The Life Cycle of an Eigenvalue Problem

From Data to Numerics

Mark Embree

Department of Mathematics
Virginia Tech

SIAM Annual Meeting

Boston · 2016

An eigenvalue problem

$$\mathbf{A}\mathbf{x} = \lambda\mathbf{x}$$

algebraic eigenvalue problem

An eigenvalue problem

$$\mathbf{A}\mathbf{x} = \lambda \mathbf{B}\mathbf{x}$$

generalized eigenvalue problem

An eigenvalue problem

$$\begin{bmatrix} \mathbf{0} & \mathbf{I} \\ -\mathbf{K} & -\mathbf{D} \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ \mathbf{v} \end{bmatrix} = \lambda \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{M} \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ \mathbf{v} \end{bmatrix}$$

generalized eigenvalue problem
with structure from a damped mechanical system

An eigenvalue problem

$$\begin{bmatrix} \mathbf{0} & \mathbf{I} \\ -\mathbf{K} & -\mathbf{D} \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ \mathbf{v} \end{bmatrix} = \lambda \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{M} \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ \mathbf{v} \end{bmatrix}$$

$$\lambda^2 \mathbf{M}\mathbf{u} + \lambda \mathbf{D}\mathbf{u} + \mathbf{K}\mathbf{u} = \mathbf{0}$$

quadratic eigenvalue problem

An eigenvalue problem

$$\begin{bmatrix} \mathbf{0} & \mathbf{I} \\ -\mathbf{K} & -\mathbf{D} \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ \mathbf{v} \end{bmatrix} = \lambda \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{M} \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ \mathbf{v} \end{bmatrix}$$

$$\lambda^2 \mathbf{M}\mathbf{u} + \lambda \mathbf{D}\mathbf{u} + \mathbf{K}\mathbf{u} = \mathbf{0}$$

quadratic eigenvalue problem

$$\mathbf{M}\mathbf{u}''(t) + \mathbf{D}\mathbf{u}'(t) + \mathbf{K}\mathbf{u}(t) = \mathbf{0}$$

An eigenvalue problem

$$\lambda^2 m(x)u(x) + \lambda d(x)u(x) - u''(x) = 0$$

quadratic eigenvalue problem

An eigenvalue problem

$$\lambda^2 m(x)u(x) + \lambda d(x)u(x) - u''(x) = 0$$

quadratic eigenvalue problem

$$m(x)u_{tt}(x,t) + d(x)u_t(x,t) - u_{xx}(x,t) = 0$$

model of a vibrating string with viscous damping

An eigenvalue problem

$$\lambda^2 m(x)u(x) + \lambda d(x)u(x) - u''(x) = 0$$

quadratic eigenvalue problem

$$m(x)u_{tt}(x,t) + d(x)u_t(x,t) - u_{xx}(x,t) = 0$$

model of a vibrating string with viscous damping

$$(\rho A)(s)\mathbf{r}_{tt}(s,t) = \left(\widehat{N}(\|\mathbf{r}_s(s,t)\|, s)\frac{\mathbf{r}_s(s,t)}{\|\mathbf{r}_s(s,t)\|}\right)_s + \mathbf{f}(s,t)$$

general nonlinear model of a perfectly flexible string

An eigenvalue problem

$$\mathbf{A}\mathbf{x} = (\lambda \mathbf{I} + \mathrm{e}^{\lambda s}\mathbf{B})\mathbf{x}$$

nonlinear eigenvalue problem

(*eigenvalue nonlinearity*)

$$\mathbf{u}'(t) = \mathbf{A}\mathbf{u}(t-s) - \mathbf{B}\mathbf{u}(t)$$

linear delay differential equation

An eigenvalue problem

$$Du(x,y) + |u(x,y)|^2 u(x,y) = \lambda u(x,y)$$

nonlinear eigenvalue problem

(*eigenvector nonlinearity*)

Gross–Pitaevskii eigenvalue problem for Bose–Einstein condensates
[Jarlebring, Kvall, Michiels, 2014]

See also Kohn–Sham eigenvalue problem in Density Functional Theory

Eigenvalues, eigenvectors, and dynamics

We most often care about eigenvalues because they give insight into *dynamics*.

Consider the diagonalizable matrix

$$\mathbf{A} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^{-1} = \sum_{j=1}^{n} \lambda_j \mathbf{v}_j \widehat{\mathbf{v}}_j^* = \sum_{j=1}^{n} \lambda_j \mathbf{P}_j.$$

Eigenvalues, eigenvectors, and dynamics

We most often care about eigenvalues because they give insight into *dynamics*.

Consider the diagonalizable matrix

$$\mathbf{A} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^{-1} = \sum_{j=1}^{n} \lambda_j \mathbf{v}_j \widehat{\mathbf{v}}_j^* = \sum_{j=1}^{n} \lambda_j \mathbf{P}_j.$$

Spectral Mapping Theorem:
The eigenvalues of $f(\mathbf{A})$ are $f(\lambda_j)$ (for f analytic on the eigenvalues):

$$f(\mathbf{A}) = \mathbf{V} f(\mathbf{\Lambda}) \mathbf{V}^{-1} = \sum_{j=1}^{n} f(\lambda_j) \mathbf{v}_j \widehat{\mathbf{v}}_j^* = \sum_{j=1}^{n} f(\lambda_j) \mathbf{P}_j.$$

Eigenvalues, eigenvectors, and dynamics

We most often care about eigenvalues because they give insight into *dynamics*.

Consider the diagonalizable matrix

$$\mathbf{A} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^{-1} = \sum_{j=1}^{n} \lambda_j \mathbf{v}_j \widehat{\mathbf{v}}_j^* = \sum_{j=1}^{n} \lambda_j \mathbf{P}_j.$$

The *discrete-time* dynamical system

$$\mathbf{x}_{k+1} = \mathbf{A}\mathbf{x}_k$$

is solved by

$$\mathbf{x}_k = \mathbf{A}^k \mathbf{x}_0 = \sum_{j=1}^{n} \left(\widehat{\mathbf{v}}_j^* \mathbf{x}_0\right) \lambda_j^k \, \mathbf{v}_j.$$

Eigenvalues, eigenvectors, and dynamics

We most often care about eigenvalues because they give insight into *dynamics*.

Consider the diagonalizable matrix

$$\mathbf{A} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^{-1} = \sum_{j=1}^{n} \lambda_j \mathbf{v}_j \widehat{\mathbf{v}}_j^* = \sum_{j=1}^{n} \lambda_j \mathbf{P}_j.$$

The *discrete-time* dynamical system

$$\mathbf{x}_{k+1} = \mathbf{A}\mathbf{x}_k$$

is solved by

$$\mathbf{x}_k = \mathbf{A}^k \mathbf{x}_0 = \sum_{j=1}^{n} \left(\widehat{\mathbf{v}}_j^* \mathbf{x}_0\right) \lambda_j^k \, \mathbf{v}_j.$$

eigenvectors give coordinate directions

influence of initial conditions

eigenvalues dictate dynamics

Eigenvalues, eigenvectors, and dynamics

We most often care about eigenvalues because they give insight into *dynamics*.

Consider the diagonalizable matrix

$$\mathbf{A} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^{-1} = \sum_{j=1}^{n} \lambda_j \mathbf{v}_j \widehat{\mathbf{v}}_j^* = \sum_{j=1}^{n} \lambda_j \mathbf{P}_j.$$

The *continuous-time* dynamical system

$$\mathbf{x}'(t) = \mathbf{A}\mathbf{x}(t)$$

is solved by

$$\mathbf{x}(t) = \mathrm{e}^{t\mathbf{A}}\mathbf{x}_0 = \sum_{j=1}^{n} (\widehat{\mathbf{v}}_j^* \mathbf{x}_0)\, \mathrm{e}^{t\lambda_j}\, \mathbf{v}_j.$$

Eigenvalues, eigenvectors, and dynamics

We most often care about eigenvalues because they give insight into *dynamics*.

Consider the diagonalizable matrix

$$\mathbf{A} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^{-1} = \sum_{j=1}^{n} \lambda_j \mathbf{v}_j \widehat{\mathbf{v}}_j^* = \sum_{j=1}^{n} \lambda_j \mathbf{P}_j.$$

The *continuous-time* dynamical system

$$\mathbf{x}'(t) = \mathbf{A}\mathbf{x}(t)$$

is solved by

$$\mathbf{x}(t) = \mathrm{e}^{t\mathbf{A}}\mathbf{x}_0 = \sum_{j=1}^{n} (\widehat{\mathbf{v}}_j^* \mathbf{x}_0)\, \mathrm{e}^{t\lambda_j}\, \mathbf{v}_j.$$

eigenvectors give coordinate directions

influence of initial conditions

eigenvalues dictate dynamics

The Life Cycle of an Eigenvalue Problem

Premise of this talk:

- "Solving an eigenvalue problem" is a broad endeavor that starts with data and ends with numerics.
- Typically we focus on one only single aspect of this endeavor.
- We gain – at least perspective, sometimes more – by thinking across multiple steps of this "life cycle".

The Life Cycle of an Eigenvalue Problem

Premise of this talk:

- "Solving an eigenvalue problem" is a broad endeavor that starts with data and ends with numerics.
- Typically we focus on one only single aspect of this endeavor.
- We gain – at least perspective, sometimes more – by thinking across multiple steps of this "life cycle".

Here we can only illustrate these steps with a few scattered vignettes.

Other related perspectives: [Collatz, 1963], [Weinberger, 1974], [Chatelin, 1983], [Babuška & Osborn, 1991], [Plum, 1997], [Liesen & Strakoš, 2013],. . . .

The Life Cycle of an Eigenvalue Problem

Five steps in the life cycle:

1. Physical problem / data $\rightarrow$ Nonlinear eigenvalue problem
2. Nonlinear eigenvalue problem $\rightarrow$ Linear eigenvalue problem
3. Linear eigenvalue problem $\rightarrow$ Large discretization matrix
4. Large discretization matrix $\rightarrow$ Small projected matrix
5. Small projected matrix $\rightarrow$ Numerical eigenvalues

Introduction

Five fundamental steps in the life-cycle of an eigenvalue problem:

1. DATA / MODEL
2. LINEARIZE
3. DISCRETIZE
4. PROJECT
5. COMPUTE

Introduction

Five fundamental steps in the life-cycle of an eigenvalue problem:

1. DATA / MODEL — *FIRST HALF OF THE TALK*
2. LINEARIZE
3. DISCRETIZE
4. PROJECT
5. COMPUTE

(2–5) — *SECOND HALF OF THE TALK*

From Physical Problem and Data to Nonlinear Eigenvalue Problem

Eigenvalues from Data: Case Study of a Vibrating String

As a prototype of this move from physical system to mathematical model, we study one of the earliest eigenvalue problems: vibrating strings.

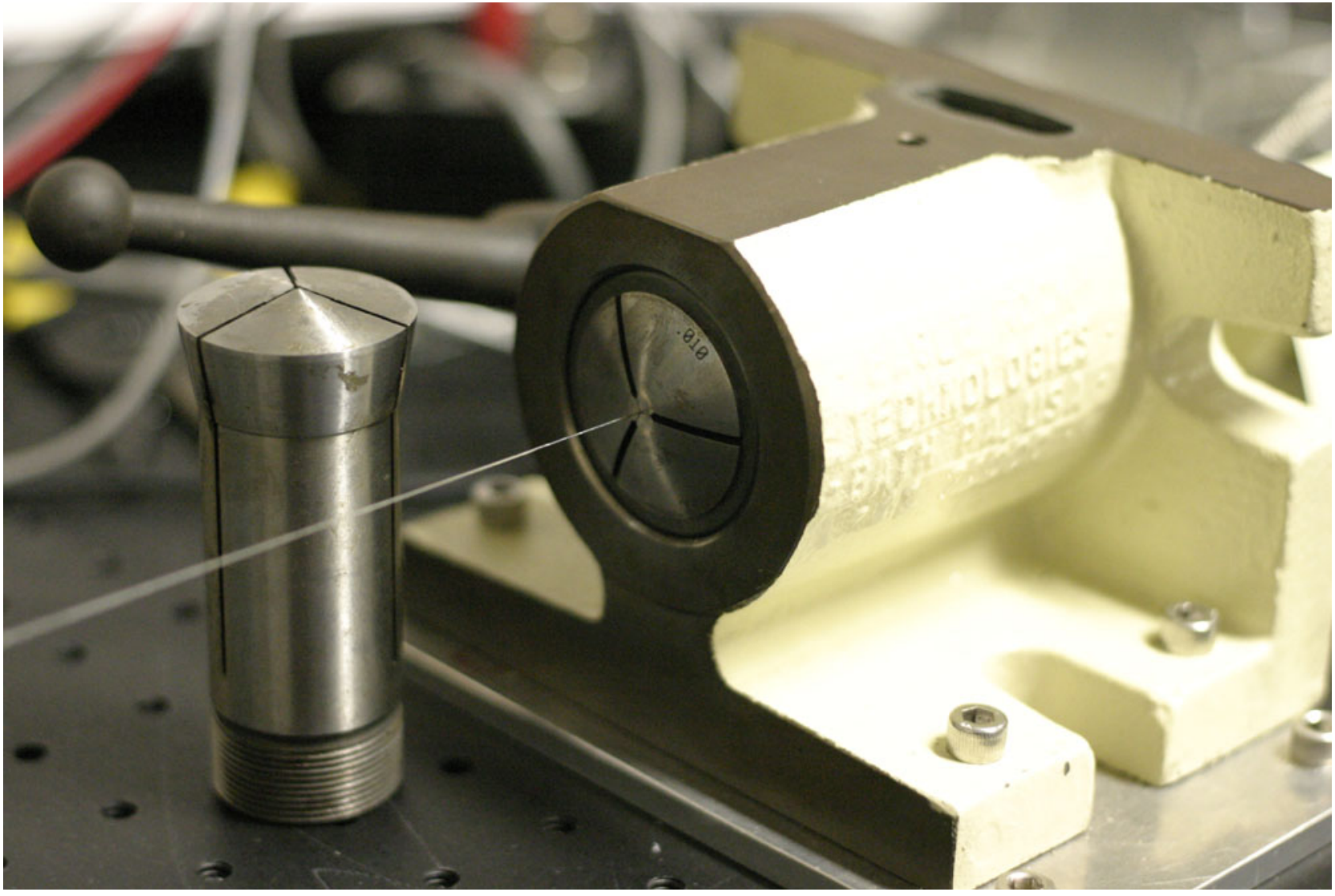
Though often seen as a trivial example, we want to emphasize the modeling challenges that arise and how they affect the resulting eigenvalue problem.

We will compare theoretical eigenvalues to those measured in our laboratory.

Work with Steve Cox and Jeffrey Hokanson.

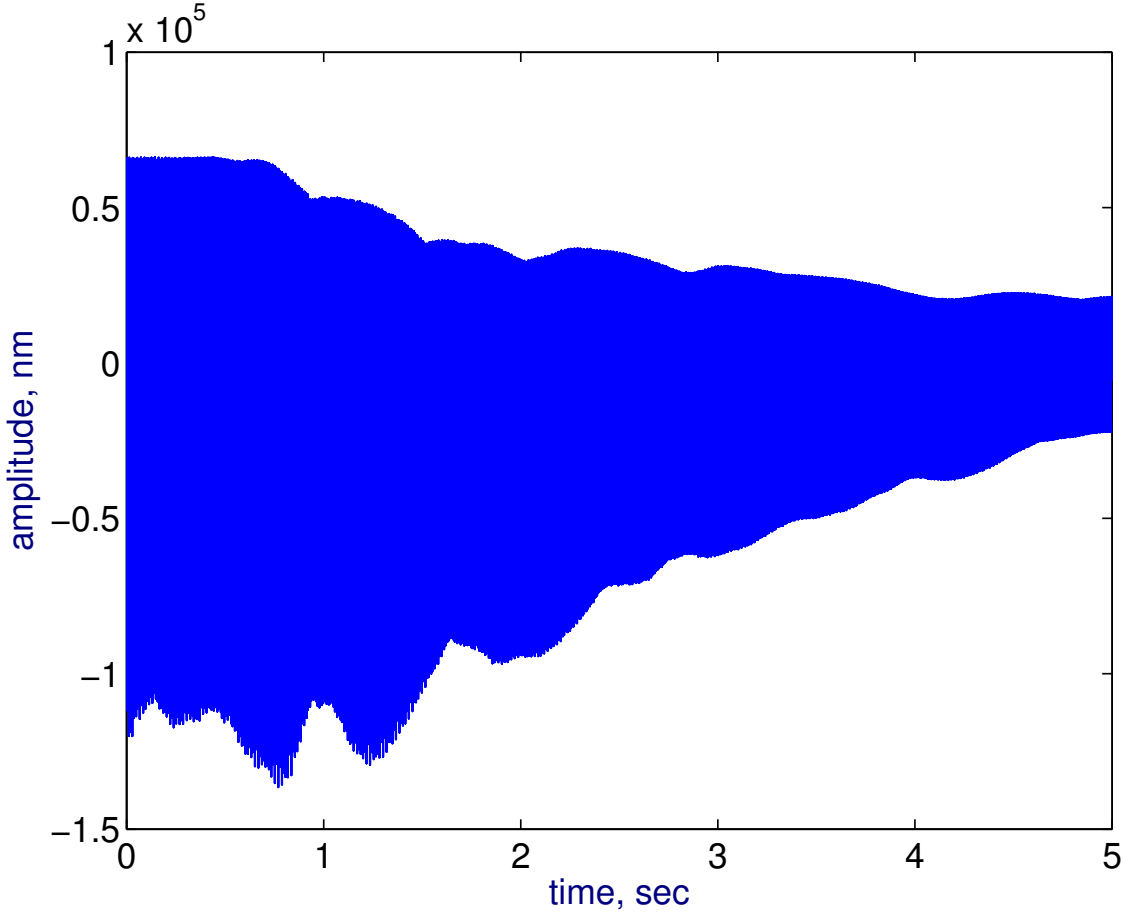
Monochord built by Sean Hardesty, Jeffrey Hokanson, and Jeffrey Bridge.

Imposing Dirichlet Boundary Conditions

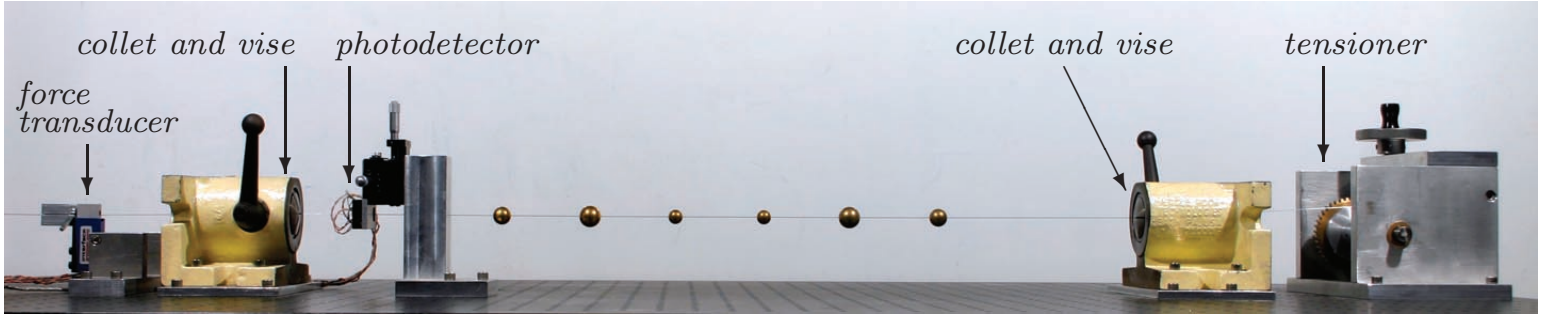


How do real strings behave?

Photodetector measurements of string displacement at one point $\widehat{x} \in [0, L]$ (undersampled).

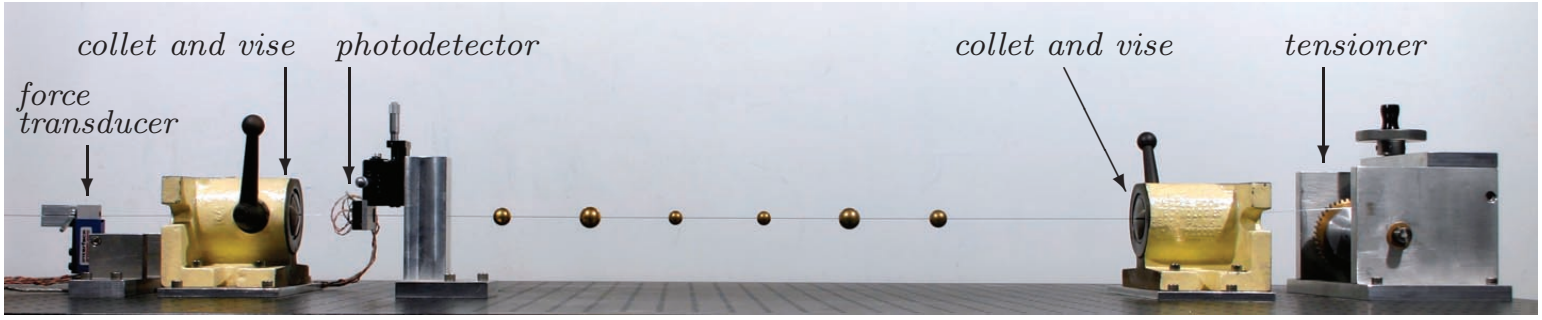


Eigenvalues from Data: Case Study of a Vibrating String



The original mathematical model was devised by Euler and Lagrange (with key contributions from D. Bernoulli and d'Alembert) in the mid-1700s [Truesdell, 1960; Antman, 2005].

Eigenvalues from Data: Case Study of a Vibrating String



The original mathematical model was devised by Euler and Lagrange (with key contributions from D. Bernoulli and d'Alembert) in the mid-1700s [Truesdell, 1960; Antman, 2005].

> *The correct derivation is simple because Euler made it so. Modern authors should be faulted not merely for doing poorly what Euler did well, but also for failing to copy from the master.*
>
> — Stuart S. Antman
> *Nonlinear Problems of Elasticity*, p. 12

Mathematical Model of a Vibrating String

Summarizing Antman:
A string is an *elastic*, *perfectly flexible* one-dimensional body drawn taut at length L with fixed ends, displaced by $\mathbf{r}(s,t) \in \mathbf{R}^3$ at $s \in [0,1]$ and time $t \geq 0$.

The boundary conditions give

$$\mathbf{r}(0,t) = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, \quad \mathbf{r}(1,t) = \begin{bmatrix} 0 \\ 0 \\ L \end{bmatrix}.$$

Mathematical Model of a Vibrating String

Summarizing Antman:
A string is an *elastic*, *perfectly flexible* one-dimensional body drawn taut at length L with fixed ends, displaced by $\mathbf{r}(s,t) \in \mathbf{R}^3$ at $s \in [0,1]$ and time $t \geq 0$.

The boundary conditions give

$$\mathbf{r}(0,t) = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, \quad \mathbf{r}(1,t) = \begin{bmatrix} 0 \\ 0 \\ L \end{bmatrix}.$$

- $\rho A(s) =$ the string's mass-density-per-length
- $\widehat{N}(\|\mathbf{r}_s(s,t)\|, s) =$ tension at (s,t)
- $(\mathbf{s}, \mathbf{t}) =$ body force per unit length (e.g., damping)

Then the displacement $\mathbf{r}(s,t)$ obeys the nonlinear partial differential equation

$$(\rho A)(s)\mathbf{r}_{tt}(s,t) = \left(\widehat{N}(\|\mathbf{r}_s(s,t)\|, s) \frac{\mathbf{r}_s(s,t)}{\|\mathbf{r}_s(s,t)\|} \right)_s + \mathbf{f}(s,t).$$

Mathematical model of a vibrating string

$$(\rho A)(s)\mathbf{r}_{tt}(s,t) = \left(\widehat{N}(\|\mathbf{r}_s(s,t)\|, s)\frac{\mathbf{r}_s(s,t)}{\|\mathbf{r}_s(s,t)\|}\right)_s + \mathbf{f}(s,t)$$

Linearize this equation about the rest state

$$\mathbf{r}(s,t) = \begin{bmatrix} 0 \\ 0 \\ sL \end{bmatrix}, \qquad \mathbf{r}_t(s,t) = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}.$$

Under appropriate assumptions, we get three scalar PDEs of the familiar form

$$u_{tt}(x,t) = c^2 u_{xx}(x,t), \qquad u(0,t) = u(L,t) = 0$$

(two describe transverse vibrations, the other describes longitudinal vibrations), which reduce via separation of variables to the eigenvalue problem

$$c^2 u''(x) = \lambda^2 u(x), \qquad u(0) = u(L) = 0.$$

Eigenvalue problem for a vibrating string

Pose the eigenvalue problem

$$u''(x) = \lambda^2 u(x), \qquad u(0) = u(L) = 0.$$

in the form $AU = \lambda U$:

$$\begin{bmatrix} 0 & I \\ \mathrm{d}^2/\mathrm{d}x^2 & 0 \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = \lambda \begin{bmatrix} u \\ v \end{bmatrix}$$

with $\mathrm{Dom}(A) = (H_0^1(0,1) \cap H^2(0,1)) \times H_0^1(0,1)$.

Eigenvalue problem for a vibrating string

Pose the eigenvalue problem

$$u''(x) = \lambda^2 u(x), \qquad u(0) = u(L) = 0.$$

in the form $AU = \lambda U$:

$$\begin{bmatrix} 0 & I \\ \mathrm{d}^2/\mathrm{d}x^2 & 0 \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = \lambda \begin{bmatrix} u \\ v \end{bmatrix}$$

with $\mathrm{Dom}(A) = (H_0^1(0,1) \cap H^2(0,1)) \times H_0^1(0,1)$.

For this model of an undamped string:

- The eigenvalues are purely imaginary: $\lambda_{\pm k} = \pm k\pi \mathrm{i}$ for $k = 1, 2, \ldots$;
- The eigenvectors are $V_{\pm k} = \begin{bmatrix} \sin(k\pi x) \\ \pm k\pi \mathrm{i} \sin(k\pi x) \end{bmatrix}$.

Eigenvalue problem for a vibrating string

The system evolves in time according to $U_t = AU$:

$$\frac{\partial}{\partial t}\begin{bmatrix} u(x,t) \\ v(x,t) \end{bmatrix} = \begin{bmatrix} 0 & I \\ \mathrm{d}^2/\mathrm{d}x^2 & 0 \end{bmatrix}\begin{bmatrix} u(x,t) \\ v(x,t) \end{bmatrix}.$$

- eigenvalues of A are purely imaginary: $\lambda_{\pm k} = \pm k\pi \mathrm{i}$ for $k = 1, 2, \ldots$;
- eigenvectors of A are $V_{\pm k} = \begin{bmatrix} \sin(k\pi x) \\ \pm k\pi \mathrm{i} \sin(k\pi x) \end{bmatrix}$.

Given initial conditions $u(x,0) = u_0(x)$, $v(x,0) = 0$, the solution is

$$U(x,t) = \sum_{\substack{k=-\infty \\ k\neq 0}}^{\infty} \langle U_0, V_k \rangle \, \mathrm{e}^{t\lambda_k} V_k(x).$$

Eigenvalue problem for a vibrating string

The system evolves in time according to $U_t = AU$:

$$\frac{\partial}{\partial t}\begin{bmatrix} u(x,t) \\ v(x,t) \end{bmatrix} = \begin{bmatrix} 0 & I \\ \mathrm{d}^2/\mathrm{d}x^2 & 0 \end{bmatrix}\begin{bmatrix} u(x,t) \\ v(x,t) \end{bmatrix}.$$

- eigenvalues of A are purely imaginary: $\lambda_{\pm k} = \pm k\pi \mathrm{i}$ for $k = 1, 2, \ldots$;
- eigenvectors of A are $V_{\pm k} = \begin{bmatrix} \sin(k\pi x) \\ \pm k\pi \mathrm{i} \sin(k\pi x) \end{bmatrix}$.

