930.

[KLS11] Liran Katzir, Edo Liberty, and Oren Somekh. Estimating sizes of social networks via biased sampling. In *Proceedings of the 20th International World Wide Web Conference (WWW)*, pages 597–606, 2011.

[KR57] Leonid Vital'evich Kantorovich and Gennadii Shlemovich Rubinshtein. On a functional space and certain extremum problems. In *Doklady Akademii Nauk*, volume 115, pages 1058–1061. Russian Academy of Sciences, 1957.

[KV17] Weihao Kong and Gregory Valiant. Spectrum estimation from samples. *Ann. Statist.*, 45(5):2218–2247, 10 2017.

[Lor66] George G. Lorentz. *Approximation of Functions*. American Mathematical Society, second edition, 1966.

[LSY16] Lin Lin, Yousef Saad, and Chao Yang. Approximating spectral densities of large matrices. *SIAM Review*, 58(1):34–65, 2016.

[LXES19] Ruipeng Li, Yuanzhe Xi, Lucas Erlandson, and Yousef Saad. The eigenvalues slicing library (EVSL): Algorithms, implementation, and software. *SIAM Journal on Scientific Computing*, 41(4):C393–C415, 2019.

[MM15] Cameron Musco and Christopher Musco. Randomized block Krylov methods for stronger and faster approximate singular value decomposition. In *Advances in Neural Information Processing Systems 28 (NeurIPS)*, pages 1396–1404, 2015.

[MM19] Michael Mahoney and Charles Martin. Traditional and heavy tailed self regularization in neural network models. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning (ICML)*, volume 97, pages 4284–4293, 2019.

[MMMW20] Raphael A. Meyer, Cameron Musco, Christopher Musco, and David Woodruff. Hutch++: Optimal stochastic trace estimation. *Proceedings of the 4th Symposium on Simplicity in Algorithms (SOSA)*, 2020.

[MMS18] Cameron Musco, Christopher Musco, and Aaron Sidford. Stability of the Lanczos method for matrix function approximation. In *Proceedings of the 29th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1605–1624, 2018.

[MNS+18] Cameron Musco, Praneeth Netrapalli, Aaron Sidford, Shashanka Ubaru, and David P. Woodruff. Spectrum approximation beyond fast matrix multiplication: Algorithms and hardness. *Proceedings of the 9th Conference on Innovations in Theoretical Computer Science (ITCS)*, 2018.

[MR95] Rajeev Motwani and Prabhakar Raghavan. *Randomized Algorithms*. Cambridge University Press, 1995.

[Par98] Beresford N. Parlett. *The symmetric eigenvalue problem*. SIAM, 1998.

[Pea94] Barak A. Pearlmutter. Fast exact multiplication by the hessian. *Neural computation*, 6(1):147–160, 1994.

[PSG18] Jeffrey Pennington, Samuel Schoenholz, and Surya Ganguli. The emergence of spectral universality in deep networks. In *Proceedings of the 21st International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 1924–1932, 2018.

[RA15] Farbod Roosta-Khorasani and Uri M. Ascher. Improved bounds on sample size for implicit matrix trace estimators. *Foundations of Computational Mathematics*, 15(5):1187–1212, 2015.

[RV13] Mark Rudelson and Roman Vershynin. Hanson-Wright inequality and sub-Gaussian concentration. *Electronic Communications in Probability*, 18, 2013.

[SEAR18] Max Simchowitz, Ahmed El Alaoui, and Benjamin Recht. Tight query complexity lower bounds for pca via finite sample deformed wigner law. In *Proceedings of the 50th Annual ACM Symposium on Theory of Computing (STOC)*, pages 1249–1259, 2018.

[Ski89] John Skilling. *The Eigenvalues of Mega-dimensional Matrices*, pages 455–466. Springer Netherlands, 1989.

[SR94] R.N. Silver and H. R oder. Densities of states of mega-dimensional hamiltonian matrices. *International Journal of Modern Physics C*, 5(4):735–753, 1994.

[SWYZ19] Xiaoming Sun, David P. Woodruff, Guang Yang, and Jialin Zhang. Querying a matrix through matrix-vector products. In *Proceedings of the 46th International Colloquium on Automata, Languages and Programming (ICALP)*, volume 132, pages 94:1–94:16, 2019.

[Tre08] Lloyd N Trefethen. Is gauss quadrature better than clenshaw–curtis? *SIAM review*, 50(1):67–87, 2008.

[Wan94] Lin-Wang Wang. Calculating the density of states and optical-absorption spectra of large quantum systems by the plane-wave moments method. *Phys. Rev. B*, 49:10154–10158, 1994.

[WCP13] Weiran Wang and Miguel A Carreira-Perpinán. Projection onto the probability simplex: An efficient algorithm with a simple proof, and an application. *arXiv preprint arXiv:1309.1541*, 2013.

[Woo14] David P. Woodruff. Sketching as a tool for numerical linear algebra. *Foundations and Trends in Theoretical Computer Science*, 10(1–2):1–157, 2014.

[WWAF06] Alexander Weiße, Gerhard Wellein, Andreas Alvermann, and Holger Fehske. The kernel polynomial method. *Reviews of modern physics*, 78(1):275, 2006.

[YGKM20] Zhewei Yao, Amir Gholami, Kurt Keutzer, and Michael W. Mahoney. Pyhessian: Neural networks through the lens of the hessian. In *IEEE BigData*, 2020.

A The Kernel Polynomial Method

In this section we show how to obtain a spectral density estimate based on a version of the kernel polynomial method that also approximates Chebyshev polynomial moments: $\mathrm{tr}(T_0(A)), \ldots, \mathrm{tr}(T_N(A))$. We again rely on Jackson's classic work on universal polynomial approximation bounds for Lipschitz functions: we take advantage of the fact that Jackson's construction of such polynomials is both linear and preserves positivity [Jac12].

A.1 Idealized Kernel Polynomial Method

As an alternative to the moment matching method presented in Section 3, a natural approach to using computed Chebyshev moments is to construct a truncated Chebyshev series approximation to s (see Definition 2.1). To do so, note that the scaled moments $\frac{1}{n}\operatorname{tr}(\bar{T}_0(A)), \ldots, \frac{1}{n}\operatorname{tr}(\bar{T}_N(A))$ are exactly equal to the first N Chebyshev series coefficients of s/w, where $w(x) = \frac{1}{\sqrt{1-x^2}}$ is as defined in Section 2. Specifically, the eigenvalues of $\bar{T}_k(A)$ are equal to $\bar{T}_k(\lambda_1), \ldots, \bar{T}_k(\lambda_n)$, where $\lambda_1, \ldots, \lambda_n$ are the eigenvalues of A. Since the trace of a diagonalizable matrix is the sum of its eigenvalues, we have $\frac{1}{n}\operatorname{tr}(\bar{T}_k(A)) = \frac{1}{n}\sum_{i=1}^{n}\bar{T}_k(\lambda_i) = \langle s, \bar{T}_k\rangle = \langle s/w, w \cdot \bar{T}_k\rangle$.

After using the scaled Chebyshev moments to construct a truncated Chebyshev series for s/w, i.e. a degree N polynomial approximation, we can then multiply the final result by w to obtain an approximation to s. Unfortunately, there are two issues with this approach: 1) it is difficult to analyze the quality of the Chebyshev series approximation, since s is not a smooth function, and 2) this approximation will not in general be a non-negative function, which is a challenge because our goal is to find *probability density* that well approximates s.

