Analysis of stochastic Lanczos quadrature for spectrum approximation

Tyler Chen

March 29 2021

Acknowledgements

Joint work with: **Tom Trogdon** (UW) and **Shashanka Ubaru** (IBM Watson)

This material is based upon work supported by the National Science Foundation Graduate Research Fellowship Program under Grant No. DGE-1762114. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author and do not necessarily reflect the views of the National Science Foundation.

Introduction

Given an $n \times n$ symmetric matrix $\mathbf{A}$, the cumulative empirical spectral measure (CESM) $\Phi = \Phi(\mathbf{A}) : \mathbb{R} \to [0, 1]$ gives the fraction of eigenvalues less than a given threshold. That is,

$$\Phi(x) = \Phi(\mathbf{A})(x) := \sum_{i=1}^{n} \frac{1}{n} \mathbb{1}[\lambda_i(\mathbf{A}) \leq x] = \mathrm{tr}(n^{-1} \mathbb{1}[\mathbf{A} \leq x]),$$

where $\mathbb{1}[\,\cdot \leq x] : \mathbb{R} \to \{0, 1\}$ is the indicator function defined by $\mathbb{1}[s \leq x] = 1$ if $s \leq x$ and $\mathbb{1}[s \leq x] = 0$ if $s > x$.

Applications

Computing $\mathrm{tr}(f(\mathbf{A}))$ is an important task. This is related to the CESM because

$$\mathrm{tr}(f(\mathbf{A})) = n \int f(s) \mathrm{d}\Phi(s).$$

So, if we can approximate Φ, we can approximate $\mathrm{tr}(f(\mathbf{A}))$.

Applications

Computing the full CESM is expensive, but lots of applications of approximate CESMs:

1. computational physics and chemistry[1]
2. matrix norms, log-determinants, Estrada indices, triangle counts in a graph[2]
3. network motifs[3]
4. estimating the number of eigenvalues in an interval[4]
5. studying properties of Hessians during neural network training[5].

[1] Ducastelle and Cyrot-Lackmann 1970; Haydock, Heine, and Kelly 1975; Wheeler and Blumstein 1972; Weiße et al. 2006; Covaci, Peeters, and Berciu 2010; Sbierski et al. 2017; Schnack, Richter, and Steinigeweg 2020.

[2] Avron 2010; Ubaru, Saad, and Seghouane 2017; Han et al. 2017; Musco et al. 2019.

[3] Dong, Benson, and Bindel 2019.

[4] Napoli, Polizzi, and Saad 2016; Xi, Li, and Saad 2018.

[5] Ghorbani, Krishnan, and Xiao 2019; Papyan 2019; Yao et al. 2020.

Weighted CESM

For any unit vector $\mathbf{v}$, define the weighted CESM $\Psi(\mathbf{A}, \mathbf{v}) : \mathbb{R} \to [0, 1]$ by

$$\Psi(\mathbf{A}, \mathbf{v})(x) := \sum_{i=1}^{n} w_i \mathbb{1}[\lambda_i(\mathbf{A}) \leq x] = \mathbf{v}^\top \mathbb{1}[\mathbf{A} \leq x]\mathbf{v}$$

where $w_i = (\mathbf{v}^\top \mathbf{u}_i)^2$ and $\mathbf{u}_i$ is the eigenvector for $\lambda_i(\mathbf{A})$.

Note that if $\mathbb{E}[\mathbf{v}\mathbf{v}^\top] = \mathbf{I}$ then

$$\mathbb{E}[\Psi(\mathbf{A}, \mathbf{v})(x)] = \Phi(\mathbf{A})(x).$$

Algorithm

Natural algorithm:

$$\Phi(x) \approx \langle \mathrm{gq}_k(\Psi_i)(x) \rangle = \frac{1}{n_{\mathbf{v}}} \sum_{i=1}^{n_{\mathbf{v}}} \mathrm{gq}_k(\Psi(\mathbf{A}, \mathbf{v}_i))(x).$$

There are clearly two separate sources of error:

1. sample error associated with randomness in the weighted CESM $\Psi(\mathbf{A}, \mathbf{v})$
2. approximation error due to using a Gaussian quadrature $\mathrm{gq}_k(\Psi(\mathbf{A}, \mathbf{v}))$ to approximate $\Psi(\mathbf{A}, \mathbf{v})$.

