Krylov Subspace Methods and Matrix Functions

new directions in design, analysis, and applications

Tyler Chen

January 11, 2024

`chen.pw/slides`

About me

I am a numerical linear algebraist who likes working with nearby communities (theoretical computer science, computational science, optimization, etc.)

Academic history:

- Currently an Assistant Professor / Courant Instructor at New York University
  - Sponsor: Chris Musco
- PhD in Applied Math at University of Washington
  - Advisors: Anne Greenbaum and Tom Trogdon
- B.S. in Math and Physics at Tufts University, minor in Studio Art

My research program

Focus: design and analysis of practically fast and theoretically justified (randomized) algorithms for fundamental linear algebra tasks

Goal: develop tools to support the advancement of knowledge in current scientific applications

Mode: collaboration with a range of fields, and involvement and training of (underrepresented) students

Hope: provide conceptually simple insights into key problems

I am interested in diverse linear algebra problems

Compressed sensing/operator learning[1]

- $O(s/\epsilon)$ matrix-vector product algorithms for relative approximation with an s-row sparse matrix (no dimension dependence and matching lower bounds!)

Stochastic Optimization[2]

- First proof of $O(\sqrt{\kappa})$ convergence of minibatch stochastic gradient descent with heavy-ball momentum

Spectrum approximation[3]

- Sharp analysis of stochastic Lanczos quadrature algorithm proving spectrum approximation in Wasserstein distance in $\tilde{O}(\mathrm{nnz}(\mathbf{A})/\epsilon)$ time

Numerical Analysis/Random Matrix Theory[4]

- First proof of forward stability of Lanczos algorithm on random matrices

[1] Amsel, T. C., Halikias, Keles, Musco, and Musco 2024.
[2] Bollapragada, T. C., and Ward 2022.
[3] T. C., Trogdon, and Ubaru 2021.
[4] T. C. and Trogdon 2023.

What is a matrix function?

An $n \times n$ symmetric matrix $\mathbf{A}$ has real eigenvalues and orthonormal eigenvectors:

$$\mathbf{A} = \sum_{i=1}^{n} \lambda_i \mathbf{u}_i \mathbf{u}_i^\top.$$

The matrix function $f(\mathbf{A})$, induced by $f : \mathbb{R} \to \mathbb{R}$ and $\mathbf{A}$, is the matrix:

$$f(\mathbf{A}) = \sum_{i=1}^{n} f(\lambda_i) \mathbf{u}_i \mathbf{u}_i^\top.$$

Typically $\mathbf{A}$ is sparse while $f(\mathbf{A})$ is dense.

What do we want?

In this talk, think of the dimension n as big! E.g. $n = 10^6$ or 10^{12}.

- For reference, if $n = 10^6$:
  - matrix requires 8 terabytes of storage (not even enough disk space)
  - 100 vectors require 0.8 gigabytes of storage (can store in RAM)

We can't store $f(\mathbf{A})$, but we might instead compute:

$$f(\mathbf{A})\mathbf{b}, \qquad \mathbf{b}^\mathsf{T} f(\mathbf{A})\mathbf{b}, \qquad \operatorname{tr}(f(\mathbf{A})) = \sum_{i=1}^{n} f(\lambda_i).$$

Example. If $f(x) = x^{-1}$, then $f(\mathbf{A}) = \mathbf{A}^{-1}$ and $f(\mathbf{A})\mathbf{b} = \mathbf{A}^{-1}\mathbf{b}$ is the solution to the linear system $\mathbf{A}\mathbf{x} = \mathbf{b}$.

What do we want?

In this talk, think of the dimension n as big! E.g. $n = 10^6$ or 10^{12}.

- For reference, if $n = 10^6$:
  - matrix requires 8 terabytes of storage (not even enough disk space)
  - 100 vectors require 0.8 gigabytes of storage (can store in RAM)

We can't store $f(\mathbf{A})$, but we might instead compute:

$$f(\mathbf{A})\mathbf{b}, \qquad \mathbf{b}^\mathsf{T} f(\mathbf{A})\mathbf{b}, \qquad \operatorname{tr}(f(\mathbf{A})) = \sum_{i=1}^{n} f(\lambda_i).$$

Example. If $f(x) = x^{-1}$, then $f(\mathbf{A}) = \mathbf{A}^{-1}$ and $f(\mathbf{A})\mathbf{b} = \mathbf{A}^{-1}\mathbf{b}$ is the solution to the linear system $\mathbf{A}\mathbf{x} = \mathbf{b}$.

Why do we care?

Applications in many fields: quantum physics/chemistry,[5] biology,[6] statistics/data science,[7] network science,[8] machine learning,[9] high performance computing,[10] etc.

Common functions: inverse, exponential, square root, sign function.

[5] Eshof, Frommer, Lippert, Schilling, and Vorst 2002; Weiße, Wellein, Alvermann, and Fehske 2006; Schnalle and Schnack 2010.

[6] Estrada 2000.

[7] Barry and Pace 1999; Gardner, Pleiss, Weinberger, Bindel, and Wilson 2018; Jin and Sidford 2019.

[8] Avron 2010; Dong, Benson, and Bindel 2019.

[9] Ghorbani, Krishnan, and Xiao 2019; Papyan 2019; Granziol, Wan, and Garipov 2019; Yao, Gholami, Keutzer, and Mahoney 2020.

[10] Polizzi 2009; Li, Xi, Erlandson, and Saad 2019.

Example application: high performance computing

State of the art parallel eigensolvers such as FEAST and EVSL work by splitting the spectrum of $\mathbf{A}$ into pieces, which can each be solved on different machines in parallel.[11]

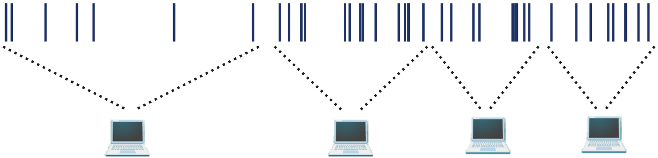
Let $\mathbb{1}[a \le x \le b] = 1$ if $x \in [a,b]$ and 0 otherwise. Then.

$$\text{number of eigenvalues in } [a,b] = \operatorname{tr}(\mathbb{1}[a \le \mathbf{A} \le b]).$$

[11]Polizzi 2009; Li, Xi, Erlandson, and Saad 2019.

Example application: high performance computing

State of the art parallel eigensolvers such as FEAST and EVSL work by splitting the spectrum of $\mathbf{A}$ into pieces, which can each be solved on different machines in parallel.[11]



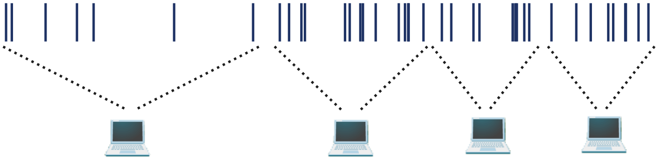
Let $\mathbb{1}[a \leq x \leq b] = 1$ if $x \in [a, b]$ and 0 otherwise. Then.

$$\text{number of eigenvalues in } [a, b] = \mathrm{tr}(\mathbb{1}[a \leq \mathbf{A} \leq b]).$$

[11] Polizzi 2009; Li, Xi, Erlandson, and Saad 2019.

Example application: high performance computing

State of the art parallel eigensolvers such as FEAST and EVSL work by splitting the spectrum of $\mathbf{A}$ into pieces, which can each be solved on different machines in parallel.[11]



Let $\mathbb{1}[a \leq x \leq b] = 1$ if $x \in [a, b]$ and 0 otherwise. Then.

$$\text{number of eigenvalues in } [a, b] = \operatorname{tr}(\mathbb{1}[a \leq \mathbf{A} \leq b]).$$

[11]Polizzi 2009; Li, Xi, Erlandson, and Saad 2019.

Part I: Rethinking how we think about existing algorithms

Many linear algebra algs are extremely effective in practice, but have limited theory.

- Analysis of Minibatch-SGD with Heavyball Momentum[12]
- Analysis of Stochastic Lanczos Quadrature and Kernel Polynomial Method[13]
- Stability of Lanczos-based methods[14]
- Analysis of Lanczos-FA[15]

[12] Bollapragada, T. C., and Ward 2022.
[13] T. C., Trogdon, and Ubaru 2021; T. C., Trogdon, and Ubaru 2022.
[14] T. C. and Trogdon 2023; T. C. 2023.
[15] T. C., Greenbaum, Musco, and Musco 2022; Xu and T. C. 2022; Amsel, T. C., Greenbaum, Musco, and Musco 2023.

Krylov subspace methods[16]

Krylov subspace methods are among the most widely used algorithms for solving large linear systems $\mathbf{Ax} = \mathbf{b}$; i.e. approximating $\mathbf{A}^{-1}\mathbf{b}$.

KSMs work by iteratively constructing a basis for the Krylov subspace:

$$\mathcal{K}_k(\mathbf{A}, \mathbf{b}) = \operatorname{span}\{\mathbf{b}, \mathbf{Ab}, \mathbf{A}^2\mathbf{b}, \ldots, \mathbf{A}^{k-1}\mathbf{b}\}.$$

Elements of the Krylov subspace are polynomials of $\mathbf{A}$ applied to $\mathbf{b}$:

$$c_0\mathbf{b} + c_1\mathbf{Ab} + \cdots + c_{k-1}\mathbf{A}^{k-1}\mathbf{b} = p(\mathbf{A})\mathbf{b},$$

where $p(x) = c_0 + c_1 x + \cdots + c_{k-1}x^{k-1}$.