A common approach for dealing with the second issue is to instead use a *damped Chebyshev expansion* [WWAF06], where the Chebyshev coefficients are slightly reweighted to ensure that the resulting polynomial is always non-negative. Such non-negativity preserving damping schemes follow from the connection between Chebyshev and Fourier series: we refer the reader to Appendix C for details. In short, by the convolution theorem, Fourier series truncation corresponds to convolution with a function whose Fourier support is bounded. If this function is also non-negative, convolution preserves non-negativity of the function being approximated, leading to truncated series that is guaranteed to be positive. One such damping schemes was introduced in classic work of Jackson [Jac12]. For any positive integer z, let $N = 4z$. Then, for $k = 0, \ldots, N$, define the coefficient

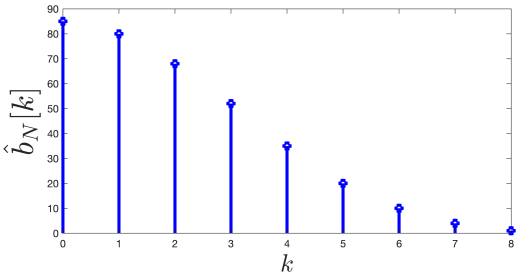

Figure 5: Jackson coefficients for $N = 8$.

$$\hat{b}_N[k] = \sum_{j=-\frac{N}{2}-1}^{\frac{N}{2}+1-k} \left(\frac{N}{2} + 1 - |j|\right) \cdot \left(\frac{N}{2} + 1 - |j+k|\right). \tag{12}$$

While (12) may look opaque, $\hat{b}_N[0], \ldots, \hat{b}_N[N]$ are actually equal to the result of a simple discrete convolution operation. Let $g \in \mathcal{F}(\mathbb{Z}, \mathbb{R})$ have $g[j] = 1$ for $j = -z, \ldots, z$, and $g[j] = 0$ otherwise. Then let $\hat{b}_N = (g * g) * (g * g)$ and $\hat{b}_N[0], \ldots, \hat{b}_N[N]$ be the values corresponding to non-negative indices.[11] See Fig. 5 for an illustration of these coefficients. They are all positive and $\hat{b}_N[0] > \hat{b}_N[1] > \ldots > \hat{b}_N[N]$. Jackson suggests approximating a function using the following truncation based on these coefficients:

Definition A.1 (Jackson damped Chebyshev series)**.** Let $f \in \mathcal{F}([-1, 1], \mathbb{R})$ have Chebyshev series $\sum_{k=0}^{\infty}\langle f, w \cdot \bar{T}_k\rangle \cdot \bar{T}_k$. The Jackson approximation to f is a degree N polynomial $\bar{f}_N$ obtained via

[11] This formulation allows the coefficients to be easily computed in most high-level programming languages. E.g., in `MATLAB` we can compute `g = ones(2*z+1,1); c = conv(conv(g,g),conv(g,g)); b = c(N+1:2*N+1);`.

the following truncation with modified coefficients:

$$\bar{f}_N(x) \coloneqq \sum_{k=0}^{N} \frac{\hat{b}_N[k]}{\hat{b}_N[0]} \langle f, w \cdot \bar{T}_k \rangle \bar{T}_k(x). \tag{13}$$

Note that $\hat{b}_N[0]/\hat{b}_N[0] = 1$, and all other terms are strictly less than one. It is not hard to show this damped series preserves positivity. We prove the following fact as Lemma C.7 in the appendix:

Fact A.2. *If $f : [-1, 1] \to \mathbb{R}^{\geq 0}$ is a non-negative function, then the polynomial $\bar{f}_N(x)$ defined in (13) is non-negative for all $x \in [-1, 1]$.*

Beyond preserving non-negativity, as claimed in Fact 3.2, the Jackson damped Chebyshev approximation is more well-known for the fact that it provably provides a good *uniform* polynomial approximation to any Lipschitz function. For completeness, we give a proof of this fact as Theorem C.6 in the appendix. With Facts A.2 and 3.2 in place, we are ready to introduced the basic kernel polynomial method for approximating the spectral density s as Algorithm 5. This algorithm is identical to the "Jackson Kernel" KPM from [WWAF06]. Recall that, for now, we assume we have access to *exact* Chebyshev moment of the spectral density s for our matrix A. In Section A.2 we prove that Algorithm 5 is robust to using approximate moments.

Algorithm 5 Idealized Jackson Damped Kernel Polynomial Method

Input: Symmetric $A \in \mathbb{R}^{n \times n}$ with spectral density $s : [-1, 1] \to \mathbb{R}^{\geq 0}$, degree $N \in 4\mathbb{N}^+$.
Output: Density function $q : [-1, 1] \to \mathbb{R}^{\geq 0}$.

1: For $k = 0, \ldots, N$ compute $\tau_k = \frac{1}{n} \operatorname{tr}(\bar{T}_k(A)) = \langle s, \bar{T}_k \rangle$.
2: For $k = 0, \ldots, N$ compute $\hat{b}_N[k]$ as is (12).
3: Return $q = w \cdot \sum_{k=0}^{N} \frac{\hat{b}_N[k]}{\hat{b}_N[0]} \cdot \tau_k \cdot \bar{T}_k$.

Lemma A.3. *If $N \geq \frac{18}{\epsilon}$, then the function $q : [-1, 1] \to \mathbb{R}^{\geq 0}$ returned by Algorithm 5 is a probability density and satisfies:*

$$W_1(s, q) \leq \epsilon.$$

Proof. We first prove that q is a probability density. To see that it is positive, note that $h = \sum_{k=0}^{N} \frac{\hat{b}_N[k]}{\hat{b}_N[0]} \cdot \tau_k \cdot \bar{T}_k$ is a Jackson approximation to the positive function s/w, so is must be non-negative by Fact A.2. Since w is also non-negative, we conclude that $q = w \cdot h$ is as well. Then we consider q's integral. We need to show that $\int_{-1}^{1} q(x)dx = 1 = \int_{-1}^{1} s(x)dx$. Since $\bar{T}_0$ is a scaling of the constant function, it suffices to show that $\langle \bar{T}_0, q \rangle = \langle \bar{T}_0, s \rangle$. We have:

$$\langle \bar{T}_0, q \rangle = \tau_0 \cdot \langle \bar{T}_0, w \cdot \bar{T}_0 \rangle = \langle \bar{T}_0, s \rangle \cdot 1.$$

The first step follows directly from the orthogonality of the Chebyshev polynomials under the weight function w, which implies that $\langle \bar{T}_0, w \cdot \bar{T}_k \rangle = 0$ for all $k > 0$. We also use that $\langle \bar{T}_0, w \cdot \bar{T}_0 \rangle = 1$.

Next, we prove the approximation guarantee. Referring to the formulation of Wasserstein-1 distance from equation (3), we have that $W_1(s, q) = \sup \langle s - q, f \rangle$ where f is a 1-Lipschitz function. So, we want to show that any 1-Lipschitz f has small inner product with the difference between s and its degree-N Jackson approximation, q. To do so, we show that this inner product is actually

exactly equal to the inner product between s and a degree-N Jackson approximation to f. Since f is 1-Lipschitz, this approximation is guaranteed to be have small error. This key equivalency follows because, like a standard Chebyshev series approximation, the Jackson approximation can be viewed as the output of a symmetric linear operator applied to s.

