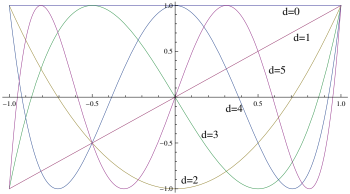
Faster (Spectral) Algorithms via Approximation Theory

Nisheeth K. Vishnoi
EPFL



Based on a recent monograph with Sushant Sachdeva (Yale)

Simons Institute, Dec. 3, 2014

The Goal

Many algorithms today rely on our ability to quickly compute good approximations to matrix-function-vector products: e.g.,

- $A^s v$, $A^{-1} v$, $\exp(-A) v$, ...
- or top few eigenvalues and eigenvectors.

The Goal

Many algorithms today rely on our ability to quickly compute good approximations to matrix-function-vector products: e.g.,

- $A^s v$, $A^{-1} v$, $\exp(-A) v$, ...
- or top few eigenvalues and eigenvectors.

Demonstrate

How to reduce the problem of computing these primitives to a **small number** of those of the form Bu where B is a matrix closely related to A (often A itself) and u is some vector.

The Goal

Many algorithms today rely on our ability to quickly compute good approximations to matrix-function-vector products: e.g.,

- $A^s v$, $A^{-1} v$, $\exp(-A)v$, ...
- or top few eigenvalues and eigenvectors.

Demonstrate

How to reduce the problem of computing these primitives to a **small number** of those of the form Bu where B is a matrix closely related to A (often A itself) and u is some vector.

- *A key feature of these algorithms is that if the matrix-vector product for A can be computed quickly, e.g., when A is sparse, then Bu can also be computed in essentially the same time.*

The Goal

Many algorithms today rely on our ability to quickly compute good approximations to matrix-function-vector products: e.g.,

- $A^s v$, $A^{-1} v$, $\exp(-A)v$, ...
- or top few eigenvalues and eigenvectors.

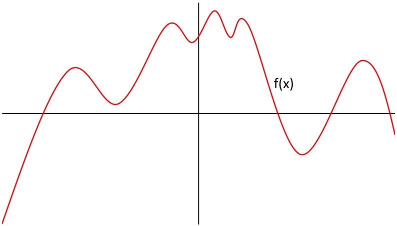
Demonstrate

How to reduce the problem of computing these primitives to a **small number** of those of the form Bu where B is a matrix closely related to A (often A itself) and u is some vector.

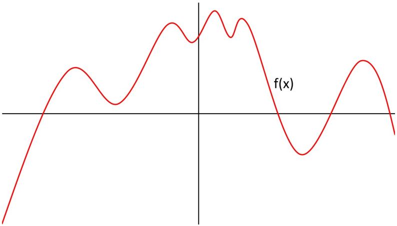
- *A key feature of these algorithms is that if the matrix-vector product for A can be computed quickly, e.g., when A is sparse, then Bu can also be computed in essentially the same time.*

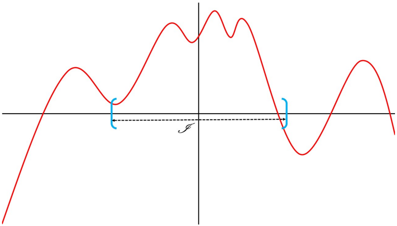
The classical area in analysis of **approximation theory** provides the right framework to study these questions.

Approximation Theory

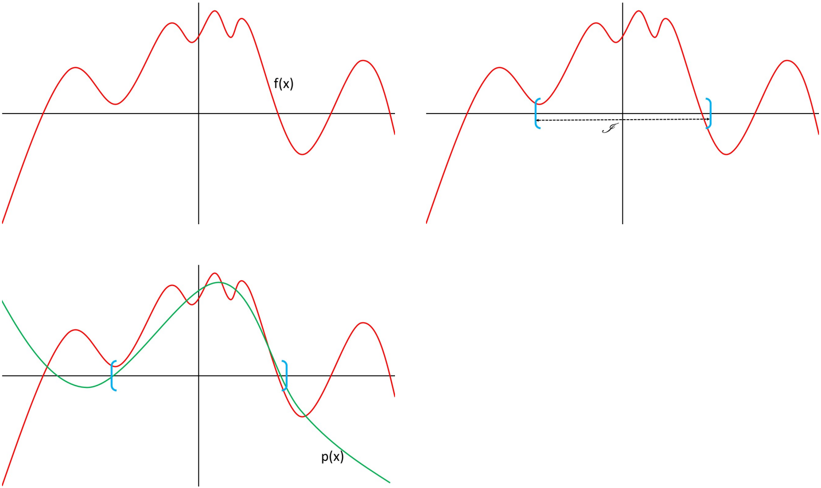


Approximation Theory

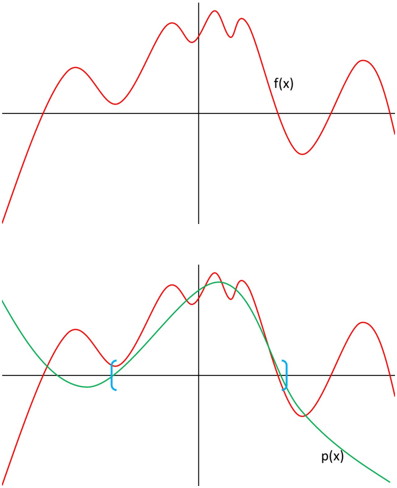




Approximation Theory



Approximation Theory



Approximation Theory

How **well** can functions be approximated by **simpler** ones?

Approximation Theory

How **well** can functions be approximated by **simpler** ones?

Uniform (Chebyshev) Approximation by Polynomials/Rationals

For $f : \mathbb{R} \mapsto \mathbb{R}$ and an interval $\mathcal{I}$, what is the closest a degree d polynomial/rational function can remain to $f(x)$ **throughout** $\mathcal{I}$

Approximation Theory

How **well** can functions be approximated by **simpler** ones?

Uniform (Chebyshev) Approximation by Polynomials/Rationals

For $f : \mathbb{R} \mapsto \mathbb{R}$ and an interval $\mathcal{I}$, what is the closest a degree d polynomial/rational function can remain to $f(x)$ **throughout** $\mathcal{I}$

$$\inf_{p \in \Sigma_d} \sup_{x \in \mathcal{I}} |f(x) - p(x)|.$$

$$\inf_{p,q \in \Sigma_d} \sup_{x \in \mathcal{I}} |f(x) - {}^{p(x)}/_{q(x)}|.$$

Σ_d: set of all polynomials of degree at most d.

Approximation Theory

How **well** can functions be approximated by **simpler** ones?