[16] IEEE Top 10 algorithms of 20th century!

Error bounds for linear system solvers

The convergence of KSMs used to approximate $\mathbf{A}^{-1}\mathbf{b}$ are well understood.

Popular KSMs for linear systems, like Conjugate Gradient, efficiently compute iterates $\mathbf{x}_k$ which satisfy strong error guarantees:

$$
\begin{aligned}
\|\mathbf{A}^{-1}\mathbf{b} - \mathbf{x}_k\| &= \operatorname*{argmin}_{\mathbf{x}\in\mathcal{K}_k(\mathbf{A},\mathbf{b})} \|\mathbf{A}^{-1}\mathbf{b} - \mathbf{x}\| && \text{optimality} \\
&\lesssim \min_{\deg(p)<k} \max_{x\in\operatorname{spec}(\mathbf{A})} |x^{-1} - p(x)| && \text{bound on eigenvalues} \\
&\lesssim \exp\left(-\frac{2k}{\sqrt{\lambda_{\max}/\lambda_{\min}}} \right). && \text{bound on spectral interval}
\end{aligned}
$$

We also have very good techniques for posteriori error estimates; entire books![17]

[17] Meurant and Tichy 2024.

The Lanczos method for matrix function approximation

The Lanczos algorithm[18] iteratively constructs a basis $\mathbf{Q}_k = [\mathbf{q}_0, \ldots, \mathbf{q}_{k-1}]$ for the Krylov subspace and a symmetric tridiagonal matrix matrix $\mathbf{T}_k$ of recurrence coefficients.

Given a function $f(x)$, we define the Lanczos-FA iterate

$$\mathsf{lan\text{-}FA}_k(f) = \mathbf{Q}_k f(\mathbf{T}_k)\mathbf{Q}_k^\mathsf{T}\mathbf{b}.$$

Fact. If $f(x) = x^{-1}$ and $\mathbf{A}$ is positive definite, then $\mathsf{lan\text{-}FA}_k(f)$ is mathematically equivalent to the CG iterate (so we have error bounds and estimates).

For other functions the algorithm is still widely used, and performs remarkably well in practice. However, less theory is known about the error.

[18] Lanczos 1950.

The Lanczos method for matrix function approximation

The Lanczos algorithm[18] iteratively constructs a basis $\mathbf{Q}_k = [\mathbf{q}_0, \ldots, \mathbf{q}_{k-1}]$ for the Krylov subspace and a symmetric tridiagonal matrix matrix $\mathbf{T}_k$ of recurrence coefficients.

Given a function $f(x)$, we define the Lanczos-FA iterate

$$\textsf{lan-FA}_k(f) = \mathbf{Q}_k f(\mathbf{T}_k)\mathbf{Q}_k^\mathsf{T}\mathbf{b}.$$

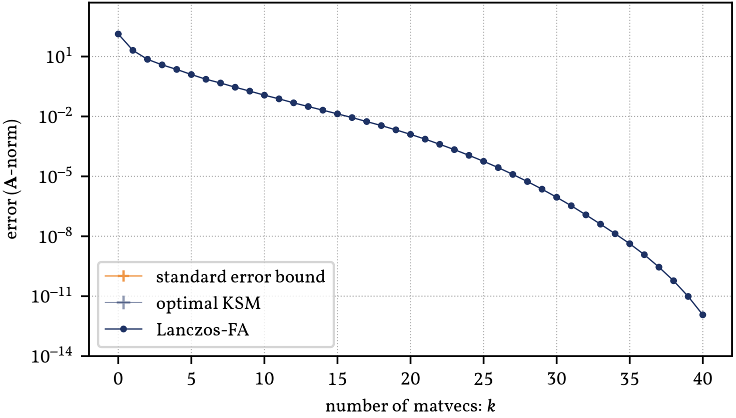
Fact. If $f(x) = x^{-1}$ and $\mathbf{A}$ is positive definite, then $\textsf{lan-FA}_k(f)$ is mathematically equivalent to the CG iterate (so we have error bounds and estimates).

For other functions the algorithm is still widely used, and performs remarkably well in practice. However, less theory is known about the error.

[18] Lanczos 1950.

Why does Lanczos-FA work so well? example: matrix square root)

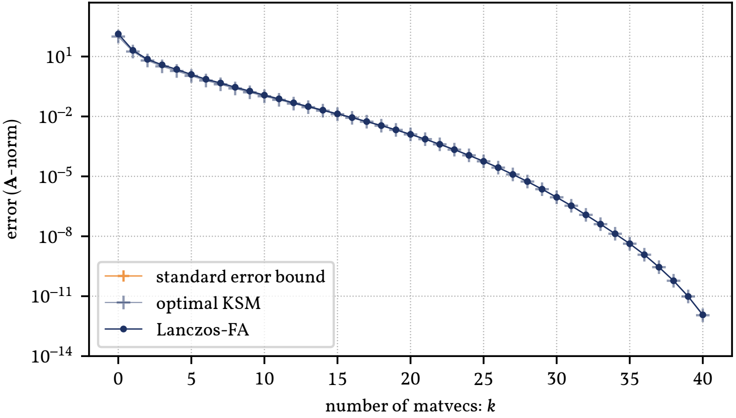
Amazingly, despite being the method of choice for 30+ years, we still don't know why Lanczos-FA works so well!



[18] standard bound is from ideas in Saad 1992 and guarantees linear convergence

Why does Lanczos-FA work so well? example: matrix square root)

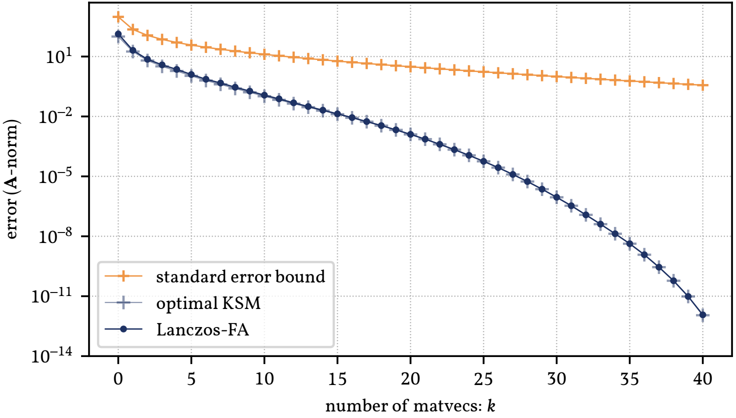
Amazingly, despite being the method of choice for 30+ years, we still don't know why Lanczos-FA works so well!



[18] standard bound is from ideas in Saad 1992 and guarantees linear convergence

Why does Lanczos-FA work so well? example: matrix square root)

Amazingly, despite being the method of choice for 30+ years, we still don't know why Lanczos-FA works so well!



[18]standard bound is from ideas in Saad 1992 and guarantees linear convergence

Key question:

Why does Lanczos-FA work so well?

A reduction to linear systems

Theorem (T. C., Greenbaum, Musco, and Musco 2022). Suppose f is analytic on an neighborhood of the eigenvalues of $\mathbf{A}$ and $\mathbf{T}_k$. Let Γ be a contour containing the eigenvalues of $\mathbf{A}$ and $\mathbf{T}_k$. Then, there is a function $C(w,z)$ (which can be computed using limited information about $\mathbf{A}$) such that, for any fixed w,

$$\|f(\mathbf{A})\mathbf{b} - \mathsf{lan\text{-}FA}_k(f)\| \leq \underbrace{\left(\frac{1}{2\pi} \oint_\Gamma |f(z)||C(w,z)|\mathrm{d}z \right)}_{\text{integral term}} \underbrace{\|\mathsf{err}_k(w)\|}_{\text{linear system error}}.$$

This decouples the error into:

- an integral term we can bound or approximate numerically
- and an error term for CG (which we know a lot about)

A reduction to linear systems

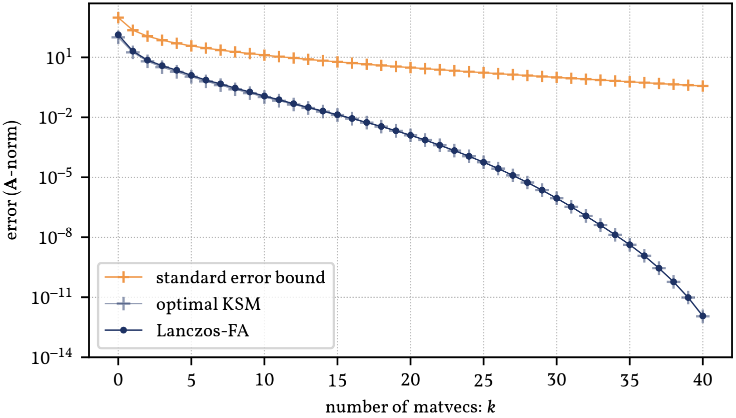
Theorem (T. C., Greenbaum, Musco, and Musco 2022). Suppose f is analytic on an neighborhood of the eigenvalues of $\mathbf{A}$ and $\mathbf{T}_k$. Let Γ be a contour containing the eigenvalues of $\mathbf{A}$ and $\mathbf{T}_k$. Then, there is a function $C(w,z)$ (which can be computed using limited information about $\mathbf{A}$) such that, for any fixed w,

$$\|f(\mathbf{A})\mathbf{b} - \mathsf{lan\text{-}FA}_k(f)\| \leq \underbrace{\left(\frac{1}{2\pi} \oint_\Gamma |f(z)||C(w,z)|\mathrm{d}z \right)}_{\text{integral term}} \underbrace{\|\mathsf{err}_k(w)\|}_{\text{linear system error}}.$$