Formally, we introduce notation for several linear operators needed to analyze (13). Let $\bar{\mathcal{T}} : \mathcal{F}([-1,1],\mathbb{R}) \to \mathcal{F}(\mathbb{N},\mathbb{R})$ be the operator mapping a function $f \in \mathcal{F}([-1,1],\mathbb{R})$ to its inner-product with the normalized Chebyshev polynomials. Define the transpose operator $\bar{\mathcal{T}}^* : \mathcal{F}(\mathbb{N},\mathbb{R}) \to \mathcal{F}([-1,1],\mathbb{R})$ to satisfy $\langle \bar{\mathcal{T}} f, g\rangle = \langle f, \bar{\mathcal{T}}^* g\rangle$ for any $g \in \mathcal{F}(\mathbb{N},\mathbb{R})$. Concretely, for $i \in \mathbb{N}$ and $x \in [-1,1]$,

$$[\bar{\mathcal{T}} f][i] \coloneqq \int_{-1}^{1} \bar{T}_i(x) f(x) dx \qquad \text{and} \qquad [\bar{\mathcal{T}}^* g](x) \coloneqq \sum_{i=0}^{\infty} \bar{T}_i(x) g[i]. \tag{14}$$

We also define operators $\mathcal{W} : \mathcal{F}([-1,1],\mathbb{R}) \to \mathcal{F}([-1,1],\mathbb{R})$ and $\mathcal{I} : \mathcal{F}(\mathbb{N},\mathbb{R}) \to \mathcal{F}(\mathbb{N},\mathbb{R})$ as follows:

$$[\mathcal{W} f](x) \coloneqq w(x) f(x) = \frac{1}{\sqrt{1-x^2}} f(x) \qquad \text{and} \qquad [\mathcal{I} g][i] \coloneqq g[i].$$

Note that $\mathcal{I}$ is an identity operator. For any $N \in 4\mathbb{N}$, we define $\mathcal{B}_N : \mathcal{F}(\mathbb{N},\mathbb{R}) \to \mathcal{F}(\mathbb{N},\mathbb{R})$ as:

$$[\mathcal{B}_N g](i) \coloneqq \begin{cases} \frac{\hat{\beta}_N[i]}{\hat{\beta}_N[0]} g(i) & \text{for } 0 \le i \le N \\ 0 & i > N. \end{cases}$$

The operators $\mathcal{W}$, $\mathcal{I}$, and $\mathcal{B}_N$ are all commutative with respect to the inner-products in their respective spaces. Specifically, for $f_1, f_2 \in \mathcal{F}([-1,1],\mathbb{R})$ and $g_1, g_2 \in \mathcal{F}(\mathbb{N},\mathbb{R})$, $\langle f_1, \mathcal{W} f_2\rangle = \langle \mathcal{W} f_1, f_2\rangle$, $\langle g_1, \mathcal{I} g_2\rangle = \langle \mathcal{I} g_1, g_2\rangle$, and $\langle g_1, \mathcal{B}_N g_2\rangle = \langle \mathcal{B}_N g_1, g_2\rangle$. Also note that by orthogonality of the Chebyshev polynomials under w, $\bar{\mathcal{T}}^* \bar{\mathcal{T}} \mathcal{W}$ is the identity operator on $\mathcal{F}([-1,1],\mathbb{R})$ and so is $\mathcal{W} \bar{\mathcal{T}}^* \bar{\mathcal{T}}$.

With these operators defined, the remainder of the proof is short. We have via direct calculation:

$$\begin{aligned} \langle f, s - q\rangle &= \langle f, s - \mathcal{W} \bar{\mathcal{T}}^* \mathcal{B}_N \bar{\mathcal{T}} s\rangle \\ &= \langle f, \mathcal{W} \bar{\mathcal{T}}^* (\mathcal{I} - \mathcal{B}_N) \bar{\mathcal{T}} s\rangle = \langle \bar{\mathcal{T}}^* (\mathcal{I} - \mathcal{B}_N) \bar{\mathcal{T}} \mathcal{W} f, s\rangle = \langle f - \bar{\mathcal{T}}^* \mathcal{B}_N \bar{\mathcal{T}} \mathcal{W} f, s\rangle. \end{aligned}$$

Note that $\bar{\mathcal{T}}^* \mathcal{B}_N \bar{\mathcal{T}} \mathcal{W} f$ is exactly the degree-N Jackson approximation to f. So by Fact **??**, if f is a 1-Lipschitz function, $\|f - \bar{\mathcal{T}}^* \mathcal{B}_N \bar{\mathcal{T}} \mathcal{W} f\|_\infty \le 18/N$. Since s is a non-negative function that integrates to 1, it follows that $\langle f, s-q\rangle = \langle f - \bar{\mathcal{T}}^* \mathcal{B}_N \bar{\mathcal{T}} \mathcal{W} f, s\rangle \le 18/N$. Since $W(s,q) = \sup_{\text{1-Lipschitz } f} \langle f, s-q\rangle$, we conclude that $W(s,q) \le \epsilon$ as long as as long as $N \ge 18/\epsilon$. □

Remark. Given access to the Chebyshev polynomial moments, $\mathrm{tr}(\bar{T}_0(A)), \ldots, \mathrm{tr}(\bar{T}_N(A))$, Algorithm 5 can be implemented in $O(1/\epsilon)$ additional time. The function it returns is an $O(1/\epsilon)$ degree polynomial times the closed form function w. The polynomial can be represented as a sum of Chebyshev polynomials, or converted to standard monomial form in $O(1/\epsilon^2)$ time. The function is easily plotted or integrated over a range – see discussion around Fact B.2 for more details.

A.2 Full Kernel Polynomial Method

Since it is not possible to efficiently compute the exact Chebyshev polynomial moments, we need to show that the kernel polynomial method can work with approximations to these moments,

Algorithm 6 Jackson Damped Kernel Polynomial Method

Input: Symmetric $A \in \mathbb{R}^{n\times n}$ with spectral density $s : [-1,1] \to \mathbb{R}^{\geq 0}$, degree parameter $N \in 4\mathbb{N}^+$, algorithm $\mathcal{M}(A)$ that computes moment approximations $\tilde{\tau}_1, \ldots, \tilde{\tau}_N$ with the guarantee that $|\tilde{\tau}_k - \frac{1}{n}\operatorname{tr}(\bar{T}_k(A))| \leq 1/N^2$ for all k.

Output: Density function $q : [-1,1] \to \mathbb{R}^{\geq 0}$.

1: For $k = 1, \ldots, N$ use $\mathcal{M}$ to compute $\tilde{\tau}_1, \ldots, \tilde{\tau}_N$ as above. Set $\tilde{\tau}_0 = 1/\sqrt{\pi}$.

2: For $k = 0, \ldots, N$ compute $\hat{b}_N[k]$ as is (12).

3: Compute polynomial $\tilde{s}_N = w \cdot \sum_{k=0}^{N} \frac{\hat{b}_N[k]}{\hat{b}_N[0]} \cdot \tilde{\tau}_k \cdot \bar{T}_k$.