Uniform (Chebyshev) Approximation by Polynomials/Rationals

For $f : \mathbb{R} \mapsto \mathbb{R}$ and an interval $\mathcal{I}$, what is the closest a degree d polynomial/rational function can remain to $f(x)$ **throughout** $\mathcal{I}$

$$\inf_{p \in \Sigma_d} \sup_{x \in \mathcal{I}} |f(x) - p(x)|.$$

$$\inf_{p,q \in \Sigma_d} \sup_{x \in \mathcal{I}} |f(x) - {}^{p(x)}/_{q(x)}|.$$

Σ_d: set of all polynomials of degree at most d.

- 150+ years of fascinating history, deep results and many applications.

Approximation Theory

How **well** can functions be approximated by **simpler** ones?

Uniform (Chebyshev) Approximation by Polynomials/Rationals

For $f : \mathbb{R} \mapsto \mathbb{R}$ and an interval $\mathcal{I}$, what is the closest a degree d polynomial/rational function can remain to $f(x)$ **throughout** $\mathcal{I}$

$$\inf_{p \in \Sigma_d} \sup_{x \in \mathcal{I}} |f(x) - p(x)|.$$

$$\inf_{p,q \in \Sigma_d} \sup_{x \in \mathcal{I}} |f(x) - p(x)/q(x)|.$$

Σ_d: set of all polynomials of degree at most d.

- 150+ years of fascinating history, deep results and many applications.
- Interested in fundamental functions such as x^s, e^{-x} and $1/x$ over finite and infinite intervals such as $[-1, 1]$, $[0, n]$, $[0, \infty)$.

Approximation Theory

How **well** can functions be approximated by **simpler** ones?

Uniform (Chebyshev) Approximation by Polynomials/Rationals

For $f : \mathbb{R} \mapsto \mathbb{R}$ and an interval $\mathcal{I}$, what is the closest a degree d polynomial/rational function can remain to $f(x)$ **throughout** $\mathcal{I}$

$$\inf_{p \in \Sigma_d} \sup_{x \in \mathcal{I}} |f(x) - p(x)|.$$

$$\inf_{p,q \in \Sigma_d} \sup_{x \in \mathcal{I}} |f(x) - {}^{p(x)}/_{q(x)}|.$$

Σ_d: set of all polynomials of degree at most d.

- 150+ years of fascinating history, deep results and many applications.
- Interested in fundamental functions such as x^s, e^{-x} and $^1/_x$ over finite and infinite intervals such as $[-1, 1]$, $[0, n]$, $[0, \infty)$.
- For our applications good enough approximations suffice.

Algorithms/Numerical Linear Alg.- $f(A)v$, Eigenvalues, ..

A simple example:

Compute $A^s v$ where A is symmetric with eigenvalues in $[-1, 1]$, v is a vector and s is a large positive integer.

A simple example:

Compute $A^s v$ where A is symmetric with eigenvalues in $[-1, 1]$, v is a vector and s is a large positive integer.

- The straightforward way to compute $A^s v$ takes time $O(ms)$ where m is the number of non-zero entries in A.

A simple example:

Compute $A^s v$ where A is symmetric with eigenvalues in $[-1, 1]$, v is a vector and s is a large positive integer.

- The straightforward way to compute $A^s v$ takes time $O(ms)$ where m is the number of non-zero entries in A.
- **Suppose** x^s can be δ-approximated over the interval $[-1, 1]$ by a degree d polynomial $p_{s,d}(x) = \sum_{i=0}^{d} a_i x^i$.

A simple example:

Compute $A^s v$ where A is symmetric with eigenvalues in $[-1, 1]$, v is a vector and s is a large positive integer.

- The straightforward way to compute $A^s v$ takes time $O(ms)$ where m is the number of non-zero entries in A.
- **Suppose** x^s can be δ-approximated over the interval $[-1, 1]$ by a degree d polynomial $p_{s,d}(x) = \sum_{i=0}^{d} a_i x^i$.
- **Candidate approximation** to $A^s v$: $\sum_{i=0}^{d} a_i A^i v$.

A simple example:

Compute $A^s v$ where A is symmetric with eigenvalues in $[-1, 1]$, v is a vector and s is a large positive integer.

- The straightforward way to compute $A^s v$ takes time $O(ms)$ where m is the number of non-zero entries in A.
- **Suppose** x^s can be δ-approximated over the interval $[-1, 1]$ by a degree d polynomial $p_{s,d}(x) = \sum_{i=0}^{d} a_i x^i$.
- **Candidate approximation** to $A^s v$: $\sum_{i=0}^{d} a_i A^i v$.
- The time to compute $\sum_{i=0}^{d} a_i A^i v$ is $O(md)$.

A simple example:

Compute $A^s v$ where A is symmetric with eigenvalues in $[-1, 1]$, v is a vector and s is a large positive integer.

- The straightforward way to compute $A^s v$ takes time $O(ms)$ where m is the number of non-zero entries in A.
- **Suppose** x^s can be δ-approximated over the interval $[-1, 1]$ by a degree d polynomial $p_{s,d}(x) = \sum_{i=0}^{d} a_i x^i$.
- **Candidate approximation** to $A^s v$: $\sum_{i=0}^{d} a_i A^i v$.
- The time to compute $\sum_{i=0}^{d} a_i A^i v$ is $O(md)$.
- $\| \sum_{i=0}^{d} a_i A^i v - A^s v\| \leq \delta \|v\|$ since
  - all the eigenvalues of A lie in $[-1, 1]$, and
  - $p_{s,d}$ is δ-close to x^s in the entire interval $[-1, 1]$.

A simple example:

Compute $A^s v$ where A is symmetric with eigenvalues in $[-1, 1]$, v is a vector and s is a large positive integer.

- The straightforward way to compute $A^s v$ takes time $O(ms)$ where m is the number of non-zero entries in A.
- **Suppose** x^s can be δ-approximated over the interval $[-1, 1]$ by a degree d polynomial $p_{s,d}(x) = \sum_{i=0}^{d} a_i x^i$.
- **Candidate approximation** to $A^s v$: $\sum_{i=0}^{d} a_i A^i v$.
- The time to compute $\sum_{i=0}^{d} a_i A^i v$ is $O(md)$.
- $\| \sum_{i=0}^{d} a_i A^i v - A^s v \| \leq \delta \|v\|$ since
  - all the eigenvalues of A lie in $[-1, 1]$, and
  - $p_{s,d}$ is δ-close to x^s in the entire interval $[-1, 1]$.

How small can d be?

Example: Approximating the Monomial

For any s, for any $\delta > 0$, and $d \sim \sqrt{s \log(1/\delta)}$, there is a polynomial $p_{s,d}$ s.t. $\sup_{x \in [-1,1]} |p_{s,d}(x) - x^s| \leq \delta$.

Example: Approximating the Monomial

For any s, for any $\delta > 0$, and $d \sim \sqrt{s \log (1/\delta)}$, there is a polynomial $p_{s,d}$ s.t. $\sup_{x \in [-1,1]} |p_{s,d}(x) - x^s| \leq \delta$.

- **Simulating Random Walks**: If A is random walk matrix of a graph, we can simulate s steps of a random walk in $m\sqrt{s}$ time.