Given initial conditions $u(x,0) = u_0(x)$, $v(x,0) = 0$, the solution is

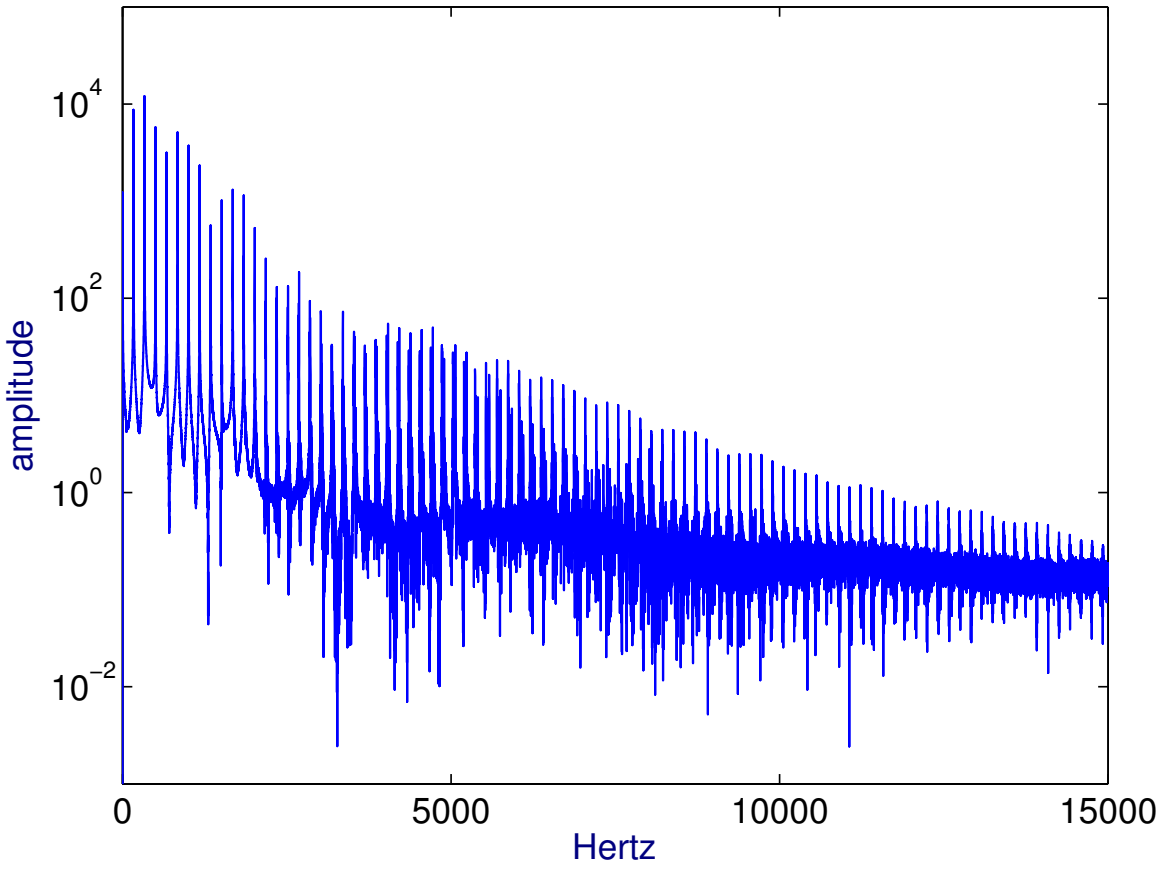
$$U(x,t) = \sum_{\substack{k=-\infty \\ k\neq 0}}^{\infty} \langle U_0, V_k \rangle \, \mathrm{e}^{t\lambda_k} V_k(x).$$

$\mathrm{e}^{t\lambda_k} = \cos(k\pi t) \pm \mathrm{i}\sin(k\pi t) \implies$ *oscillation*

rapidly decaying in $|k|$

How does a real string behave?

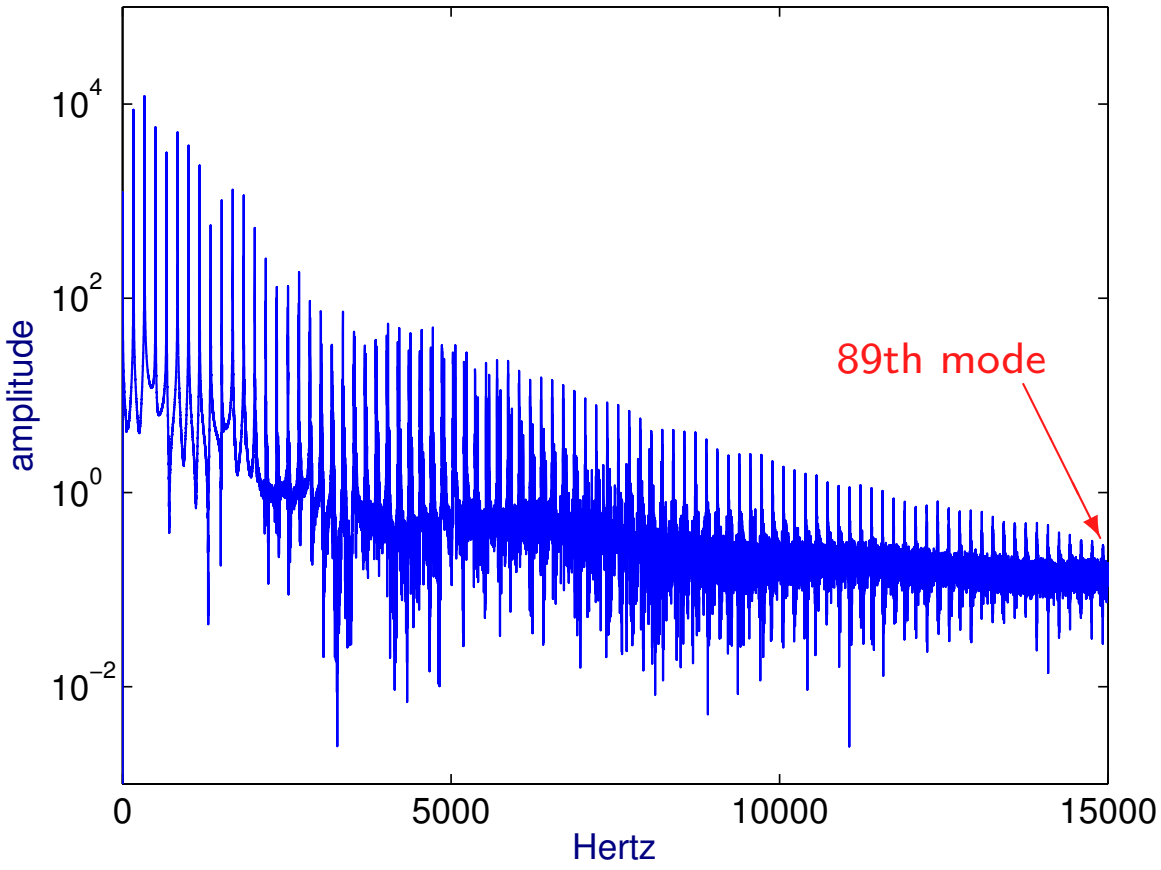
To expose the imaginary parts of the eigenvalues of a real string ("$\lambda_{\pm k} \approx \pm k\pi$i"), take the FFT of the displacement:



Each peak corresponds to a conjugate pair of eigenvalues.

How does a real string behave?

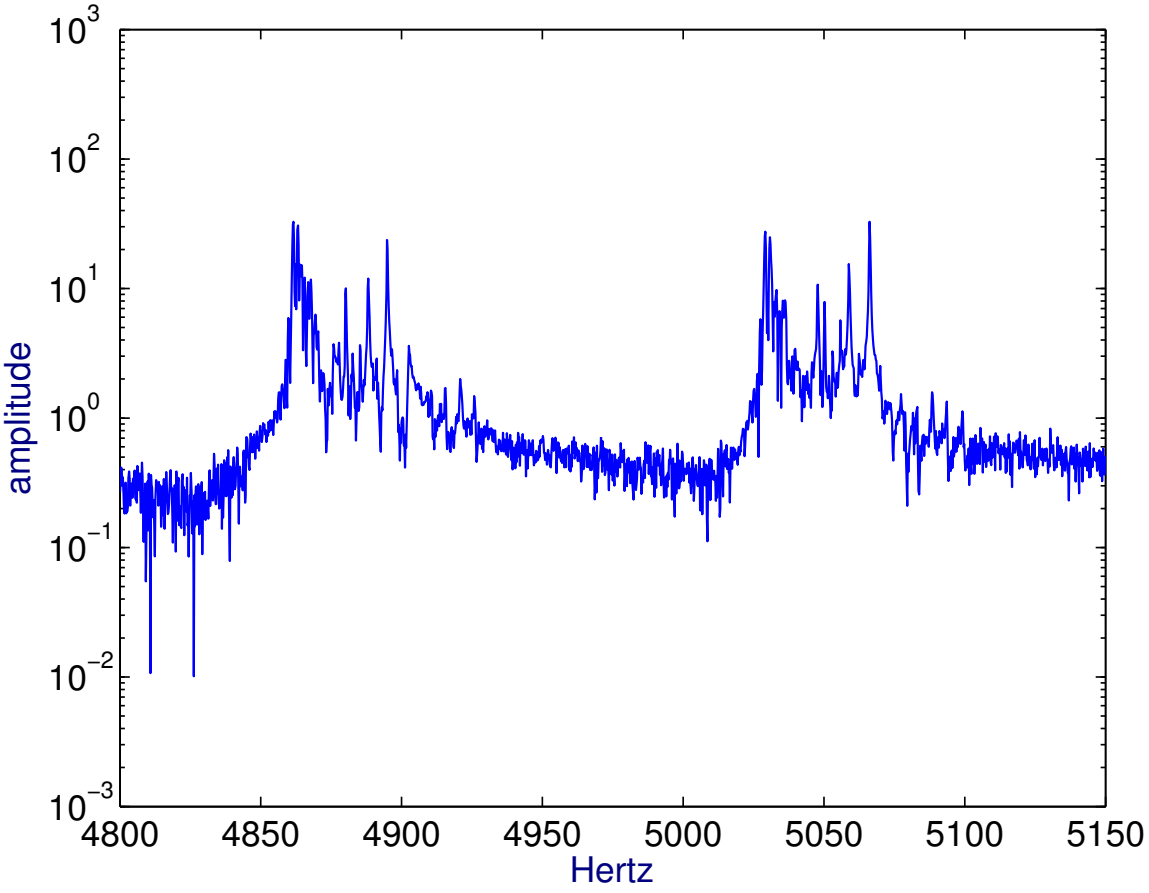
To expose the imaginary parts of the eigenvalues of a real string ("$\lambda_{\pm k} \approx \pm k\pi$i"), take the FFT of the displacement:



Each peak corresponds to a conjugate pair of eigenvalues.

How does a real string behave?

To determine the imaginary part of λ_k, zoom in on the FFT....



What look like strong "peaks" on the previous plot are considerably more intricate.

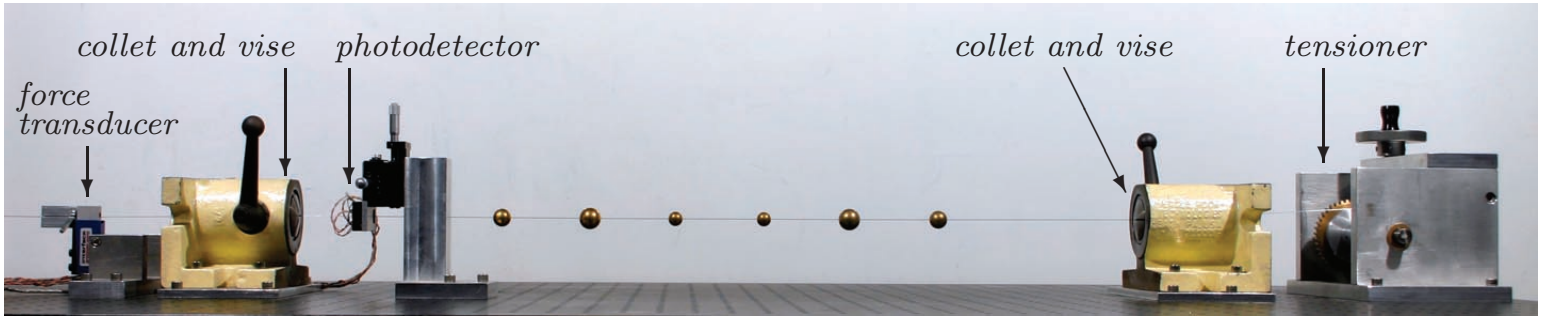
Inverse Eigenvalue Problems

Given the (measured) eigenvalues of an object,
can we determine the "shape" of that object?

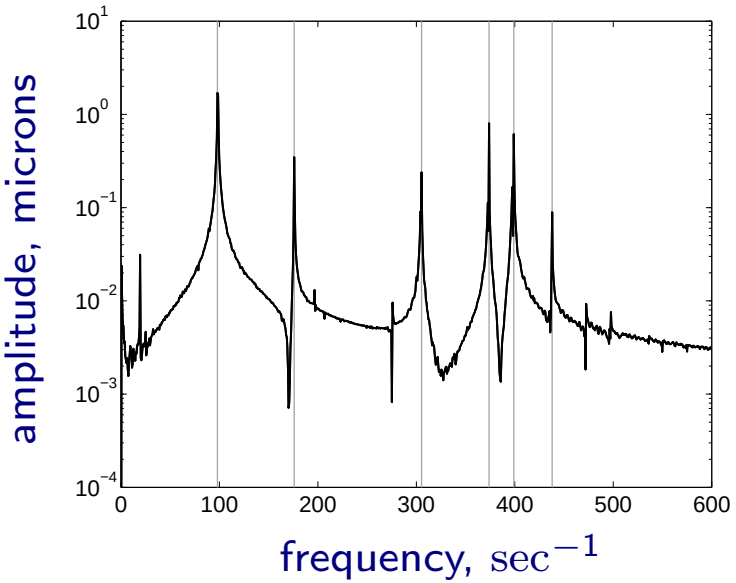
Inverse Eigenvalue Problems

Given the (measured) eigenvalues of an object,
can we determine the "shape" of that object?

- What constraints make the problem well posed ?
  In 1966, Mark Kac famously asked, "Can One Hear the Shape of a Drum?"
- Often *symmetry* is the key ingredient to determine a unique solution.
- In the 1950s, Mark Krein use the continued fractions work of Stieltjes [1894] to show how to *discover the location of n beads arranged symmetrically on a string from the n eigenvalues*.
- In [Cox, E., Hokanson 2012], we put this algorithm to the test.
  Much data available at: `www.caam.rice.edu/~beads`

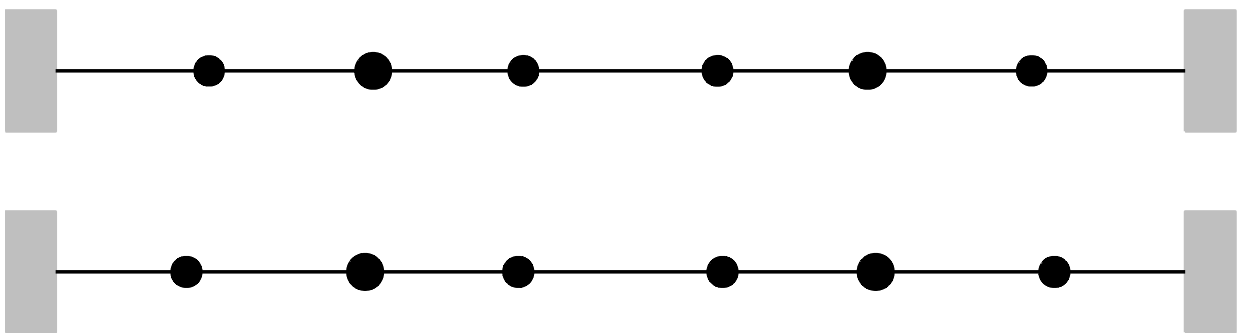


Inverse Eigenvalue Problems: "Hearing" Beads on a String



experimental eigenvalues

$$\lambda_1 = \ \ 98\mathrm{i}\ \mathrm{sec}^{-1}$$
$$\lambda_2 = 176\mathrm{i}\ \mathrm{sec}^{-1}$$
$$\lambda_3 = 305\mathrm{i}\ \mathrm{sec}^{-1}$$
$$\lambda_4 = 374\mathrm{i}\ \mathrm{sec}^{-1}$$
$$\lambda_5 = 399\mathrm{i}\ \mathrm{sec}^{-1}$$
$$\lambda_6 = 438\mathrm{i}\ \mathrm{sec}^{-1}$$



	M_1 (g)	M_2 (g)	M_3 (g)	L_0 (cm)	L_1 (cm)	L_2 (cm)	L_3 (cm)
recovered	16.6	30.4	17.1	15.3	16.3	14.9	19.3
measured	17.8	30.8	17.8	13.0	17.8	15.2	20.3

Complex eigenvalues for damped strings

$$U(x,t) = \sum_{\substack{k=-\infty \\ k\neq 0}}^{\infty} \langle U_0, V_k \rangle \, \mathrm{e}^{t\lambda_k} V_k(x).$$

- Thus far we have been computing purely imaginary λ_k by finding peaks.
- Purely imaginary eigenvalues correspond to vibrations that never die out,

$$\mathrm{e}^{t\lambda_k} = \mathrm{e}^{\mathrm{i}\pi k t} = \cos(\pi k t) + \mathrm{i}\sin(\pi k t).$$

Complex eigenvalues for damped strings

$$U(x,t) = \sum_{\substack{k=-\infty \\ k\neq 0}}^{\infty} \langle U_0, V_k \rangle \, \mathrm{e}^{t\lambda_k} V_k(x).$$

- Thus far we have been computing purely imaginary λ_k by finding peaks.
- Purely imaginary eigenvalues correspond to vibrations that never die out,

$$\mathrm{e}^{t\lambda_k} = \mathrm{e}^{\mathrm{i}\pi k t} = \cos(\pi k t) + \mathrm{i}\sin(\pi k t).$$

- To model decay, add *damping* to the model, giving $\mathrm{Re}(\lambda_k) \neq 0$.

viscous damping:	$u_{tt} = u_{xx} - 2a(x)u_t$
Kelvin–Voigt:	$u_{tt} = u_{xx} + (a(x)u_{xt})_x$
magnetic damping:	$u_{tt} = u_{xx} - a(x)\int_0^\pi a(s)u_t(s,t)\,\mathrm{d}s$
stiff strings:	$u_{tt} = c^2 u_{xx} - \kappa^2 u_{xxxx} - 2a(x)u_t + 2b(x)u_{xxt}$

A model of viscous damping

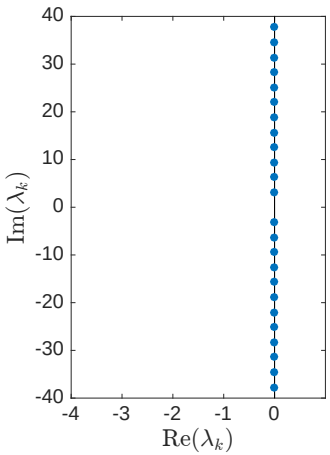
$$u_{tt}(x,t) = u_{xx}(x,t) - 2a(x)u_t(x,t)$$

$$\begin{bmatrix} 0 & I \\ \mathrm{d}^2/\mathrm{d}x^2 & -2a \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = \lambda \begin{bmatrix} u \\ v \end{bmatrix}$$

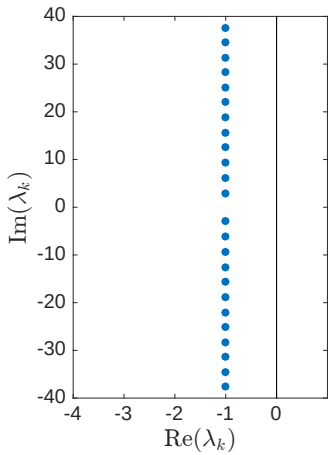
- Eigenvalues of A: $\lambda_{\pm k} = -a \pm \sqrt{a^2 - k^2\pi^2}$

For constant damping parameter a:

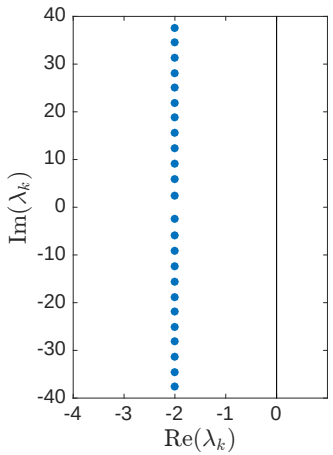
If $0 \leq a \leq \pi$, then $\mathrm{Re}(\lambda_k) = -a$ for all eigenvalues.



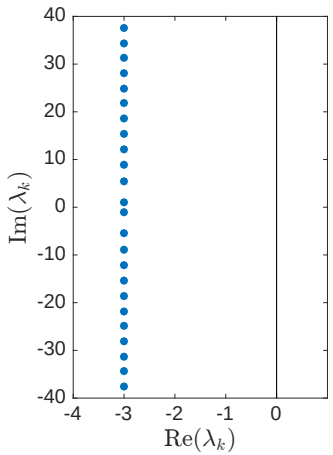
$a = 0$



$a = 1$



$a = 2$



$a = 3$

Finding complex eigenvalues data

How do we compute complex eigenvalues from vibration measurements?

- Take measurements at uniform times, $u_j \approx u(\widehat{x}, t_j)$ for $j = 1, \ldots, m$.
- Fit measurements to a sum of p exponentials:

$$u_j \approx \sum_{k=1}^{p} c_k \exp(\lambda_k t_j).$$

Find linear parameters $\{c_k\}$ and nonlinear parameters $\{\lambda_k\}$.

Finding complex eigenvalues data

How do we compute complex eigenvalues from vibration measurements?

- Take measurements at uniform times, $u_j \approx u(\widehat{x}, t_j)$ for $j = 1, \ldots, m$.
- Fit measurements to a sum of p exponentials:

$$u_j \approx \sum_{k=1}^{p} c_k \exp(\lambda_k t_j).$$

Find linear parameters $\{c_k\}$ and nonlinear parameters $\{\lambda_k\}$.

A multitude of methods exist for the exponential fitting problem.

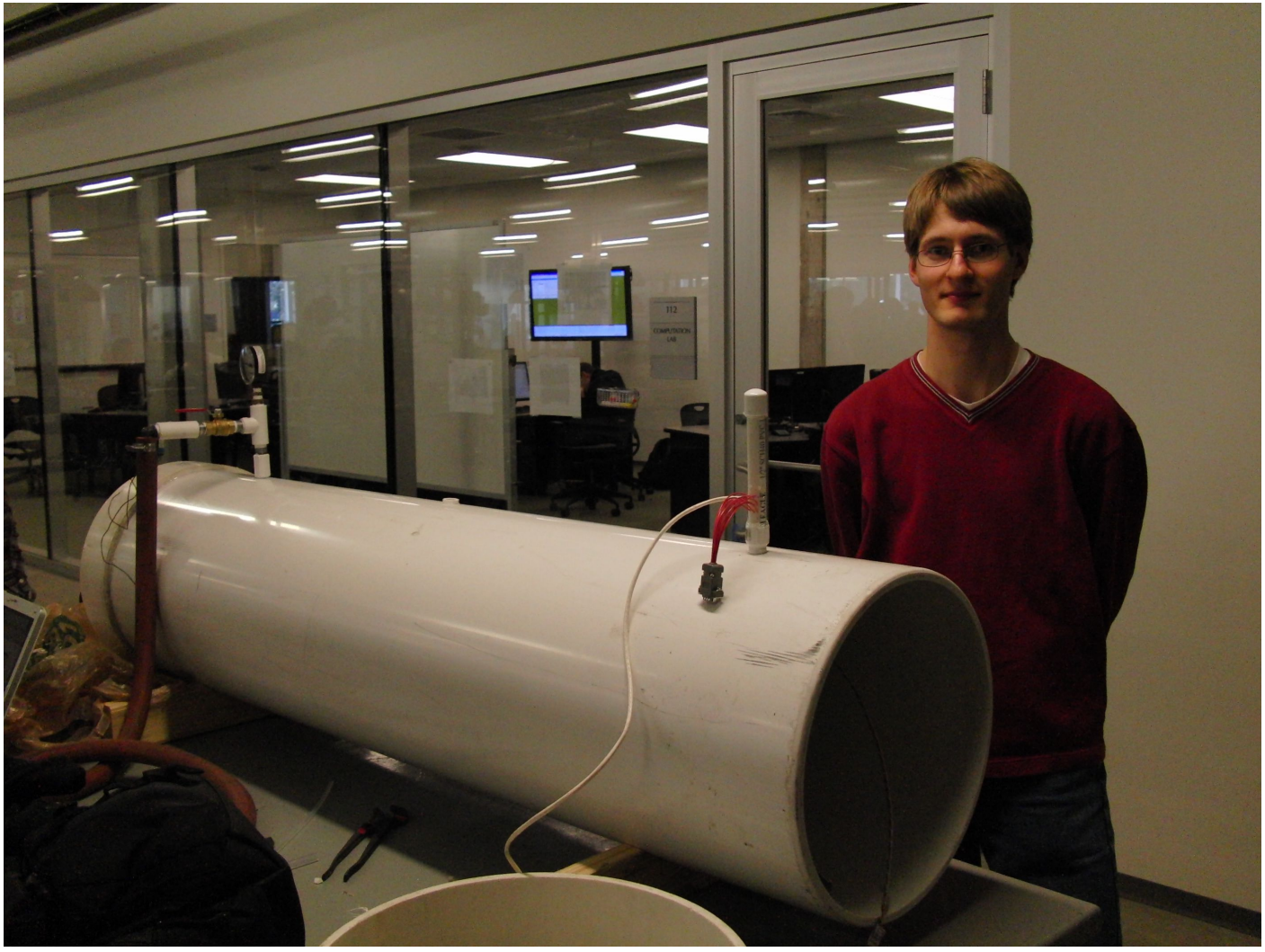
- *Improvements of Prony's method* (e.g., HSVD, HTLS)
  (Solve a Hankel linear system and companion matrix eigenvalue problem.)
- *Nonlinear least squares methods* (e.g., VARPRO)

$$\min_{\mathbf{c},\boldsymbol{\lambda}\in\mathbf{C}^p} \|\mathbf{u} - \mathbf{V}(\boldsymbol{\lambda})\mathbf{c}\|_2.$$

For efficiency, we compress via a specialized *matrix sketching* method [Hokanson 2013, 2015]:

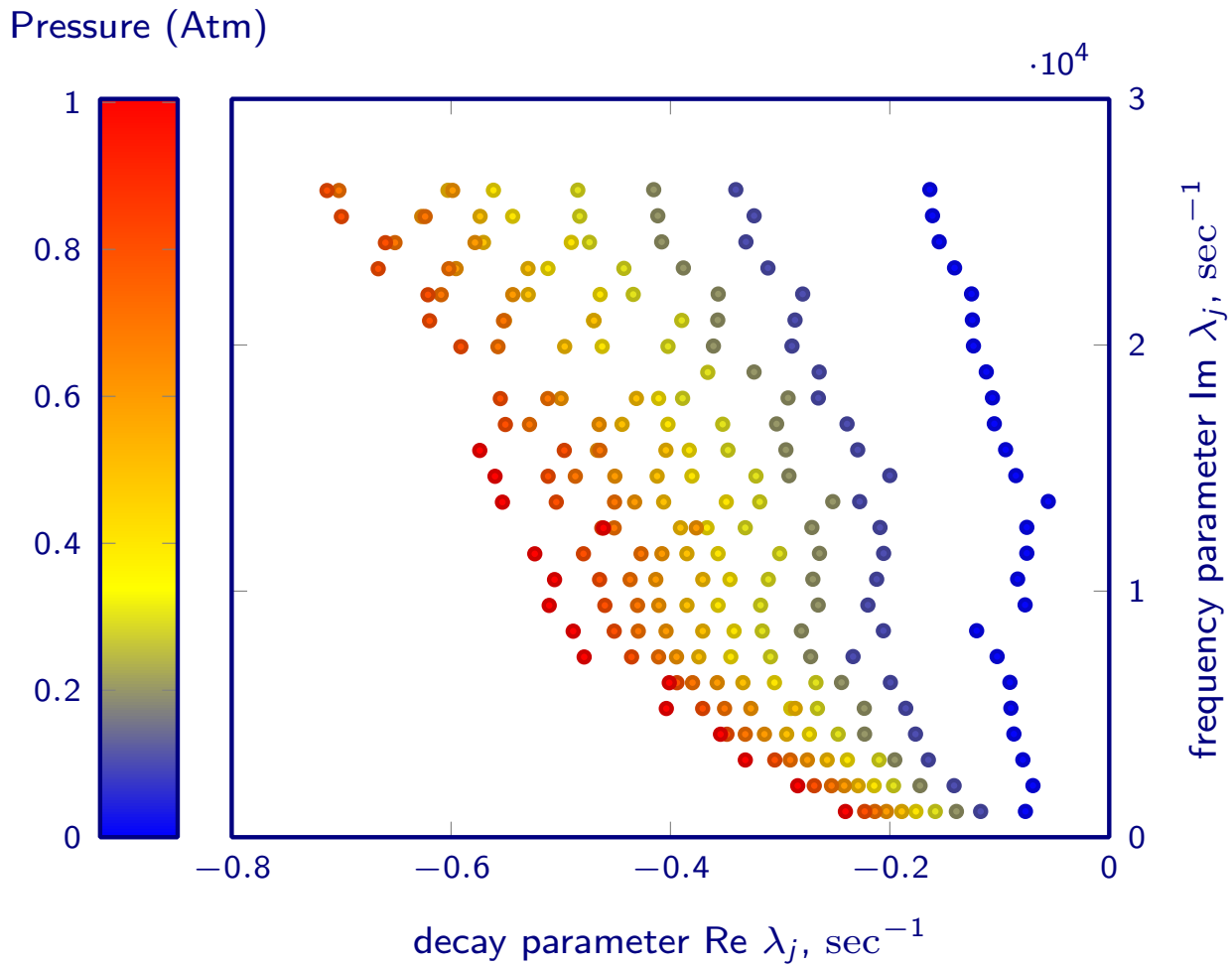
$$\min_{\mathbf{c},\boldsymbol{\lambda}\in\mathbf{C}^p} \|\mathbf{W}^*(\mathbf{u} - \mathbf{V}(\boldsymbol{\lambda})\mathbf{c})\|_2.$$

Is the viscous damping model accurate ?