4: Return the probability density $q = \left(\tilde{s}_N + \frac{w\sqrt{2}}{N\sqrt{\pi}}\right) / \left(1 + \frac{\sqrt{2\pi}}{N}\right)$.

computed e.g. using a stochastic trace estimator as described in Section 4. Here, we first prove a general result on the accuracy of approximation needed to ensure we obtain a good spectral density estimation. Specifically, we analyze the following "robust" version of Algorithm 5.

The final transformation of $\tilde{s}_N$ in Line 4 of Algorithm 6 ensures that we return a proper density, since error incurred by approximating $\frac{1}{n}\operatorname{tr}(\bar{T}_k(A)) = \langle s, \bar{T}_k\rangle$ could leave the function with negative values. So, we shift by a small positive function, and rescale to maintain unit integral. Our main result on the error of Algorithm 6, which parallels Lemma A.3 for Algorithm 5, is as follows:

Lemma A.4. *If $N \geq \frac{18}{\epsilon}$, then the function $q : [-1,1] \to \mathbb{R}^{\geq 0}$ returned by Algorithm 6 is a probability density and satisfies:*

$$W_1(s, q) \leq 2\epsilon.$$

Proof. We first prove that q is a probability distribution. Let s_N denote the ideal distribution returned by Algorithm 5 if exact Chebyshev moments were used. I.e.,

$$s_N = w \cdot \sum_{k=0}^{N} \frac{\hat{b}_N[k]}{\hat{b}_N[0]} \cdot \tau_k \cdot \bar{T}_k$$

where $\tau_k = \frac{1}{n}\operatorname{tr}(\bar{T}_k(A)) = \langle s, \bar{T}_k\rangle$. Note that for any density s, $\tau_0 = \langle s, \bar{T}_0\rangle = 1/\sqrt{\pi}$. Let $\Delta_k = \tilde{\tau}_k - \tau_k$. We have $\tilde{s}_N(x) = s_N + \sum_{k=1}^{N} \Delta_k \frac{\hat{b}_N[k]}{\hat{b}_N[0]} w(x)\bar{T}_k(x)$. Define functions $\eta = s_N/w$ and $\tilde{\eta} = \tilde{s}/w$. It follows that for any $x \in [-1,1]$,

$$\left|\tilde{\eta}(x) - \eta(x)\right| = \left|\sum_{k=1}^{N} \frac{\hat{b}_N[k]}{\hat{b}_N[0]} \Delta_k \bar{T}_k(x)\right| \leq \frac{\sqrt{2}}{N\sqrt{\pi}}. \tag{15}$$

The last inequality uses that $0 \leq \hat{b}_N[k]/\hat{b}_N[0] \leq 1$ and for $x \in [-1,1]$, $\bar{T}_k(x) \leq \sqrt{2/\pi}$ for $k \geq 1$. Since η is a non-negative function, from (15) we can conclude that the function $\tilde{\eta} + \frac{\sqrt{2}}{N\sqrt{\pi}}$ is non-negitive, and thus $w \cdot (\tilde{\eta} + \frac{\sqrt{2}}{N\sqrt{\pi}}) = \tilde{s}_N + w\frac{\sqrt{2}}{N\sqrt{\pi}}$ is also non-negative. The density of this function is $\int_{-1}^{1} \tilde{s}_N(x)dx + \frac{\sqrt{2}}{N\sqrt{\pi}} \int_{-1}^{1} w(x)dx = 1 + \frac{\sqrt{2\pi}}{N}$, so dividing by $1 + \frac{\sqrt{2\pi}}{N}$ gives a probability density.

Next we prove the approximation guarantee. By Lemma A.3 we know that $W_1(s, s_N) \leq \epsilon$, so if we can show that $W_1(s_N, q) \leq \epsilon$, then by triangle inequality we will have shown that $W_1(s, q) \leq W_1(s, s_N) + W_1(s_N, q) \leq 2\epsilon$.

To bound $W_1(s_N, q)$, we need to show that $\langle f, s_N - q\rangle \leq \epsilon$ for any 1-Lipschitz function $f \in \mathcal{F}([-1,1],\mathbb{R})$. Without loss of generality, we can assume that $\int_{-1}^{1} f(x)dx = 0$, as the 1-Lipschitz function $f' = f - \int_{-1}^{1} f(x)dx$ satisfies $\langle f, s_N - q\rangle = \langle f', s_N - q\rangle$ (since s_N and q are both probability densities). If $\int_{-1}^{1} f(x)dx = 0$, $f(x)$ must be zero for some $x \in [-1,1]$, and since it is also 1-Lipschitz we can in turn bound $\|f\|_\infty \leq 1$.[12] We can then bound the inner product:

$$\langle f, s_N - q\rangle \leq \|f(\bar{s}_N - q)\|_1 \leq \|f\|_\infty \|\bar{s}_N(x) - q(x)\|_1 \leq \left\| w \cdot \left(\eta - \frac{\tilde{\eta} + \frac{\sqrt{2}}{N\sqrt{\pi}}}{1 + \frac{\sqrt{2\pi}}{N}}\right)\right\|_1$$
$$\leq \underbrace{\left\| w \cdot \left(\eta - \tilde{\eta} - \frac{\sqrt{2}}{N\sqrt{\pi}}\right)\right\|_1}_{z_1} + \underbrace{\left\| \frac{\sqrt{2\pi}}{N} \cdot w \cdot \left(\tilde{\eta} + \frac{\sqrt{2}}{N\sqrt{\pi}}\right)\right\|_1}_{z_2}$$

The last inequality uses the fact that $1 - \frac{1}{1+\gamma} \leq \gamma$ for $0 \leq \gamma \leq 1$, which we apply with $\gamma = \frac{\sqrt{2\pi}}{N}$. Using the fact that $\|w\|_1 = \int_{-1}^{1} \frac{1}{\sqrt{1-x^2}} dx = \pi$ and the bound on $\|\eta - \tilde{\eta}\|_\infty$ from (15), we have

$$z_1 \leq \|w\|_1 \cdot \left\|\eta - \tilde{\eta} - \frac{\sqrt{2}}{N\sqrt{\pi}}\right\|_\infty \leq \frac{2\pi\sqrt{2}}{N\sqrt{\pi}}.$$

Examining z_2, recall that we showed earlier that $w(\tilde{\eta} + \frac{\sqrt{2}}{N\sqrt{\pi}}) = \tilde{s}_N + \frac{\sqrt{2}}{N\sqrt{\pi}} w$ has ℓ_1 norm $1 + \frac{\sqrt{2\pi}}{N}$. So we have $z_2 \leq \frac{\sqrt{2\pi}}{N}(1 + \frac{\sqrt{2}}{N\sqrt{\pi}}) \leq \frac{2\sqrt{2\pi}}{N}$ for all $N \geq 1$.