Assuming the indicator of error d : (set of dists) $\times$ (set of dists) $\to \mathbb{R}_{\geq 0}$ satisfies the triangle inequality, we have

$$d(\Phi, \langle \tilde{\Psi}_i \rangle) \leq d(\Phi, \langle \Psi_i \rangle) + \langle d(\Psi_i, \mathrm{gq}_k(\Psi_i)) \rangle.$$

Goal

Determine the runtime (number of samples $n_{\mathbf{v}}$ and the number of Lanczos iterations k) required to obtain a Wasserstein distance of t between true CESM Φ and output of algorithm.

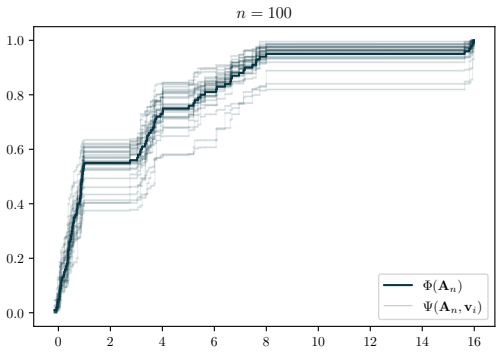
To do this we will study

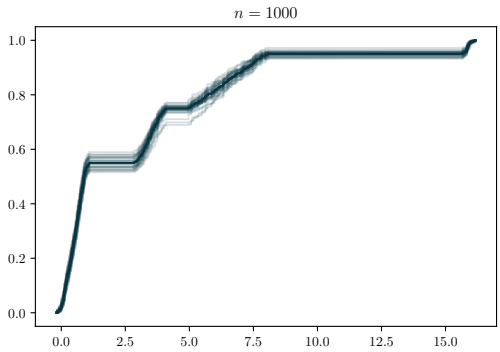
1. $\mathbb{P}\left[d_{\mathrm{W}}(\Phi, \langle \Psi_i \rangle) > t\right]$ as function of $n_{\mathbf{v}}$
2. $d_{\mathrm{W}}(\Psi_i, \mathrm{gq}_k(\Psi_i))$ as function of k

Wasserstein distance of distributions μ and ν,

$$d_{\mathrm{W}}(\mu, \nu) = \int |\mu(s) - \nu(s)| \mathrm{d}s = \sup \left\{ \int f(s) \mathrm{d}(\mu(s) - \nu(s)) : f \text{ is 1-Lipshitz} \right\}$$

Weighted distribution





Gaussian Quadrature/Lanczos

Let μ be a distribution function. A (discrete) distribution function ν corresponding to a set of points θ_i and weights d_i, $i = 1, 2, \ldots, k$ is said to be a Gaussian quadrature rule of degree k for μ if, for all polynomials p of degree at most $2k - 1$,

$$\int p(s)\mathrm{d}\mu(s) = \int p(s)\mathrm{d}\nu(s), \qquad \nu(x) = \sum_{i=1}^{k} d_i \mathbb{1}[\theta_i \leq x].$$

We denote such a distribution by $\mathrm{gq}_k(\mu)$.

Gaussian Quadrature/Lanczos

To compute a degree k Gaussian quadrature, compute upper left $k \times k$ principle submatrix $[\mathbf{T}]_{:k,:k}$ of Jacobi matrix for orthogonal polynomials of μ.

- nodes are eigenvalues
- weights are squares of first components of eigenvectors

If $\mu = \Psi(\mathbf{A}, \mathbf{v})$, this is can be done by Lanczos with $\mathbf{A}, \mathbf{v}$ Then

$$\mathrm{gq}_k(\Psi(\mathbf{A}, \mathbf{v})) = \Psi([\mathbf{T}]_{:k,:k}, \hat{\mathbf{e}}).$$

Sample complexity

By the unitary invariance property of Gaussian vectors $\mathbf{U}^\top \mathbf{v}$ is distributed like $\mathbf{v}$, so $\mathbf{v}^\top \mathbf{u}_i$ is distributed like $[\mathbf{v}]_i$, where $[\mathbf{v}]_i$ is the i-th coordinate of $\mathbf{v}$. Since $\mathbf{v}$ is obtained by sampling a Gaussian vector and normalizing,

$$w_i \sim \frac{X_i}{X_1 + \cdots + X_n},$$

where $X_1, \ldots, X_n$ are iid χ^2_1 random variables.