Jeffrey Bridge and his vacuum chamber.

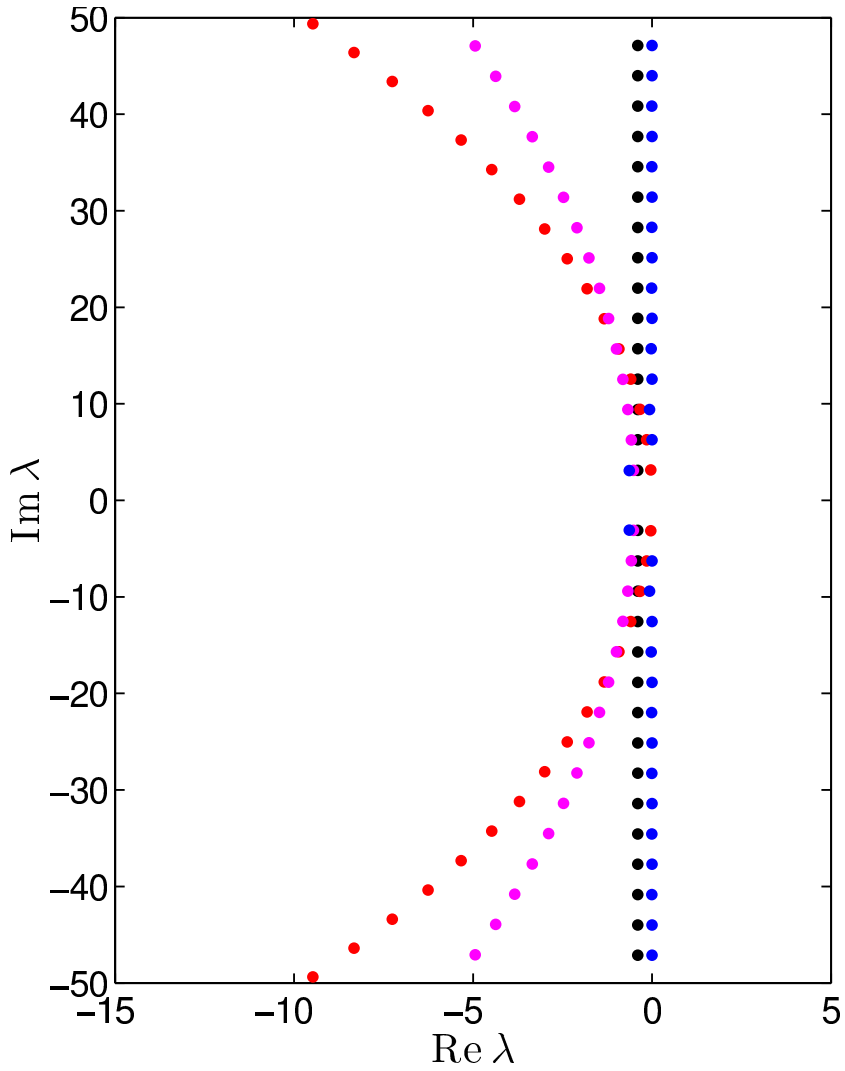
Is the viscous damping model accurate ?



Eigenvalues derived by Hokanson from a piano wire in the vacuum chamber.

Viscous damping model predicts that, for a given pressure, all eigenvalues should have the same real part.

Canonical Models of Damping in Strings



For certain constant damping coefficients:

- Kelvin–Voigt damping
- stiff string
- viscous damping
- magnetic damping

Damping models are difficult to differentiate from low-frequency eigenvalues, but these are the only ones that can be reliably estimated.

A Nonnormal Inverse Eigenvalue Problem for String Design

Can you design a guitar string that sounds the way you desire?

A Nonnormal Inverse Eigenvalue Problem for String Design

Can you design a guitar string that sounds the way you desire?

Can you find a viscous damping coefficient $a(x)$ to give desired eigenvalues?

A Nonnormal Inverse Eigenvalue Problem for String Design

Can you design a guitar string that sounds the way you desire?

Can you find a viscous damping coefficient $a(x)$ to give desired eigenvalues?

A string design problem:

- Specify eigenvalues $\{\lambda_k\} \subset \mathbf{C}$ that sound pleasant.
  If $\mathrm{Re}(\lambda_k) \approx 0$, *the tone* $\mathrm{Im}(\lambda_k)$ *will persist for a while.*
  If $\mathrm{Re}(\lambda_k) \ll 0$, *the tone* $\mathrm{Im}(\lambda_k)$ *will die out quickly.*

- Find a damping coefficient $a(x)$ such that

$$\text{eigenvalues of } \begin{bmatrix} 0 & I \\ \mathrm{d}^2/\mathrm{d}x^2 & -2a(x) \end{bmatrix} = \{\lambda_k\}$$

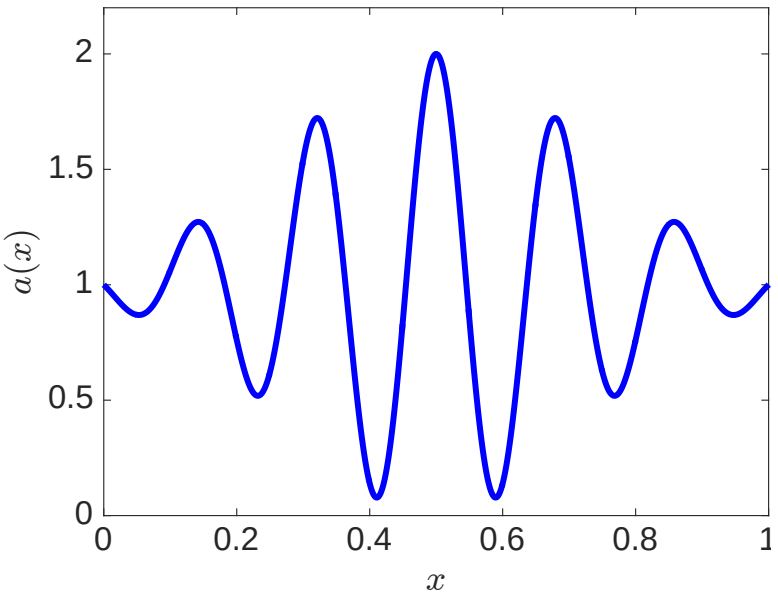
- This is an *inverse eigenvalue problem for a nonnormal operator*.
  This area is in a primitive state, compared to 70 years of work on inverse eigenvalue problems for self-adjoint operators.

A Nonnormal Inverse Eigenvalue Problem for String Design

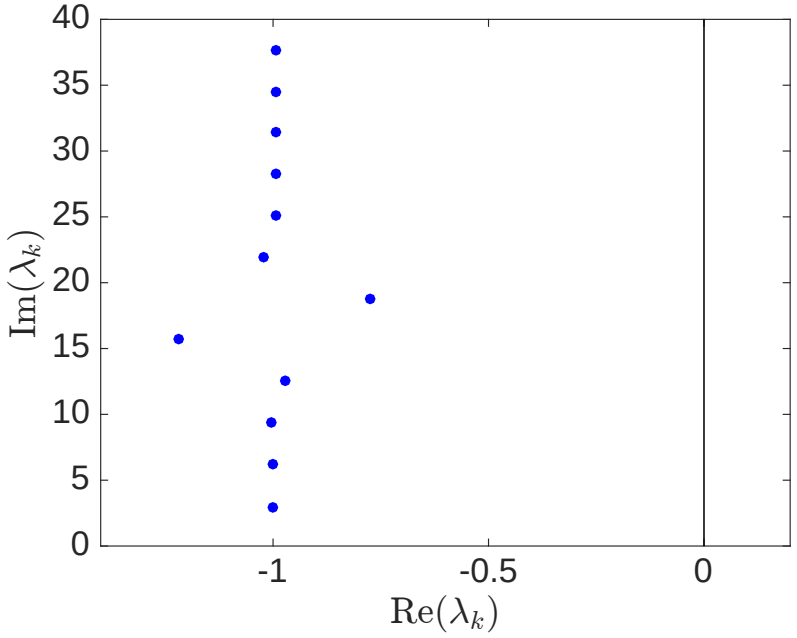
In [Cox, E. 2011], we use spectral asymptotics to derive the simple estimate

$$a(x) \approx a_0 + 2\sum_{k=1}^{\infty}(a_0 + \operatorname{Re}\lambda_k(a))\cos(2k\pi x).$$

This formula can lose accuracy for problems with heavily damped/unstable low-frequency eigenvalues, but works well in many more benign circumstances.



true damping function



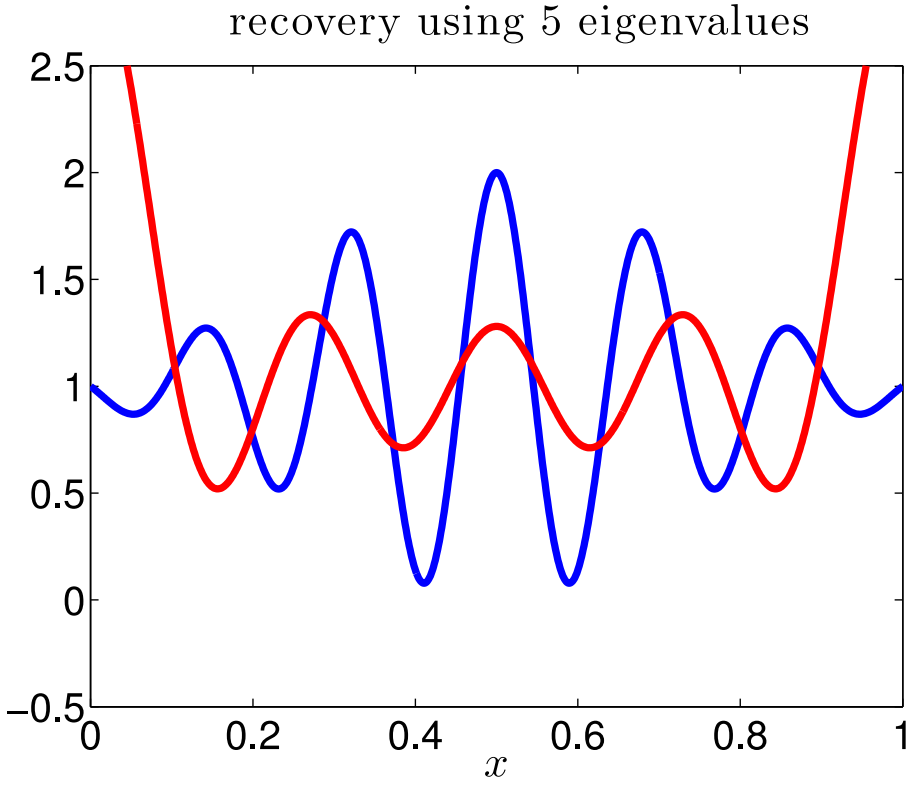
eigenvalue data

A Nonnormal Inverse Eigenvalue Problem for String Design

In [Cox, E. 2011], we use spectral asymptotics to derive the simple estimate

$$a(x) \approx a_0 + 2\sum_{k=1}^{\infty}(a_0 + \operatorname{Re}\lambda_k(a))\cos(2k\pi x).$$

This formula can lose accuracy for problems with heavily damped/unstable low-frequency eigenvalues, but works well in many more benign circumstances.

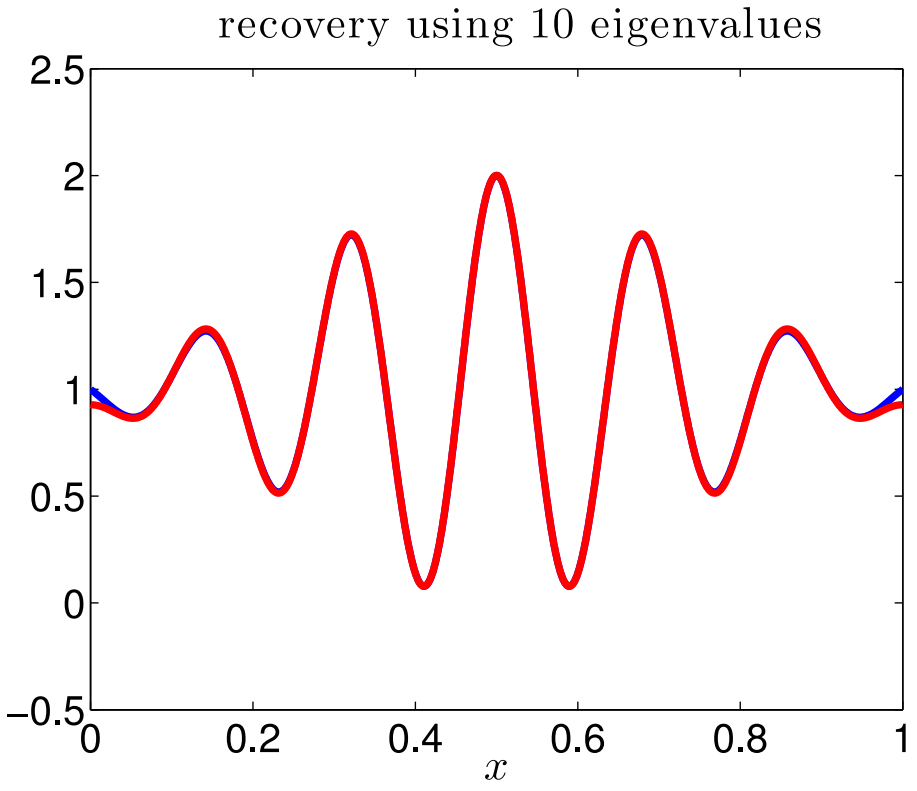


A Nonnormal Inverse Eigenvalue Problem for String Design

In [Cox, E. 2011], we use spectral asymptotics to derive the simple estimate

$$a(x) \approx a_0 + 2\sum_{k=1}^{\infty}(a_0 + \operatorname{Re}\lambda_k(a))\cos(2k\pi x).$$

This formula can lose accuracy for problems with heavily damped/unstable low-frequency eigenvalues, but works well in many more benign circumstances.

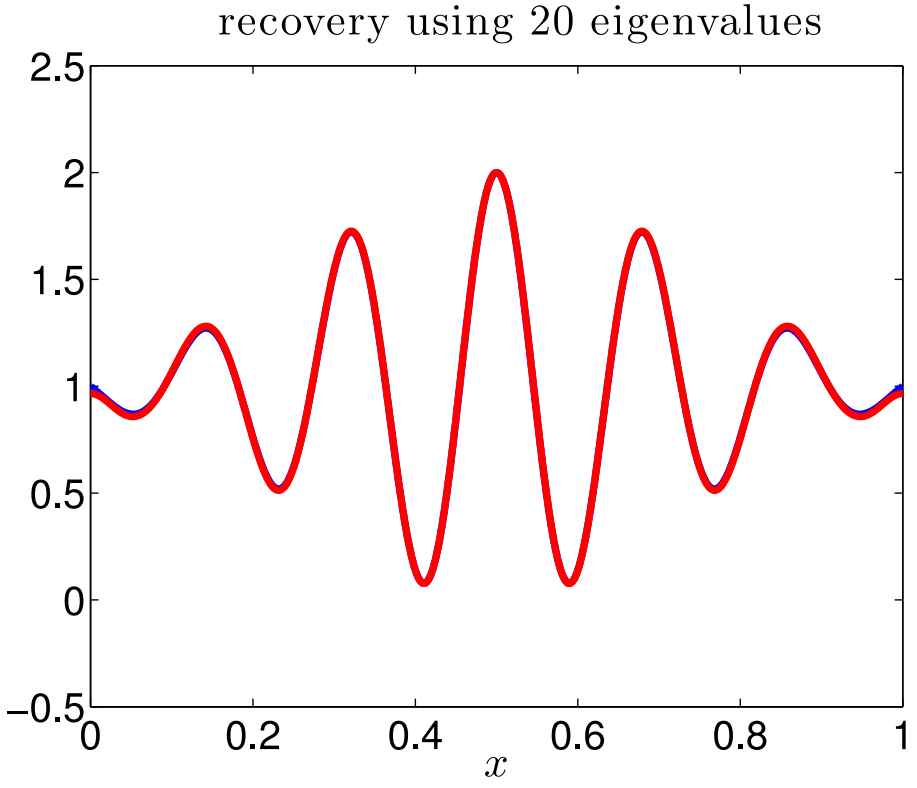


A Nonnormal Inverse Eigenvalue Problem for String Design

In [Cox, E. 2011], we use spectral asymptotics to derive the simple estimate

$$a(x) \approx a_0 + 2\sum_{k=1}^{\infty}(a_0 + \operatorname{Re}\lambda_k(a))\cos(2k\pi x).$$

This formula can lose accuracy for problems with heavily damped/unstable low-frequency eigenvalues, but works well in many more benign circumstances.



Operational Modal Analysis for building vibrations

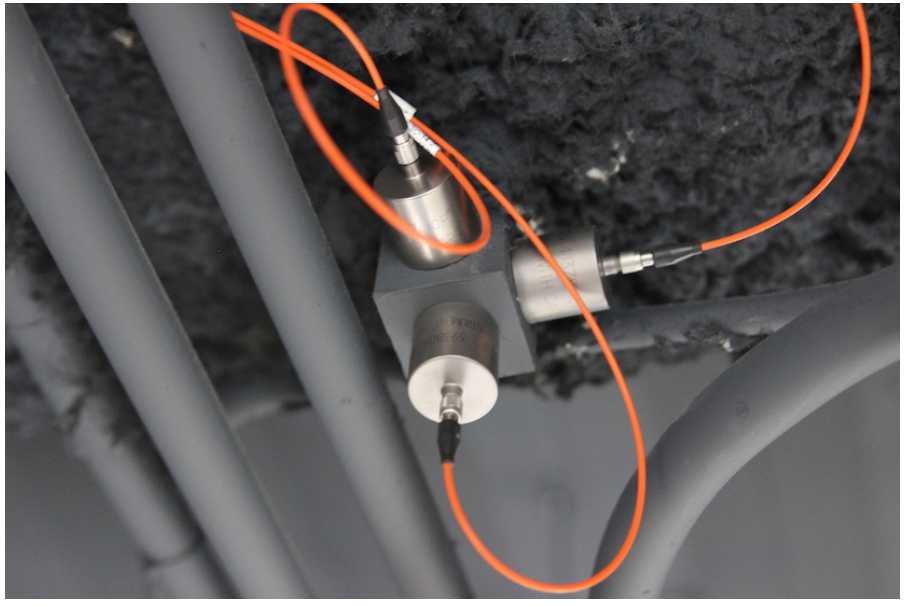
Virginia Tech opened Goodwin Hall in Fall 2015.

- The "flagship building for the College of Engineering".
- 155,000 square feet (classrooms, research and teaching labs, offices).
- 212 accelerometers welded to the steel structure during construction; 25,600 samples/second per accelerometer.

Instrumenting a smart building

The instrumentation and analysis of Goodwin Hall has been led by Prof. Pablo Tarazaga and his Virginia Tech Smart Infrastructure Lab (including Prof. Mary Kasarda, Dustin Bales, Bryan Joyce, Sriram Milladi, Austin Phoenix, Mico Woolard).

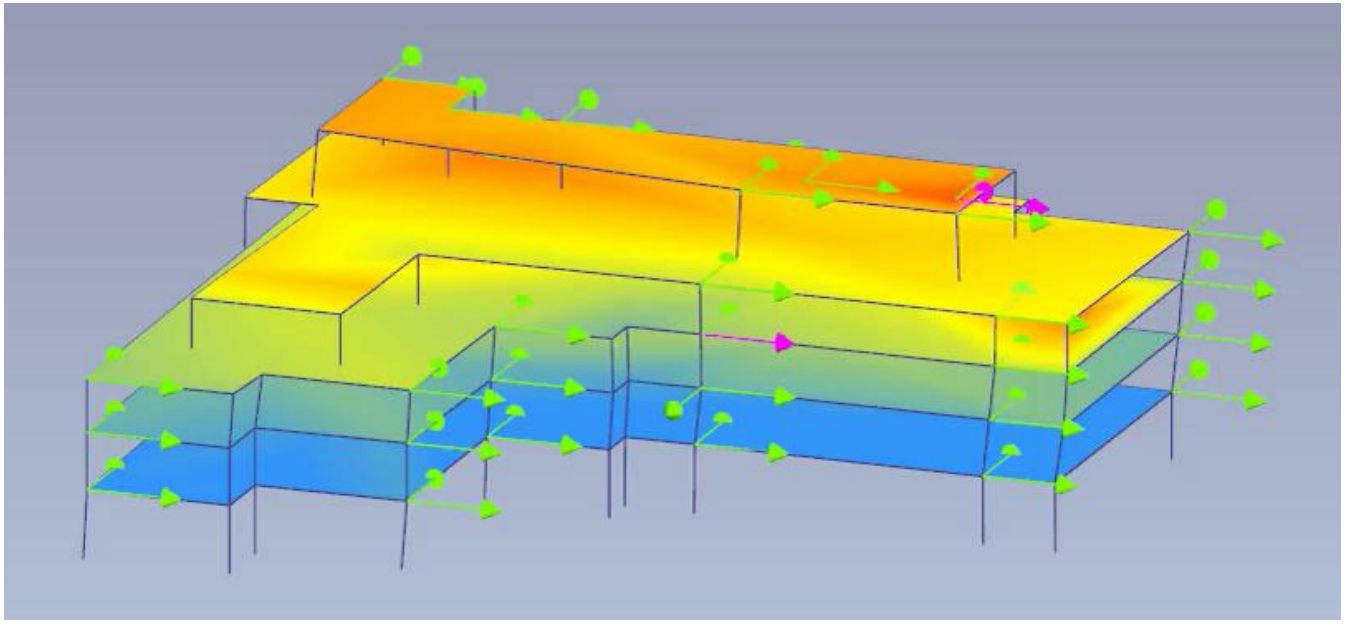
3-axis accelerometers deployed in Goodwin Hall:



Eigenvectors of Goodwin Hall

Can we use vibration data to perform modal analysis of Goodwin Hall?

First eigenmode of Goodwin Hall:

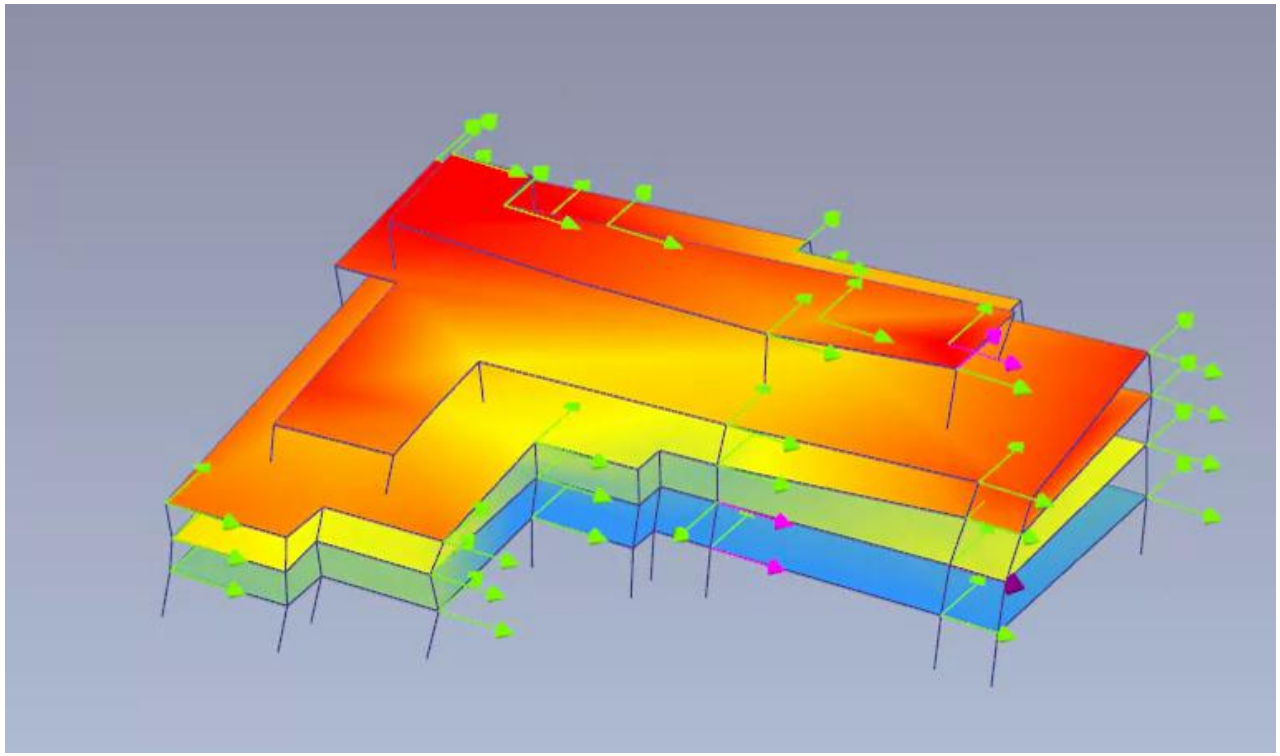


Mode estimate from accelerometer data using Aretemis software.
Computed by Pablo Tarazaga and the VT Smart Infrastructure Laboratory.

Eigenvectors of Goodwin Hall

Can we use vibration data to perform modal analysis of Goodwin Hall?

Second eigenmode of Goodwin Hall:



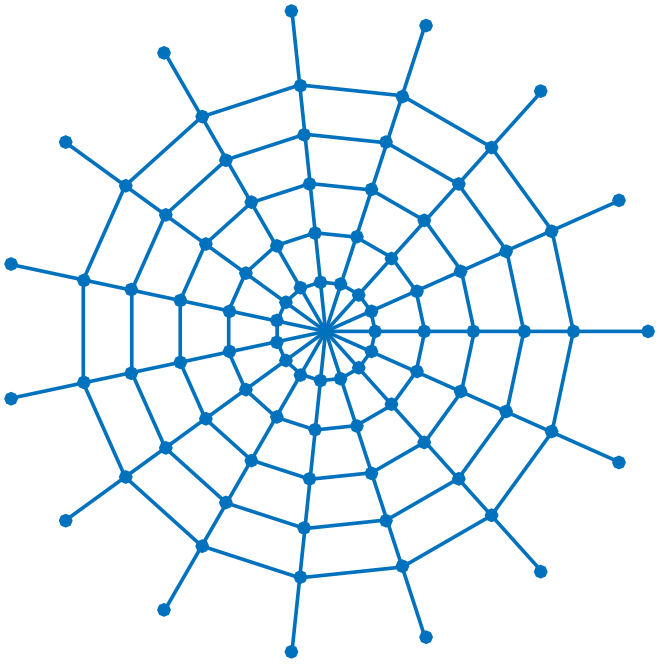
Mode estimate from accelerometer data using Aretemis software.
Computed by Pablo Tarazaga and the VT Smart Infrastructure Laboratory.

Networks of strings: Spider Webs

Future work: modal analysis for spider webs.

Biological background: Fritz Vollrath, Oxford Silk Group, ...

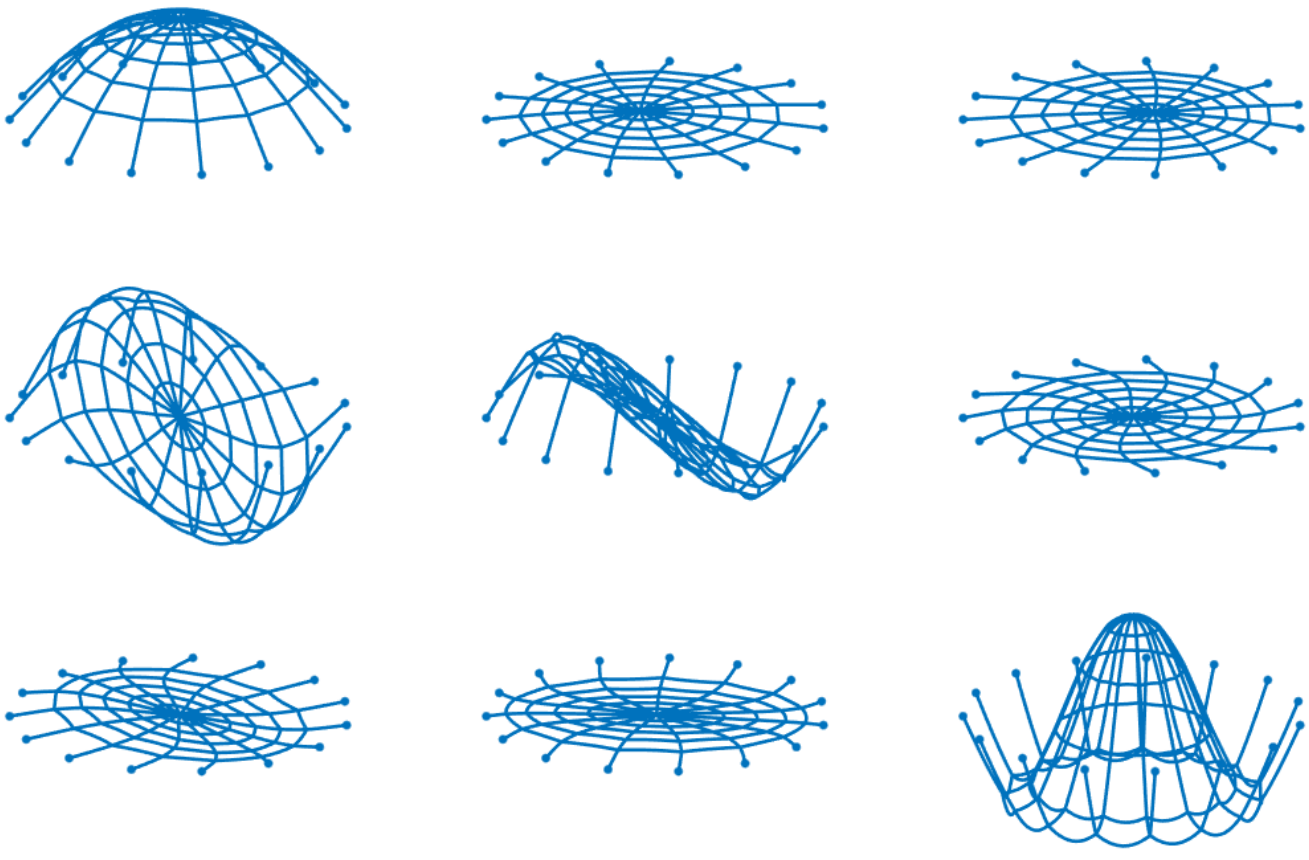
PDEs on linked networks: [Schmidt 1992], [Lagnese, Leugering, Schmidt 1994], . . . , [Arioli, Benzi 2015].