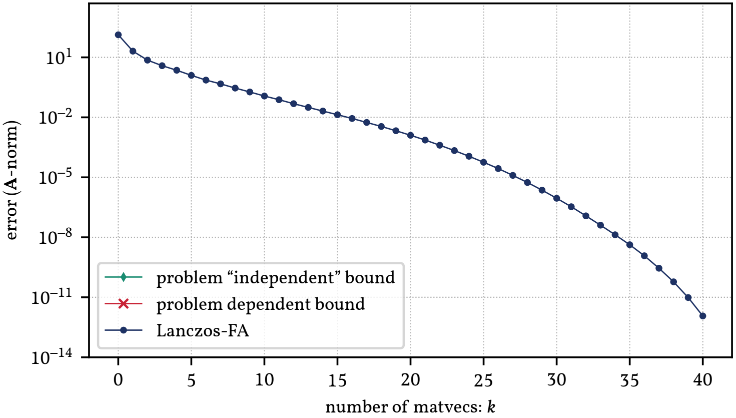
This decouples the error into:

- an integral term we can bound or approximate numerically
- and an error term for CG (which we know a lot about)

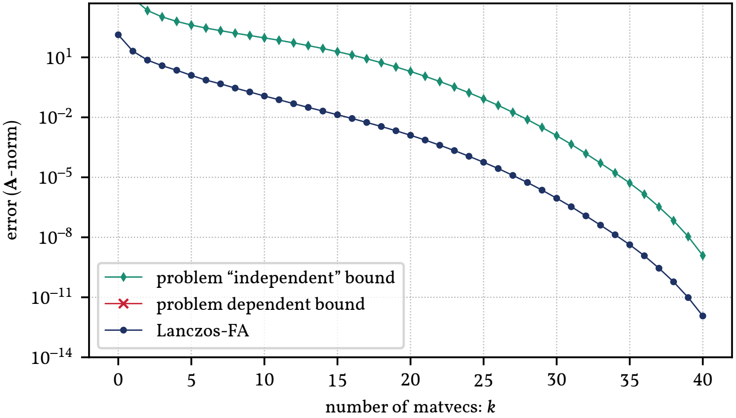
New bounds! (example: matrix square root)



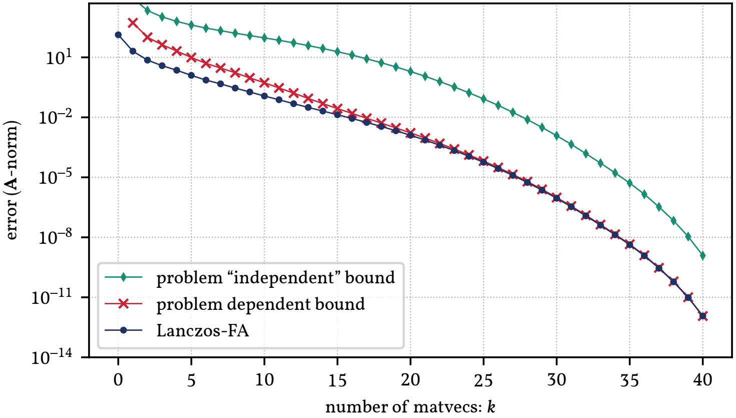
New bounds! (example: matrix square root)



New bounds! (example: matrix square root)



New bounds! (example: matrix square root)



A reduction to linear systems

From Cauchy integral formula:

$$f(x) = -\frac{1}{2\pi \boldsymbol{i}} \oint_\Gamma \frac{f(z)}{x - z}\, \mathrm{d}z.$$

This gives matrix versions:

$$f(\mathbf{A})\mathbf{b} = -\frac{1}{2\pi \boldsymbol{i}} \oint_\Gamma f(z)(\mathbf{A} - z\mathbf{I})^{-1}\mathbf{b}\, \mathrm{d}z.$$

$$\textsf{lan-FA}_k(f) = -\frac{1}{2\pi \boldsymbol{i}} \oint_\Gamma f(z)\mathbf{Q}(\mathbf{T} - z\mathbf{I})^{-1}\mathbf{Q}^\mathsf{T}\mathbf{b}\, \mathrm{d}z.$$

Define $\textsf{err}_k(z) = (\mathbf{A} - z\mathbf{I})^{-1}\mathbf{b} - \mathbf{Q}(\mathbf{T} - z\mathbf{I})^{-1}\mathbf{Q}^\mathsf{T}\mathbf{b}$. Then,

$$f(\mathbf{A})\mathbf{b} - \textsf{lan-FA}_k(f) = -\frac{1}{2\pi \boldsymbol{i}} \oint_\Gamma f(z)\, \textsf{err}_k(z)\, \mathrm{d}z.$$

A reduction to linear systems

From Cauchy integral formula:

$$f(x) = -\frac{1}{2\pi \boldsymbol{i}} \oint_{\Gamma} \frac{f(z)}{x - z} \, \mathrm{d}z.$$

This gives matrix versions:

$$f(\mathbf{A})\mathbf{b} = -\frac{1}{2\pi \boldsymbol{i}} \oint_{\Gamma} f(z)(\mathbf{A} - z\mathbf{I})^{-1}\mathbf{b} \, \mathrm{d}z.$$

$$\textsf{lan-FA}_k(f) = -\frac{1}{2\pi \boldsymbol{i}} \oint_{\Gamma} f(z)\mathbf{Q}(\mathbf{T} - z\mathbf{I})^{-1}\mathbf{Q}^\mathsf{T}\mathbf{b} \, \mathrm{d}z.$$

Define $\textsf{err}_k(z) = (\mathbf{A} - z\mathbf{I})^{-1}\mathbf{b} - \mathbf{Q}(\mathbf{T} - z\mathbf{I})^{-1}\mathbf{Q}^\mathsf{T}\mathbf{b}$. Then,

$$f(\mathbf{A})\mathbf{b} - \textsf{lan-FA}_k(f) = -\frac{1}{2\pi \boldsymbol{i}} \oint_{\Gamma} f(z) \, \textsf{err}_k(z) \, \mathrm{d}z.$$

A reduction to linear systems

From Cauchy integral formula:

$$f(x) = -\frac{1}{2\pi \boldsymbol{i}} \oint_\Gamma \frac{f(z)}{x-z} \, \mathrm{d}z.$$

This gives matrix versions:

$$f(\mathbf{A})\mathbf{b} = -\frac{1}{2\pi \boldsymbol{i}} \oint_\Gamma f(z)(\mathbf{A} - z\mathbf{I})^{-1}\mathbf{b} \, \mathrm{d}z.$$

$$\textsf{lan-FA}_k(f) = -\frac{1}{2\pi \boldsymbol{i}} \oint_\Gamma f(z)\mathbf{Q}(\mathbf{T} - z\mathbf{I})^{-1}\mathbf{Q}^\mathsf{T}\mathbf{b} \, \mathrm{d}z.$$

Define $\textsf{err}_k(z) = (\mathbf{A} - z\mathbf{I})^{-1}\mathbf{b} - \mathbf{Q}(\mathbf{T} - z\mathbf{I})^{-1}\mathbf{Q}^\mathsf{T}\mathbf{b}$. Then,

$$f(\mathbf{A})\mathbf{b} - \textsf{lan-FA}_k(f) = -\frac{1}{2\pi \boldsymbol{i}} \oint_\Gamma f(z) \, \textsf{err}_k(z) \, \mathrm{d}z.$$

Some basic facts and a key lemma

Lemma 1. The CG residual to $\mathbf{A}\mathbf{x} = \mathbf{b}$ is in the direction of the Lanczos vector $\mathbf{q}_k$.

Lemma 2. For any z, $\mathcal{K}_k(\mathbf{A} - z\mathbf{I}, \mathbf{b}) = \mathcal{K}_k(\mathbf{A}, \mathbf{b})$.

Define the residual and error for the iterate $\mathbf{x}_k(z) = \mathbf{Q}_k(\mathbf{T}_k - z\mathbf{I})\mathbf{Q}_k^\top\mathbf{b}$:

$$\mathsf{res}_k(z) = \mathbf{b} - (\mathbf{A} - z\mathbf{I})\mathbf{x}_k(z), \qquad \mathsf{err}_k(z) = (\mathbf{A} - z\mathbf{I})^{-1}\mathbf{b} - \mathbf{x}_k(z).$$

Corollary. With $h_{w,z}(x) = (x - w)/(x - z)$, we have

$$\mathsf{res}_k(z) = c(w,z)\mathsf{res}_k(w), \qquad \mathsf{err}_k(z) = c(w,z)h_{w,z}(\mathbf{A})\mathsf{err}_k(w).$$

Some basic facts and a key lemma

Lemma 1. The CG residual to $\mathbf{A}\mathbf{x} = \mathbf{b}$ is in the direction of the Lanczos vector $\mathbf{q}_k$.

Lemma 2. For any z, $\mathcal{K}_k(\mathbf{A} - z\mathbf{I}, \mathbf{b}) = \mathcal{K}_k(\mathbf{A}, \mathbf{b})$.