Let $m = n\Phi(x)$ (number of eigenvalues at most x). Then,

$$\Psi(\mathbf{A}, \mathbf{v})(x) = \sum_{i=1}^{m} w_i \sim \frac{X_1 + \cdots + X_m}{X_1 + \cdots + X_n} \sim \mathrm{Beta}\left(\frac{m}{2}, \frac{n-m}{2}\right).$$

Sample complexity

From this we obtain

$$\mathbb{P}\left[|\Phi(x) - \Psi(\mathbf{A}, \mathbf{v})(x)| > t\right] \leq 2\exp\left(-(n+2)t^2\right)$$

so

$$\mathbb{P}\left[|\Phi(x) - \langle\Psi(\mathbf{A}, \mathbf{v})(x)\rangle| > t\right] \leq 2\exp\left(-n_{\mathbf{v}}(n+2)t^2\right)$$

and

$$\mathbb{P}\left[|\Phi(x) - \langle\Psi(\mathbf{A}, \mathbf{v})(x)\rangle| > t,\ \forall x\right] \leq 2n\exp\left(-n_{\mathbf{v}}(n+2)t^2\right).$$

Finally,

$$\mathbb{P}\left[d_{\mathrm{W}}(\Phi, \langle\Psi(\mathbf{A}, \mathbf{v})\rangle) > t\|\mathbf{A}\|\right] \leq 2n\exp\left(-n_{\mathbf{v}}(n+2)t^2\right).$$

Quadrature Error

Suppose μ and ν are two probability distribution functions supported on $[a, b]$ whose moments are equal up to degree $k - 1$. Then,

$$d_{\mathrm{W}}(\mu, \nu) \leq (b - a)(1 + \pi^2/2)k^{-1} < 6(b - a)k^{-1}.$$

By properties of Gaussian quadrature, Ψ_i and $\mathrm{gq}_k(\Psi_i)$ share the $2k - 1$ moments. Thus, defining $l(\mathbf{A}) = |\lambda_{\max}(\mathbf{A}) - \lambda_{\min}(\mathbf{A})|$,

$$d_{\mathrm{W}}(\Psi_i, \mathrm{gq}_k(\Psi_i)) \leq 3\, l(\mathbf{A})\, k^{-1}$$

By triangle inequality,

$$d_{\mathrm{W}}(\langle \Psi_i \rangle, \langle \mathrm{gq}_k(\Psi_i) \rangle) \leq 3\, l(\mathbf{A})\, k^{-1}$$

Putting it together

We have obtained

$$\mathbb{P}\left[d_{\mathrm{W}}(\Phi, \langle\Psi(\mathbf{A}, \mathbf{v})\rangle > t\|\mathbf{A}\|\right] \leq 2n \exp\left(-n_{\mathbf{v}}(n+2)t^2\right).$$
$$d_{\mathrm{W}}(\langle\Psi_i\rangle, \langle \mathrm{gq}_k(\Psi_i)\rangle) \leq 3\, l(\mathbf{A})\, k^{-1}$$