Work with Cox, Chan, LiKamWa, Morrell, Tarazaga.

Networks of strings: Spider Webs

First nine modes of a model web



From Nonlinear Eigenvalue Problem to Linear Eigenvalue Problem

Linearization of Nonlinear Eigenvalue Problems

By *linearization*, we mean converting some kind of nonlinear eigenvalue problem into a generalized eigenvalue problem (perhaps approximating).

- *Polynomial Eigenvalue Problems*
  - Generalized eigenvalue problems
  - Quadratic eigenvalue problems in damped systems

- *eigenvalue nonlinearities*
  - Exponential eigenvalue problems from delay differential equations
  - Nonlinearities induced by boundary conditions, e.g. in fiber optics

- *eigenvector nonlinearities*
  - Kohn–Sham eigenvalue problem in Density Functional Theory

A robust spectral theory exists for polynomial eigenvalue problems [Gohberg, Lancaster, Rodman, 1982], [Tisseur and Meerbergen, 2001].

General nonlinear problems pose greater challenges (finite dimensional problems can have infinitely many eigenvalues; definition of spectrum, . . .); *seven* distinct definitions of spectrum in [Appell, De Pascale, Vignoli, 2004].

Linearization of a Quadratic Eigenvalue Problem

The wave operator with viscous damping considered previously corresponds to the quadratic eigenvalue problem

$$u''(x) + 2\lambda a(x)u'(x) = \lambda^2 u(x),$$

which is discretized in the form

$$(\mathbf{K} + \lambda\mathbf{D} + \lambda^2\mathbf{M})\mathbf{u} = 0.$$

This equation can be "linearized" by introducing $\mathbf{v} = \lambda\mathbf{u}$:

$$\begin{bmatrix} \mathbf{0} & \mathbf{I} \\ -\mathbf{K} & -\mathbf{D} \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ \mathbf{v} \end{bmatrix} = \lambda \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{M} \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ \mathbf{v} \end{bmatrix}.$$

Linearization of a Quadratic Eigenvalue Problem

The wave operator with viscous damping considered previously corresponds to the quadratic eigenvalue problem

$$u''(x) + 2\lambda a(x)u'(x) = \lambda^2 u(x),$$

which is discretized in the form

$$(\mathbf{K} + \lambda\mathbf{D} + \lambda^2\mathbf{M})\mathbf{u} = 0.$$

This equation can be "linearized" by introducing $\mathbf{v} = \lambda\mathbf{u}$:

$$\begin{bmatrix} \mathbf{0} & \mathbf{I} \\ -\mathbf{K} & -\mathbf{D} \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ \mathbf{v} \end{bmatrix} = \lambda \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{M} \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ \mathbf{v} \end{bmatrix}.$$

However, other choices are available, e.g., if $\mathbf{M} = \mathbf{L}\mathbf{L}^*$, then

$$\begin{bmatrix} \mathbf{0} & \mathbf{I} \\ -\mathbf{L}^{-1}\mathbf{K}\mathbf{L}^{-*} & -\mathbf{L}^{-1}\mathbf{D}\mathbf{L}^{-*} \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ \mathbf{v} \end{bmatrix} = \lambda \begin{bmatrix} \mathbf{u} \\ \mathbf{v} \end{bmatrix}.$$

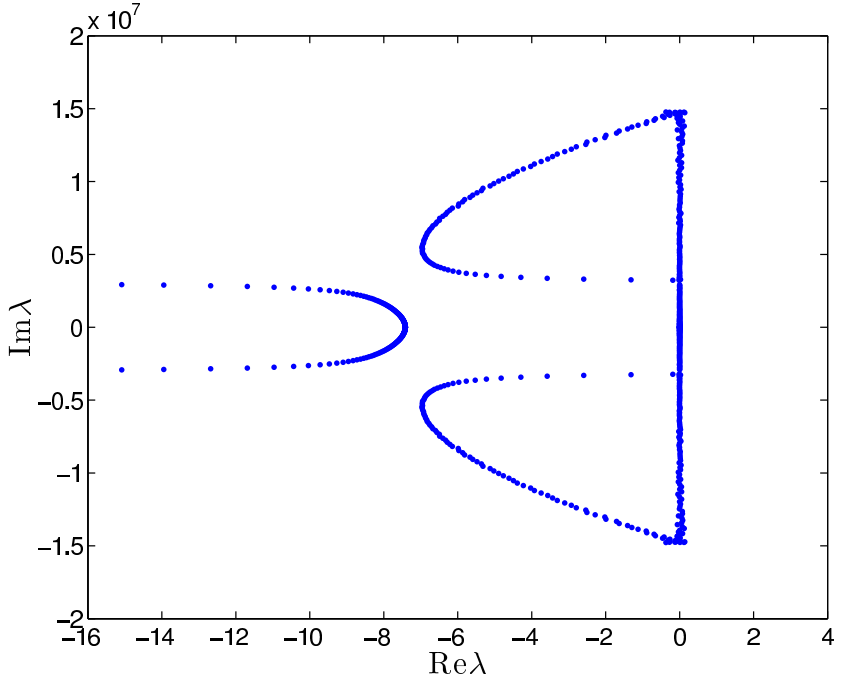
$$\begin{bmatrix} \mathbf{0} & \mathbf{M}^{-1} \\ -\mathbf{K} & \mathbf{D}\mathbf{M}^{-1} \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ \mathbf{v} \end{bmatrix} = \lambda \begin{bmatrix} \mathbf{u} \\ \mathbf{v} \end{bmatrix}.$$

The linearization can significantly influence eigenvalue computations [Mackey, Mackey, Mehl, Mehrmann, 2007], [Higham, Mackey, Tisseur, Garvey, 2008].

Linearization of a Quadratic Eigenvalue Problem

A beam with pointwise damping, from [Higham, Mackey, Tisseur, Garvey, 2008]

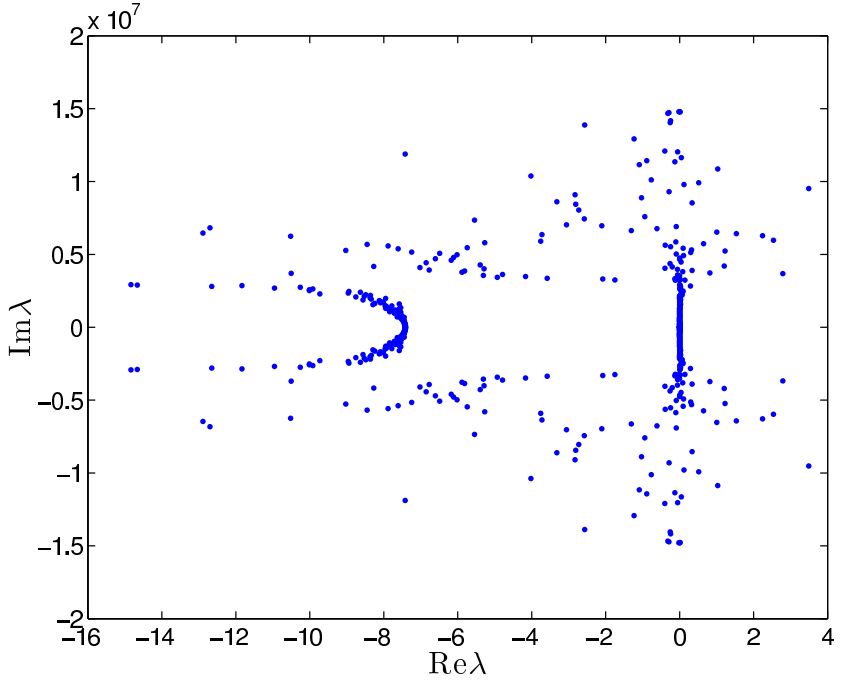
$$\begin{bmatrix} -\mathbf{D} & -\mathbf{K} \\ \mathbf{I} & \mathbf{0} \end{bmatrix}, \quad \begin{bmatrix} \mathbf{M} & \mathbf{0} \\ \mathbf{0} & \mathbf{I} \end{bmatrix}$$



Linearization of a Quadratic Eigenvalue Problem

A beam with pointwise damping, from [Higham, Mackey, Tisseur, Garvey, 2008]

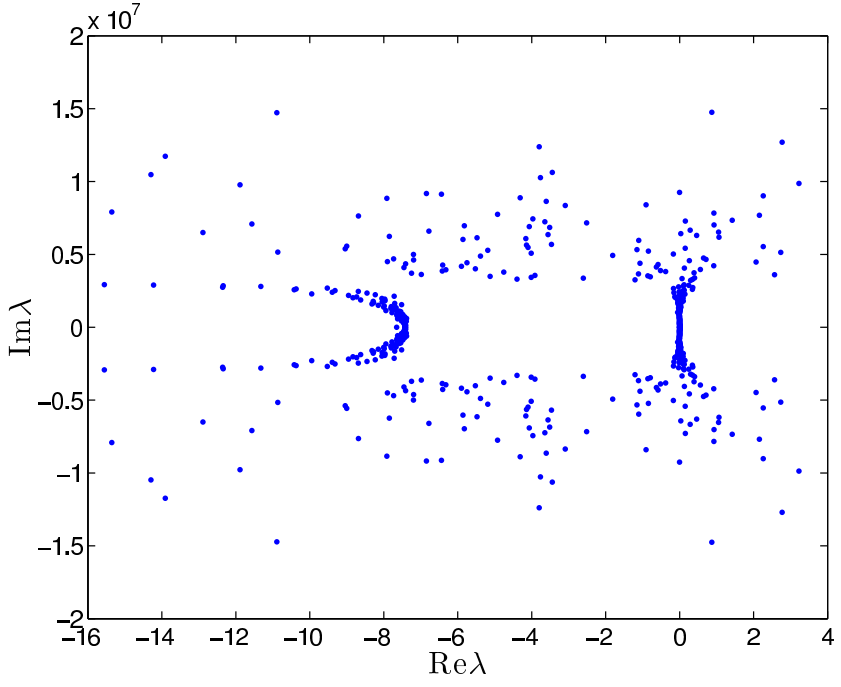
$$\begin{bmatrix} -\mathbf{D} & -\mathbf{K} \\ -\mathbf{K} & \mathbf{0} \end{bmatrix}, \quad \begin{bmatrix} \mathbf{M} & \mathbf{0} \\ \mathbf{0} & -\mathbf{K} \end{bmatrix}$$



Linearization of a Quadratic Eigenvalue Problem

A beam with pointwise damping, from [Higham,Mackey,Tisseur, Garvey, 2008]

$$\begin{bmatrix} -\mathbf{M} & \mathbf{0} \\ \mathbf{0} & -\mathbf{K} \end{bmatrix}, \begin{bmatrix} \mathbf{0} & \mathbf{M} \\ \mathbf{M} & \mathbf{D} \end{bmatrix}$$

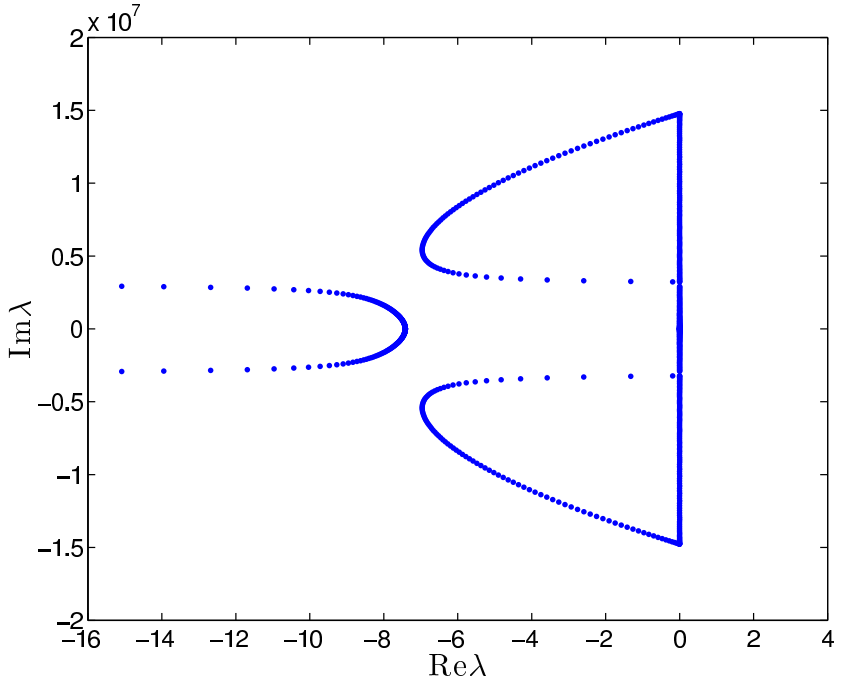


Linearization of a Quadratic Eigenvalue Problem

A beam with pointwise damping, from [Higham, Mackey, Tisseur, Garvey, 2008]

Higham et al. show that the instability can be cured by applying a clever coefficient scaling of [Fan, Lin, Van Dooren 2004].

$$\begin{bmatrix} -\mathbf{M} & \mathbf{0} \\ \mathbf{0} & -\mathbf{K} \end{bmatrix}, \qquad \begin{bmatrix} \mathbf{0} & \mathbf{M} \\ \mathbf{M} & \mathbf{D} \end{bmatrix}$$

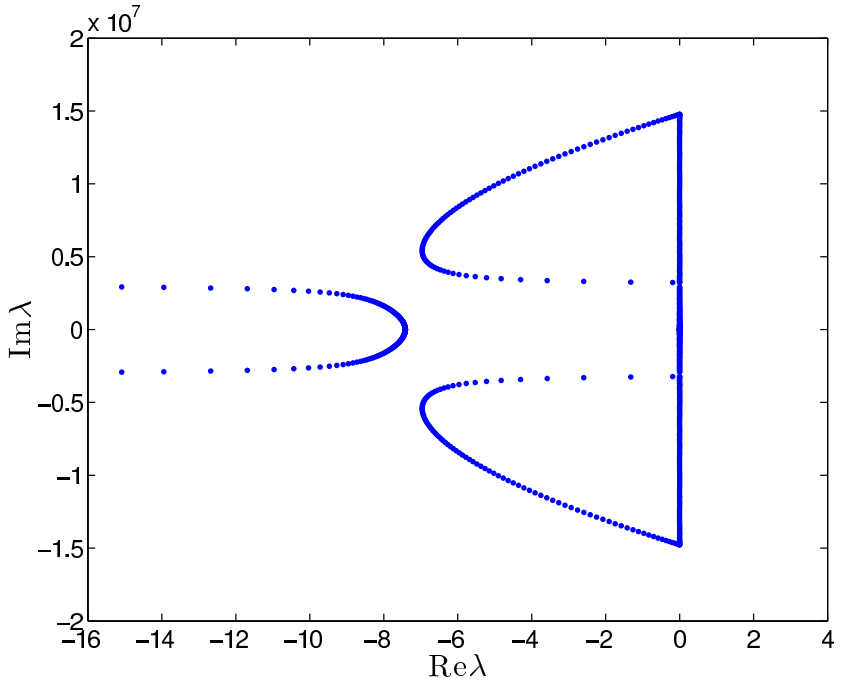


Linearization of a Quadratic Eigenvalue Problem

A beam with pointwise damping, from [Higham, Mackey, Tisseur, Garvey, 2008]

Another perspective: Transform coordinates so that the 2-norm of the linearization represents the *energy norm* of the original quadratic problem.

$$\begin{bmatrix} \mathbf{0} & \mathbf{M}^{-1} \\ -\mathbf{K} & -\mathbf{D}\mathbf{M}^{-1} \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ \mathbf{v} \end{bmatrix} = \lambda \begin{bmatrix} \mathbf{u} \\ \mathbf{v} \end{bmatrix}$$



Eigenvalue conditioning of discretization now matches that of the operator.

From Linear Operator Eigenvalue Problem to Large Discretization Matrix

Convergence Theory for Discretized Operators

Approximate the linear operator A with an $N \times N$ discretization matrix $\mathbf{A}_N$.

How fast do eigenvalues of $\mathbf{A}_N$ converge to those of A?

Many results toward this end were obtained in the 1960s–1980s, with antecedents in the Weinstein/Aronszajn theory of intermediate problems for self-adjoint operators.

- Compact integral operators [Anselone, 1965], [Atkinson, 1975], [Osborn, 1975]
  Convergence tracks the accuracy of the discretizing quadrature rule
- Differential operators in variational form [Osborn, 1976, . . .]
  Convergence tracks quality of eigenfunction approximation in FE space
- Abstract theory based on operator convergence, $\mathbf{A}_N \to A$.

Surveys include [Chatelin, 1983]; [Babuška and Osborn, 1971], [Ahues, Largillier, Limaye, 2001]; [Boffi, 2010].

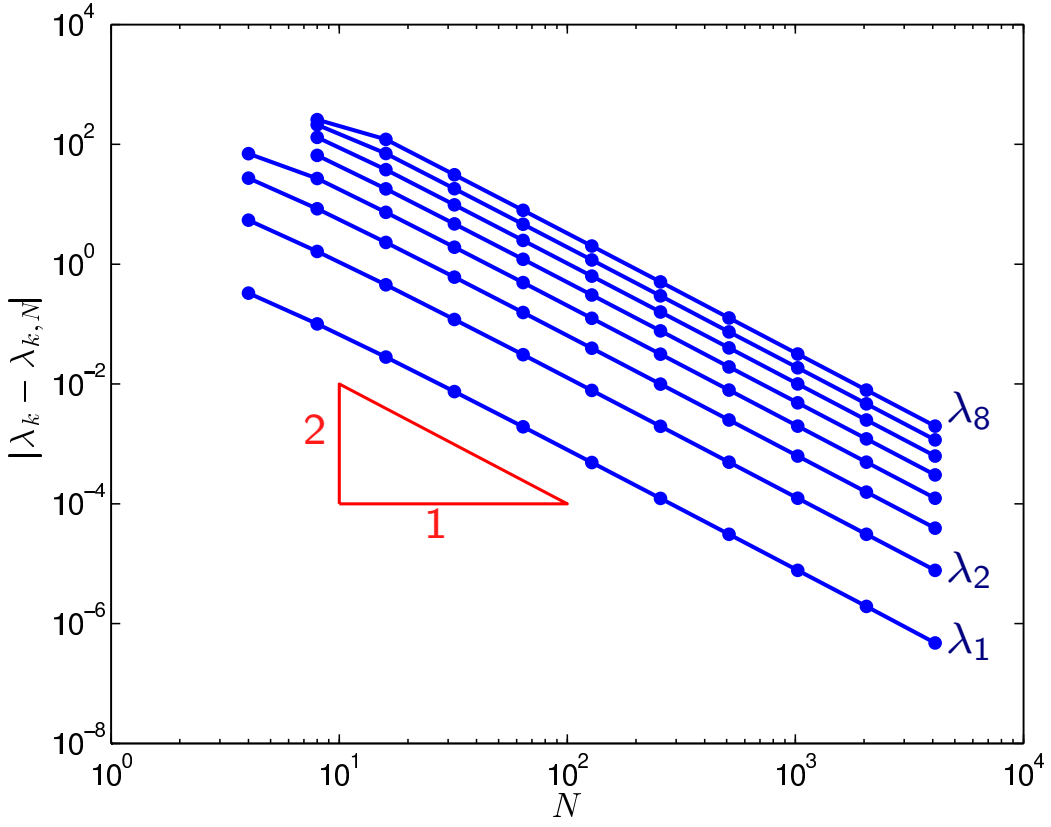
Key improvements include mesh refinement for troublesome eigenfunctions; multiple/defective eigenvalues; *a posteriori* error adaptivity, *hp* refinement, etc.

Standard Convergence Behavior for a Self-Adjoint Operator

Dirichlet Laplacian: $Au = -u''$ on $L^2[0,1]$ with $u(0) = u(1) = 0$.

Eigenvalues: $\lambda_k = k^2\pi^2$. Eigenfunctions: $u_k(x) = \sin(k\pi x)$.

Finite elements give a generalized eigenvalue problem: $\mathbf{Ku} = \lambda\mathbf{Mu}$.
For piecewise linear elements:

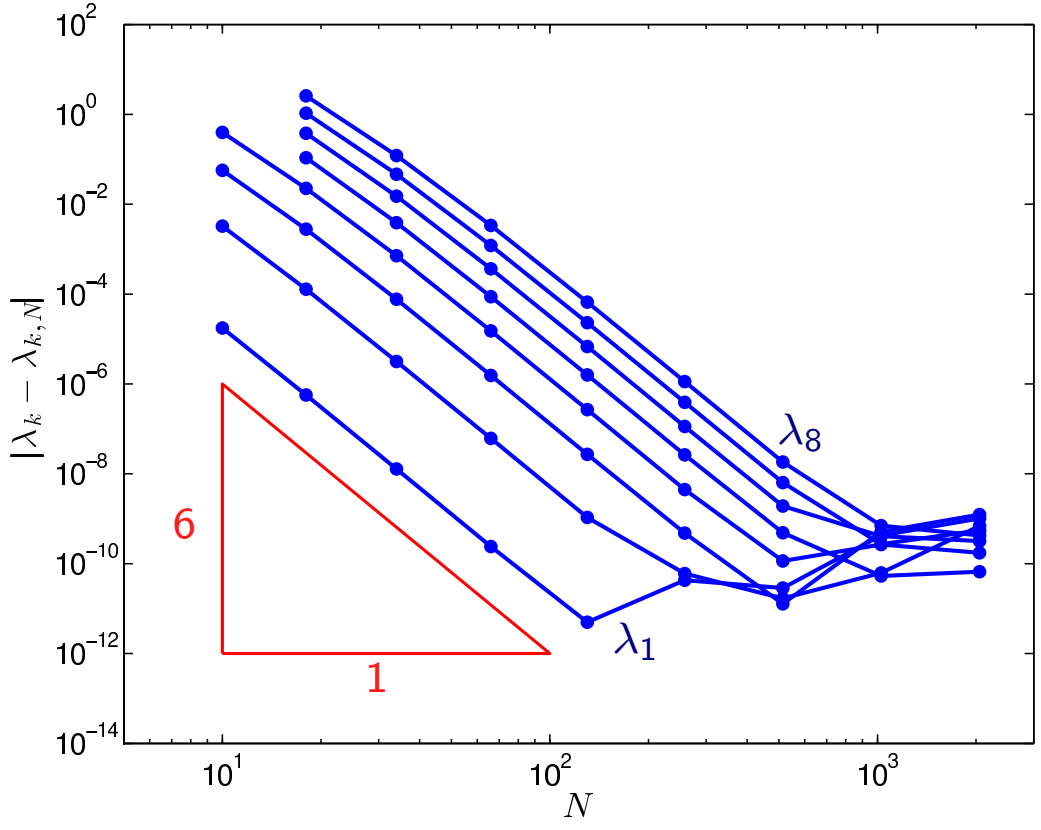


Standard Convergence Behavior for a Self-Adjoint Operator

Dirichlet Laplacian: $Au = -u''$ on $L^2[0,1]$ with $u(0) = u(1) = 0$.

Eigenvalues: $\lambda_k = k^2\pi^2$. Eigenfunctions: $u_k(x) = \sin(k\pi x)$.

Finite elements give a generalized eigenvalue problem: $\mathbf{Ku} = \lambda\mathbf{Mu}$.
For piecewise cubic Hermite elements:



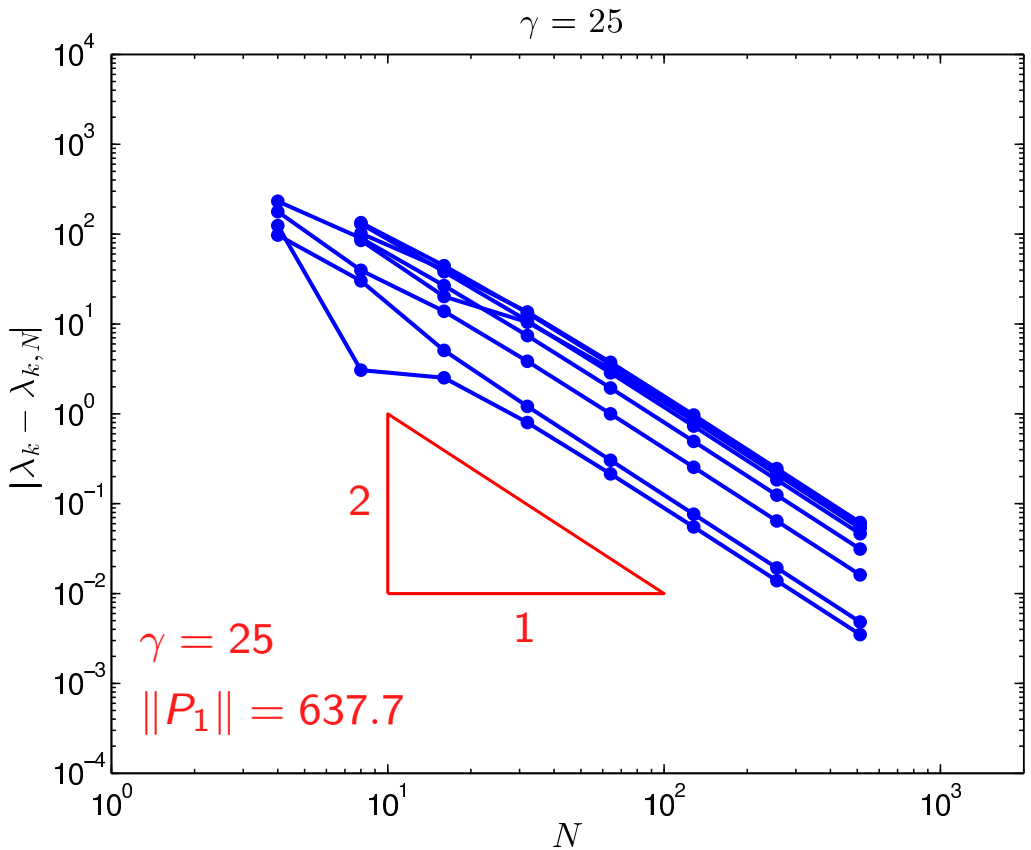
Standard Convergence Behavior for a Non-Self-Adjoint Operator

Convection–Diffusion: $Au = -u'' + \gamma u'$ on $L^2[0,1]$ with $u(0) = u(1) = 0$.

Eigenvalues: $\lambda_k = k^2\pi^2 + \gamma^2/4$. Eigenfunctions: $u_k(x) = e^{\gamma x/2}\sin(k\pi x)$.

Norms of spectral projectors: $\|P_j\| = \dfrac{\sqrt{e^{\gamma} + e^{-\gamma} - 2}}{\gamma + \gamma^3/(4k^2\pi^2)}$.