Example: Approximating the Monomial

For any s, for any $\delta > 0$, and $d \sim \sqrt{s \log (1/\delta)}$, there is a polynomial $p_{s,d}$ s.t. $\sup_{x \in [-1,1]} |p_{s,d}(x) - x^s| \leq \delta$.

- **Simulating Random Walks**: If A is random walk matrix of a graph, we can simulate s steps of a random walk in $m\sqrt{s}$ time.
- **Conjugate Gradient Method**: Given $Ax = b$ with eigenvalues of A in $(0, 1]$, one can find y s.t. $\|y - A^{-1}b\|_A \leq \delta \|A^{-1}b\|_A$ in time roughly $m\sqrt{\kappa(A) \log 1/\delta}$.

Example: Approximating the Monomial

For any s, for any $\delta > 0$, and $d \sim \sqrt{s \log\left({}^{1}\!/_{\delta}\right)}$, there is a polynomial $p_{s,d}$ s.t. $\sup_{x \in [-1,1]} |p_{s,d}(x) - x^s| \leq \delta$.

- **Simulating Random Walks**: If A is random walk matrix of a graph, we can simulate s steps of a random walk in $m\sqrt{s}$ time.
- **Conjugate Gradient Method**: Given $Ax = b$ with eigenvalues of A in $(0, 1]$, one can find y s.t. $\|y - A^{-1}b\|_A \leq \delta \|A^{-1}b\|_A$ in time roughly $m\sqrt{\kappa(A) \log {}^{1}\!/_{\delta}}$.
- **Quadratic speedup over the Power Method**: Given A, in time $\sim {}^{m}\!/_{\sqrt{\delta}}$ can compute a value $\mu \in [(1-\delta)\lambda_1(A), \lambda_1(A)]$.

Chebyshev Polynomials

The Chebyshev polynomial of deg. d is defined recursively to be:

$$T_d(x) \stackrel{\text{def}}{=} 2xT_{d-1}(x) - T_{d-2}(x)$$

for $d \geq 2$ with $T_0(x) \stackrel{\text{def}}{=} 1$, $T_1(x) \stackrel{\text{def}}{=} x$.

Chebyshev Polynomials

The Chebyshev polynomial of deg. d is defined recursively to be:

$$T_d(x) \stackrel{\text{def}}{=} 2xT_{d-1}(x) - T_{d-2}(x)$$

for $d \geq 2$ with $T_0(x) \stackrel{\text{def}}{=} 1$, $T_1(x) \stackrel{\text{def}}{=} x$.

Averaging Property

$$xT_d(x) = \frac{T_{d+1}(x) + T_{d-1}(x)}{2}.$$

Chebyshev Polynomials

The Chebyshev polynomial of deg. d is defined recursively to be:

$$T_d(x) \stackrel{\text{def}}{=} 2xT_{d-1}(x) - T_{d-2}(x)$$

for $d \geq 2$ with $T_0(x) \stackrel{\text{def}}{=} 1$, $T_1(x) \stackrel{\text{def}}{=} x$.

Averaging Property

$$xT_d(x) = \frac{T_{d+1}(x) + T_{d-1}(x)}{2}.$$

Boundedness Property

For any θ, and any integer d, $T_d(\cos\theta) = \cos(d\theta)$.

Chebyshev Polynomials

The Chebyshev polynomial of deg. d is defined recursively to be:

$$T_d(x) \stackrel{\text{def}}{=} 2xT_{d-1}(x) - T_{d-2}(x)$$

for $d \geq 2$ with $T_0(x) \stackrel{\text{def}}{=} 1$, $T_1(x) \stackrel{\text{def}}{=} x$.

Averaging Property

$$xT_d(x) = \frac{T_{d+1}(x) + T_{d-1}(x)}{2}.$$

Boundedness Property

For any θ, and any integer d, $T_d(\cos\theta) = \cos(d\theta)$.
Thus, $|T_d(x)| \leq 1$ for all $x \in [-1, 1]$.

Back to Approximating Monomials

$D_s \stackrel{\text{def}}{=} \sum_{i=1}^{s} Y_i$ where $Y_1, \ldots, Y_s$ i.i.d. ± 1 w.p. $1/2$ ($D_0 \stackrel{\text{def}}{=} 0$).

Back to Approximating Monomials

$D_s \stackrel{\text{def}}{=} \sum_{i=1}^{s} Y_i$ where $Y_1, \ldots, Y_s$ i.i.d. ± 1 w.p. $1/2$ ($D_0 \stackrel{\text{def}}{=} 0$).

Thus, $\Pr\left[|D_s| \geq \sqrt{2s \log(2/\delta)}\right] \leq \delta.$

Back to Approximating Monomials

$D_s \stackrel{\text{def}}{=} \sum_{i=1}^{s} Y_i$ where $Y_1, \ldots, Y_s$ i.i.d. ± 1 w.p. $1/2$ ($D_0 \stackrel{\text{def}}{=} 0$).

Thus, $\Pr\left[|D_s| \geq \sqrt{2s \log(2/\delta)}\right] \leq \delta$.

Key Claim: $\mathop{\mathbf{E}}_{Y_1,\ldots,Y_s}[T_{D_s}(x)] = x^s$.

Back to Approximating Monomials

$D_s \stackrel{\text{def}}{=} \sum_{i=1}^{s} Y_i$ where $Y_1, \ldots, Y_s$ i.i.d. ± 1 w.p. $1/2$ ($D_0 \stackrel{\text{def}}{=} 0$).

Thus, $\Pr\left[|D_s| \geq \sqrt{2s \log(2/\delta)}\right] \leq \delta$.

Key Claim: $\mathop{\mathbf{E}}_{Y_1,\ldots,Y_s}[T_{D_s}(x)] = x^s$.

$$\begin{aligned} x^{s+1} &= x \cdot \mathop{\mathbf{E}}_{Y_1,\ldots,Y_s} T_{D_s}(x) = \mathop{\mathbf{E}}_{Y_1,\ldots,Y_s}[x \cdot T_{D_s}(x)] \\ &= \mathop{\mathbf{E}}_{Y_1,\ldots,Y_s}[1/2(T_{D_s+1}(x) + T_{D_s-1}(x))] = \mathop{\mathbf{E}}_{Y_1,\ldots,Y_{s+1}}[T_{D_{s+1}}(x)]. \end{aligned}$$

Back to Approximating Monomials

$D_s \stackrel{\text{def}}{=} \sum_{i=1}^{s} Y_i$ where $Y_1, \ldots, Y_s$ i.i.d. ± 1 w.p. $1/2$ ($D_0 \stackrel{\text{def}}{=} 0$).

Thus, $\Pr\left[|D_s| \geq \sqrt{2s\log(2/\delta)}\right] \leq \delta$.