Thus, if

$$n_{\mathbf{v}} > 4(n+2)^{-1}t^{-2}\log(2n\eta^{-1}), \qquad k > 4t^{-1}$$

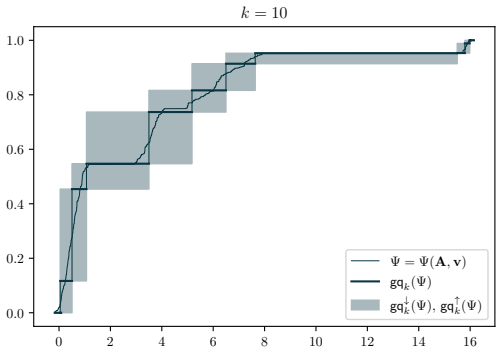
then

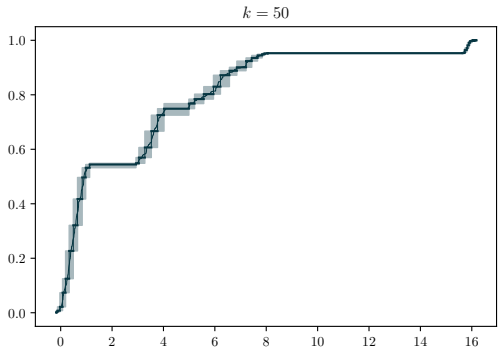
$$\mathbb{P}\left[d_{\mathrm{W}}(\Phi, \langle \mathrm{gq}_k(\Psi_i)\rangle) > t\, l(\mathbf{A})\right] < \eta.$$

An a posteriori approach

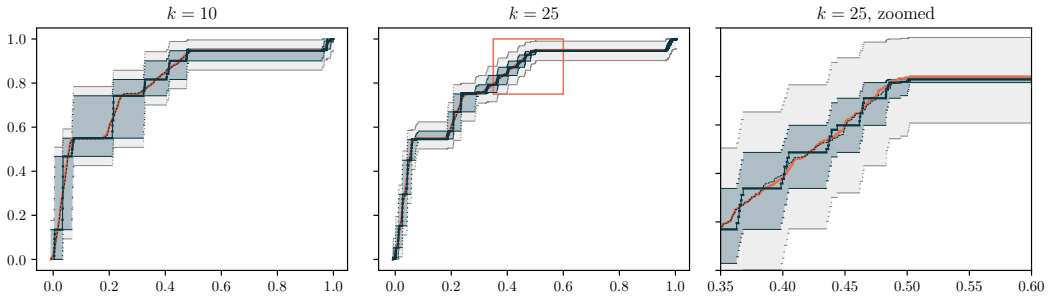
Karlin and Shapley 1972, Theorem 22.1: Suppose μ and ν are two probability distribution functions constant on the complement of $[a, b]$ whose moments are equal up to degree $k - 1$. Define $\gamma : [a, b] \to [0, 1]$ by $\gamma(x) = \mu(x) - \nu(x)$. Then γ is identically zero or changes sign at least $k - 1$ times.

An a posteriori approach

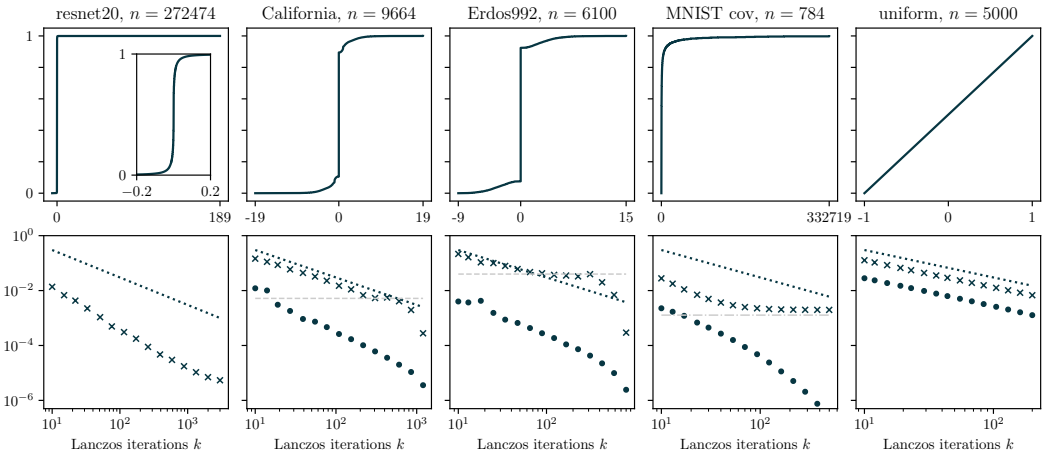




An a posteriori approach



Numerical examples



Conclusion

- If $t \gg n^{-1/2}$, then runtime is $t^{-1}(T_{\mathrm{mv}} + n)$
- Can prove matching lower bound for sample complexity, and in certain setups, iteration complexity
- Comparison with other related algorithms worth exploring