Define the residual and error for the iterate $\mathbf{x}_k(z) = \mathbf{Q}_k(\mathbf{T}_k - z\mathbf{I})\mathbf{Q}_k^\top \mathbf{b}$:

$$\mathsf{res}_k(z) = \mathbf{b} - (\mathbf{A} - z\mathbf{I})\mathbf{x}_k(z), \qquad \mathsf{err}_k(z) = (\mathbf{A} - z\mathbf{I})^{-1}\mathbf{b} - \mathbf{x}_k(z).$$

Corollary. With $h_{w,z}(x) = (x - w)/(x - z)$, we have

$$\mathsf{res}_k(z) = c(w, z)\mathsf{res}_k(w), \qquad \mathsf{err}_k(z) = c(w, z)h_{w,z}(\mathbf{A})\mathsf{err}_k(w).$$

An error bound

Using the previous result:

$$f(\mathbf{A})\mathbf{b} - \text{lan-FA}_k(f) = \left(-\frac{1}{2\pi \boldsymbol{i}} \oint_\Gamma f(z) \text{err}_k(z) \, \mathrm{d}z \right).$$

Take norm, move norm into integral, and get:

Theorem (T. C., Greenbaum, Musco, and Musco 2022).

$$\|f(\mathbf{A})\mathbf{b} - \text{lan-FA}_k(f)\| \leq \underbrace{\left(\frac{1}{2\pi} \oint_\Gamma |f(z)||C(w,z)| \mathrm{d}z \right)}_{\text{integral term}} \underbrace{\|\text{err}_k(w)\|}_{\text{linear system error}}.$$

An error bound

Using the previous result:

$$f(\mathbf{A})\mathbf{b} - \textsf{lan-FA}_k(f) = \left(-\frac{1}{2\pi \boldsymbol{i}} \oint_\Gamma f(z) c(w,z)\, h_{w,z}(\mathbf{A})\,\mathrm{d}z \right) \textsf{err}_k(w).$$

Take norm, move norm into integral, and get:

Theorem (T. C., Greenbaum, Musco, and Musco 2022).

$$\|f(\mathbf{A})\mathbf{b} - \textsf{lan-FA}_k(f)\| \leq \underbrace{\left(\frac{1}{2\pi} \oint_\Gamma |f(z)||C(w,z)|\mathrm{d}z \right)}_{\text{integral term}} \underbrace{\|\textsf{err}_k(w)\|}_{\text{linear system error}}.$$

An error bound

Using the previous result:

$$f(\mathbf{A})\mathbf{b} - \textsf{lan-FA}_k(f) = \left(-\frac{1}{2\pi \boldsymbol{i}} \oint_\Gamma f(z) c(w,z)\, h_{w,z}(\mathbf{A})\, \mathrm{d}z \right) \textsf{err}_k(w).$$

Take norm, move norm into integral, and get:

Theorem (T. C., Greenbaum, Musco, and Musco 2022).

$$\|f(\mathbf{A})\mathbf{b} - \textsf{lan-FA}_k(f)\| \leq \underbrace{\left(\frac{1}{2\pi} \oint_\Gamma |f(z)||C(w,z)| \mathrm{d}z \right)}_{\text{integral term}} \underbrace{\|\textsf{err}_k(w)\|}_{\text{linear system error}}.$$

There’s still more!

Generalizations of T. C., Greenbaum, Musco, and Musco 2022:

- Xu and T. C. 2022: block Lanczos algorithm[19]
- Simunec 2023: rational Krylov methods

In Amsel, T. C., Greenbaum, Musco, and Musco 2023, we show that Lanczos-FA is nearly-optimal for certain classes of functions.

We have made progress over the past several years, but the remarkable performance of Lanczos-FA still defies understanding!

[19] Work with an undergrad at UW!

Part II: Designing better algorithms

We can improve existing linear algebra algorithms and design new ones.

- High performance Conjugate Gradient algorithms[20]
- Memory efficient / optimal KSMs[21]
- Krylov-aware low-rank approximation and trace estimation[22]
- Spectrum-adaptive Kernel Polynomial Method[23]

[20] T. C. and Carson 2020.
[21] T. C., Greenbaum, Musco, and Musco 2023.
[22] T. C. and Hallman 2023; Persson, T. C., and Musco 2023.
[23] T. C. 2023.

Low-rank approximation

Since $f(\mathbf{A})$ is dense, we can't store it explicitly if n is big. If we need access to $f(\mathbf{A})$ for some application, we might try to get a low-rank approximation:

$$f(\mathbf{A}) \approx \mathbf{W}\mathbf{X}\mathbf{W}^\top, \text{ where } \mathbf{W} \text{ is } n \times k \text{ and } \mathbf{X} \text{ is } k \times k, \text{ and } k \ll n.$$

KSMs like Lanczos-FA essentially give black-box matrix-vector products with matrix functions: $\mathbf{b} \mapsto f(\mathbf{A})\mathbf{b}$.

This lets us run existing matrix-free low-rank approximation algorithms.

Randomized low-rank approximation

Suppose we wish to obtain a low-rank approximation to a symmetric matrix $\mathbf{B}$.

- Compute a (low-dimension) subspace $\mathbf{K}$
- Project $\mathbf{X}$ onto $\mathbf{K}$

Algorithm 1 Randomized SVD (two-sided)

1: Sample a standard Gaussian $n \times k$ matrix $\mathbf{\Omega}$
2: Form $\mathbf{K} = \mathbf{B\Omega}$ ▷ k matvecs with $\mathbf{B}$
3: Compute $\mathbf{W} = \textsc{orth}(\mathbf{K})$
4: Form $\mathbf{X} = \mathbf{W}^\mathsf{T}\mathbf{BW}$ ▷ k matvecs with $\mathbf{B}$
5: **return** $\mathbf{WXW}^\mathsf{T}$

The result $\mathbf{WXW}^\mathsf{T}$ is a nearly optimal rank k approximation to $\mathbf{B}$.[24]

Algorithms of this flavor are widely used in all areas of computational science.

[24] Halko, Martinsson, and Tropp 2011; Tropp and Webber 2023.

Randomized low-rank approximation

Suppose we wish to obtain a low-rank approximation to a symmetric matrix $\mathbf{B}$.

- Compute a (low-dimension) subspace $\mathbf{K}$
- Project $\mathbf{X}$ onto $\mathbf{K}$

Algorithm 2 Randomized SVD (two-sided)

1: Sample a standard Gaussian $n \times k$ matrix $\mathbf{\Omega}$
2: Form $\mathbf{K} = \mathbf{B}\mathbf{\Omega}$ ▷ k matvecs with $\mathbf{B}$
3: Compute $\mathbf{W} = \textsc{orth}(\mathbf{K})$
4: Form $\mathbf{X} = \mathbf{W}^\mathsf{T}\mathbf{B}\mathbf{W}$ ▷ k matvecs with $\mathbf{B}$
5: **return** $\mathbf{W}\mathbf{X}\mathbf{W}^\mathsf{T}$

The result $\mathbf{W}\mathbf{X}\mathbf{W}^\mathsf{T}$ is a nearly optimal rank k approximation to $\mathbf{B}$.[24]

Algorithms of this flavor are widely used in all areas of computational science.

[24] Halko, Martinsson, and Tropp 2011; Tropp and Webber 2023.

Key question:

How to do low-rank approximation of matrix functions?

Randomized SVD for matrix functions (black-box version)

Algorithm 3 Low-rank approximation for matrix functions

1: Sample a standard Gaussian $n \times k$ matrix $\mathbf{\Omega}$
2: Form $\mathbf{K} \approx f(\mathbf{A})\mathbf{\Omega}$ from $\mathcal{K}_s(\mathbf{A}, \mathbf{\Omega})$ $\triangleright$ $(s-1)k$ matvces with $\mathbf{A}$
3: Compute $\mathbf{W} = \textsc{orth}(\mathbf{K})$
4: Form $\mathbf{X} \approx \mathbf{W}^\top f(\mathbf{A})\mathbf{W}$ from $\mathcal{K}_{r+1}(\mathbf{A}, \mathbf{W})$ $\triangleright$ rk matvces with $\mathbf{A}$
5: **return** $\mathbf{W}\mathbf{X}\mathbf{W}^\top \approx \mathbf{W}\mathbf{W}^\top f(\mathbf{A})\mathbf{W}\mathbf{W}^\top$

As we send $s, r \to \infty$, algorithm converges to the exact randomized SVD.

Look into black box

The main costs are matvecs with $\mathbf{A}$:

1. comptuing $\mathbf{K} \approx f(\mathbf{A})\boldsymbol{\Omega}$ from $\mathcal{K}_s(\mathbf{A}, \boldsymbol{\Omega})$ and
2. computing $\mathbf{X} \approx \mathbf{W}^\top f(\mathbf{A})\mathbf{W}$ from $\mathcal{K}_{r+1}(\mathbf{A}, \mathbf{W})$, where $\mathbf{W} = \textsc{orth}(\mathbf{K})$.

Note that:

- We can instead take: $\mathbf{K} \approx f(\mathbf{A})^q\boldsymbol{\Omega}$ or even $\mathbf{K} \approx [\boldsymbol{\Omega}, f(\mathbf{A})\boldsymbol{\Omega}, \ldots, f(\mathbf{A})^q\boldsymbol{\Omega}]$.
- Best error if we use the whole Krylov subspace: $\mathbf{K} = [\boldsymbol{\Omega}, \mathbf{A}\boldsymbol{\Omega}, \ldots, \mathbf{A}^s\boldsymbol{\Omega}]$.