Compiling the bounds on z_1 and z_2, we have that for all 1-Lipschitz f, $\langle f, s_N - q\rangle \leq \frac{4\sqrt{2\pi}}{N} \leq \frac{11}{N}$, and thus $W_1(\bar{s}_N, q) \leq \frac{11}{N}$. For $N \geq \frac{18}{\epsilon}$ we conclude that $W_1(\bar{s}_N, q) \leq \epsilon$. Applying triangle quality as discussed above completes the proof. □

Lemma A.4 is exactly analogous to Lemma 3.4. We can take advantage of the result by using the Hutchinson's based method from Section 4 or the sublinear time method from Section 5 to obtain the approximations for the Chebyshev moments required by Algorithm 6. The end result is that we can obtain the same bounds as Theorem 1.4 and Theorem 1.3 with $\ell = \max(1, \frac{C'}{n}\epsilon^{-4}\log^2(\frac{1}{\epsilon\delta}))$ and $\epsilon_{\mathsf{MV}} = C''\epsilon^{-4}$, respectively. The slightly worse ϵ dependence follows from the fact that Algorithm 6 has a more stringent requirement on the approximate Chebyshev moments used than Algorithm 1.

B Approximate Eigenvalues from Spectral Density Estimate

Algorithm 5 and Algorithm 6 in the previous sections output a closed form representation of a distribution q which is close in Wasserstein-1 distance to s. In particular, the distribution output is continuous. Alternatively, we describe a simple greedy algorithm (Algorithm 7) that recovers a list of n eigenvalues $\tilde{\Lambda} = [\tilde{\lambda}_1, \ldots, \tilde{\lambda}_n]$ such that $\|\Lambda - \tilde{\Lambda}\|_1 \leq 3n\epsilon$, which implies that the discrete distribution associated with $\tilde{\Lambda}$ is 3ϵ close to s in Wassersetin-1 distance. Formally:

[12] Let z maximize $f(x)$. Since f is 1-Lipschitz we have $f(z) \leq |x - z| - f(x)$ for all x. Integrating both sides from -1 to 1, we have $2f(z) \leq (z^2 + 1) - 0 \leq 2$. So, $f(z) \leq 1$.

Theorem B.1. *Let s be a spectral density and let q be a density on $[-1,1]$ such $W_1(s,q) \leq \epsilon$ for $\epsilon \in (0,1)$. As long as q can be integrated over any subinterval of $[-1,1]$ (e.g., has a closed form antiderivative), there is an algorithm (Algorithm 7) that computes $1/\epsilon$ such integrals and in $O\left(n + 1/\epsilon\right)$ additional time outputs a list of n values $\tilde{\Lambda} = [\tilde{\lambda}_1, \ldots, \tilde{\lambda}_n]$ such that $\|\Lambda - \tilde{\Lambda}\|_1 \leq 3n\epsilon$.*

At a high-level, Algorithm 7 computes a grid with spacing ϵ for the interval $[-1,1]$, "snaps" the mass of the continuous density onto the nearest point in the grid, and then readjusts the resulting point masses to a distribution where every point mass is divisible by $1/n$ (and can therefore be represented by a certain number of eigenvalues). It does so by iteratively shifting fractional masses to the next point in the grid so that the mass at the current point is divisible by $1/n$.

The method requires computing the mass $\int_a^b q(x)dx$ where $-1 \leq a < b \leq 1$. For Algorithms 5 and 6, q is written as $q = w \cdot p$ where p is a degree N polynomial written as a sum of the first $N+1$ Chebyshev polynomials. So to compute the integral $\int_a^b q(x)dx$, we just need to compute the integral $\int_a^b T_k(x)w(x)dx$ for any $k \in 0, \ldots, N$. We can do so using the following closed form expression (see Appendix D for a short derivation):

Fact B.2. *For $k \in \mathbb{N}^{>0}$ and $-1 \leq a < b \leq 1$ we have that*

$$\int_a^b \frac{T_k(x)}{\sqrt{1-x^2}}dx = \frac{-\cos(k \sin^{-1} b)}{k} - \frac{-\cos(k \sin^{-1} a)}{k}$$

For $k = 0$, $T_k(x) = 1$ for all x and we have that $\int_a^b T_k(x)w(x)dx = \sin^{-1}(b) - \sin^{-1}(a)$.

Using the above fact, when $q = w \cdot p$ for a degree N polynomial p, we can compute $\int_a^b q(x)dx$ in $O(N)$ time. In our main results $N = O(1/\epsilon)$, so this cost is small.

Algorithm 7 Approximate Eigenvalues from Spectral Density

Input: Spectral density $q : [-1,1] \to \mathbb{R}^+$, integer n.
Output: Vector $\tilde{\Lambda} = [\tilde{\lambda}_1, \ldots, \tilde{\lambda}_n]$.

1: compute $\vec{v} = (v_{-1+\epsilon}, v_{-1+2\epsilon} \ldots, v_0, v_\epsilon, \ldots, v_1)$ such that $v_t = \int_{t-\epsilon}^t q(x)dx$
2: **for** t in $(-1+\epsilon, -1+2\epsilon \ldots, 0, \epsilon, \ldots, 1)$ **do**
3: $\quad r \leftarrow v_t - \lfloor v_t \rfloor_{1/n}$ ▷ $\lfloor v_t \rfloor_{1/n}$ is the largest value $\leq v_t$ that is divisible by $\frac{1}{n}$
4: $\quad v_{t+\epsilon} \leftarrow r + v_{t+\epsilon}$
5: $\quad$ Set $n \cdot \lfloor v_t \rfloor_{1/n}$ values in $\hat{\Lambda}$ to be t
6: return $\hat{\Lambda}$

Proof of Theorem B.1. Consider the output $\tilde{\Lambda}$ of Algorithm 7 with input q and n. Notice that $W_1(v,q) \leq \epsilon$ by the definition of v and the earthmover's definition of the Wasserstein distance. Hence, by triangle inequality, we have that $W_1(v,s) \leq 2\epsilon$. Let $\tilde{v}$ be the vector of masses after the shifting procedure (Line 4) in the for-loop of the algorithm. Notice that $\tilde{v}$ is the distribution corresponding to having n equally weighted point-masses on the points in $\tilde{\Lambda}$. Since the procedure in Line 4 moves at most $1/n$ mass at most ϵ distance in each iteration, we have $W_1(v,\tilde{v}) \leq \epsilon$ by the earthmover's distance definition of the Wasserstein-1 distance. It follows then that $W_1(\tilde{v}, s) \leq 3\epsilon$.

□

We note that there are other options beyond Algorithm 7 for discretizing a continuous density return by the Jackson damped kernel polynomial method – i) the optimal discretization of a continuous density on the interval $[-1, 1]$ into n equally-weighted point-masses, and ii) an algorithm by [CKSV18] that can be seen as a combination of Algorithm 7 and the optimal method.

Optimal Discretization. Given the continuous density q, consider the discrete density that results from the following procedure:

1. Initialize $t = -1$, then repeat the following steps until $t = 1$.
2. Let $t' \geq t$ be the smallest value such that $\int_t^{t'} q(x)dx = \frac{1}{n}$.
3. Place a point-mass at $\mathbf{E}_{x\sim q}\left[x \mid x \in [t, t']\right]$. I.e. a point-mass is placed in the interval $[t, t']$ at the point given by the conditional distribution of q on the interval.
4. Update $t \leftarrow t'$.