Finite elements again give $\mathbf{Ku} = \lambda\mathbf{Mu}$, but now $\mathbf{K}$ is nonsymmetric.
For piecewise linear elements:



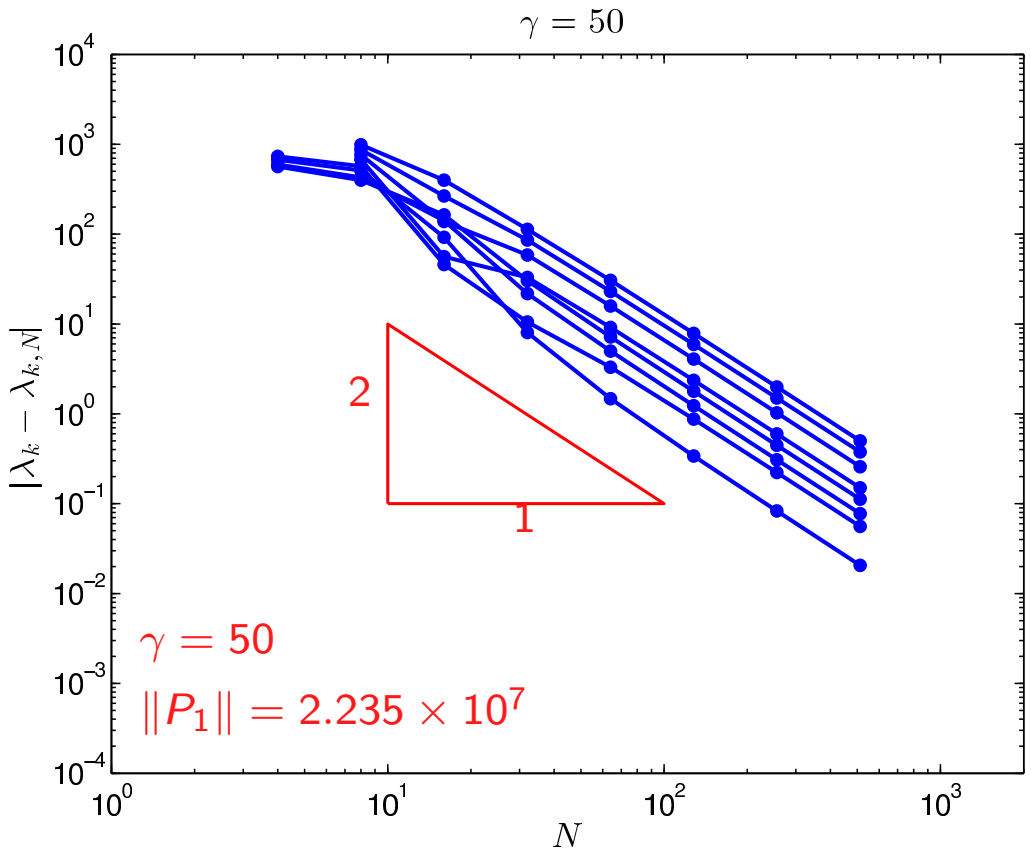
Standard Convergence Behavior for a Non-Self-Adjoint Operator

Convection–Diffusion: $Au = -u'' + \gamma u'$ on $L^2[0,1]$ with $u(0) = u(1) = 0$.

Eigenvalues: $\lambda_k = k^2\pi^2 + \gamma^2/4$. Eigenfunctions: $u_k(x) = e^{\gamma x/2}\sin(k\pi x)$.

Norms of spectral projectors: $\|P_j\| = \dfrac{\sqrt{e^{\gamma} + e^{-\gamma} - 2}}{\gamma + \gamma^3/(4k^2\pi^2)}$.

Finite elements again give $\mathbf{Ku} = \lambda\mathbf{Mu}$, but now $\mathbf{K}$ is nonsymmetric.
For piecewise linear elements:



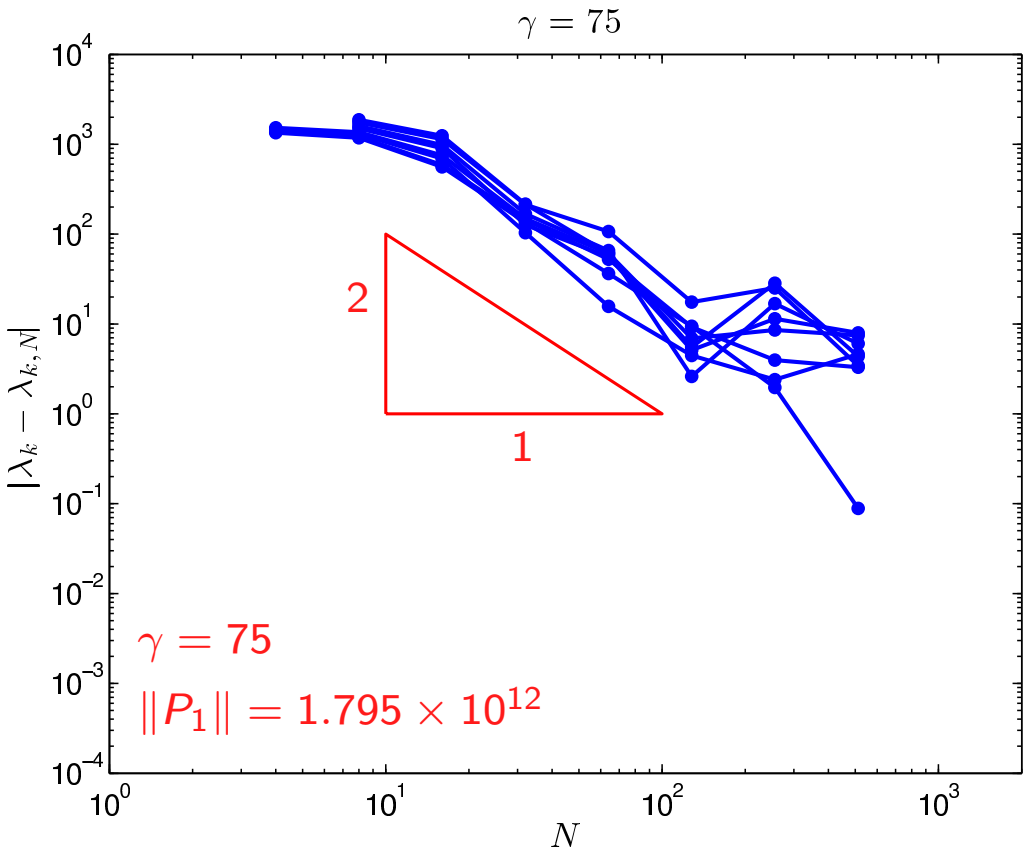
Standard Convergence Behavior for a Non-Self-Adjoint Operator

Convection–Diffusion: $Au = -u'' + \gamma u'$ on $L^2[0,1]$ with $u(0) = u(1) = 0$.

Eigenvalues: $\lambda_k = k^2\pi^2 + \gamma^2/4$. Eigenfunctions: $u_k(x) = e^{\gamma x/2}\sin(k\pi x)$.

Norms of spectral projectors: $\|P_j\| = \dfrac{\sqrt{e^{\gamma} + e^{-\gamma} - 2}}{\gamma + \gamma^3/(4k^2\pi^2)}$.

Finite elements again give $\mathbf{Ku} = \lambda\mathbf{Mu}$, but now $\mathbf{K}$ is nonsymmetric.
For piecewise linear elements:



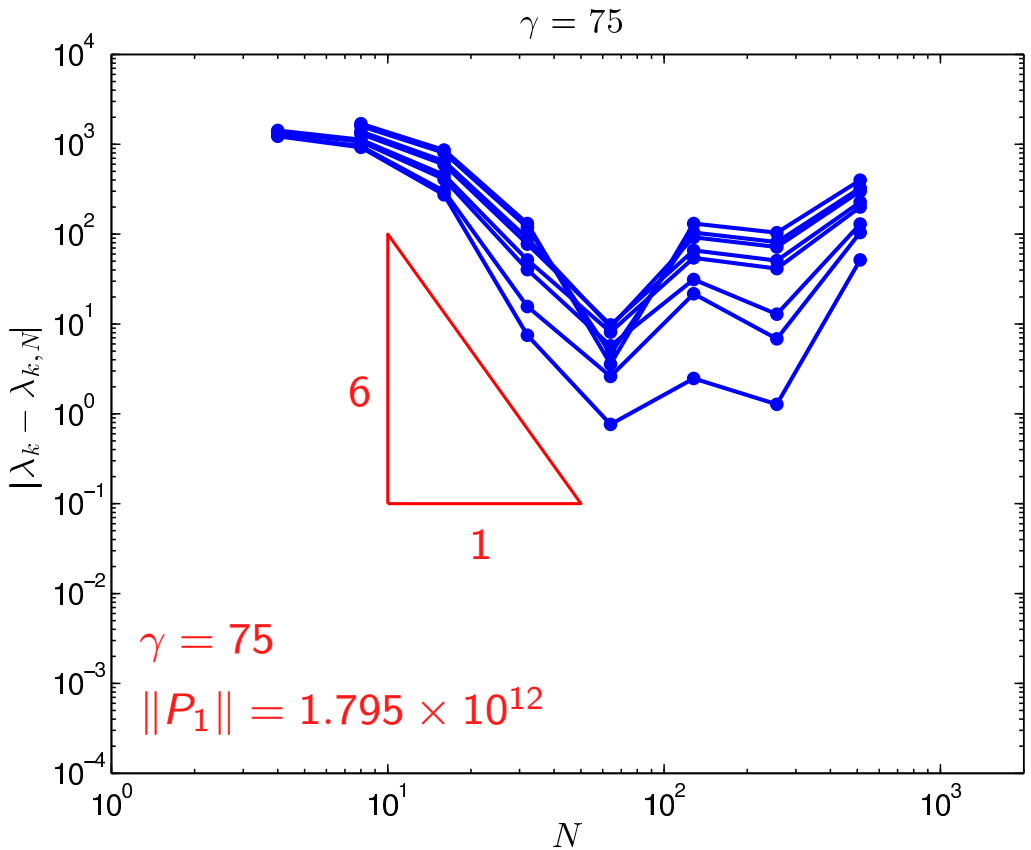
Standard Convergence Behavior for a Non-Self-Adjoint Operator

Convection–Diffusion: $Au = -u'' + \gamma u'$ on $L^2[0,1]$ with $u(0) = u(1) = 0$.

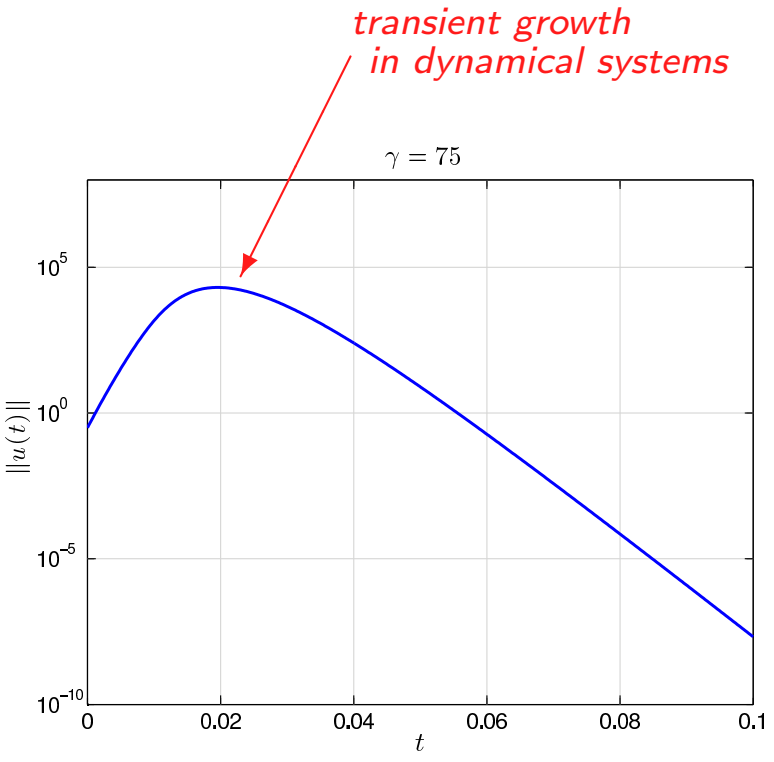
Eigenvalues: $\lambda_k = k^2\pi^2 + \gamma^2/4$. Eigenfunctions: $u_k(x) = e^{\gamma x/2}\sin(k\pi x)$.

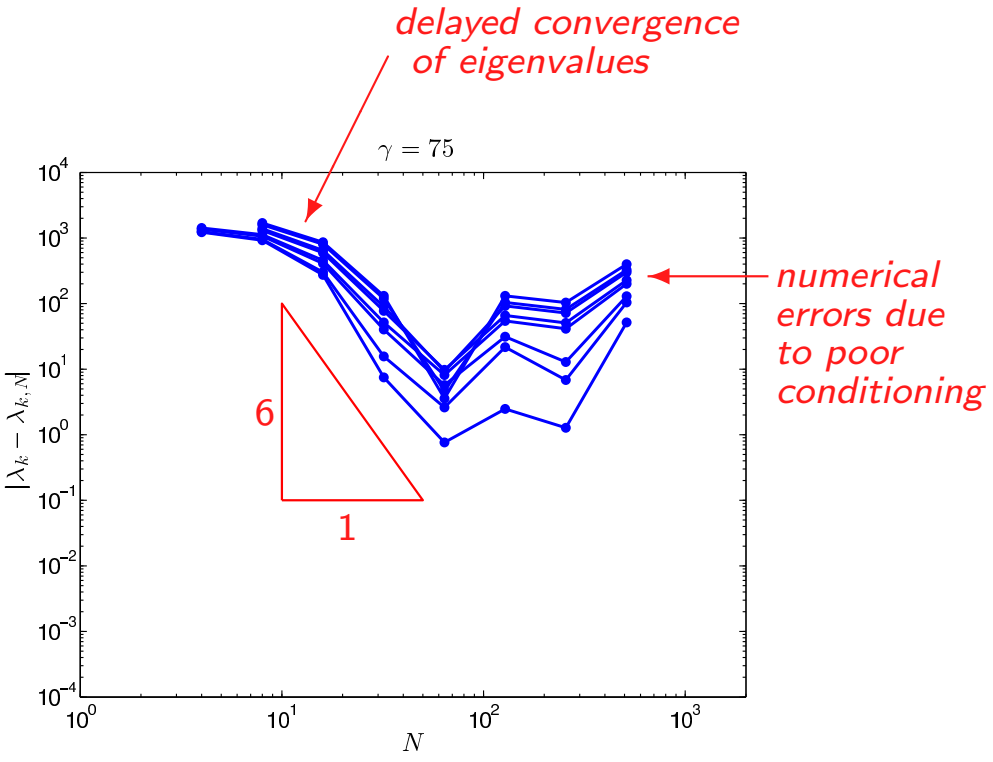
Norms of spectral projectors: $\|P_j\| = \dfrac{\sqrt{e^{\gamma} + e^{-\gamma} - 2}}{\gamma + \gamma^3/(4k^2\pi^2)}$.

Finite elements again give $\mathbf{Ku} = \lambda\mathbf{Mu}$, but now $\mathbf{K}$ is nonsymmetric.
For piecewise cubic Hermite elements:



Three Influences of Nonnormality





$$u_t(t) = u_{xx}(x,t) - \gamma u_x(x,t) + cu(x,t)$$

Discretization for Bulk Spectral Properties

Sometimes one needs more than just a few eigenvalues;
one seeks bulk spectral properties, e.g., density of states, fractal dimension.

Discretization for Bulk Spectral Properties

Sometimes one needs more than just a few eigenvalues;
one seeks bulk spectral properties, e.g., density of states, fractal dimension.

Example: The Fibonacci Hamiltonian, a one-dimensional quasicrystal model [Kohmoto, Kadanoff, Tang 1983; Ostlund et al. 1983], ...

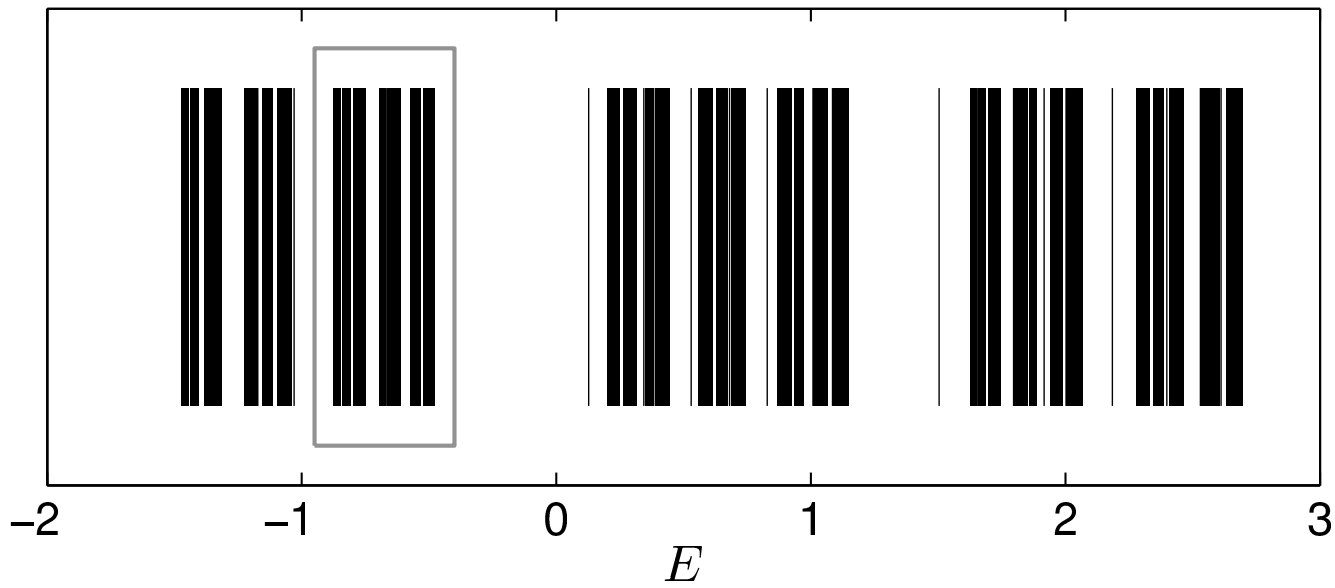
$$H_\lambda = \begin{bmatrix}
\ddots & \ddots & \ddots & & & & & & & & \\
 & 1 & 0 & 1 & & & & & & & \\
 & & 1 & \lambda & 1 & & & & & & \\
 & & & 1 & \lambda & 1 & & & & & \\
 & & & & 1 & 0 & 1 & & & & \\
 & & & & & 1 & \lambda & 1 & & & \\
 & & & & & & 1 & 0 & 1 & & \\
 & & & & & & & 1 & \lambda & 1 & \\
 & & & & & & & & 1 & \lambda & 1 \\
 & & & & & & & & & \ddots & \ddots & \ddots
\end{bmatrix}$$

Key: The potential (diagonal) is *quasiperiodic*.

How does the 'coupling constant' λ affect the spectrum?

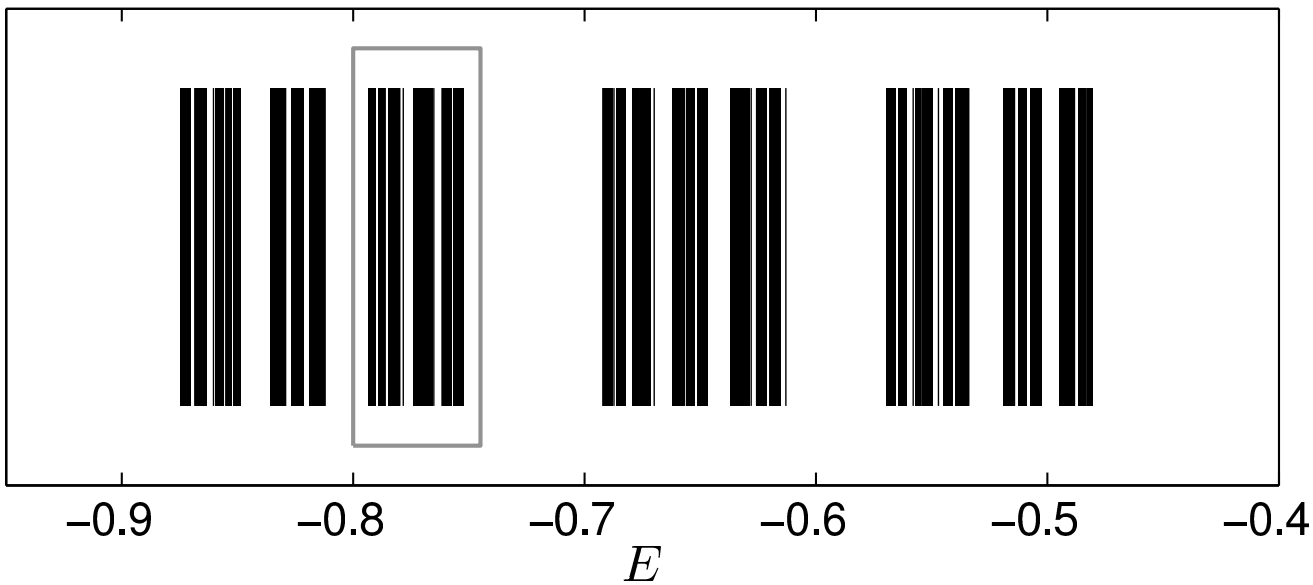
Discretization for Bulk Spectral Properties

Eigenvalues of an $N \times N$ section of H_λ: $\lambda = 1$ and $N = 100{,}000$.



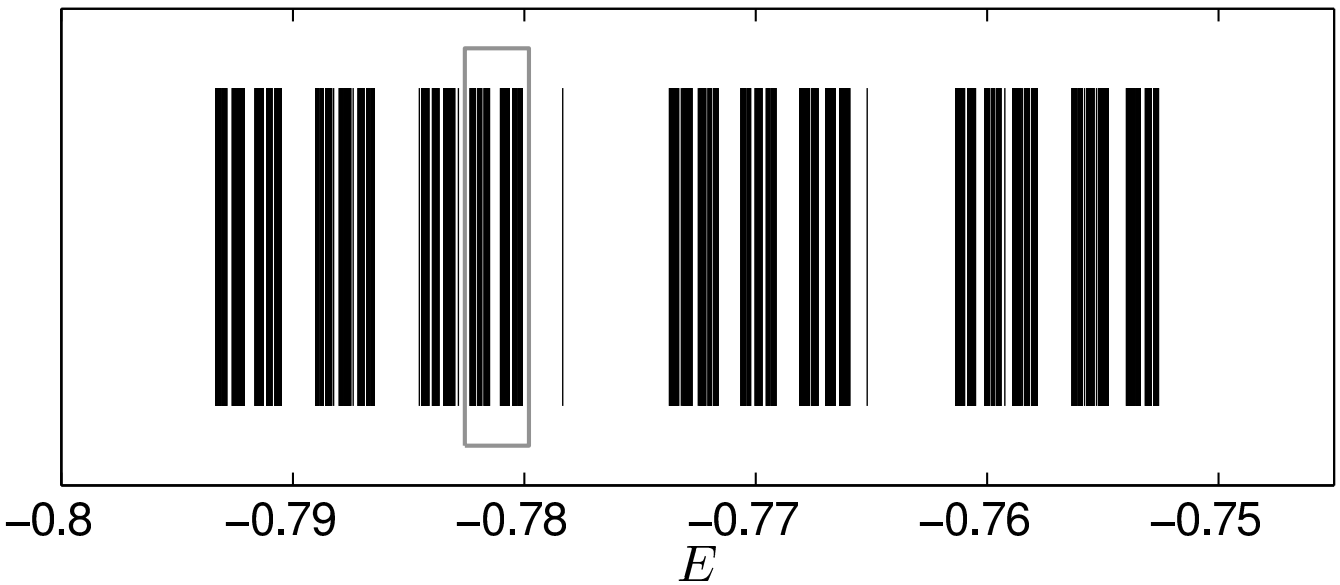
Discretization for Bulk Spectral Properties

Eigenvalues of an $N \times N$ section of H_λ: $\lambda = 1$ and $N = 100{,}000$.



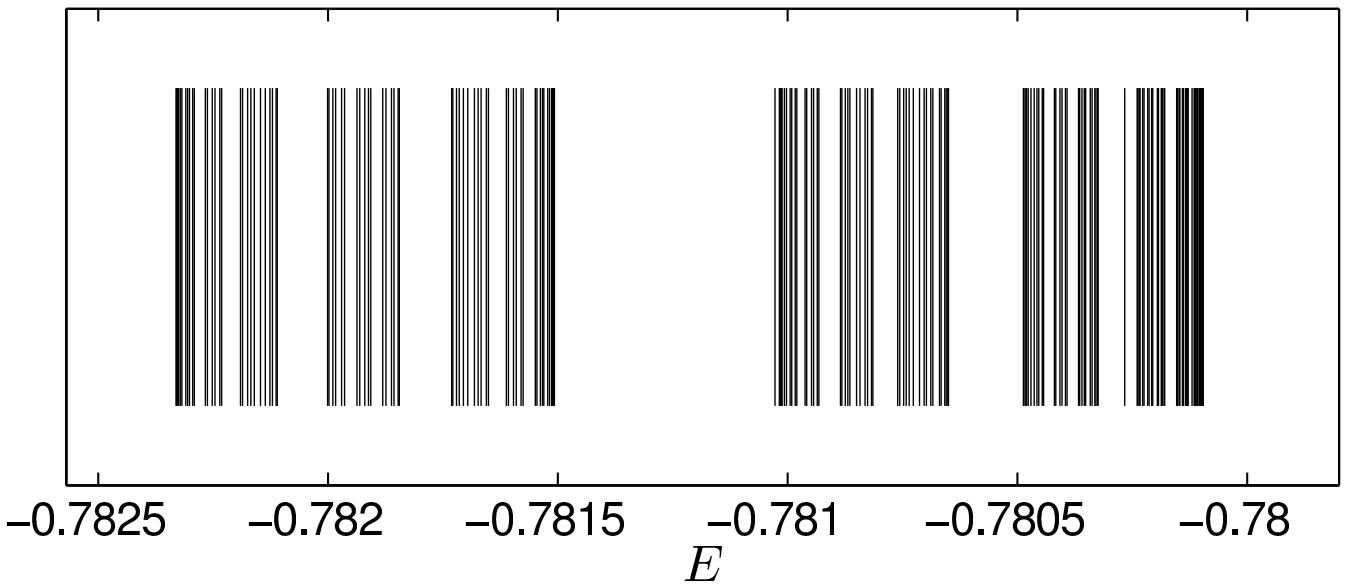
Discretization for Bulk Spectral Properties

Eigenvalues of an $N \times N$ section of H_λ: $\lambda = 1$ and $N = 100{,}000$.



Discretization for Bulk Spectral Properties

Eigenvalues of an $N \times N$ section of H_λ: $\lambda = 1$ and $N = 100{,}000$.



Discretization for Bulk Spectral Properties

$$H_\lambda = \begin{bmatrix} \ddots & \ddots & \ddots & & & & & & & & \\ & 1 & 0 & 1 & & & & & & & \\ & & 1 & \lambda & 1 & & & & & & \\ & & & 1 & \lambda & 1 & & & & & \\ & & & & 1 & 0 & 1 & & & & \\ & & & & & 1 & \lambda & 1 & & & \\ & & & & & & 1 & 0 & 1 & & \\ & & & & & & & 1 & \lambda & 1 & \\ & & & & & & & & 1 & \lambda & \ddots \\ & & & & & & & & & \ddots & \ddots \end{bmatrix}$$

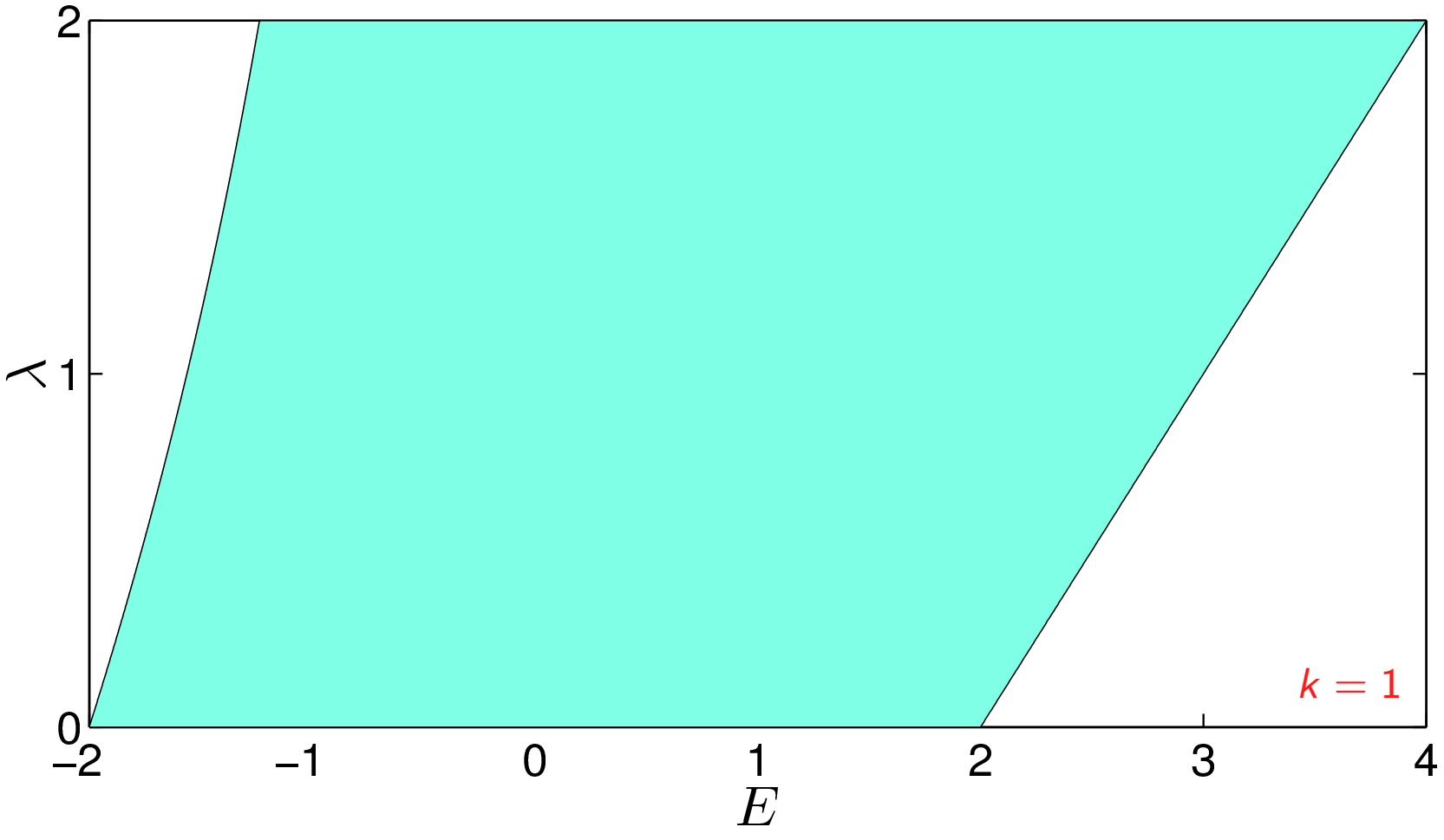
- For any $\lambda > 0$, *the spectrum is a Cantor set* Sütő [1987, 1989].
- If the potential (diagonal) had period p, one could compute the spectrum by solving two $p \times p$ matrix eigenvalue problems (Floquet theory).
- Sütő [1987] proved the spectra of successive *period-k approximations* $H_{k,\lambda}$ cover the spectrum of H_λ,

$$\sigma(H_\lambda) \subset \Sigma_{k,\lambda} := \sigma(H_{k,\lambda}) \cup \sigma(H_{k+1,\lambda}),$$

  and converge to it.
- Thus we infer bulk spectral properties from finite dimensional problems.