Key Claim: $\mathop{\mathbf{E}}_{Y_1,\ldots,Y_s}[T_{D_s}(x)] = x^s$.

$$\begin{aligned} x^{s+1} &= x \cdot \mathop{\mathbf{E}}_{Y_1,\ldots,Y_s} T_{D_s}(x) = \mathop{\mathbf{E}}_{Y_1,\ldots,Y_s}[x \cdot T_{D_s}(x)] \\ &= \mathop{\mathbf{E}}_{Y_1,\ldots,Y_s}[1/2(T_{D_s+1}(x) + T_{D_s-1}(x))] = \mathop{\mathbf{E}}_{Y_1,\ldots,Y_{s+1}}[T_{D_{s+1}}(x)]. \end{aligned}$$

Our Approximation to x^s:

$$p_{s,d}(x) \stackrel{\text{def}}{=} \mathop{\mathbf{E}}_{Y_1,\ldots,Y_s}\left[T_{D_s}(x) \cdot 1_{|D_s| \leq d}\right] \quad \text{for} \quad d = \sqrt{2s\log(2/\delta)}.$$

Back to Approximating Monomials

$D_s \stackrel{\text{def}}{=} \sum_{i=1}^{s} Y_i$ where $Y_1, \ldots, Y_s$ i.i.d. ± 1 w.p. $1/2$ ($D_0 \stackrel{\text{def}}{=} 0$).

Thus, $\Pr\left[|D_s| \geq \sqrt{2s \log(2/\delta)}\right] \leq \delta$.

Key Claim: $\mathop{\mathbf{E}}_{Y_1,\ldots,Y_s}[T_{D_s}(x)] = x^s$.

$$\begin{aligned} x^{s+1} &= x \cdot \mathop{\mathbf{E}}_{Y_1,\ldots,Y_s} T_{D_s}(x) = \mathop{\mathbf{E}}_{Y_1,\ldots,Y_s}[x \cdot T_{D_s}(x)] \\ &= \mathop{\mathbf{E}}_{Y_1,\ldots,Y_s}[1/2(T_{D_s+1}(x) + T_{D_s-1}(x))] = \mathop{\mathbf{E}}_{Y_1,\ldots,Y_{s+1}}[T_{D_{s+1}}(x)]. \end{aligned}$$

Our Approximation to x^s:

$$p_{s,d}(x) \stackrel{\text{def}}{=} \mathop{\mathbf{E}}_{Y_1,\ldots,Y_s}\left[T_{D_s}(x) \cdot 1_{|D_s| \leq d}\right] \quad \text{for} \quad d = \sqrt{2s \log(2/\delta)}.$$

$$\begin{aligned} \sup_{x \in [-1,1]} |p_{s,d}(x) - x^s| &= \sup_{x \in [-1,1]} \left| \mathop{\mathbf{E}}_{Y_1,\ldots,Y_s}\left[T_{D_s}(x) \cdot 1_{|D_s| > d}\right] \right| \\ &\leq \mathop{\mathbf{E}}_{Y_1,\ldots,Y_s}\left[1_{|D_s| > d} \cdot \sup_{x \in [-1,1]} |T_{D_s}(x)|\right] \leq \mathop{\mathbf{E}}_{Y_1,\ldots,Y_s}\left[1_{|D_s| > d}\right] \leq \delta. \end{aligned}$$

A General Recipe?

Suppose $f(x)$ is δ-approximated by a Taylor polynomial $\sum_{s=0}^{k} c_s x^s$, then one may instead try the approx. (with suitably shifted $p_{s,d}$)

$$\sum_{s=0}^{k} c_s p_{s,\sqrt{s \log 1/\delta}}(x)$$

.

A General Recipe?

Suppose $f(x)$ is δ-approximated by a Taylor polynomial $\sum_{s=0}^{k} c_s x^s$, then one may instead try the approx. (with suitably shifted $p_{s,d}$)

$$\sum_{s=0}^{k} c_s p_{s,\sqrt{s \log 1/\delta}}(x)$$

.

Approximating the Exponential

For every $b > 0$, and δ, there is a polynomial $r_{b,\delta}$ s.t. $\sup_{x \in [0,b]} |e^{-x} - r_{b,\delta}(x)| \leq \delta$; degree $\sim \sqrt{b \log 1/\delta}$. (Taylor -$\Omega(b)$.)

A General Recipe?

Suppose $f(x)$ is δ-approximated by a Taylor polynomial $\sum_{s=0}^{k} c_s x^s$, then one may instead try the approx. (with suitably shifted $p_{s,d}$)

$$\sum_{s=0}^{k} c_s p_{s,\sqrt{s \log 1/\delta}}(x)$$

.

Approximating the Exponential

For every $b > 0$, and δ, there is a polynomial $r_{b,\delta}$ s.t. $\sup_{x\in[0,b]} |e^{-x} - r_{b,\delta}(x)| \le \delta$; degree $\sim \sqrt{b \log 1/\delta}$. (Taylor -$\Omega(b)$.)

- Implies $\tilde{O}(m\sqrt{\|A\| \log 1/\delta})$ time algorithm to compute a δ-approximation to $e^{-A}v$ for a PSD A. *Useful in solving SDPs.*

A General Recipe?

Suppose $f(x)$ is δ-approximated by a Taylor polynomial $\sum_{s=0}^{k} c_s x^s$, then one may instead try the approx. (with suitably shifted $p_{s,d}$)

$$\sum_{s=0}^{k} c_s p_{s,\sqrt{s \log 1/\delta}}(x)$$

.

Approximating the Exponential

For every $b > 0$, and δ, there is a polynomial $r_{b,\delta}$ s.t. $\sup_{x \in [0,b]} |e^{-x} - r_{b,\delta}(x)| \leq \delta$; degree $\sim \sqrt{b \log 1/\delta}$. (Taylor -$\Omega(b)$.)

- Implies $\tilde{O}(m\sqrt{\|A\| \log 1/\delta})$ time algorithm to compute a δ-approximation to $e^{-A}v$ for a PSD A. *Useful in solving SDPs.*
- When A is a graph Laplacian, implies an optimal spectral algorithm for Balanced Separator that runs in time $\tilde{O}(m/\sqrt{\gamma})$. ($\gamma$ is the target conductance) [Orecchia-Sachdeva-V. 2012].

A General Recipe?

Suppose $f(x)$ is δ-approximated by a Taylor polynomial $\sum_{s=0}^{k} c_s x^s$, then one may instead try the approx. (with suitably shifted $p_{s,d}$)

$$\sum_{s=0}^{k} c_s p_{s,\sqrt{s \log 1/\delta}}(x)$$

.

Approximating the Exponential

For every $b > 0$, and δ, there is a polynomial $r_{b,\delta}$ s.t. $\sup_{x \in [0,b]} |e^{-x} - r_{b,\delta}(x)| \leq \delta$; degree $\sim \sqrt{b \log 1/\delta}$. (Taylor -$\Omega(b)$.)