But wait…

- If $\mathbf{K}$ (and hence $\mathbf{W}$) has more columns, approximating $\mathbf{X} \approx \mathbf{W}^\top f(\mathbf{A})\mathbf{W}$ from $\mathcal{K}_{r+1}(\mathbf{A}, \mathbf{W})$ is ostensibly more expensive.

Look into black box

The main costs are matvecs with $\mathbf{A}$:

1. comptuing $\mathbf{K} \approx f(\mathbf{A})\mathbf{\Omega}$ from $\mathcal{K}_s(\mathbf{A}, \mathbf{\Omega})$ and
2. computing $\mathbf{X} \approx \mathbf{W}^\top f(\mathbf{A})\mathbf{W}$ from $\mathcal{K}_{r+1}(\mathbf{A}, \mathbf{W})$, where $\mathbf{W} = \textsc{orth}(\mathbf{K})$.

Note that:

- We can instead take: $\mathbf{K} \approx f(\mathbf{A})^q\mathbf{\Omega}$ or even $\mathbf{K} \approx [\mathbf{\Omega}, f(\mathbf{A})\mathbf{\Omega}, \dots, f(\mathbf{A})^q\mathbf{\Omega}]$.
- Best error if we use the whole Krylov subspace: $\mathbf{K} = [\mathbf{\Omega}, \mathbf{A}\mathbf{\Omega}, \dots, \mathbf{A}^s\mathbf{\Omega}]$.

But wait…

- If $\mathbf{K}$ (and hence $\mathbf{W}$) has more columns, approximating $\mathbf{X} \approx \mathbf{W}^\top f(\mathbf{A})\mathbf{W}$ from $\mathcal{K}_{r+1}(\mathbf{A}, \mathbf{W})$ is ostensibly more expensive.

Krylov subspaces of Krylov subspaces are Krylov subspaces

In general, if $\mathbf{K}$ (and hence $\mathbf{W}$) have sk columns, approximating $\mathbf{X} \approx \mathbf{W}^\top f(\mathbf{A})\mathbf{W}$ from $\mathcal{K}_{r+1}(\mathbf{A}, \mathbf{W})$ is ostensibly s-times expensive than if $\mathbf{K}$ has k columns.

Theorem. Suppose $\mathbf{Q}_s = [\boldsymbol{\Omega}\ \mathbf{A}\boldsymbol{\Omega} \cdots \mathbf{A}^{s-1}\boldsymbol{\Omega}]$. Then, $\mathcal{K}_{s+r}(\mathbf{A}, \boldsymbol{\Omega}) = \mathcal{K}_{r+1}(\mathbf{A}, \mathbf{Q}_s)$.

Proof.

$$\begin{aligned}
\mathcal{K}_{r+1}(\mathbf{A}, \mathbf{Q}_s) &= \operatorname{range}\left(\begin{bmatrix} \mathbf{Q}_s & \mathbf{A}\mathbf{Q}_s & \cdots & \mathbf{A}^r\mathbf{Q}_s \end{bmatrix}\right) \\
&= \operatorname{range}\left(\begin{bmatrix} \boldsymbol{\Omega} & \mathbf{A}\boldsymbol{\Omega} & \cdots & \mathbf{A}^{s-1}\boldsymbol{\Omega} \\ & \mathbf{A}\boldsymbol{\Omega} & \mathbf{A}^2\boldsymbol{\Omega} & \cdots & \mathbf{A}^{s}\boldsymbol{\Omega} \\ & & \mathbf{A}^r\boldsymbol{\Omega} & \mathbf{A}^{r+1}\boldsymbol{\Omega} & \cdots & \mathbf{A}^{s+r-1}\boldsymbol{\Omega} \end{bmatrix}\right) \\
&= \operatorname{range}\left(\begin{bmatrix} \boldsymbol{\Omega} & \mathbf{A}\boldsymbol{\Omega} & \cdots & \mathbf{A}^{s+r-1}\boldsymbol{\Omega} \end{bmatrix}\right) = \mathcal{K}_{s+r}(\mathbf{A}, \boldsymbol{\Omega}).
\end{aligned}$$

Krylov subspaces of Krylov subspaces are Krylov subspaces

In general, if $\mathbf{K}$ (and hence $\mathbf{W}$) have sk columns, approximating $\mathbf{X} \approx \mathbf{W}^\top f(\mathbf{A})\mathbf{W}$ from $\mathcal{K}_{r+1}(\mathbf{A}, \mathbf{W})$ is ostensibly s-times expensive than if $\mathbf{K}$ has k columns.

Theorem. Suppose $\mathbf{Q}_s = [\mathbf{\Omega}\ \mathbf{A\Omega}\ \cdots\ \mathbf{A}^{s-1}\mathbf{\Omega}]$. Then, $\mathcal{K}_{s+r}(\mathbf{A}, \mathbf{\Omega}) = \mathcal{K}_{r+1}(\mathbf{A}, \mathbf{Q}_s)$.

Proof.

$$
\begin{aligned}
\mathcal{K}_{r+1}(\mathbf{A}, \mathbf{Q}_s) &= \operatorname{range}\left(\begin{bmatrix} \mathbf{Q}_s & \mathbf{A}\mathbf{Q}_s & \cdots & \mathbf{A}^r\mathbf{Q}_s \end{bmatrix}\right) \\
&= \operatorname{range}\left(\begin{bmatrix} \mathbf{\Omega} & \mathbf{A\Omega} & \cdots & \mathbf{A}^{s-1}\mathbf{\Omega} \\ \mathbf{A\Omega} & \mathbf{A}^2\mathbf{\Omega} & \cdots & \mathbf{A}^{s}\mathbf{\Omega} \\ \mathbf{A}^r\mathbf{\Omega} & \mathbf{A}^{r+1}\mathbf{\Omega} & \cdots & \mathbf{A}^{s+r-1}\mathbf{\Omega} \end{bmatrix}\right) \\
&= \operatorname{range}\left(\begin{bmatrix} \mathbf{\Omega} & \mathbf{A\Omega} & \cdots & \mathbf{A}^{s+r-1}\mathbf{\Omega} \end{bmatrix}\right) = \mathcal{K}_{s+r}(\mathbf{A}, \mathbf{\Omega}).
\end{aligned}
$$

Krylov subspaces of Krylov subspaces are Krylov subspaces

In general, if $\mathbf{K}$ (and hence $\mathbf{W}$) have sk columns, approximating $\mathbf{X} \approx \mathbf{W}^\top f(\mathbf{A})\mathbf{W}$ from $\mathcal{K}_{r+1}(\mathbf{A}, \mathbf{W})$ is ostensibly s-times expensive than if $\mathbf{K}$ has k columns.

Theorem. Suppose $\mathbf{Q}_s = [\boldsymbol{\Omega}\ \mathbf{A}\boldsymbol{\Omega}\ \cdots\ \mathbf{A}^{s-1}\boldsymbol{\Omega}]$. Then, $\mathcal{K}_{s+r}(\mathbf{A}, \boldsymbol{\Omega}) = \mathcal{K}_{r+1}(\mathbf{A}, \mathbf{Q}_s)$.

Proof.

$$\begin{aligned}
\mathcal{K}_{r+1}(\mathbf{A}, \mathbf{Q}_s) &= \operatorname{range}\left(\begin{bmatrix} \mathbf{Q}_s & \mathbf{A}\mathbf{Q}_s & \cdots & \mathbf{A}^r\mathbf{Q}_s \end{bmatrix}\right) \\
&= \operatorname{range}\left(\left[\begin{matrix} \boldsymbol{\Omega} & \mathbf{A}\boldsymbol{\Omega} & \cdots & \mathbf{A}^{s-1}\boldsymbol{\Omega} \\ \mathbf{A}\boldsymbol{\Omega} & \mathbf{A}^2\boldsymbol{\Omega} & \cdots & \mathbf{A}^{s}\boldsymbol{\Omega} \\ \mathbf{A}^r\boldsymbol{\Omega} & \mathbf{A}^{r+1}\boldsymbol{\Omega} & \cdots & \mathbf{A}^{s+r-1}\boldsymbol{\Omega} \end{matrix}\right]\right) \\
&= \operatorname{range}\left(\begin{bmatrix} \boldsymbol{\Omega} & \mathbf{A}\boldsymbol{\Omega} & \cdots & \mathbf{A}^{s+r-1}\boldsymbol{\Omega} \end{bmatrix}\right) = \mathcal{K}_{s+r}(\mathbf{A}, \boldsymbol{\Omega}).
\end{aligned}$$

Krylov-aware low-rank approximation[26]

Algorithm 4 Low-rank approximation for matrix functions

1: Sample a standard Gaussian $n \times k$ matrix $\mathbf{\Omega}$
2: Form $\mathbf{K} \approx f(\mathbf{A})\mathbf{\Omega}$ from $\mathcal{K}_s(\mathbf{A}, \mathbf{\Omega})$ $\triangleright$ $(s-1)k$ matvces with $\mathbf{A}$
3: Compute $\mathbf{W} = \textsc{orth}(\mathbf{K})$
4: Form $\mathbf{X} \approx \mathbf{W}^\mathsf{T} f(\mathbf{A})\mathbf{W}$ from $\mathcal{K}_{r+1}(\mathbf{A}, \mathbf{W})$ $\triangleright$ rk matvces with $\mathbf{A}$
5: **return** $\mathbf{W}\mathbf{X}\mathbf{W}^\mathsf{T} \approx \mathbf{W}\mathbf{W}^\mathsf{T} f(\mathbf{A})\mathbf{W}\mathbf{W}^\mathsf{T}$

Some effort need worked out to implement this efficiently and stably.