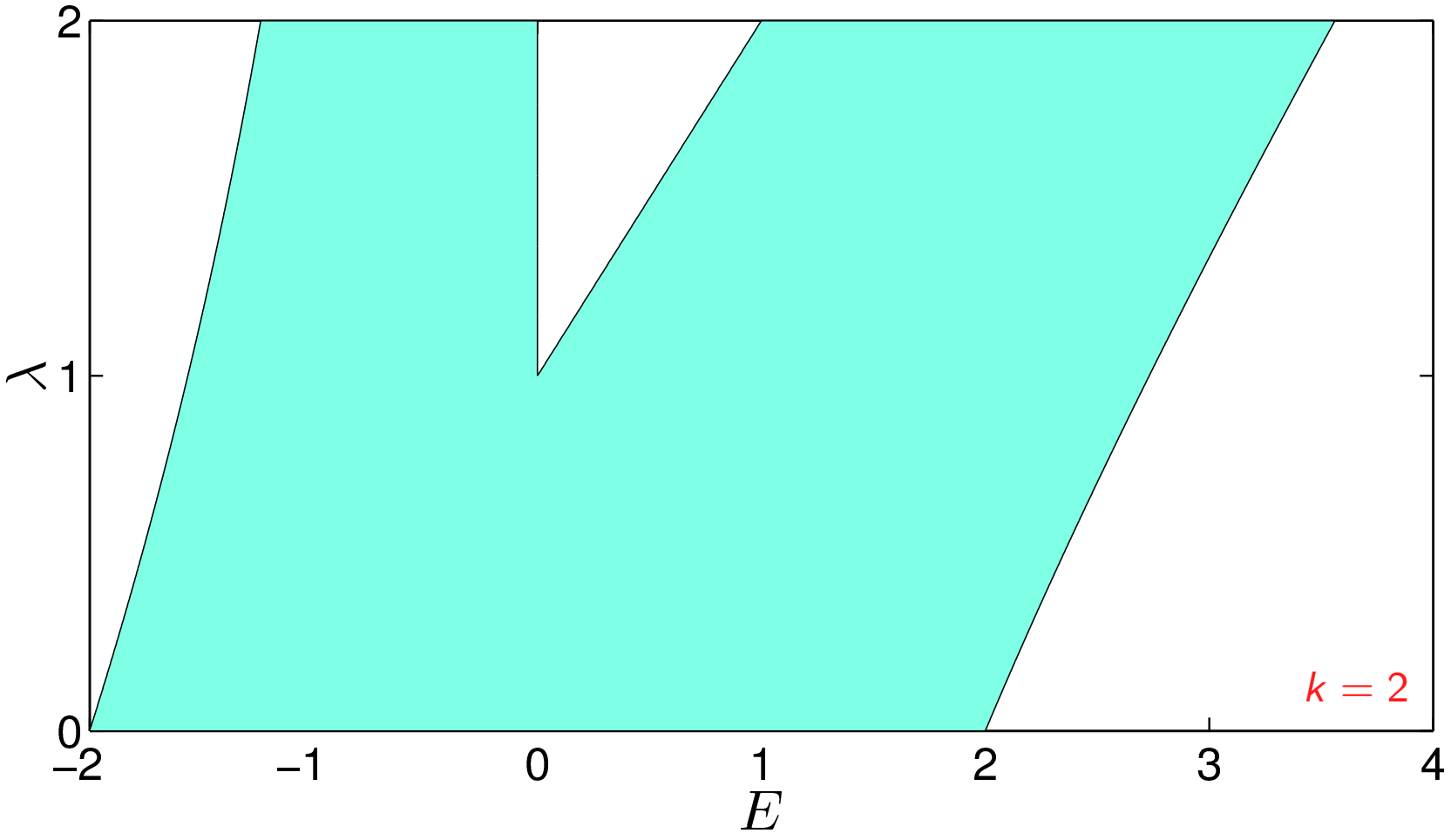
Upper Bounds on the Spectrum

Sütő's upper bound $\Sigma_{\lambda,k}$ as a function of λ.



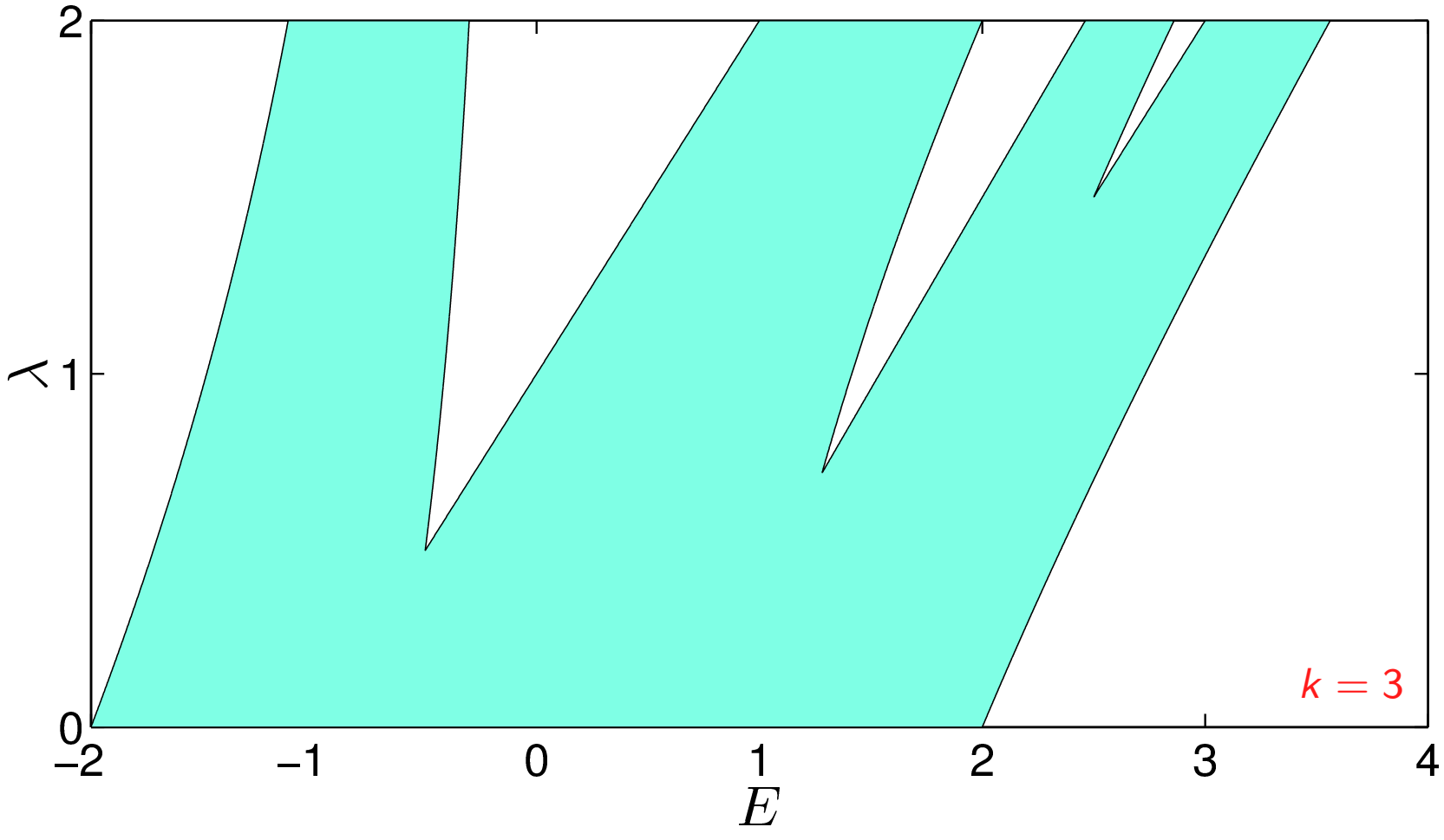
Upper Bounds on the Spectrum

Sütő's upper bound $\Sigma_{\lambda,k}$ as a function of λ.



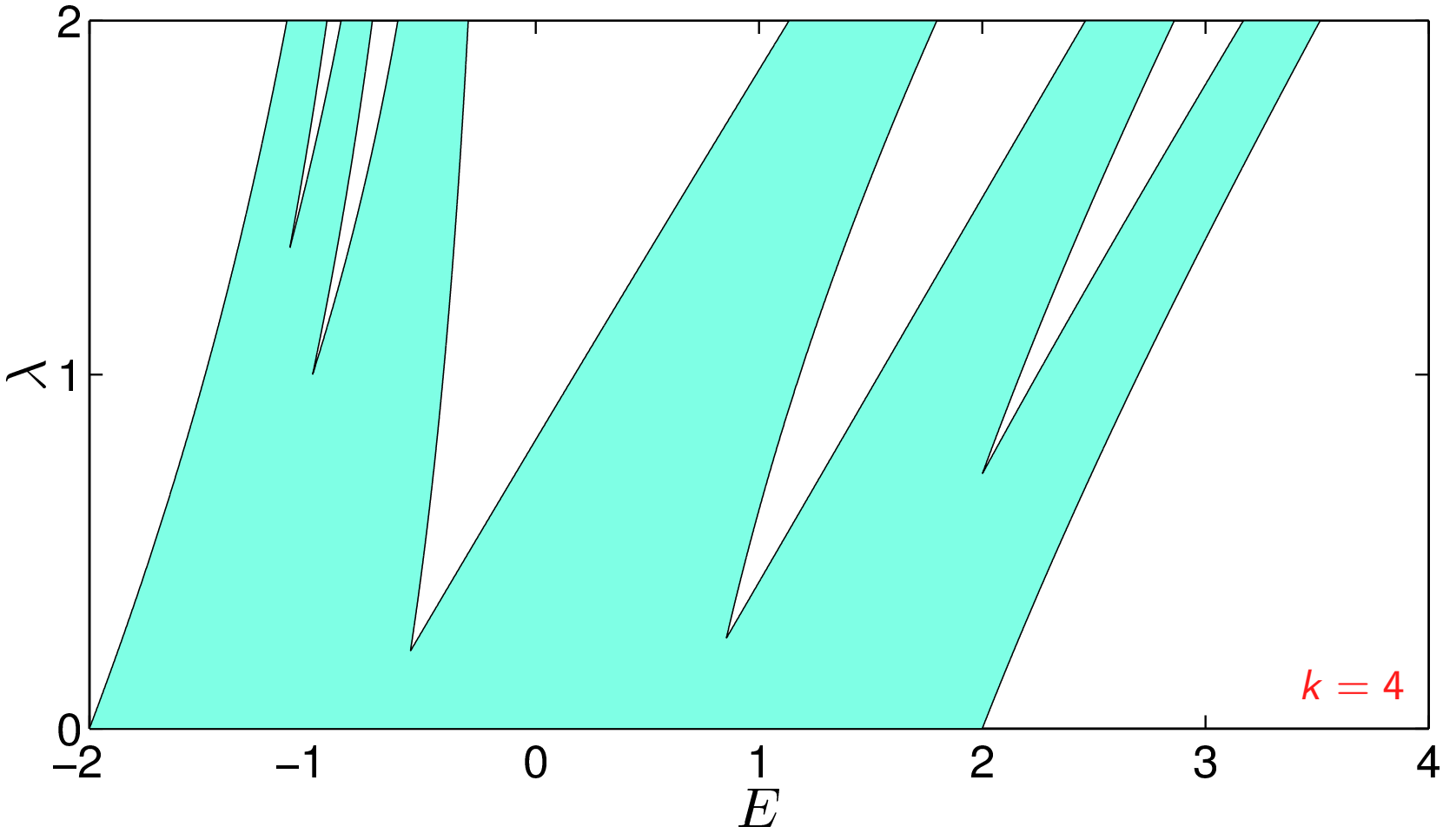
Upper Bounds on the Spectrum

Sütő's upper bound $\Sigma_{\lambda,k}$ as a function of λ.



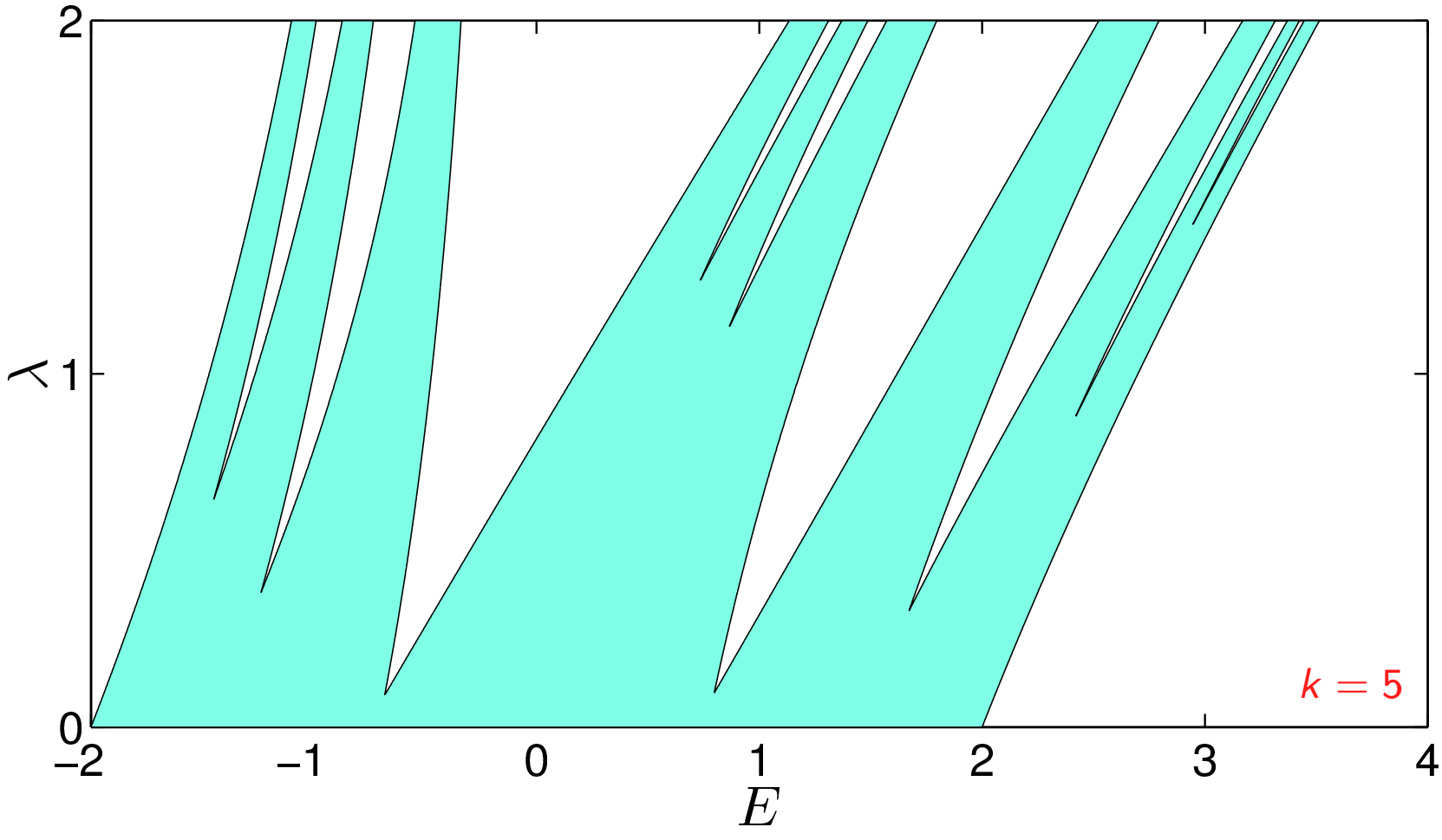
Upper Bounds on the Spectrum

Sütő's upper bound $\Sigma_{\lambda,k}$ as a function of λ.



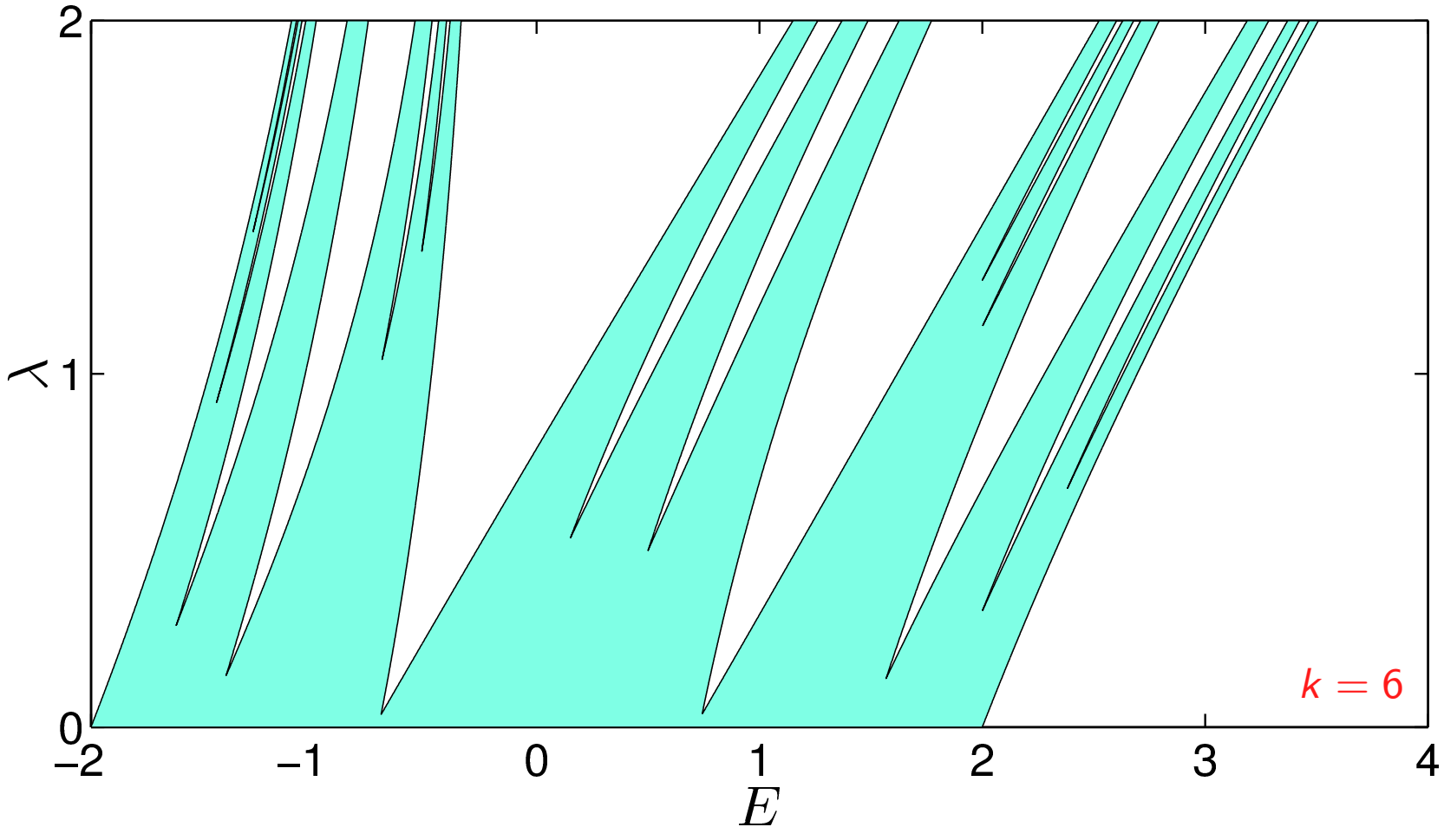
Upper Bounds on the Spectrum

Sütő's upper bound $\Sigma_{\lambda,k}$ as a function of λ.



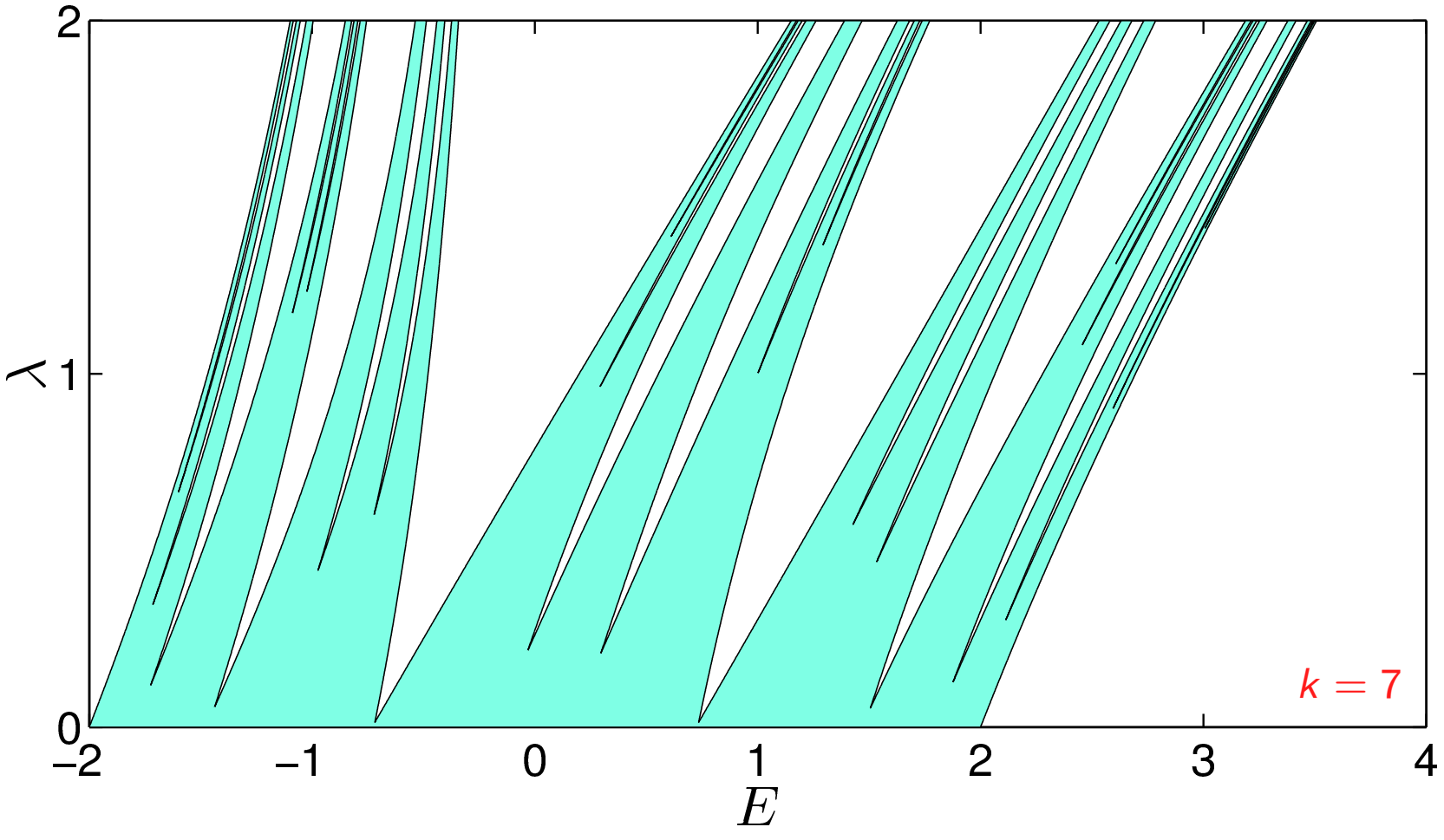
Upper Bounds on the Spectrum

Sütő's upper bound $\Sigma_{\lambda,k}$ as a function of λ.



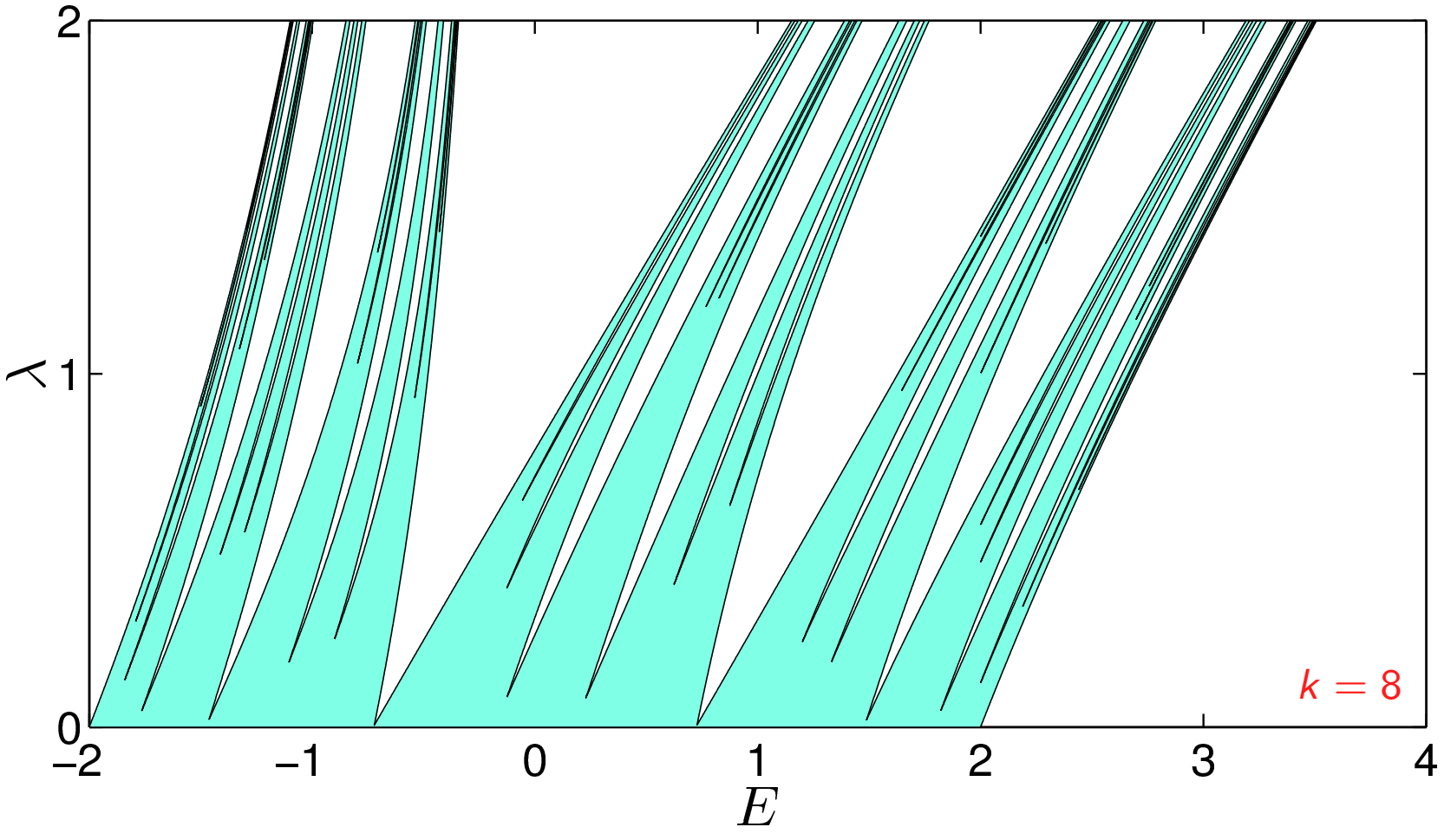
Upper Bounds on the Spectrum

Sütő's upper bound $\Sigma_{\lambda,k}$ as a function of λ.