- Implies $\tilde{O}(m\sqrt{\|A\| \log 1/\delta})$ time algorithm to compute a δ-approximation to $e^{-A}v$ for a PSD A. *Useful in solving SDPs.*
- When A is a graph Laplacian, implies an optimal spectral algorithm for Balanced Separator that runs in time $\tilde{O}(m/\sqrt{\gamma})$. ($\gamma$ is the target conductance) [Orecchia-Sachdeva-V. 2012].

How far can polynomial approximations take us?

Lower Bounds for Polynomial Approximations

Bad News [see Sachdeva-V. 2014]

- Polynomial approx. to x^s on $[-1, 1]$ requires degree $\Omega(\sqrt{s})$.
- Polynomials approx. to e^{-x} on $[0, b]$ requires degree $\Omega(\sqrt{b})$.

Lower Bounds for Polynomial Approximations

Bad News [see Sachdeva-V. 2014]

- Polynomial approx. to x^s on $[-1, 1]$ requires degree $\Omega(\sqrt{s})$.
- Polynomials approx. to e^{-x} on $[0, b]$ requires degree $\Omega(\sqrt{b})$.

Markov's Theorem (inspired by a prob. of Mendeleev in Chemistry)

Any degree-d polynomial p s.t. $|p(x)| \leq 1$ over $[-1, 1]$ must have its derivative $|p^{(1)}(x)| \leq d^2$ for all $x \in [-1, 1]$.

Lower Bounds for Polynomial Approximations

Bad News [see Sachdeva-V. 2014]

- Polynomial approx. to x^s on $[-1,1]$ requires degree $\Omega(\sqrt{s})$.
- Polynomials approx. to e^{-x} on $[0,b]$ requires degree $\Omega(\sqrt{b})$.

Markov's Theorem (inspired by a prob. of Mendeleev in Chemistry)

Any degree-d polynomial p s.t. $|p(x)| \leq 1$ over $[-1,1]$ must have its derivative $|p^{(1)}(x)| \leq d^2$ for all $x \in [-1,1]$.

- Chebyshev polynomials are a tight example for this theorem.

Lower Bounds for Polynomial Approximations

Bad News [see Sachdeva-V. 2014]

- Polynomial approx. to x^s on $[-1, 1]$ requires degree $\Omega(\sqrt{s})$.
- Polynomials approx. to e^{-x} on $[0, b]$ requires degree $\Omega(\sqrt{b})$.

Markov's Theorem (inspired by a prob. of Mendeleev in Chemistry)

Any degree-d polynomial p s.t. $|p(x)| \leq 1$ over $[-1, 1]$ must have its derivative $|p^{(1)}(x)| \leq d^2$ for all $x \in [-1, 1]$.

- Chebyshev polynomials are a tight example for this theorem.

Bypass this barrier via rational functions!

Example: Approximating the Exponential

For all integers $d \geq 0$, there is a degree-d polynomial $S_d(x)$ s.t. $\sup_{x \in [0,\infty)} \left| e^{-x} - \frac{1}{S_d(x)} \right| \leq 2^{-\Omega(d)}$.

Example: Approximating the Exponential

For all integers $d \geq 0$, there is a degree-d polynomial $S_d(x)$ s.t. $\sup_{x\in[0,\infty)} \left| e^{-x} - \frac{1}{S_d(x)} \right| \leq 2^{-\Omega(d)}$.

$S_d(x) \stackrel{\text{def}}{=} \sum_{k=0}^{d} \frac{x^k}{k!}$. (Proof by induction.)

Example: Approximating the Exponential

For all integers $d \geq 0$, there is a degree-d polynomial $S_d(x)$ s.t. $\sup_{x \in [0,\infty)} \left| e^{-x} - \frac{1}{S_d(x)} \right| \leq 2^{-\Omega(d)}$.

$S_d(x) \stackrel{\text{def}}{=} \sum_{k=0}^{d} \frac{x^k}{k!}$. (Proof by induction.)

- No dependence on the length of the interval!

Example: Approximating the Exponential

For all integers $d \geq 0$, there is a degree-d polynomial $S_d(x)$ s.t. $\sup_{x \in [0,\infty)} \left| e^{-x} - \frac{1}{S_d(x)} \right| \leq 2^{-\Omega(d)}$.

$$S_d(x) \stackrel{\text{def}}{=} \sum_{k=0}^{d} \frac{x^k}{k!}.$$ (Proof by induction.)

- No dependence on the length of the interval!
- Hence, for any $\delta > 0$, we have a rational function of degree $O(\log 1/\delta)$ that is a δ-approximation to e^{-x}. For most applications, an error of $\delta = 1/\mathrm{poly}(n)$ suffices, so we can choose $d = O(\log n)$.

Example: Approximating the Exponential

For all integers $d \geq 0$, there is a degree-d polynomial $S_d(x)$ s.t. $\sup_{x \in [0,\infty)} \left| e^{-x} - \frac{1}{S_d(x)} \right| \leq 2^{-\Omega(d)}$.

$S_d(x) \stackrel{\text{def}}{=} \sum_{k=0}^{d} \frac{x^k}{k!}$. (Proof by induction.)

- No dependence on the length of the interval!
- Hence, for any $\delta > 0$, we have a rational function of degree $O(\log 1/\delta)$ that is a δ-approximation to e^{-x}. For most applications, an error of $\delta = 1/\text{poly}(n)$ suffices, so we can choose $d = O(\log n)$.
- Thus, $(S_d(A))^{-1} v$ δ-approximates $e^{-A} v$.

Example: Approximating the Exponential

For all integers $d \geq 0$, there is a degree-d polynomial $S_d(x)$ s.t. $\sup_{x \in [0,\infty)} \left| e^{-x} - \frac{1}{S_d(x)} \right| \leq 2^{-\Omega(d)}$.

$$S_d(x) \stackrel{\text{def}}{=} \sum_{k=0}^{d} \frac{x^k}{k!}. \text{ (Proof by induction.)}$$

- No dependence on the length of the interval!
- Hence, for any $\delta > 0$, we have a rational function of degree $O(\log 1/\delta)$ that is a δ-approximation to e^{-x}. For most applications, an error of $\delta = 1/\mathrm{poly}(n)$ suffices, so we can choose $d = O(\log n)$.
- Thus, $(S_d(A))^{-1} v$ δ-approximates $e^{-A}v$.

How do we compute $(S_d(A))^{-1} v$?

Why any Rational Approximation is not Enough?

Factor $S_d(x) = \alpha_0 \prod_{i=1}^{d}(x - \beta_i)$ and output $\alpha_0 \prod_{i=1}^{d}(A - \beta_i I)^{-1} v$.

Why any Rational Approximation is not Enough?

Factor $S_d(x) = \alpha_0 \prod_{i=1}^{d}(x - \beta_i)$ and output $\alpha_0 \prod_{i=1}^{d}(A - \beta_i I)^{-1} v$.

- Since d is $O(\log n)$, sufficient to compute $(A - \beta_i I)^{-1} u$.