Deeper theoretical analysis[25]

[25] Persson, T. C., and Musco 2023.
[26] T. C. and Hallman 2023.

Krylov-aware low-rank approximation[26]

Algorithm 5 Krylov-aware low-rank approximation

1: Sample a standard Gaussian $n \times k$ matrix $\mathbf{\Omega}$
2: Form basis $\mathbf{K}$ for $\mathcal{K}_s(\mathbf{A}, \mathbf{\Omega})$ $\triangleright$ $(s-1)k$ matvces with $\mathbf{A}$
3: Compute $\mathbf{W} = \textsc{orth}(\mathbf{K})$
4: Form $\mathbf{X} \approx \mathbf{W}^\mathsf{T} f(\mathbf{A})\mathbf{W}$ from $\mathcal{K}_{r+1}(\mathbf{A}, \mathbf{W}) = \mathcal{K}_{s+r}(\mathbf{A}, \mathbf{\Omega})$ $\triangleright$ rk matvces with $\mathbf{A}$
5: **return** $\mathbf{W}\mathbf{X}\mathbf{W}^\mathsf{T} \approx \mathbf{W}\mathbf{W}^\mathsf{T} f(\mathbf{A})\mathbf{W}\mathbf{W}^\mathsf{T}$

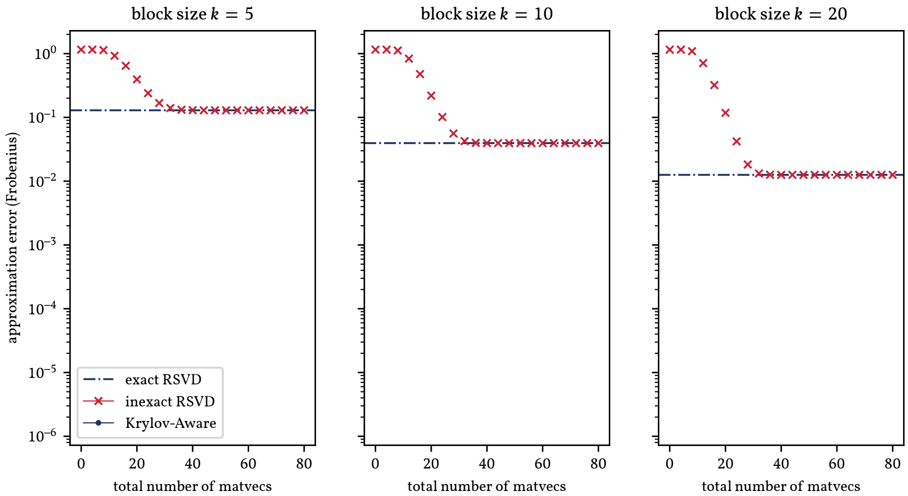
Some effort need worked out to implement this efficiently and stably.

Deeper theoretical analysis[25]

[25] Persson, T. C., and Musco 2023.
[26] T. C. and Hallman 2023.

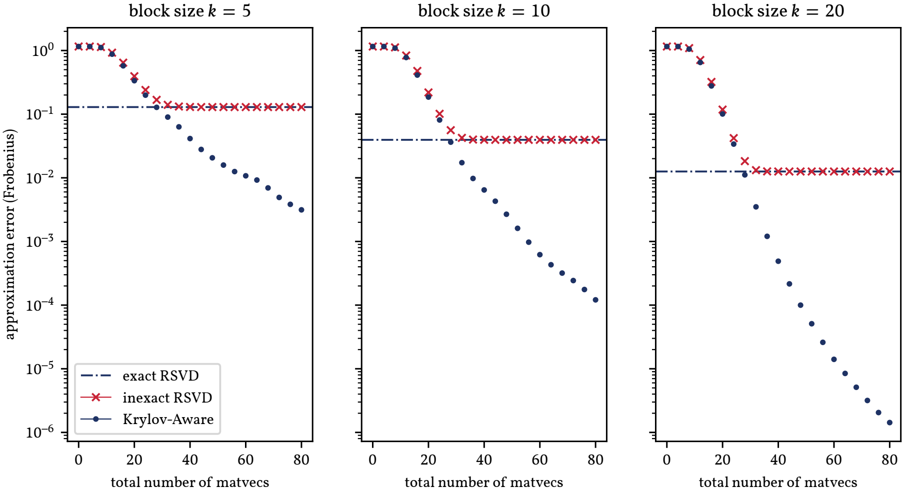
Numerical experiment: exponential function

Setup: $f(x) = \exp(-\beta x)$, $\mathbf{A}$ Hamiltonian of a spin system



Numerical experiment: exponential function

Setup: $f(x) = \exp(-\beta x)$, $\mathbf{A}$ Hamiltonian of a spin system



Part III: Advancing basic science

There is a ton of potential for NLA to advance basic science.

- T. C. and Cheng 2022
- T. C. 2023
- T. C., Chen, Li, Nzeuton, Pan, and Wang 2023
- T. C., Trogdon, and Ubaru 2021

Quantum equilibrium thermodynamics

Consider a quantum system consisting of subsystems (s) and (b) with Hamiltonian

$$\mathbf{H} = \bar{\mathbf{H}}_\mathrm{s} + \bar{\mathbf{H}}_\mathrm{b} + \mathbf{H}_\mathrm{sb}, \qquad \bar{\mathbf{H}}_\mathrm{s} = \mathbf{H}_\mathrm{s} \otimes \mathbf{I}_\mathrm{b}, \quad \bar{\mathbf{H}}_\mathrm{b} = \mathbf{I}_\mathrm{s} \otimes \mathbf{H}_\mathrm{b}. \tag{1}$$

In thermal equilibrium at interver temperature β, the state of the system is described by a density matrix

$$\boldsymbol{\rho}_\mathrm{t}(\beta) = \frac{\exp(-\beta\mathbf{H})}{Z_\mathrm{t}(\beta)}, \qquad Z_\mathrm{t}(\beta) = \mathrm{tr}(\exp(-\beta\mathbf{H}); \tag{2}$$

The denisty matrix for subsystem (s) is given by

$$\boldsymbol{\rho}^*(\beta) = \mathrm{tr}_\mathrm{b}(\boldsymbol{\rho}_\mathrm{t}(\beta)) = \frac{\mathrm{tr}_\mathrm{b}(\exp(-\beta\mathbf{H}))}{\mathrm{tr}(\exp(-\beta\mathbf{H}))}, \tag{3}$$

where $\mathrm{tr}_b(\,\cdot\,)$ is the *partial trace* over subsystem (b).[27]

[27] Campisi, Zueco, and Talkner 2010; Ingold, Hänggi, and Talkner 2009; Talkner and Hänggi 2020.

von Neumann entropy of Heisenberg spin chains

The von Neumann entropy $-\mathrm{tr}(\boldsymbol{\rho}^*(\beta)\ln(\boldsymbol{\rho}^*(\beta)))$ is a measure of the entanglement betweeen subsystems (s) and (b).

Understanding the von Neumann entropy for a range of a system with Hamiltonian $\mathbf{H}(\theta)$ at a range of parameter values θ and inverse temperatures β is of interest.

We will consider a special case

$$\mathbf{H} = \sum_{i,j}\left[J^{\mathrm{x}}_{i,j}\boldsymbol{\sigma}^{\mathrm{x}}_i\boldsymbol{\sigma}^{\mathrm{x}}_j + J^{\mathrm{y}}_{i,j}\boldsymbol{\sigma}^{\mathrm{y}}_i\boldsymbol{\sigma}^{\mathrm{y}}_j + J^{\mathrm{z}}_{i,j}\boldsymbol{\sigma}^{\mathrm{z}}_i\boldsymbol{\sigma}^{\mathrm{z}}_j\right] + \frac{h}{2}\sum_{i=1}^{N}\boldsymbol{\sigma}^{\mathrm{z}}_i.$$

where h is the magnetic field strength.

Subsystem (s) corresponds to $i = 1, 2$ and subsystem (b) corresponds to the rest of the spins.

Key question:

How to compute reduced density matrices numerically?

A starting point: stochastic trace estimation

If $\mathbf{b}$ is a standard Gaussian random vector:

$$\mathbb{E}[\mathbf{b}^\mathsf{T} f(\mathbf{A})\mathbf{b}] = \operatorname{tr}(f(\mathbf{A})), \qquad \mathbb{V}[\mathbf{b}^\mathsf{T} f(\mathbf{A})\mathbf{b}] = 2\|f(\mathbf{A})\|_\mathsf{F}^2.$$

It's standard to use a KSM to approximate products $\mathbf{b} \mapsto \mathbf{b}^\mathsf{T} f(\mathbf{A})\mathbf{b}$.

Lots of work balancing the cost of the KSM with the variance of the estimator[28].

[28] Han, Malioutov, Avron, and Shin 2017; Ubaru, Chen, and Saad 2017; T. C., Trogdon, and Ubaru 2021; T. C., Trogdon, and Ubaru 2022; Braverman, Krishnan, and Musco 2022.