The values $\tilde{\Lambda} = \tilde{\lambda}_1, \ldots, \tilde{\lambda}_n$ given by the point-masses computed by the aforementioned procedure is a optimal discretization of q into n equally-weighted point-masses on $[-1, 1]$ in terms of Wasserstein-1 distance. To see why this is the case, consider the first $1/n$ fraction of the mass of the density q, i.e. the smallest $t > -1$ such that $\int_{-1}^{t} q(x)dx = 1/n$. The policy minimizing the earthmover's distance to any n equally-weighted point-wise masses must "move" the mass of q on the interval $[-1, t]$ to the point-mass closest to -1. Hence, it is sufficient to restrict our attention to the interval $[-1, t]$ when computing the smallest point-mass, i.e. the mass closest to -1. Now that we are constrained to looking at the interval $[-1, t]$ one can check that the point-mass minimizing the earthmover's distance to q, restricted to $[-1, t]$, is the point-mass at $\mathbf{E}_{x\sim q}\left[x \mid x \in [-1, t]\right]$. The optimality of the procedure follows from making this argument inductively for all n point-masses.

We note that all steps of the procedure takes roughly $O(n)$ time, although a numerical integration technique or binary search would need to be used to find each t' to sufficiently high accuracy.

A result combining the greedy discretization in Algorithm 7 and the optimal discretization is given in [CKSV18]. They compute a fractional discretization on an ϵ-spaced grid of $[-1, 1]$, as in Algorithm 7, but then compute the eigenvalues using the conditional expectation of every $1/n$ fraction of mass based on the discrete density on the grid.

C Positive Polynomial Approximation

In this section, we introduce Jackson's powerful result from 1912 on the uniform approximation of Lipschitz continuous periodic functions by low-degree trigonometric polynomials [Jac12, Jac30]. This result will directly translate to the result for algebraic polynomials needed to analyze the kernel polynomial method. We start with basic definitions and preliminaries below.

C.1 Fourier Series Preliminaries

Definition C.1 (Fourier Series)**.** A function f with period 2π that is integrable on the length of that period can be written via the Fourier series:

$$f(x) = \frac{\alpha_0}{2} + \sum_{k=1}^{\infty} \alpha_k \cos(kx) + \beta_k \sin(kx)$$

where

$$\alpha_k = \frac{1}{\pi} \int_{-\pi}^{\pi} f(x) \cos(kx) dx \qquad\qquad \beta_k = \frac{1}{\pi} \int_{-\pi}^{\pi} f(x) \sin(kx) dx.$$

Equivalently we can write f in *exponential form* as:

$$f(x) = \sum_{k=-\infty}^{\infty} \hat{f}_k e^{ikx}$$

where $i = \sqrt{-1}$, $\hat{f}_0 = \alpha_0/2$, $\hat{f}_k = \hat{f}^*_{|k|}$ for $k < 0$, and for $k > 0$,

$$\hat{f}_k = \frac{1}{2}(\alpha_k - i\beta_k).$$

If the Fourier series of a periodic function f has $\hat{f}_k = 0$ for $k > N$ (equivalently, $\alpha_k = \beta_k = 0$ for $k > N$), we say that f is a degree N trigonometric polynomial.

In working with Fourier series, we require the two standard convolution theorems:

Claim C.2 (First Convolution Theorem)**.** *Let f, g be integrable 2π-periodic functions with exponential form Fourier series coefficients $[\hat{f}_k]_{k=-\infty}^{\infty}$ and $[\hat{g}_k]_{k=-\infty}^{\infty}$, respectively. Let h be their convolution:*

$$h(x) = [f * g](x) = \int_{-\pi}^{\pi} f(u) g(x-u) du.$$

The exponential form Fourier series coefficients of h, $[\hat{h}_k]_{k=-\infty}^{\infty}$, satisfy:

$$\hat{h}_k = 2\pi \cdot \hat{f}_k \hat{g}_k$$

Claim C.3 (Second Convolution Theorem)**.** *Let f, g be integrable 2π-periodic functions with exponential form Fourier series coefficients $[\hat{f}_k]_{k=-\infty}^{\infty}$ and $[\hat{g}_k]_{k=-\infty}^{\infty}$, respectively. Let h be their product:*

$$h(x) = f(x) \cdot g(x).$$

The exponential form Fourier series coefficients of h, $[\hat{h}_k]_{k=-\infty}^{\infty}$, satisfy:

$$\hat{h}_k = \sum_{j=-\infty}^{\infty} \hat{f}_j \cdot \hat{g}_{k-j}$$

In other words, the Fourier coefficients of h are the discrete convolution of those of f and g.

C.2 Jackson's Theorem for Trigonometric Polynomials

We seek a low-degree trigonometric polynomial $\tilde{f}$ that is a good *uniform* approximation to any sufficiently smooth periodic function f. I.e., we want $\|f - \tilde{f}\|_\infty < \epsilon$ where $\|z\|_\infty$ denotes $\|z\|_\infty = \max_x z(x)$. A natural choice for $\tilde{f}$ is the truncated Fourier series $\sum_{k=-N}^{N} c_k e^{ikx}$, but this does not lead to good uniform approximation in general. Instead, Jackson showed that better accuracy can be obtained with a Fourier series with *damped coefficients*, which is equivalent to the *convolution* of f with an appropriately chosen "bump" function (aka kernel), defined below:

Definition C.4 (Jackson Kernel)**.** For any positive integer m, let b be the $2m-2$ degree trigonometric polynomial:

$$b = \left(\frac{\sin(mx/2)}{\sin(x/2)}\right)^4 = \sum_{k=-2m+2}^{2m-2} \hat{b}_k e^{ikx},$$

which has exponential form coefficients $\hat{b}_{-2m+2}, \ldots, \hat{b}_0, \ldots, \hat{b}_{2m-2}$ equal to:

$$\hat{b}_{-k} = \hat{b}_k = \sum_{j=-m}^{m-k} (m - |j|) \cdot (m - |j+k|) \qquad \text{for } k = 0, \ldots, 2m-2. \tag{16}$$

When m is odd it is easy to see that b is a degree $2m-2$ trigonometric polynomial. Specifically, for odd m we have the well known Fourier series of the periodic sinc function $s(x) = \frac{\sin(mx/2)}{\sin(x/2)} = \sum_{k=-(m-1)/2}^{(m-1)/2} e^{ikx}$. We then apply the convolution theorem (Claim C.3) to $s(x) \cdot s(x)$. to see that $s^2(x) = \left(\frac{\sin(mx/2)}{\sin(x/2)}\right)^2$ is an $m-1$ degree trigonometric polynomial with coefficients $c_{-k} = c_k = m-k$. Applying it again to $s^2(x) \cdot s^2(x)$ yields (16). For a derivation of (16) when m is even, we refer the reader to [Jac30] or [Lor66].