Why any Rational Approximation is not Enough?

Factor $S_d(x) = \alpha_0 \prod_{i=1}^{d} (x - \beta_i)$ and output $\alpha_0 \prod_{i=1}^{d} (A - \beta_i I)^{-1} v$.

- Since d is $O(\log n)$, sufficient to compute $(A - \beta_i I)^{-1} u$.
- When A is Laplacian, and $\beta_i \leq 0$, then $A - \beta_i I$ is **SDD!**

Why any Rational Approximation is not Enough?

Factor $S_d(x) = \alpha_0 \prod_{i=1}^{d}(x - \beta_i)$ and output $\alpha_0 \prod_{i=1}^{d}(A - \beta_i I)^{-1} v$.

- Since d is $O(\log n)$, sufficient to compute $(A - \beta_i I)^{-1} u$.
- When A is Laplacian, and $\beta_i \leq 0$, then $A - \beta_i I$ is **SDD!**
- β_is could be complex:
  $S_d(x)$ has exactly one real zero $x_d \in [-d, -1]$ if d is odd, and no real zeros if d is even. Also, zeros of $S_d(x)$ grow linearly in magnitude with d.

Why any Rational Approximation is not Enough?

Factor $S_d(x) = \alpha_0 \prod_{i=1}^{d}(x - \beta_i)$ and output $\alpha_0 \prod_{i=1}^{d}(A - \beta_i I)^{-1} v$.

- Since d is $O(\log n)$, sufficient to compute $(A - \beta_i I)^{-1} u$.
- When A is Laplacian, and $\beta_i \leq 0$, then $A - \beta_i I$ is **SDD!**
- β_is could be complex:
  $S_d(x)$ has exactly one real zero $x_d \in [-d, -1]$ if d is odd, and no real zeros if d is even. Also, zeros of $S_d(x)$ grow linearly in magnitude with d.
- However, since S_d has real coefficients, its complex roots appear as conjugates. Hence, the task reduces to computing $(A^2 - (\beta_i + \bar{\beta}_i)A + |\beta_i|^2 I)^{-1} u$.

Why any Rational Approximation is not Enough?

Factor $S_d(x) = \alpha_0 \prod_{i=1}^{d}(x - \beta_i)$ and output $\alpha_0 \prod_{i=1}^{d}(A - \beta_i I)^{-1} v$.

- Since d is $O(\log n)$, sufficient to compute $(A - \beta_i I)^{-1} u$.
- When A is Laplacian, and $\beta_i \leq 0$, then $A - \beta_i I$ is **SDD!**
- β_is could be complex:
  $S_d(x)$ has exactly one real zero $x_d \in [-d, -1]$ if d is odd, and no real zeros if d is even. Also, zeros of $S_d(x)$ grow linearly in magnitude with d.
- However, since S_d has real coefficients, its complex roots appear as conjugates. Hence, the task reduces to computing $(A^2 - (\beta_i + \bar{\beta}_i)A + |\beta_i|^2 I)^{-1} u$.
- The matrix $(A^2 - (\beta_i + \bar{\beta}_i)A + |\beta_i|^2 I)$ is PSD but the condition number can be comparable to that of A.

Why any Rational Approximation is not Enough?

Factor $S_d(x) = \alpha_0 \prod_{i=1}^{d}(x - \beta_i)$ and output $\alpha_0 \prod_{i=1}^{d}(A - \beta_i I)^{-1} v$.

- Since d is $O(\log n)$, sufficient to compute $(A - \beta_i I)^{-1} u$.
- When A is Laplacian, and $\beta_i \leq 0$, then $A - \beta_i I$ is **SDD!**
- β_is could be complex:
  $S_d(x)$ has exactly one real zero $x_d \in [-d, -1]$ if d is odd, and no real zeros if d is even. Also, zeros of $S_d(x)$ grow linearly in magnitude with d.
- However, since S_d has real coefficients, its complex roots appear as conjugates. Hence, the task reduces to computing $(A^2 - (\beta_i + \bar{\beta}_i)A + |\beta_i|^2 I)^{-1} u$.
- The matrix $(A^2 - (\beta_i + \bar{\beta}_i)A + |\beta_i|^2 I)$ is PSD but the condition number can be comparable to that of A.

Desire: A rational approximation with negative poles.

Rational Approximation with Negative Poles

- How about $(1 + {}^{x}\!/_{d})^{-d}$? Converges to e^{-x} unif. over $[0, \infty)$.

Rational Approximation with Negative Poles

- How about $(1 + {}^{x}\!/_{d})^{-d}$? Converges to e^{-x} unif. over $[0, \infty)$.
- Convergence rate slow: at $x = 1$ error is $\Theta({}^{1}\!/_{d})$.

Rational Approximation with Negative Poles

- How about $(1 + {}^{x}\!/_{d})^{-d}$? Converges to e^{-x} unif. over $[0, \infty)$.
- Convergence rate slow: at $x = 1$ error is $\Theta({}^{1}\!/_{d})$.
- More generally, for every rational function of the form ${}^{1}\!/_{p_d(x)}$, where p_d is a degree-d polynomial with real roots:

$$\sup_{x \in [0,\infty)} \left| e^{-x} - {}^{1}\!/_{p_d(x)} \right| = \Omega({}^{1}\!/_{d^2}).$$

Rational Approximation with Negative Poles

- How about $(1+x/d)^{-d}$? Converges to e^{-x} unif. over $[0,\infty)$.
- Convergence rate slow: at $x=1$ error is $\Theta(1/d)$.
- More generally, for every rational function of the form $1/p_d(x)$, where p_d is a degree-d polynomial with real roots:

$$\sup_{x\in[0,\infty)} \left|e^{-x} - 1/p_d(x)\right| = \Omega(1/d^2).$$

Saff-Schönhage-Varga 1975

For every d, there exists a degree-d polynomial p_d s.t.,

$$\sup_{x\in[0,\infty)} \left|e^{-x} - p_d\left(\frac{1}{1+x/d}\right)\right| \leq 2^{-\Omega(d)}.$$

Rational Approximation with Negative Poles

- How about $(1 + x/d)^{-d}$? Converges to e^{-x} unif. over $[0, \infty)$.
- Convergence rate slow: at $x = 1$ error is $\Theta(1/d)$.
- More generally, for every rational function of the form $1/p_d(x)$, where p_d is a degree-d polynomial with real roots:

$$\sup_{x \in [0,\infty)} \left| e^{-x} - 1/p_d(x) \right| = \Omega(1/d^2).$$

Saff-Schönhage-Varga 1975

For every d, there exists a degree-d polynomial p_d s.t.,

$$\sup_{x \in [0,\infty)} \left| e^{-x} - p_d\left(\frac{1}{1+x/d}\right) \right| \leq 2^{-\Omega(d)}.$$

Sachdeva-V. 2014

Moreover, the coefficients of p_d are bounded by $d^{O(d)}$, and can be approximated up to an error of $d^{-\Theta(d)}$ using $\mathrm{poly}(d)$ arithmetic operations, where all intermediate numbers $\mathrm{poly}(d)$ bits.