Partial traces

Suppose $\mathbf{A}$ is a $d_\mathrm{s} d_\mathrm{b} \times d_\mathrm{s} d_\mathrm{b}$ matrtix partitioned as:

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}_{1,1} & \mathbf{A}_{1,2} & \cdots & \mathbf{A}_{1,d_\mathrm{s}} \\ \mathbf{A}_{2,1} & \mathbf{A}_{2,2} & \cdots & \mathbf{A}_{2,d_\mathrm{s}} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{A}_{d_\mathrm{s},1} & \mathbf{A}_{d_\mathrm{s},2} & \cdots & \mathbf{A}_{d_\mathrm{s},d_\mathrm{s}} \end{bmatrix},$$

Partial traces

Then the partial trace (wrt. this partitioning) is defined as:

$$\mathrm{tr}_{\mathrm{b}}(\mathbf{A}) = \begin{bmatrix} \mathrm{tr}(\mathbf{A}_{1,1}) & \mathrm{tr}(\mathbf{A}_{1,2}) & \cdots & \mathrm{tr}(\mathbf{A}_{1,d_{\mathrm{s}}}) \\ \mathrm{tr}(\mathbf{A}_{2,1}) & \mathrm{tr}(\mathbf{A}_{2,2}) & \cdots & \mathrm{tr}(\mathbf{A}_{2,d_{\mathrm{s}}}) \\ \vdots & \vdots & \ddots & \vdots \\ \mathrm{tr}(\mathbf{A}_{d_{\mathrm{s}},1}) & \mathrm{tr}(\mathbf{A}_{d_{\mathrm{s}},2}) & \cdots & \mathrm{tr}(\mathbf{A}_{d_{\mathrm{s}},d_{\mathrm{s}}}) \end{bmatrix}.$$

An algorithm for partial traces[29]

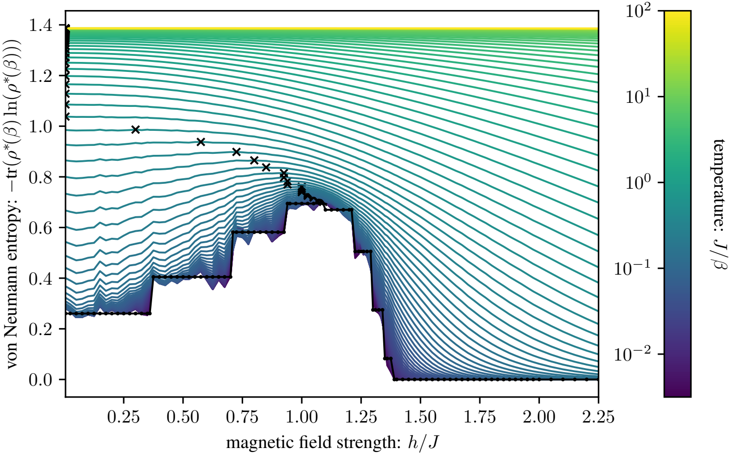
We can use a randomized estimator:

$$(\mathbf{I}_{d_\mathrm{s}} \otimes \mathbf{b})^\top \mathbf{A} (\mathbf{I}_{d_\mathrm{s}} \otimes \mathbf{b}) = \begin{bmatrix} \mathbf{b}^\top \mathbf{A}_{1,1} \mathbf{b} & \mathbf{b}^\top \mathbf{A}_{1,2} \mathbf{b} & \cdots & \mathbf{b}^\top \mathbf{A}_{1,d_\mathrm{s}} \mathbf{b} \\ \mathbf{b}^\top \mathbf{A}_{2,1} \mathbf{b} & \mathbf{b}^\top \mathbf{A}_{2,2} \mathbf{b} & \cdots & \mathbf{b}^\top \mathbf{A}_{2,d_\mathrm{s}} \mathbf{b} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{b}^\top \mathbf{A}_{d_\mathrm{s},1} \mathbf{b} & \mathbf{b}^\top \mathbf{A}_{d_\mathrm{s},2} \mathbf{b} & \cdots & \mathbf{b}^\top \mathbf{A}_{d_\mathrm{s},d_\mathrm{s}} \mathbf{b} \end{bmatrix}.$$

Then use a KSM to approximate products with $\mathbf{A} = f(\mathbf{H})$.

[29] T. C. and Cheng 2022.

von Neumann entropy phase plot[30]



[30] T. C. and Cheng 2022.

Partial trace estimator: variance reduction

For any matrix $\widetilde{\mathbf{A}}$,

$$\mathrm{tr}_b(\mathbf{A}) = \mathrm{tr}_b(\widetilde{\mathbf{A}}) + \mathrm{tr}_b(\mathbf{A} - \widetilde{\mathbf{A}}).$$

So we might try to use the estimator

$$\mathrm{tr}_b(\mathbf{A}) \approx \mathrm{tr}_b(\widetilde{\mathbf{A}}) + \widehat{\mathrm{tr}}_b^{\,m}(\mathbf{A} - \widetilde{\mathbf{A}}).$$

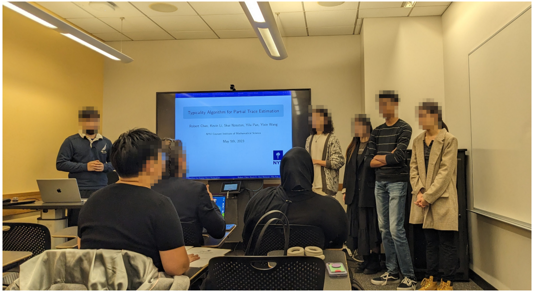
which will have reduced variance if $\|\mathbf{A} - \widetilde{\mathbf{A}}\|_\mathsf{F}^2 \ll \|\mathbf{A}\|_\mathsf{F}^2$.

This residual trick is widely used in regular trace estimation.[31]

But there are a number of numerical issues with generalizing to partial traces of matrix functions.

[31] Girard 1987; Weiße, Wellein, Alvermann, and Fehske 2006; Lin 2016; Morita and Tohyama 2020; Meyer, Musco, Musco, and Woodruff 2021.

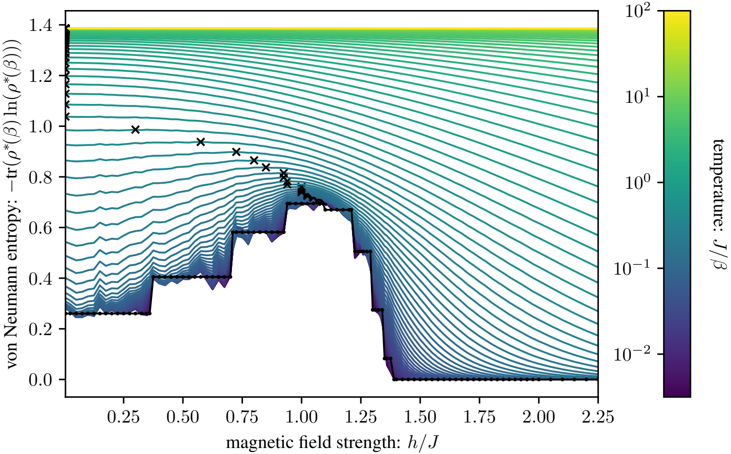
Student involvement



students were a major part of this project, and were able to:

- write and receive grant for research funding
- present at NYU undergrad conference, SIAM NY-NJ-PA annual meeting, Alan Edelman's birthday conference
- perform numerical experiments on NYU's Greene supercomputer

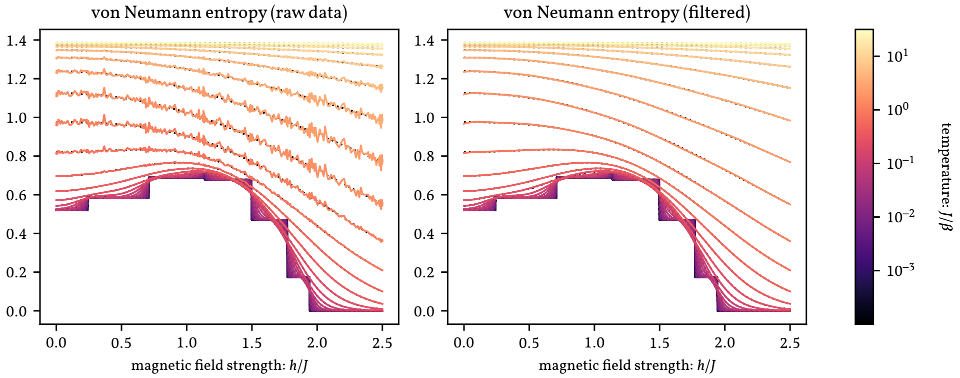
von Neumann entropy phase plot[32]



[32]T. C. and Cheng 2022.

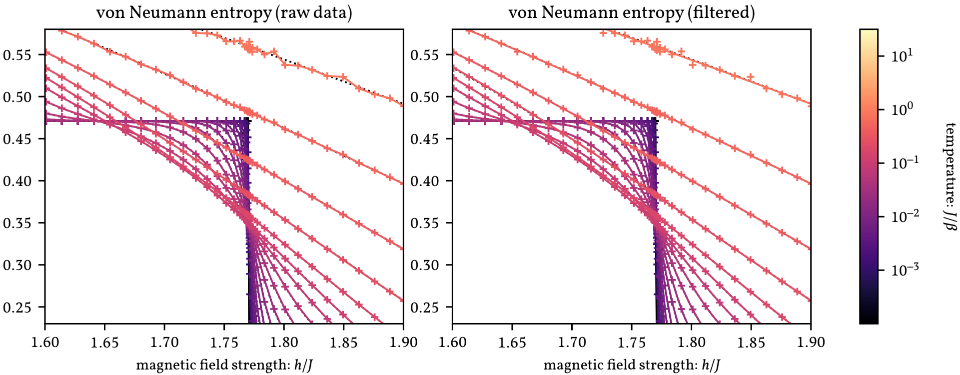
von Neumann entropy phase plot: improved algorithm[33]

We can compute these phase plots, which are more accurate at low temperature, orders of magnitude faster.