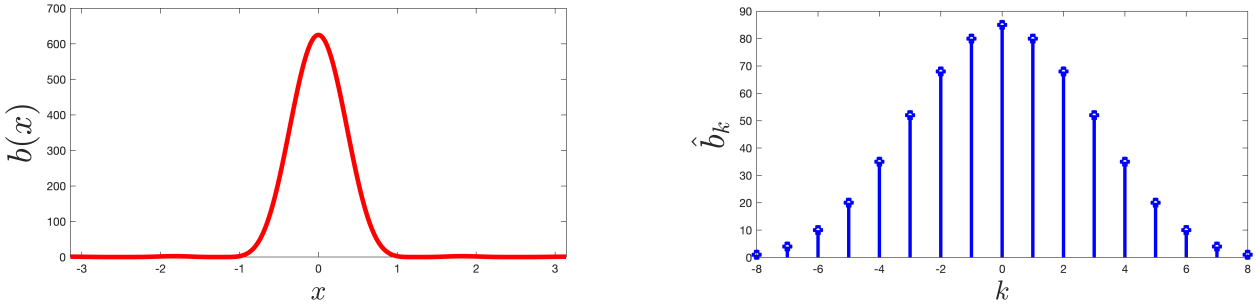

Figure 6: Jackson's bump function $b(x)$ for $m = 5$, alongside its Fourier series coefficients.

Jackson's main result is as follows. We include a short proof for completeness.

Theorem C.5 (Jackson [Jac12], see also [Jac30])**.** *Let f be a 2π-periodic, Lipschitz continuous function with Lipschitz constant λ. I.e., $|f(x) - f(y)| \leq \lambda|x-y|$ for all x, y. For integer m, let b be the bump function from Definition 16, with k^{th} Fourier ceofficients $\hat{b}_k$. The function $\tilde{f}(x) = \frac{1}{2\pi\hat{b}_0}\int_{-\pi}^{\pi} b(u)f(x-u)du$ satisfies:*

$$\|\tilde{f} - f\|_\infty \leq 9\frac{\lambda}{m}.$$

$\tilde{f}$ is a $2m-2$ degree trigonometric polynomial, and by the convolution theorem, its exponential form Fourier series coefficients are given by $\hat{\tilde{f}}_k = \frac{\hat{b}_k}{\hat{b}_0} \cdot \hat{f}_k$ for $k = -2m+2, \ldots, 2m-2$.

Remark. The function $\tilde{f}$ takes the form of a *damped* truncation of f's Fourier series: $\frac{\hat{b}_0}{\hat{b}_0} = 1$ and $\frac{\hat{b}_k}{\hat{b}_0}$ falls off towards zero as $k \to 2m-2$. After $2m-2$, the Fourier series coefficients from f are fully truncated to 0.

Proof. Recalling that $\hat{b}_0 = \frac{1}{2\pi}\int_{-\pi}^{\pi} b(x)dx$, we have that $\int_{-\pi}^{\pi} \frac{1}{2\pi\hat{b}_0} b(u)du = 1$, and thus

$$|\tilde{f}(x) - f(x)| \leq \int_{-\pi}^{\pi} \frac{1}{2\pi\hat{b}_0} b(u) \cdot |f(x) - f(x-u)|du.$$

By our Lipschitz assumption of f, we can bound $|f(x) - f(x-u)| \leq \lambda|u|$ and thus have:

$$\max_x |\tilde{f}(x) - f(x)| = \|\tilde{f} - f\|_\infty \leq \lambda \cdot \frac{\int_{-\pi}^{\pi} |u| b(u)du}{2\pi\hat{b}_0} = \lambda \cdot \frac{\int_0^{\pi} ub(u)du}{\int_0^{\pi} b(u)du}. \tag{17}$$

In the last equality, we use that b is symmetric about zero. We have that $2 \cdot \sin\left(\frac{u}{2}\right) \leq u \leq \pi \cdot \sin\left(\frac{u}{2}\right)$ for $x \in [0, \pi]$ and thus:

$$\int_0^{\pi} ub(u)du \leq \pi^4 \int_0^{\pi} u\frac{\sin(mu/2)^4}{u^4}du = \pi^4 m^2 \int_0^{\pi m} \frac{\sin(v/2)^4}{v^3}dv \leq \pi^4 m^2 \int_0^{\infty} \frac{\sin(v/2)^4}{v^3}dv.$$

The last integral evaluates of $\frac{\ln 2}{4}$, so overall we have $\int_0^{\pi} b(u) \cdot udu \leq \frac{\pi^4 \ln 2}{4} \cdot m^2$. Moreover we can check that:

$$\int_0^{\pi} b(u)du = \pi \cdot (\frac{2}{3}m^3 + \frac{1}{3}m) \geq \frac{2\pi}{3}m^3.$$

Plugging into (17) we have that:

$$\|\tilde{f} - f\|_\infty \leq 8.06\frac{\lambda}{m}.$$

The result follows. We note that the constant above is loose: numerical results suggest the bound can be improved to $\leq \frac{\pi}{2}\frac{\lambda}{m}$. □

Theorem C.5 translates to a result for *algebraic polynomials* via a standard transformation between Fourier series and Chebyshev series, which we detail below.

C.3 Jackson's Theorem for Algebraic Polynomials

Theorem C.6. *Let $f \in \mathcal{F}([-1,1], \mathbb{R})$ be a Lipschitz continuous function on $[-1,1]$ with Lipschitz constant λ. I.e., $|f(x) - f(y)| \leq \lambda|x - y|$ for all x, y. For integer m, let $\hat{b}_0, \ldots, \hat{b}_{2m-2}$ be the coefficients from (16). Let $c_k = \langle f, w \cdot \bar{T}_k \rangle$ be the k^{th} coefficient in f's Chebyshev polynomial expansion, where w and $\bar{T}_k$ are as defined in Section 2. The degree $(2m-2)$ algebraic polynomial*

$$\tilde{f}(x) = \sum_{n=0}^{2m-2} \frac{\hat{b}_k}{\hat{b}_0} c_k \cdot \bar{T}_k(x)$$

satisfies $\|\tilde{f} - f\|_\infty \leq 9\frac{\lambda}{m}$.

Proof. To translate from the trigonometric case to the algebraic setting, we will use the identity that for all k,

$$T_k(\cos\theta) = \cos(k\theta). \tag{18}$$

Consider any function $r \in \mathcal{F}([-1,1],\mathbb{R})$ with Chebyshev expansion coefficients $c_0, c_1, \ldots$, where $c_k = \langle r, w \cdot \bar{T}_k \rangle$. Transform r into a periodic function as follows: let $g(\theta) = r(\cos\theta)$ for $\theta \in [-\pi, 0]$ and let $h(\theta) = g(-|\theta|)$ for $\theta \in [-\pi, \pi]$. The function $h(\theta)$ is periodic, and also even, so its Fourier series has all coefficients $\beta_1, \beta_2, \ldots$ equal to 0. We thus have that

$$h(\theta) = \sum_{n=0}^{\infty} \alpha_k \cos(n\theta),$$

where

$$\alpha_0 = \frac{1}{2\pi}\int_{-\pi}^{\pi} h(\theta)\cos(k\theta)d\theta = \frac{1}{\pi}\int_{-\pi}^{0} g(\theta)\cos(k\theta)d\theta$$

and, for $n > 0$,

$$\alpha_k = \frac{1}{\pi}\int_{-\pi}^{\pi} h(\theta)\cos(k\theta)d\theta = \frac{2}{\pi}\int_{-\pi}^{0} g(\theta)\cos(k\theta)d\theta.$$

Using (18) and the fact that $\frac{d}{dx}\cos^{-1}(x) = \frac{1}{\sqrt{1-x^2}}$, we have:

$$\int_{-\pi}^{0} g(\theta)\cos(k\theta)d\theta = \int_{-1}^{1} r(x)T_k(x)\frac{1}{\sqrt{1-x^2}}dx.$$