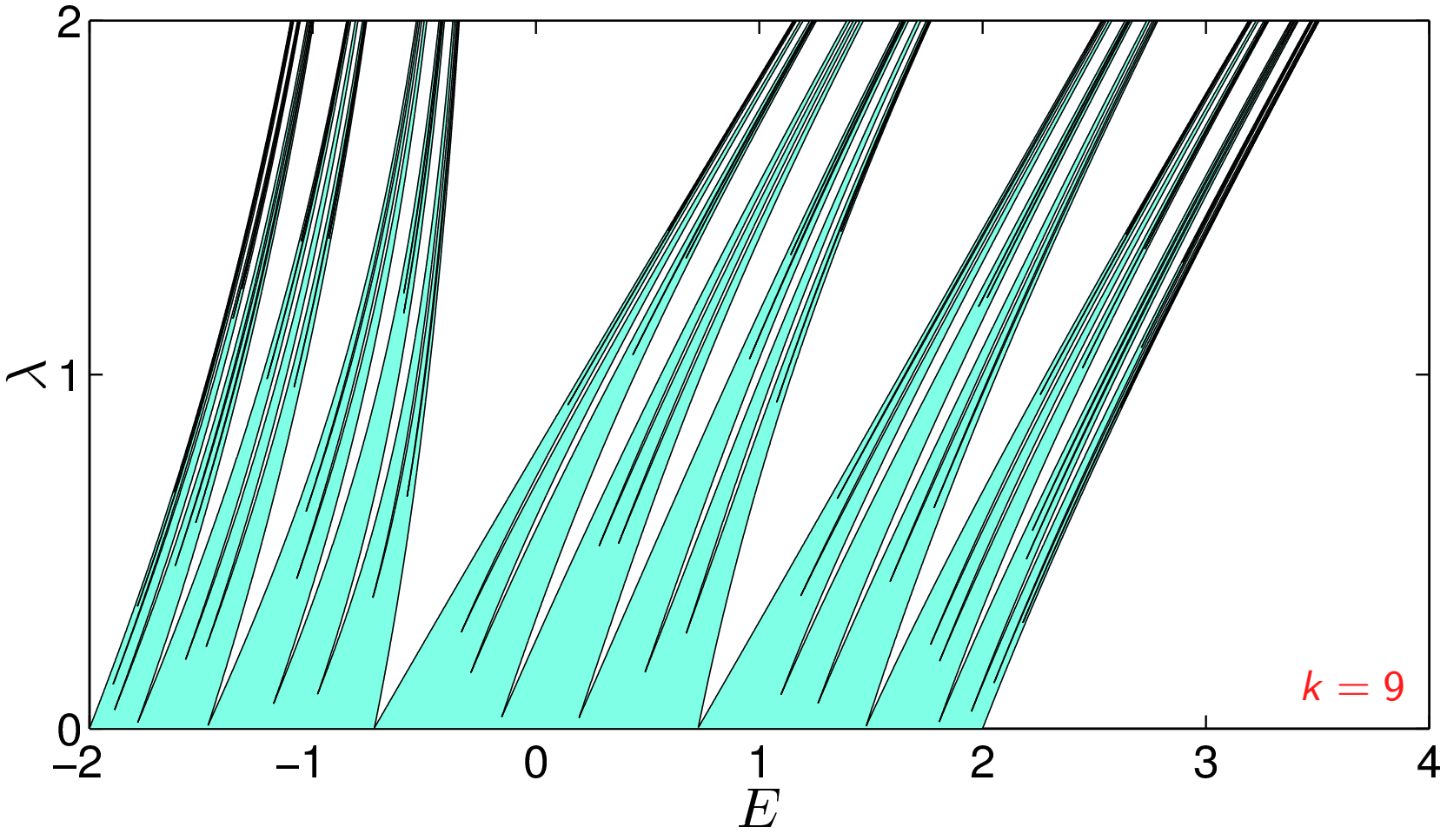
Upper Bounds on the Spectrum

Sütő's upper bound $\Sigma_{\lambda,k}$ as a function of λ.



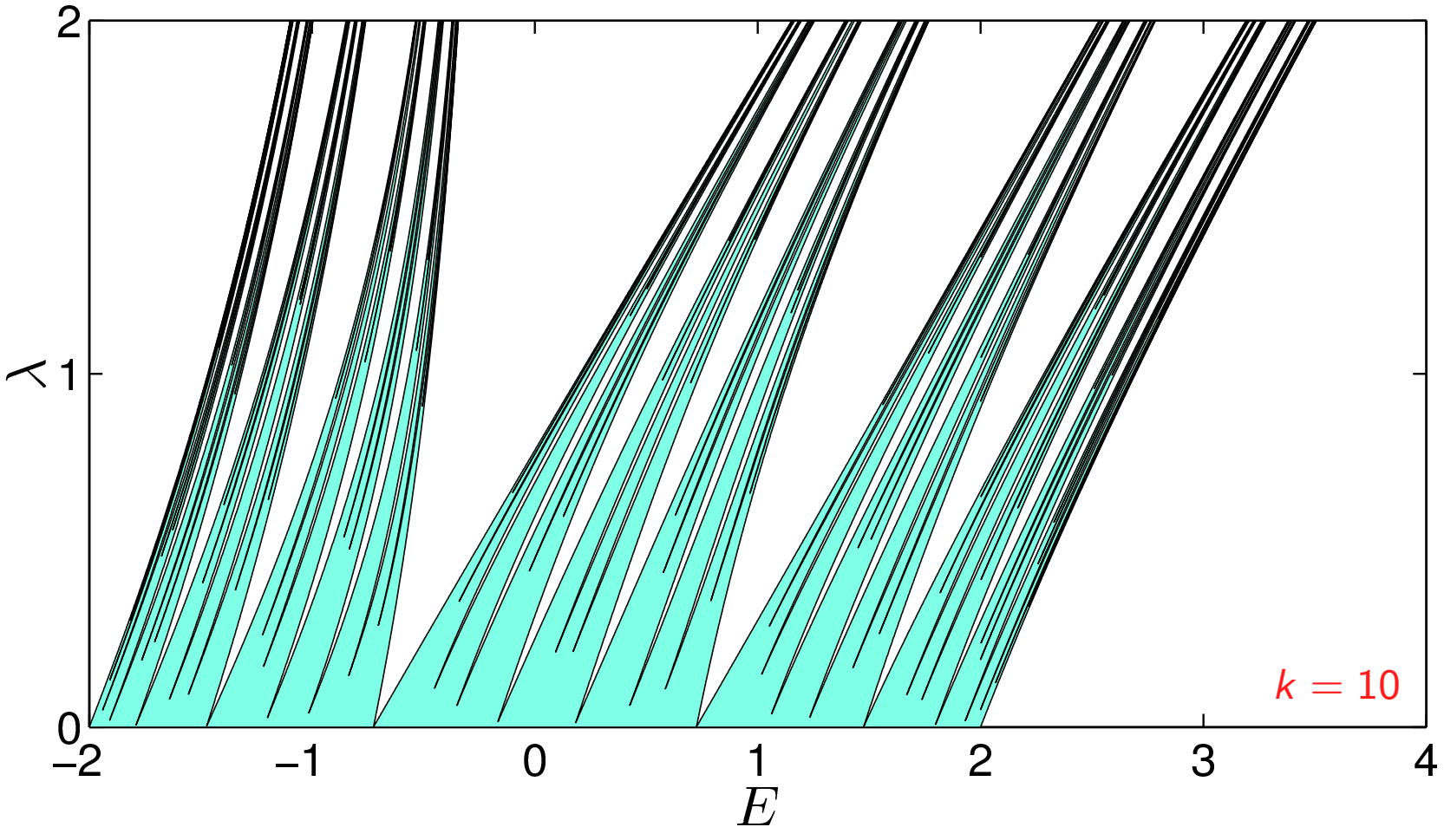
Upper Bounds on the Spectrum

Sütő's upper bound $\Sigma_{\lambda,k}$ as a function of λ.



Upper Bounds on the Spectrum

Sütő's upper bound $\Sigma_{\lambda,k}$ as a function of λ.



From Large Discretization Matrix to Small Projected Matrix

Projection to a Small Matrix

> *The size of the matrix problem will thus be much larger than the number of eigenvalues we are attempting to calculate. The matrix eigenvalue solver, a crucial component of the complete computational procedure, should therefore be designed to effectively find the low eigenvalues of large sparse, generalized matrix problems. . . . Because the extraction of the eigenvalues is very expensive, various "tricks" are used in engineering practice to reduce the sizes of the matrices under consideration.*
>
> — Babuška and Osborn, 1991

In the common situation where we seek only some subset of well-converged eigenvalues and eigenvectors, we wish to automatically compress $\mathbf{A}_N$ onto a subspace $\mathcal{V}_k$ that captures those eigenvectors:

$$\mathbf{H}_k := \mathbf{V}_k^* \mathbf{A}_N \mathbf{V}_k \in \mathbf{C}^{k \times k}$$

for $k \ll n$, where the columns of $\mathbf{V}_k$ for an orthonormal basis for the approximation. $\mathbf{H}_k$ is a *generalized Rayleigh quotient*.

Krylov Subspace Projection

The most fruitful automatic choices for $\mathcal{V}_k$ derive from *Krylov subspaces*,

$$\mathcal{K}_k(\mathbf{A}, \mathbf{v}) = \text{span}\{\mathbf{v}, \mathbf{A}\mathbf{v}, \ldots, \mathbf{A}^{k-1}\mathbf{v}\};$$

the basis comprises iterates of the power method [Lanczos, 1950], [Arnoldi, 1951].

Krylov Subspace Projection

The most fruitful automatic choices for $\mathcal{V}_k$ derive from *Krylov subspaces*,

$$\mathcal{K}_k(\mathbf{A}, \mathbf{v}) = \text{span}\{\mathbf{v}, \mathbf{A}\mathbf{v}, \ldots, \mathbf{A}^{k-1}\mathbf{v}\};$$

the basis comprises iterates of the power method [Lanczos, 1950], [Arnoldi, 1951].

Since Krylov subspaces are *shift invariant*,

$$\mathcal{K}_k(\mathbf{A} - \mu\mathbf{I}, \mathbf{v}) = \mathcal{K}_k(\mathbf{A}, \mathbf{v}),$$

we expect $\mathcal{K}_k(\mathbf{A}, \mathbf{v})$ to be rich in eigenvectors corresponding to any eigenvalues that can be made largest in magnitude by some choice of shift μ.

Krylov Subspace Projection

The most fruitful automatic choices for $\mathcal{V}_k$ derive from *Krylov subspaces*,

$$\mathcal{K}_k(\mathbf{A}, \mathbf{v}) = \text{span}\{\mathbf{v}, \mathbf{A}\mathbf{v}, \ldots, \mathbf{A}^{k-1}\mathbf{v}\};$$

the basis comprises iterates of the power method [Lanczos, 1950], [Arnoldi, 1951].

Since Krylov subspaces are *shift invariant*,

$$\mathcal{K}_k(\mathbf{A} - \mu\mathbf{I}, \mathbf{v}) = \mathcal{K}_k(\mathbf{A}, \mathbf{v}),$$

we expect $\mathcal{K}_k(\mathbf{A}, \mathbf{v})$ to be rich in eigenvectors corresponding to any eigenvalues that can be made largest in magnitude by some choice of shift μ.

Suppose we seek the eigenpair $(\lambda, \mathbf{u})$ of $\mathbf{A}$, and λ has multiplicity 1.
One can show the convergence obeys

$$\sin \angle(\mathbf{u}, \mathcal{K}_k(\mathbf{A}, \mathbf{v})) \leq \frac{1}{\|\mathbf{P}\mathbf{v}\|} \min_{\substack{\psi \in \mathcal{P}_{k-1} \\ \psi(\lambda)=1}} \|(\mathbf{I} - \mathbf{P})\psi(\mathbf{A})\|,$$

where $\mathbf{P}$ is the spectral projector associated with $(\lambda, \mathbf{u})$
[Saad, 1980]; cf. subspace bounds in [Beattie, E., Rossi, 2004].

Convergence of Krylov Subspace Projection

$$\sin \angle(\mathbf{u}, \mathcal{K}_k(\mathbf{A}, \mathbf{v})) \leq \frac{1}{\|\mathbf{P}\mathbf{v}\|} \min_{\substack{\psi \in \mathcal{P}_{k-1} \\ \psi(\lambda)=1}} \|(\mathbf{I} - \mathbf{P})\psi(\mathbf{A})\|,$$

Suppose $\mathbf{A}$ is Hermitian and we seek leftmost eigenvalue λ_1, where

$$\lambda_1 < \lambda_2 \leq \cdots \leq \lambda_N.$$

The error bound suggests the progress made at each iteration is like

$$\gamma := \frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1}, \qquad \text{where } \kappa := \frac{\lambda_N - \lambda_1}{\lambda_2 - \lambda_1}.$$

Convergence of Krylov Subspace Projection

$$\sin \angle(\mathbf{u}, \mathcal{K}_k(\mathbf{A}, \mathbf{v})) \leq \frac{1}{\|\mathbf{P}\mathbf{v}\|} \min_{\substack{\psi \in \mathcal{P}_{k-1} \\ \psi(\lambda)=1}} \|(\mathbf{I} - \mathbf{P})\psi(\mathbf{A})\|,$$

Suppose $\mathbf{A}$ is Hermitian and we seek leftmost eigenvalue λ_1, where

$$\lambda_1 < \lambda_2 \leq \cdots \leq \lambda_N.$$

The error bound suggests the progress made at each iteration is like

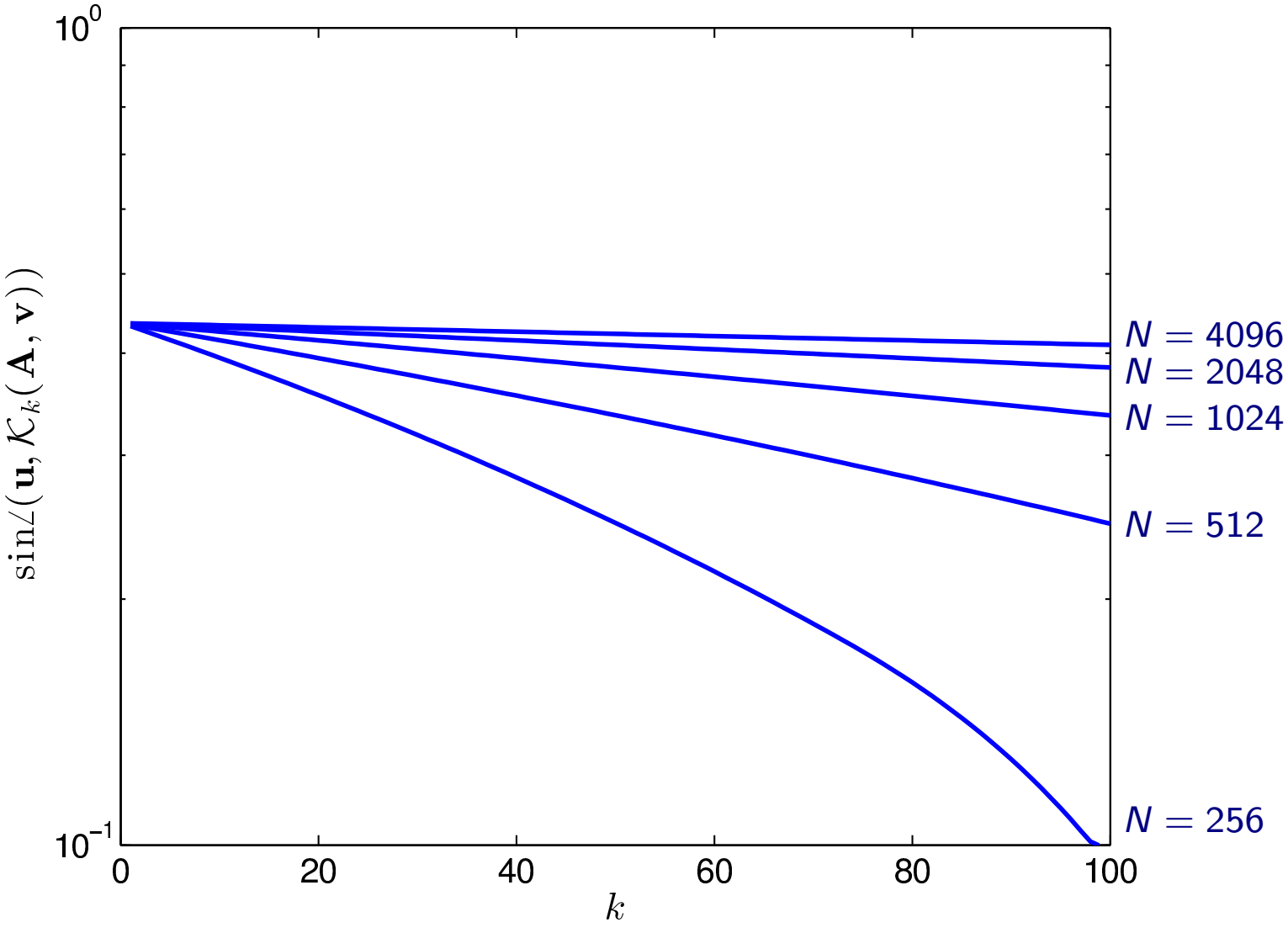
$$\gamma := \frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1}, \qquad \text{where } \kappa := \frac{\lambda_N - \lambda_1}{\lambda_2 - \lambda_1}.$$

When $\mathbf{A}$ discretizes an unbounded operator, we expect $\lambda_N = \|\mathbf{A}\| \to \infty$ as $N \to \infty$. Hence, *the convergence rate goes to zero as $N \to \infty$*.

This illustrates why Krylov subspace methods applied to $\mathbf{A}$ perform poorly for many PDE eigenvalue problems.

Convergence of Krylov Subspace Projection

We see this effect for the simple 1d Laplacian considered earlier.



Convergence of Krylov Subspace Projection

The problem becomes immediately apparent if we attempt to run Krylov subspace projection *on the operator itself*,

$$\mathcal{K}_k(A, v) = \text{span}\{v, Av, \ldots, A^{k-1}v\}.$$

For $Au = -u''$ with Dirichlet boundary conditions, $u(0) = u(1) = 1$, we choose a starting vector $v \in \text{Dom}(A)$, i.e.,

$$v(0) = v(1) = 0.$$

However, in general $Av \notin \text{Dom(A)}$, so we cannot build the next Krylov direction $A^2v = A(Av)$. The Lanczos algorithm breaks down at the third step.

Convergence of Krylov Subspace Projection

The problem becomes immediately apparent if we attempt to run Krylov subspace projection *on the operator itself*,

$$\mathcal{K}_k(A, v) = \text{span}\{v, Av, \ldots, A^{k-1}v\}.$$

For $Au = -u''$ with Dirichlet boundary conditions, $u(0) = u(1) = 1$, we choose a starting vector $v \in \text{Dom}(A)$, i.e.,

$$v(0) = v(1) = 0.$$

However, in general $Av \notin \text{Dom(A)}$, so we cannot build the next Krylov direction $A^2v = A(Av)$. The Lanczos algorithm breaks down at the third step.

The operator setting suggests that we instead apply Lanczos to A^{-1}:

$$\mathcal{K}_k(A^{-1}, v) = \text{span}\{v, A^{-1}v, \ldots, A^{-(k-1)}v\}.$$

In this case, A^{-1} is a beautiful compact operator, and

$$(A^{-1}v)(x) = \iint v + C_0 + C_1 x,$$

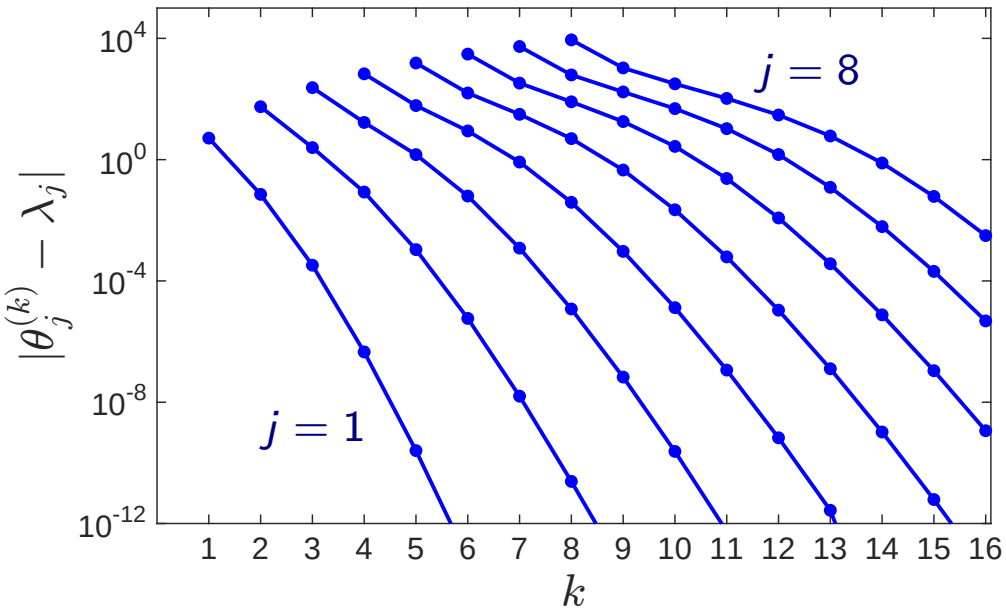
where we choose C_0 and C_1 so that

$$(A^{-1}v)(0) = (A^{-1}v)(1) = 0.$$

Convergence of Krylov Subspace Projection

We run the Lanczos algorithm on A^{-1} exactly in Mathematica.
Denote the eigenvalues of $\mathbf{H}_k$ as

$$\theta_1^{(k)} \leq \theta_2^{(k)} \leq \cdots \leq \theta_k^{(k)}.$$

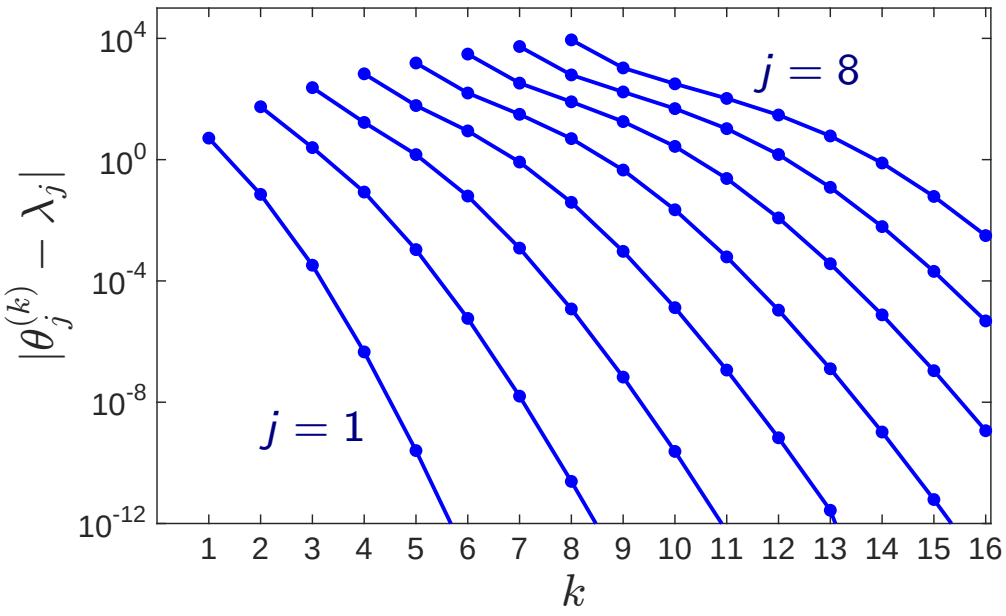


Cf. [Winther, 1980], [Nevanlinna, 1993], [Moret, 1997], [Olver, 2009] for CG and GMRES applied to operators, and [Kirby, 2010] for the Riesz map setting.

Convergence of Krylov Subspace Projection

We run the Lanczos algorithm on A^{-1} exactly in Mathematica.
Denote the eigenvalues of $\mathbf{H}_k$ as

$$\theta_1^{(k)} \leq \theta_2^{(k)} \leq \cdots \leq \theta_k^{(k)}.$$



Cf. [Winther, 1980], [Nevanlinna, 1993], [Moret, 1997], [Olver, 2009] for CG and GMRES applied to operators, and [Kirby, 2010] for the Riesz map setting.

This mode of computation is preferred for discretization matrices as well: the *shift-invert Arnoldi method* uses

$$\mathcal{K}_k((\mathbf{A} - \mu\mathbf{I})^{-1}, \mathbf{v}) = \text{span}\{\mathbf{v}, (\mathbf{A} - \mu\mathbf{I})^{-1}\mathbf{v}, \ldots, (\mathbf{A} - \mu\mathbf{I})^{-(k-1)}\mathbf{v}\}.$$

From Small Projected Matrix to Computed Eigenvalues

Computation of Eigenvalues for Small Matrices

Robust software exists for the solution of the symmetric and nonsymmetric eigenvalue problem. LAPACK software provides gold-standard implementations of the best algorithms.

Aside from a few lingering issues (e.g., superb orthogonality of eigenvectors for symmetric matrices; accelerating convergence via aggressive deflation), the dense eigenvalue problem is mainly regarded as a solved problem.

However, the main outstanding challenge is critical to many applications: *the high relative accuracy of small eigenvalues*.

There is a broad literature on this subject, e.g., [Demmel, Veselić, Drmač, Slapničar, . . .] In the context of differential operators, see [Qiang Ye, 2009. . .].

Standard Eigensolvers and Relative Accuracy

Discretize $Au = -u''$ on $[0,1]$ with $u(0) = u(1) = 0$ using second order finite differences with mesh size $h = 1/(N+1)$:

$$\mathbf{A}_N = \frac{1}{h^2} \begin{bmatrix} 2 & -1 & & \\ -1 & 2 & \ddots & \\ & \ddots & 2 & -1 \\ & & -1 & 2 \end{bmatrix}.$$

Standard Eigensolvers and Relative Accuracy

Discretize $Au = -u''$ on $[0,1]$ with $u(0) = u(1) = 0$ using second order finite differences with mesh size $h = 1/(N+1)$:

$$\mathbf{A}_N = \frac{1}{h^2}\begin{bmatrix} 2 & -1 & & \\ -1 & 2 & \ddots & \\ & \ddots & 2 & -1 \\ & & -1 & 2 \end{bmatrix}.$$

The eigenvalues of this matrix are well-known:

$$\lambda_{k,N} = 2h^{-2}\Big(1 - \cos\Big(k\pi h\Big)\Big), \qquad k = 1, \ldots, N.$$

For small k,

$$\lambda_{k,N} = k^2\pi^2 + \mathcal{O}(h^2),$$

approximating to $\mathcal{O}(h^2)$ the true eigenvalue $\lambda_k = k^2\pi^2$ of A.

Standard Eigensolvers and Relative Accuracy

Discretize $Au = -u''$ on $[0,1]$ with $u(0) = u(1) = 0$ using second order finite differences with mesh size $h = 1/(N+1)$:

$$\mathbf{A}_N = \frac{1}{h^2}\begin{bmatrix} 2 & -1 & & \\ -1 & 2 & \ddots & \\ & \ddots & 2 & -1 \\ & & -1 & 2 \end{bmatrix}.$$

The eigenvalues of this matrix are well-known:

$$\lambda_{k,N} = 2h^{-2}\Big(1 - \cos\Big(k\pi h\Big)\Big), \qquad k = 1, \ldots, N.$$

For small k,

$$\lambda_{k,N} = k^2\pi^2 + \mathcal{O}(h^2),$$

approximating to $\mathcal{O}(h^2)$ the true eigenvalue $\lambda_k = k^2\pi^2$ of A.