[33] T. C., Chen, Li, Nzeuton, Pan, and Wang 2023.

von Neumann entropy phase plot: improved algorithm[34] (cropped)



[34] T. C., Chen, Li, Nzeuton, Pan, and Wang 2023.

My research program

Focus: design and analysis of practically fast and theoretically justified (randomized) algorithms for fundamental linear algebra tasks

Goal: develop tools to support the advancement of knowledge in current scientific applications

Mode: collaboration with a range of fields, and involvement and training of (minority) students

Hope: provide conceptually simple insights into key problems

References I

Amsel, Noah et al. (2023). *Near-Optimality Guarantees for Approximating Rational Matrix Functions by the Lanczos Method*.

Amsel, Noah et al. (2024). *Near Optimal Sparse Matrix Approximation via Matrix-Vector Products*.

Avron, Haim (2010). "Counting Triangles in Large Graphs using Randomized Matrix Trace Estimation". In: *Proceedings of KDD-LDMTA*.

Barry, Ronald Paul and R. Kelley Pace (Mar. 1999). "Monte Carlo estimates of the log determinant of large sparse matrices". In: *Linear Algebra and its Applications* 289.1-3, pp. 41–54.

Bollapragada, Raghu, T. C., and Rachel Ward (2022). *On the fast convergence of minibatch heavy ball momentum*.

Braverman, Vladimir, Aditya Krishnan, and Christopher Musco (June 2022). "Sublinear time spectral density estimation". In: *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing*. arXiv cs.DS 2104.03461. ACM.

Campisi, Michele, David Zueco, and Peter Talkner (Oct. 2010). "Thermodynamic anomalies in open quantum systems: Strong coupling effects in the isotropic XY model". In: *Chemical Physics* 375.2-3, pp. 187–194.

Dong, Kun, Austin R. Benson, and David Bindel (July 2019). "Network Density of States". In: *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM.

Eshof, J. van den et al. (2002). "Numerical methods for the QCDd overlap operator. I. Sign-function and error bounds". In: *Computer Physics Communications* 146.2, pp. 203–224.

Estrada, Ernesto (Mar. 2000). "Characterization of 3D molecular structure". In: *Chemical Physics Letters* 319.5-6, pp. 713–718.

References II

Gardner, Jacob et al. (2018). "GPyTorch: Blackbox Matrix-Matrix Gaussian Process Inference with GPU Acceleration". In: *Advances in Neural Information Processing Systems*. Ed. by S. Bengio et al. Vol. 31. Curran Associates, Inc.

Ghorbani, Behrooz, Shankar Krishnan, and Ying Xiao (Sept. 2019). "An Investigation into Neural Net Optimization via Hessian Eigenvalue Density". In: *Proceedings of the 36th International Conference on Machine Learning*. Ed. by Kamalika Chaudhuri and Ruslan Salakhutdinov. Vol. 97. Proceedings of Machine Learning Research. PMLR, pp. 2232–2241.

Girard, Didier (May 1987). *Un algorithme simple et rapide pour la validation croisée généralisée sur des problèmes de grande taille*.

Granziol, Diego, Xingchen Wan, and Timur Garipov (2019). *Deep Curvature Suite*.

Halko, N. , P. G. Martinsson, and J. A. Tropp (Jan. 2011). "Finding Structure with Randomness: Probabilistic Algorithms for Constructing Approximate Matrix Decompositions". In: *SIAM Review* 53.2, pp. 217–288.

Han, Insu et al. (Jan. 2017). "Approximating Spectral Sums of Large-Scale Matrices using Stochastic Chebyshev Approximations". In: *SIAM Journal on Scientific Computing* 39.4, A1558–A1585.

Ingold, Gert-Ludwig, Peter Hänggi, and Peter Talkner (June 2009). "Specific heat anomalies of open quantum systems". In: *Physical Review E* 79.6.

Jin, Yujia and Aaron Sidford (2019). "Principal Component Projection and Regression in Nearly Linear Time through Asymmetric SVRG". In: *Advances in Neural Information Processing Systems 32*. Ed. by H. Wallach et al. Curran Associates, Inc. , pp. 3868–3878.

References III

Lanczos, Cornelius (1950). "An iteration method for the solution of the eigenvalue problem of linear differential and integral operators". In: *Journal of research of the National Bureau of Standards* 45, pp. 255–282.

Li, Ruipeng et al. (Jan. 2019). "The Eigenvalues Slicing Library (EVSL): Algorithms, Implementation, and Software". In: *SIAM Journal on Scientific Computing* 41.4, pp. C393–C415.

Lin, Lin (Aug. 2016). "Randomized estimation of spectral densities of large matrices made accurate". In: *Numerische Mathematik* 136.1, pp. 183–213.

Meurant, Gerard and Peter Tichy (2024). *Error Norm Estimation in the Conjugate Gradient Algorithm*.

Meyer, Raphael A et al. (2021). "Hutch++: Optimal Stochastic Trace Estimation". In: *Symposium on Simplicity in Algorithms (SOSA)*. SIAM, pp. 142–155.

Morita, Katsuhiro and Takami Tohyama (Feb. 2020). "Finite-temperature properties of the Kitaev-Heisenberg models on kagome and triangular lattices studied by improved finite-temperature Lanczos methods". In: *Physical Review Research* 2.1.

Papyan, Vardan (2019). *The Full Spectrum of Deepnet Hessians at Scale: Dynamics with SGD Training and Sample Size*.

Persson, David, T. C., and Christopher Musco (2023). *Randomized block Krylov subspace methods for low rank approximation of matrix functions*.

Polizzi, Eric (Mar. 2009). "Density-matrix-based algorithm for solving eigenvalue problems". In: *Physical Review B* 79.11.

Saad, Yousef (1992). "Analysis of Some Krylov Subspace Approximations to the Matrix Exponential Operator". In: *SIAM Journal on Numerical Analysis* 29.1, pp. 209–228.

References IV

Schnalle, Roman and Jürgen Schnack (July 2010). “Calculating the energy spectra of magnetic molecules: application of real- and spin-space symmetries”. In: *International Reviews in Physical Chemistry* 29.3, pp. 403–452.

Simunec, Igor (2023). *Error bounds for the approximation of matrix functions with rational Krylov methods*.

T. C. (Sept. 2023). “A spectrum adaptive kernel polynomial method”. In: *The Journal of Chemical Physics* 159.11, p. 114101.

T. C. and Erin T. C. Carson (Jan. 2020). “Predict-and-recompute conjugate gradient variants”. In: *SIAM Journal on Scientific Computing* 42.5, A3084–A3108.

T. C. and Yu-Chen Cheng (Aug. 2022). “Numerical computation of the equilibrium-reduced density matrix for strongly coupled open quantum systems”. In: *The Journal of Chemical Physics* 157.6, p. 064106.

T. C. and Eric Hallman (Aug. 2023). “Krylov-Aware Stochastic Trace Estimation”. In: *SIAM Journal on Matrix Analysis and Applications* 44.3, pp. 1218–1244.

T. C. and Thomas Trogdon (Nov. 2023). “Stability of the Lanczos algorithm on matrices with regular spectral distributions”. In: *Linear Algebra and its Applications*.

T. C., Thomas Trogdon, and Shashanka Ubaru (July 2021). “Analysis of stochastic Lanczos quadrature for spectrum approximation”. In: *Proceedings of the 38th International Conference on Machine Learning*. Vol. 139. Proceedings of Machine Learning Research. PMLR, pp. 1728–1739.

— (2022). *Randomized matrix-free quadrature for spectrum and spectral sum approximation*.

T. C. et al. (May 2022). “Error Bounds for Lanczos-Based Matrix Function Approximation”. In: *SIAM Journal on Matrix Analysis and Applications* 43.2, pp. 787–811.

T. C. et al. (2023). *Faster randomized partial trace estimation*.

References V

T. C. et al. (May 2023). "Low-Memory Krylov Subspace Methods for Optimal Rational Matrix Function Approximation". In: *SIAM Journal on Matrix Analysis and Applications* 44.2, pp. 670–692.

Talkner, Peter and Peter Hänggi (Oct. 2020). "Colloquium : Statistical mechanics and thermodynamics at strong coupling: Quantum and classical". In: *Reviews of Modern Physics* 92.4.

Tropp, Joel A and Robert J Webber (2023). "Randomized algorithms for low-rank matrix approximation: Design, analysis, and applications". In: *arXiv preprint arXiv:2306.12418*.

Ubaru, Shashanka, Jie Chen, and Yousef Saad (Jan. 2017). "Fast Estimation of $\operatorname{tr}(f(A))$ via Stochastic Lanczos Quadrature". In: *SIAM Journal on Matrix Analysis and Applications* 38.4, pp. 1075–1099.

Weiße, Alexander et al. (Mar. 2006). "The kernel polynomial method". In: *Reviews of Modern Physics* 78.1, pp. 275–306.

Xu, Qichen and T. C. (2022). *A posteriori error bounds for the block-Lanczos method for matrix function approximation*.

Yao, Zhewei et al. (2020). *PyHessian: Neural Networks Through the Lens of the Hessian*.