Computing the Matrix Exponential- Summary

Orecchia-Sachdeva-V. 2012, Sachdeva-V. 2014

Given an **SDD** $A \succeq 0$, a vector v with $\|v\| = 1$ and δ, we compute a vector u s.t. $\|\exp(-A)v - u\| \leq \delta$, in time $\tilde{O}\left(m \log\|A\| \log 1/\delta\right)$.

Computing the Matrix Exponential- Summary

Orecchia-Sachdeva-V. 2012, Sachdeva-V. 2014

Given an **SDD** $A \succeq 0$, a vector v with $\|v\| = 1$ and δ, we compute a vector u s.t. $\|\exp(-A)v - u\| \leq \delta$, in time $\tilde{O}\left(m \log \|A\| \log 1/\delta\right)$.

Corollary [Orecchia-Sachdeva-V. 2012]

$\sqrt{\gamma}$-approximation for Balanced separator in time $\tilde{O}(m)$. Spectral guarantee for approximation, running time *independent* of γ

Computing the Matrix Exponential- Summary

Orecchia-Sachdeva-V. 2012, Sachdeva-V. 2014

Given an **SDD** $A \succeq 0$, a vector v with $\|v\| = 1$ and δ, we compute a vector u s.t. $\|\exp(-A)v - u\| \leq \delta$, in time $\tilde{O}\left(m \log\|A\| \log 1/\delta\right)$.

Corollary [Orecchia-Sachdeva-V. 2012]

$\sqrt{\gamma}$-approximation for Balanced separator in time $\tilde{O}(m)$. Spectral guarantee for approximation, running time *independent* of γ

SDD Solvers

Given $Lx = b$, L is SDD, and $\varepsilon > 0$, obtain a vector u s.t., $\|u - L^{-1}b\|_L \leq \varepsilon\|L^{-1}b\|_L$. Time required $\tilde{O}\left(m \log 1/\varepsilon\right)$

Computing the Matrix Exponential- Summary

Orecchia-Sachdeva-V. 2012, Sachdeva-V. 2014

Given an **SDD** $A \succeq 0$, a vector v with $\|v\| = 1$ and δ, we compute a vector u s.t. $\|\exp(-A)v - u\| \leq \delta$, in time $\tilde{O}\left(m \log\|A\| \log 1/\delta\right)$.

Corollary [Orecchia-Sachdeva-V. 2012]

$\sqrt{\gamma}$-approximation for Balanced separator in time $\tilde{O}(m)$. Spectral guarantee for approximation, running time *independent* of γ

SDD Solvers

Given $Lx = b$, L is SDD, and $\varepsilon > 0$, obtain a vector u s.t., $\|u - L^{-1}b\|_L \leq \varepsilon\|L^{-1}b\|_L$. Time required $\tilde{O}\left(m \log 1/\varepsilon\right)$

Are Laplacian solvers necessary for the matrix exponential?

Matrix Inversion via Exponentiation

Belykin-Monzon 2010, Sachdeva-V. 2014

For $\varepsilon, \delta \in (0,1]$, there exist $\mathrm{poly}(\log(1/\varepsilon\delta))$ numbers $0 < w_j, t_j$ s.t. for all symm. $\varepsilon I \preceq A \preceq I$, $(1-\delta)A^{-1} \preceq \sum_j w_j e^{-t_j A} \preceq (1+\delta)A^{-1}$.

Matrix Inversion via Exponentiation

Belykin-Monzon 2010, Sachdeva-V. 2014

For $\varepsilon, \delta \in (0,1]$, there exist $\mathrm{poly}(\log(1/\varepsilon\delta))$ numbers $0 < w_j, t_j$ s.t. for all symm. $\varepsilon I \preceq A \preceq I$, $(1-\delta)A^{-1} \preceq \sum_j w_j e^{-t_j A} \preceq (1+\delta)A^{-1}$.

- Weights w_j are $O(\mathrm{poly}(1/\delta\varepsilon))$, we lose only a polynomial factor in the approximation error.

Matrix Inversion via Exponentiation

Belykin-Monzon 2010, Sachdeva-V. 2014

For $\varepsilon, \delta \in (0,1]$, there exist $\mathrm{poly}(\log(1/\varepsilon\delta))$ numbers $0 < w_j, t_j$ s.t. for all symm. $\varepsilon I \preceq A \preceq I$, $(1-\delta)A^{-1} \preceq \sum_j w_j e^{-t_j A} \preceq (1+\delta)A^{-1}$.

- Weights w_j are $O(\mathrm{poly}(1/\delta\varepsilon))$, we lose only a polynomial factor in the approximation error.
- For applications polylogarithmic dependence on both $1/\delta$ and the condition number of A ($1/\varepsilon$ in this case).

Matrix Inversion via Exponentiation

Belykin-Monzon 2010, Sachdeva-V. 2014

For $\varepsilon, \delta \in (0, 1]$, there exist $\mathrm{poly}(\log(1/\varepsilon\delta))$ numbers $0 < w_j, t_j$ s.t. for all symm. $\varepsilon I \preceq A \preceq I$, $(1-\delta)A^{-1} \preceq \sum_j w_j e^{-t_j A} \preceq (1+\delta)A^{-1}$.

- Weights w_j are $O(\mathrm{poly}(1/\delta\varepsilon))$, we lose only a polynomial factor in the approximation error.
- For applications polylogarithmic dependence on both $1/\delta$ and the condition number of A ($1/\varepsilon$ in this case).
- Discretizing $x^{-1} = \int_0^\infty e^{-xt} \mathrm{d}t$ naively **needs** $\mathrm{poly}(1/(\varepsilon\delta))$ terms.

Matrix Inversion via Exponentiation

Belykin-Monzon 2010, Sachdeva-V. 2014

For $\varepsilon, \delta \in (0,1]$, there exist $\mathrm{poly}(\log(1/\varepsilon\delta))$ numbers $0 < w_j, t_j$ s.t. for all symm. $\varepsilon I \preceq A \preceq I$, $(1-\delta)A^{-1} \preceq \sum_j w_j e^{-t_j A} \preceq (1+\delta)A^{-1}$.

- Weights w_j are $O(\mathrm{poly}(1/\delta\varepsilon))$, we lose only a polynomial factor in the approximation error.
- For applications polylogarithmic dependence on both $1/\delta$ and the condition number of A ($1/\varepsilon$ in this case).
- Discretizing $x^{-1} = \int_0^\infty e^{-xt}\mathrm{d}t$ naively **needs** $\mathrm{poly}(1/(\varepsilon\delta))$ terms.
- Substituting $t = e^y$ in the above integral obtains the identity $x^{-1} = \int_{-\infty}^{\infty} e^{-xe^y+y}\mathrm{d}y$.