We conclude that the Chebyshev coefficients of r are precisely a scaling of the Fourier coefficients of h. Specifically, since $\bar{T}_0 = \sqrt{\frac{2}{\pi}}T_0$ and $\bar{T}_k = \sqrt{\frac{1}{\pi}}T_k$, we have:

$$\sqrt{\frac{2}{\pi}}c_0 = \alpha_0, \qquad \sqrt{\frac{1}{\pi}}c_k = \alpha_k \text{ for } k > 0. \tag{19}$$

With this fact in hand, Theorem C.6 follows almost immediately from Theorem C.5. Specifically, given $f \in \mathcal{F}([-1,1],\mathbb{R})$ with Chebyshev series coefficients $c_0, c_1, \ldots$, we let $g(\theta) = f(\cos\theta)$ and $h(\theta) = g(-|\theta|)$. Let $\alpha_0, \alpha_1, \ldots$ denote h's non-zero Fourier coefficients. Then, let $\tilde{h}$ be the approximation to h given by Theorem C.5. $\tilde{h}$ is a $2m-2$ degree trigonometric polynomial and is even since h is even and the bump function b is symmetric. Denote $\tilde{h}$'s non-zero Fourier coefficients by $\tilde{\alpha}_0, \ldots, \tilde{\alpha}_{2m-2}$. We have that $\tilde{\alpha}_k = \frac{\hat{b}_k}{\hat{b}_0}\alpha_k$ for $0 \le k \le 2m-2$. Finally, let $\tilde{f} \in \mathcal{F}([-1,1],\mathbb{R})$ be defined by $\tilde{f}(\cos(\theta)) = h(-\theta)$. By (19), we have that $\tilde{f}$ is a degree $2m-2$ polynomial and its Chebyshev series coefficients $\tilde{c}_0, \ldots, \tilde{c}_{2m-2}$ are exactly equal to $\frac{\hat{b}_k}{\hat{b}_0}c_k$.

Moreover, we have $\|f - \tilde{f}\|_\infty = \max_{x\in[-1,1]} |f(x) - \tilde{f}(x)| = \max_x |h(x) - \tilde{h}(x)|$. By Theorem C.5 we have $\max_x |h(x) - \tilde{h}(x)| < 9\frac{\lambda}{m}$, so we conclude that $\|f - \tilde{f}\|_\infty < 9\frac{\lambda}{m}$. □

In addition to the main result of Theorem C.6, our SDE algorithm alsos require an additional property of the damped Chebyshev approximation $\tilde{f}$:

Lemma C.7. *For any non-negative function $f \in \mathcal{F}([-1,1],\mathbb{R})$ (not necessarily Lipschitz), let $\tilde{f}$ be as in Theorem C.6. We have that $\tilde{f}$ is also non-negative on $[-1,1]$.*

Proof. Let $h(\theta)$ and $\tilde{h}(\theta)$ be the 2π perioduc functions as in the proof of Theorem C.6. I.e., $h(\theta) = g(-|\theta|)$ where $g(\theta) = f(\cos\theta)$ and $\tilde{h}$ is the truncated, Jackson-damped approximation to h from Theorem C.5. If f is non-negative, then so is h, and since $\tilde{h}$ is the convolution of h with a non-negative function, it is non-negative as well. Finally, since $\tilde{f}(\cos(\theta)) = h(-\theta)$, we conclude that $\tilde{f}(x) \geq 0$ for $x \in [-1,1]$. □

D Derivation of Fact B.2

Let $x = \sin(u)$ then we have that $dx = \cos(u)du$. Substituting the change of variable in the integral and noting the fact that $T_k(\cos\theta) = \cos(k\theta)$ for $\theta \in [-\pi,\pi]$ gives us that

$$\begin{aligned}\int_a^b \frac{T_k(x)}{\sqrt{1-x^2}}dx &= \int_{\sin^{-1}a}^{\sin^{-1}b} \frac{\cos(k\cos^{-1}\sin(u))}{\sqrt{1-\sin^2(u)}}\cos(u)du = \int_{\sin^{-1}a}^{\sin^{-1}b} \cos(k(\pi/2-u))du \\ &= \frac{-\sin(k(\pi/2-u))}{k}\Big|_{\sin^{-1}a}^{\sin^{-1}b} = \frac{-\cos(ku)}{k}\Big|_{\sin^{-1}a}^{\sin^{-1}b}\end{aligned}$$

where we used the fact that $\cos^2(u)+\sin^2(u) = 1$ and $\int\cos(u)du = \sin(u)+c$.

E Proof of Fact 3.3

Proof. We start by doing a change of variables; set $x = \cos\theta$ and note that $dx = -\sin\theta d\theta$. Substituting this into the expression for $\langle f, w\cdot\bar{T}_k\rangle$ and noting that $T_k(\cos\theta) = \cos k\theta$ gives us that

$$\sqrt{\frac{2}{\pi}}\int_{-1}^{1} f(x)\frac{T_k(x)}{\sqrt{1-x^2}}dx = \sqrt{\frac{2}{\pi}}\int_{-\pi}^{0} -f(\cos\theta)(\cos k\theta)d\theta$$

since $\sqrt{1-\cos^2\theta} = \sin\theta$ and $dx = -\sin\theta d\theta$. Integrating by parts and noting that $(f(\cos\theta)\int -\cos k\theta d\theta)|_{-\pi}^{0} = -f(\cos\theta)\frac{\sin k\theta}{k}|_{-\pi}^{0} = 0$ gives us that

$$\langle f, w\cdot\bar{T}_k\rangle = \sqrt{\frac{2}{\pi}}\int_{-\pi}^{0}\frac{\sin k\theta}{k}\,df(\cos\theta).$$

We use the definition of the Riemann-Stieltjes integral and let $M \in \mathbb{N}^+$ be a parameter and $\mathcal{P}_M = \{-\pi = x_0 \leq \cdots \leq x_M = 0\}$ be the set of all M intervals partitioning the interval $[-\pi,0]$. Then for a partition $P \in \mathcal{P}_M$ we denote norm(P) to be the length of its longest sub-interval. The Riemann-Stieltjes integral $\int_{-\pi}^{0}\sin(k\theta)\,df(\cos\theta)$ can be written as

$$\int_{-\pi}^{0}\sin k\theta\; df(\cos\theta) = \lim_{\epsilon\to 0}\sup_{\substack{M,\ P\in\mathcal{P}_M \\ \text{s.t.norm}(P)\leq\epsilon}}\sum_{i=0}^{m-1}(f(\cos x_{i+1}) - f(\cos x_i))\sin kx_i.$$

Since $f(x) \in \text{lip}_1$ and $|\sin k\theta| \leq 1$ we can bound the magnitude of the above summation as

$$\left| \sum_{i=0}^{m-1} (f(\cos x_{i+1}) - f(\cos x_i)) \sin k x_i \right| \leq \sum_{i=0}^{m-1} \lambda |\cos x_{i+1} - \cos x_i| \leq 2.$$

The last inequality follows from the fact that $\cos(\theta)$ is 1-Lipschitz. Putting these bounds together gives us that $|\langle f, w \cdot \bar{T}_k \rangle| \leq 2\lambda/k$. $\square$