Matrix Inversion via Exponentiation

Belykin-Monzon 2010, Sachdeva-V. 2014

For $\varepsilon, \delta \in (0,1]$, there exist $\mathrm{poly}(\log(1/\varepsilon\delta))$ numbers $0 < w_j, t_j$ s.t. for all symm. $\varepsilon I \preceq A \preceq I$, $(1-\delta)A^{-1} \preceq \sum_j w_j e^{-t_j A} \preceq (1+\delta)A^{-1}$.

- Weights w_j are $O(\mathrm{poly}(1/\delta\varepsilon))$, we lose only a polynomial factor in the approximation error.
- For applications polylogarithmic dependence on both $1/\delta$ and the condition number of A ($1/\varepsilon$ in this case).
- Discretizing $x^{-1} = \int_0^\infty e^{-xt}\mathrm{d}t$ naively **needs** $\mathrm{poly}(1/(\varepsilon\delta))$ terms.
- Substituting $t = e^y$ in the above integral obtains the identity $x^{-1} = \int_{-\infty}^{\infty} e^{-xe^y + y}\mathrm{d}y$.
- Discretizing this integral, we bound the error using the Euler-Maclaurin formula, Riemann zeta fn.; global error analysis!

A Heuristic for solving PSD Systems?

- **Goal**: compute $A^{-1}u$ for A psd.

A Heuristic for solving PSD Systems?

- **Goal**: compute $A^{-1}u$ for A psd.
- Write $A = A_1 + \cdots + A_k$.

A Heuristic for solving PSD Systems?

- **Goal**: compute $A^{-1}u$ for A psd.
- Write $A = A_1 + \cdots + A_k$.
- Hence, $e^{-tA} = e^{-t(A_1 + \cdots + A_k)}$.

A Heuristic for solving PSD Systems?

- **Goal**: compute $A^{-1}u$ for A psd.
- Write $A = A_1 + \cdots + A_k$.
- Hence, $e^{-tA} = e^{-t(A_1 + \cdots + A_k)}$.
- Suppose *magically*:
  1. k is *small*.

A Heuristic for solving PSD Systems?

- **Goal**: compute $A^{-1}u$ for A psd.
- Write $A = A_1 + \cdots + A_k$.
- Hence, $e^{-tA} = e^{-t(A_1+\cdots+A_k)}$.
- Suppose *magically*:
    1. k is *small*.
    2. Computing $e^{-tA_i}v$ easier for all i.

A Heuristic for solving PSD Systems?

- **Goal**: compute $A^{-1}u$ for A psd.
- Write $A = A_1 + \cdots + A_k$.
- Hence, $e^{-tA} = e^{-t(A_1+\cdots+A_k)}$.
- Suppose *magically*:
  1. k is *small*.
  2. Computing $e^{-tA_i}v$ easier for all i.
  3. $e^{-t(A_1+\cdots+A_k)} \approx e^{-tA_1} \cdots e^{-tA_k}$.

A Heuristic for solving PSD Systems?

- **Goal**: compute $A^{-1}u$ for A psd.
- Write $A = A_1 + \cdots + A_k$.
- Hence, $e^{-tA} = e^{-t(A_1 + \cdots + A_k)}$.
- Suppose *magically*:
  1. k is *small*.
  2. Computing $e^{-tA_i}v$ easier for all i.
  3. $e^{-t(A_1 + \cdots + A_k)} \approx e^{-tA_1} \cdots e^{-tA_k}$.
- Then $A^{-1}u \approx \sum_j w_j \prod_i e^{-t_j A_i} u$ (from previous slide).

A Heuristic for solving PSD Systems?

- **Goal**: compute $A^{-1}u$ for A psd.
- Write $A = A_1 + \cdots + A_k$.
- Hence, $e^{-tA} = e^{-t(A_1+\cdots+A_k)}$.
- Suppose *magically*:
  1. k is *small*.
  2. Computing $e^{-tA_i}v$ easier for all i.
  3. $e^{-t(A_1+\cdots+A_k)} \approx e^{-tA_1} \cdots e^{-tA_k}$.
- Then $A^{-1}u \approx \sum_j w_j \prod_i e^{-t_j A_i} u$ (from previous slide).

Theorem

For large enough p, $e^{(B_1+\cdots+B_k)} \approx \left(e^{\frac{B_1}{p}} \cdots e^{\frac{B_k}{p}}\right)^p$.

Conclusion

- Uniform approx. the right notion for algorithmic applications.
- Taylor series often not the best.

Conclusion

- Uniform approx. the right notion for algorithmic applications.
- Taylor series often not the best.
- Often reduce computations of $f(A)v$ to a small number of sparse matrix-vector computations.
  - Mere existence of good approximation suffices (see V. 2013).

Conclusion

- Uniform approx. the right notion for algorithmic applications.
- Taylor series often not the best.
- Often reduce computations of $f(A)v$ to a small number of sparse matrix-vector computations.
  - Mere existence of good approximation suffices (see V. 2013).
- Rational approx. can benefit from the ability to solve $Lx = b$.
- Much left to be explained in the fascinating world of rational approximations.

Conclusion

- Uniform approx. the right notion for algorithmic applications.
- Taylor series often not the best.
- Often reduce computations of $f(A)v$ to a small number of sparse matrix-vector computations.
  - Mere existence of good approximation suffices (see V. 2013).
- Rational approx. can benefit from the ability to solve $Lx = b$.
- Much left to be explained in the fascinating world of rational approximations.
- **Challenge problem:** Can we compute a δ-approximation to $W^s v$ in time $\tilde{O}(m \log s \cdot \log 1/\delta)$? ($W$ is the random walk matrix of an undirected graph.)

Conclusion

- Uniform approx. the right notion for algorithmic applications.
- Taylor series often not the best.
- Often reduce computations of $f(A)v$ to a small number of sparse matrix-vector computations.
  - Mere existence of good approximation suffices (see V. 2013).
- Rational approx. can benefit from the ability to solve $Lx = b$.
- Much left to be explained in the fascinating world of rational approximations.
- **Challenge problem:** Can we compute a δ-approximation to $W^s v$ in time $\tilde{O}(m \log s \cdot \log 1/\delta)$? ($W$ is the random walk matrix of an undirected graph.)
- Beyond $Lx = b$?

Conclusion

- Uniform approx. the right notion for algorithmic applications.
- Taylor series often not the best.
- Often reduce computations of $f(A)v$ to a small number of sparse matrix-vector computations.
  - Mere existence of good approximation suffices (see V. 2013).
- Rational approx. can benefit from the ability to solve $Lx = b$.
- Much left to be explained in the fascinating world of rational approximations.
- **Challenge problem:** Can we compute a δ-approximation to $W^s v$ in time $\tilde{O}(m \log s \cdot \log 1/\delta)$? ($W$ is the random walk matrix of an undirected graph.)
- Beyond $Lx = b$?

Thanks for your attention!