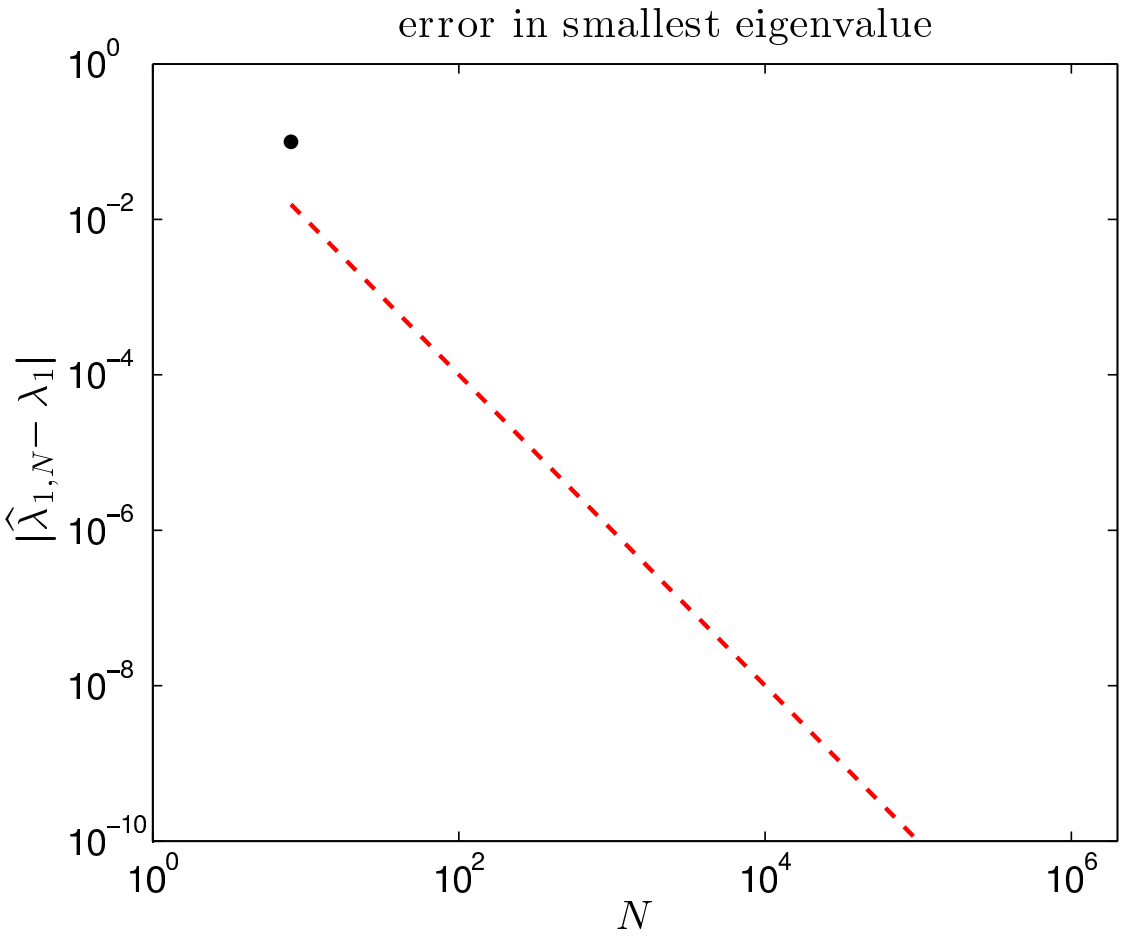
Do numerical computations recover this expected error?

Standard Eigensolvers and Relative Accuracy

Use ARPACK (via `eigs` in MATLAB) to compute the smallest eigenvalue

$$\lambda_{1,N} = \frac{2}{h^2}(1 - \cos(\pi h)) = \pi^2 + \mathcal{O}(h^2).$$

Denote the eigenvalue `eigs` computes as $\widehat{\lambda}_{1,N}$.

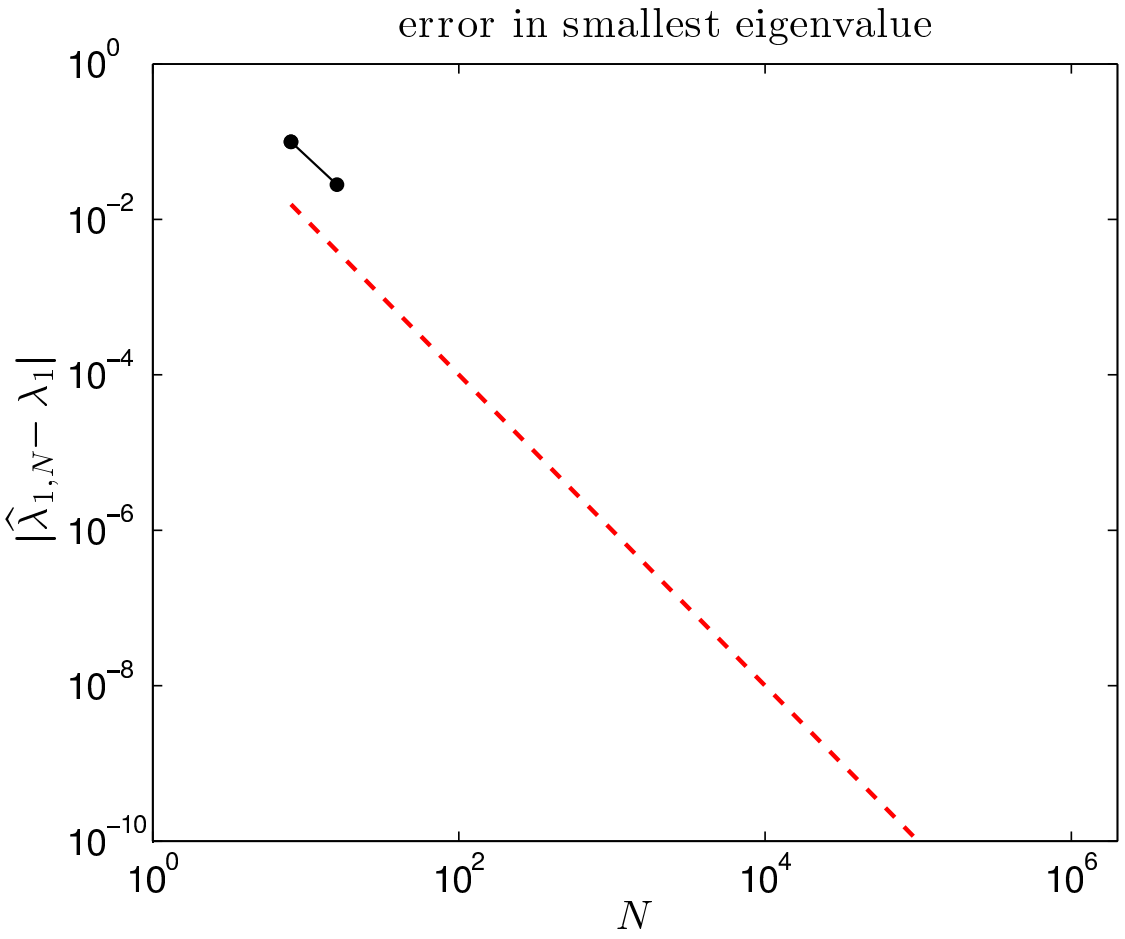


Standard Eigensolvers and Relative Accuracy

Use ARPACK (via `eigs` in MATLAB) to compute the smallest eigenvalue

$$\lambda_{1,N} = \frac{2}{h^2}(1 - \cos(\pi h)) = \pi^2 + \mathcal{O}(h^2).$$

Denote the eigenvalue `eigs` computes as $\widehat{\lambda}_{1,N}$.

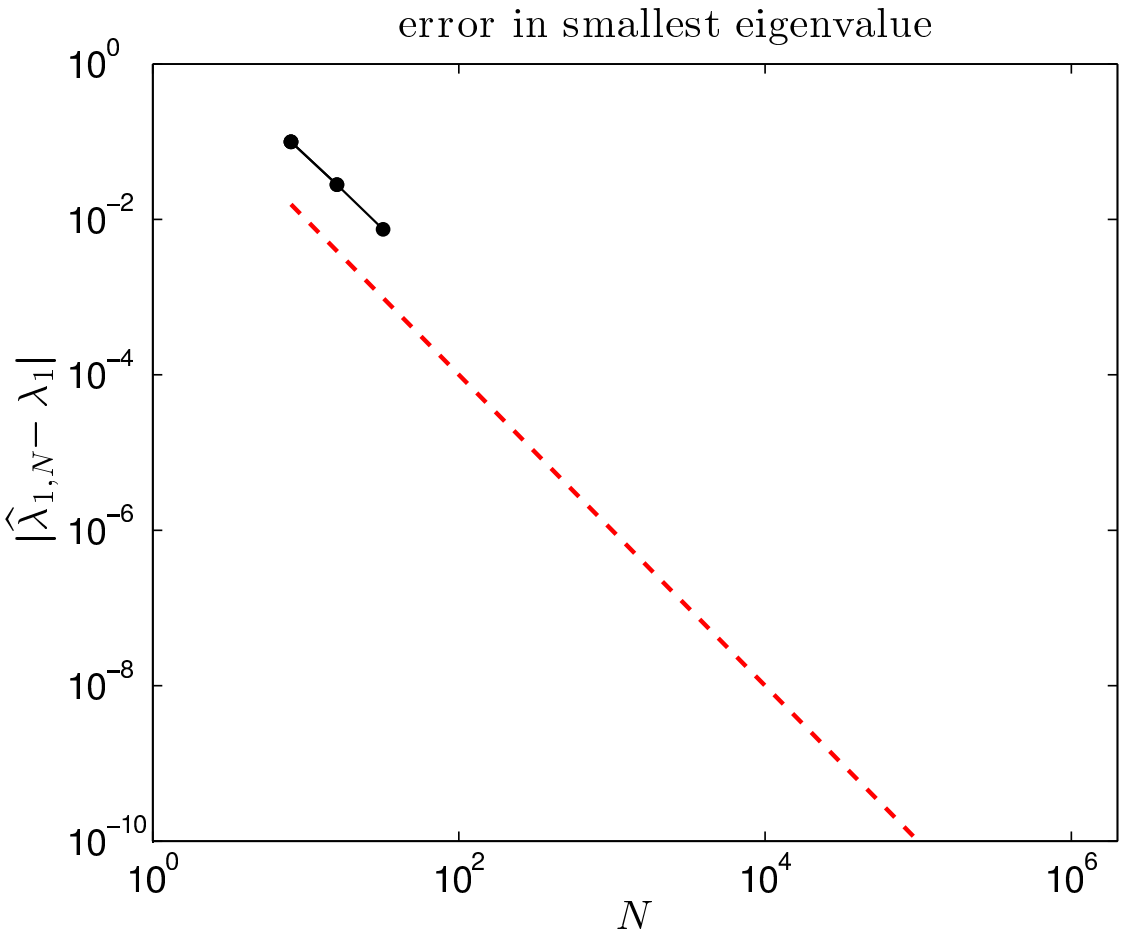


Standard Eigensolvers and Relative Accuracy

Use ARPACK (via `eigs` in MATLAB) to compute the smallest eigenvalue

$$\lambda_{1,N} = \frac{2}{h^2}(1 - \cos(\pi h)) = \pi^2 + \mathcal{O}(h^2).$$

Denote the eigenvalue `eigs` computes as $\widehat{\lambda}_{1,N}$.

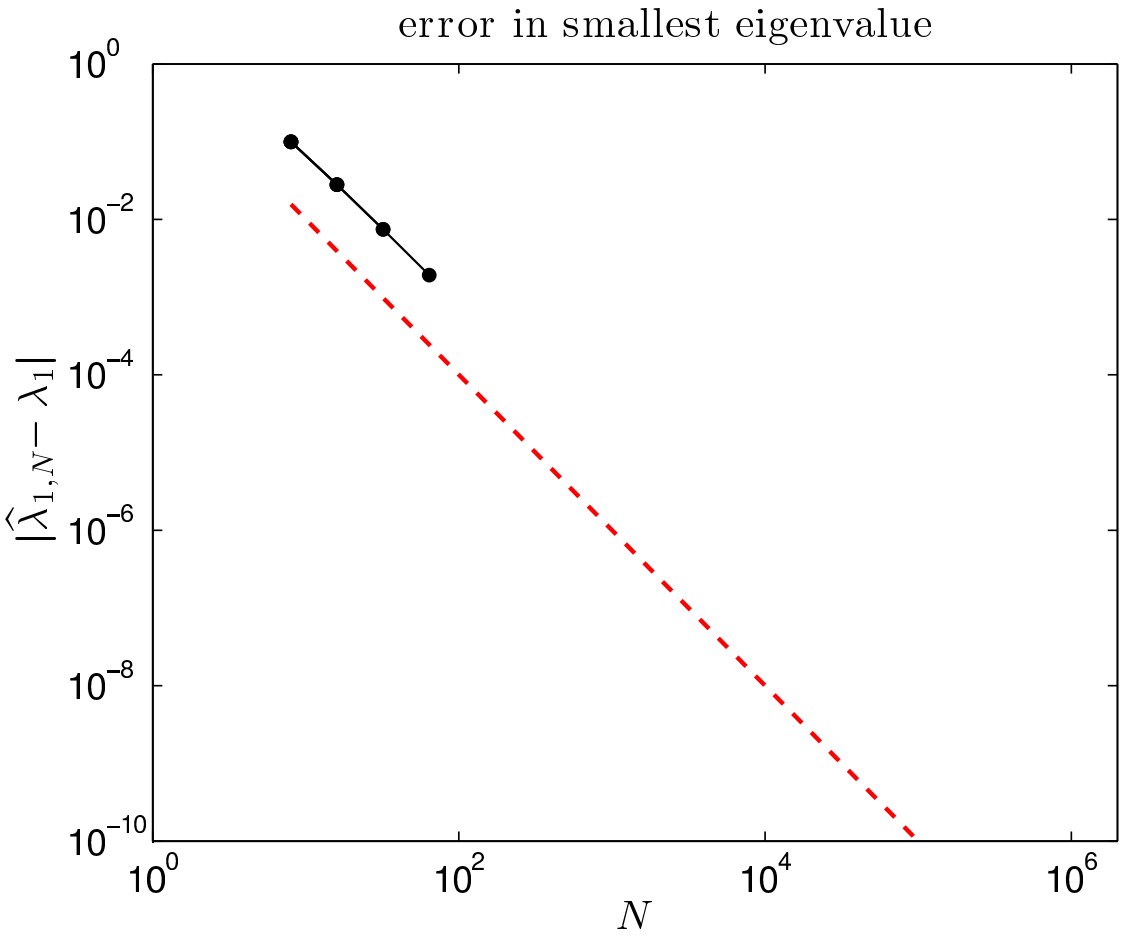


Standard Eigensolvers and Relative Accuracy

Use ARPACK (via `eigs` in MATLAB) to compute the smallest eigenvalue

$$\lambda_{1,N} = \frac{2}{h^2}(1 - \cos(\pi h)) = \pi^2 + \mathcal{O}(h^2).$$

Denote the eigenvalue `eigs` computes as $\widehat{\lambda}_{1,N}$.

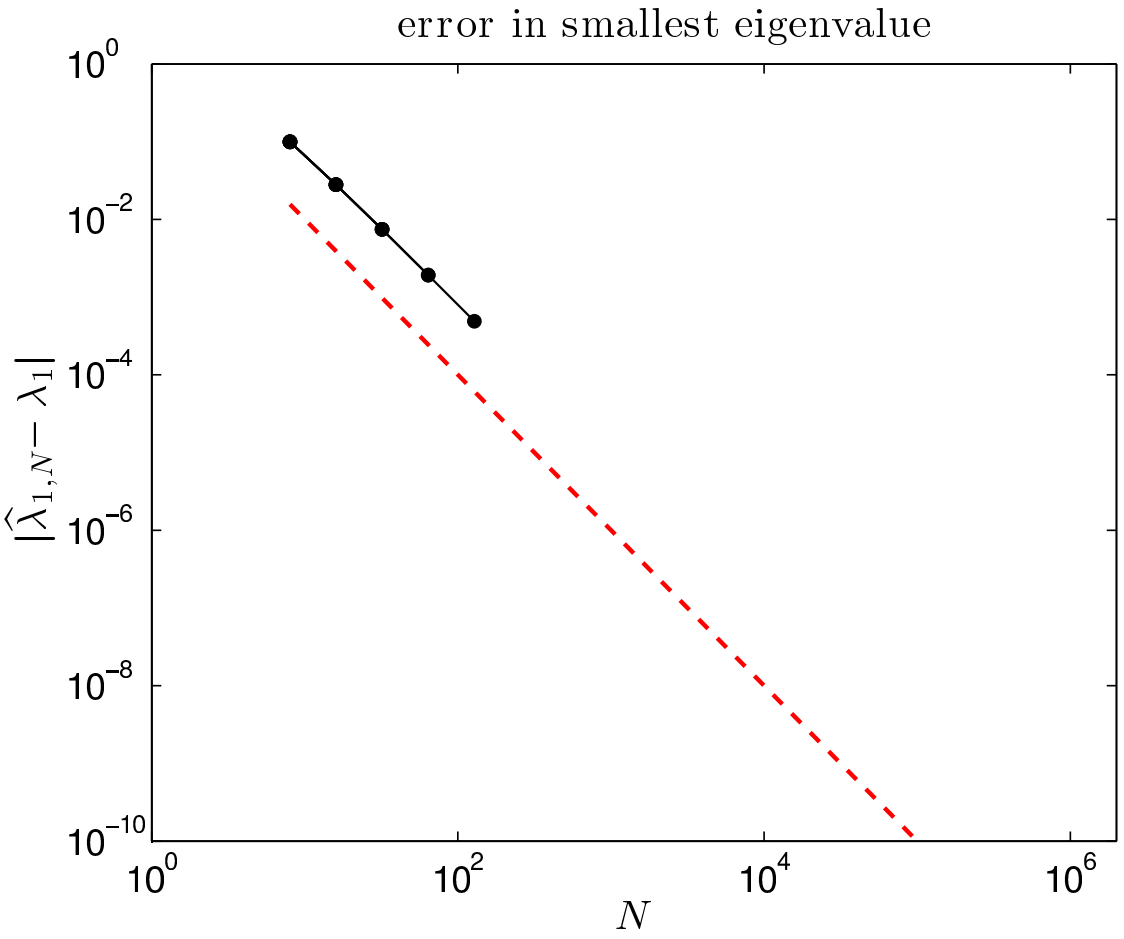


Standard Eigensolvers and Relative Accuracy

Use ARPACK (via `eigs` in MATLAB) to compute the smallest eigenvalue

$$\lambda_{1,N} = \frac{2}{h^2}(1 - \cos(\pi h)) = \pi^2 + \mathcal{O}(h^2).$$

Denote the eigenvalue `eigs` computes as $\widehat{\lambda}_{1,N}$.

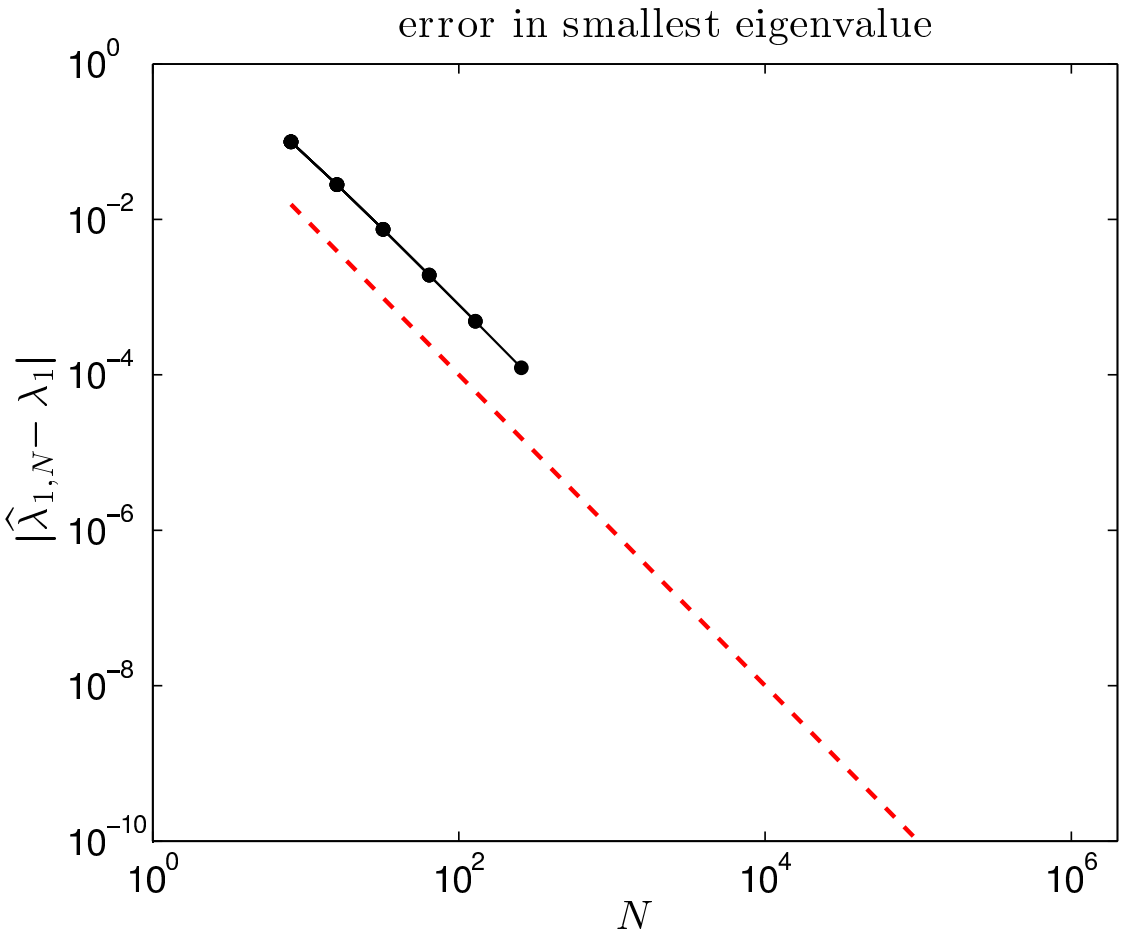


Standard Eigensolvers and Relative Accuracy

Use ARPACK (via `eigs` in MATLAB) to compute the smallest eigenvalue

$$\lambda_{1,N} = \frac{2}{h^2}(1 - \cos(\pi h)) = \pi^2 + \mathcal{O}(h^2).$$

Denote the eigenvalue `eigs` computes as $\widehat{\lambda}_{1,N}$.

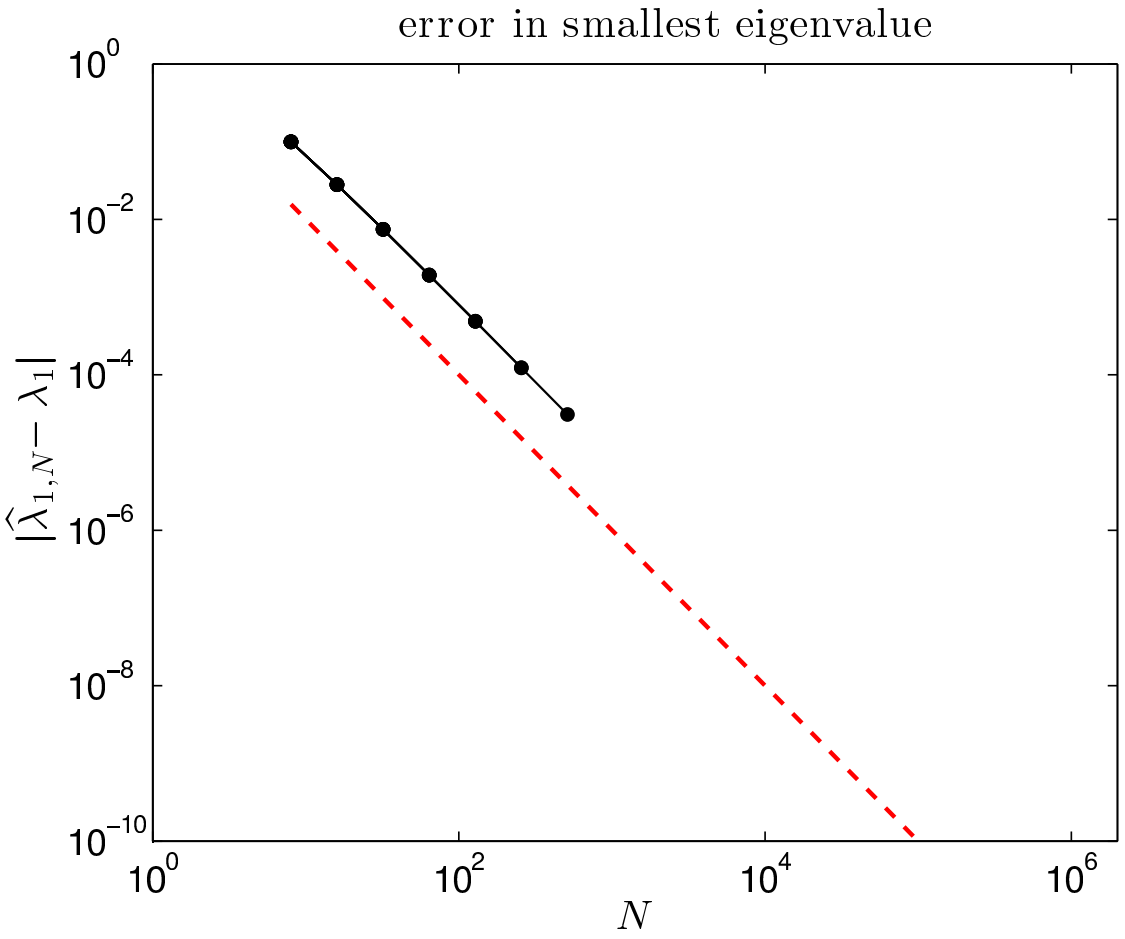


Standard Eigensolvers and Relative Accuracy

Use ARPACK (via `eigs` in MATLAB) to compute the smallest eigenvalue

$$\lambda_{1,N} = \frac{2}{h^2}(1 - \cos(\pi h)) = \pi^2 + \mathcal{O}(h^2).$$

Denote the eigenvalue `eigs` computes as $\widehat{\lambda}_{1,N}$.

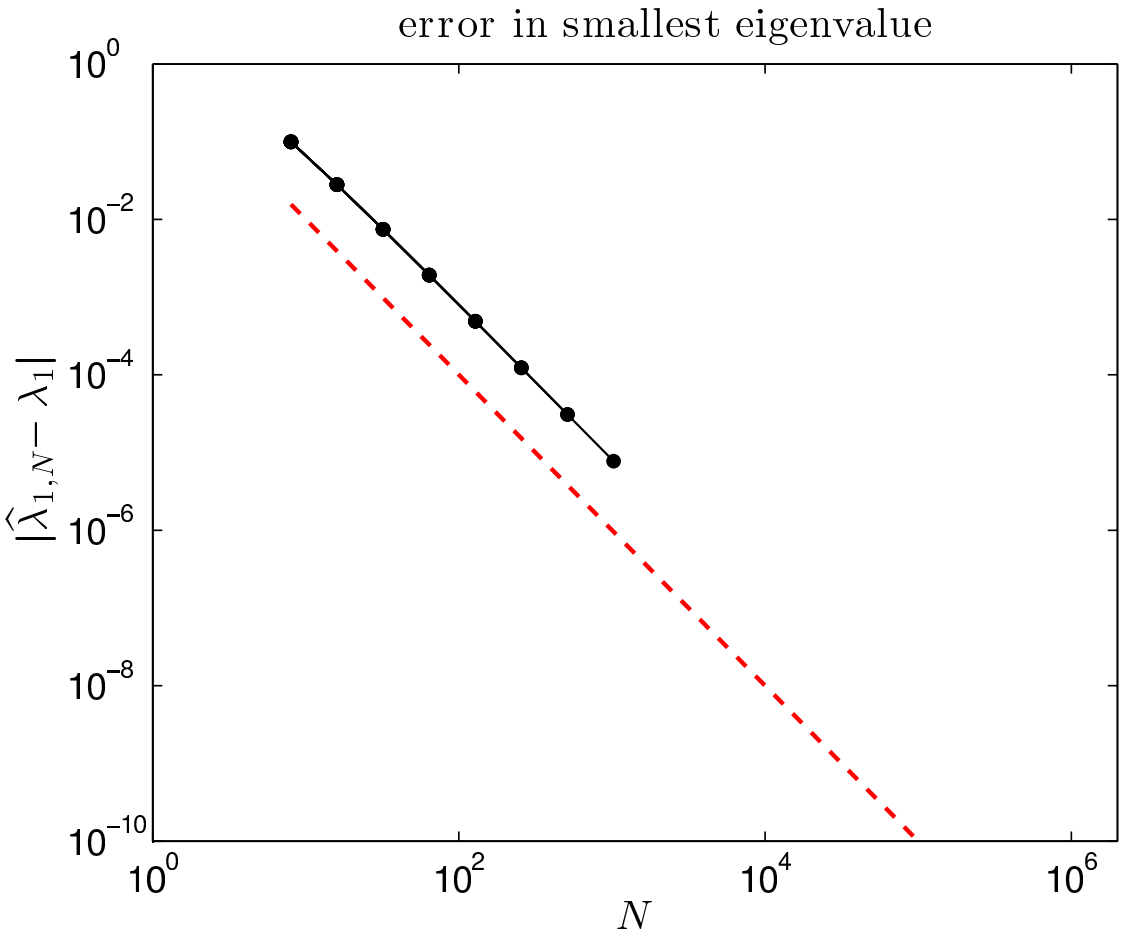


Standard Eigensolvers and Relative Accuracy

Use ARPACK (via `eigs` in MATLAB) to compute the smallest eigenvalue

$$\lambda_{1,N} = \frac{2}{h^2}(1 - \cos(\pi h)) = \pi^2 + \mathcal{O}(h^2).$$

Denote the eigenvalue `eigs` computes as $\widehat{\lambda}_{1,N}$.

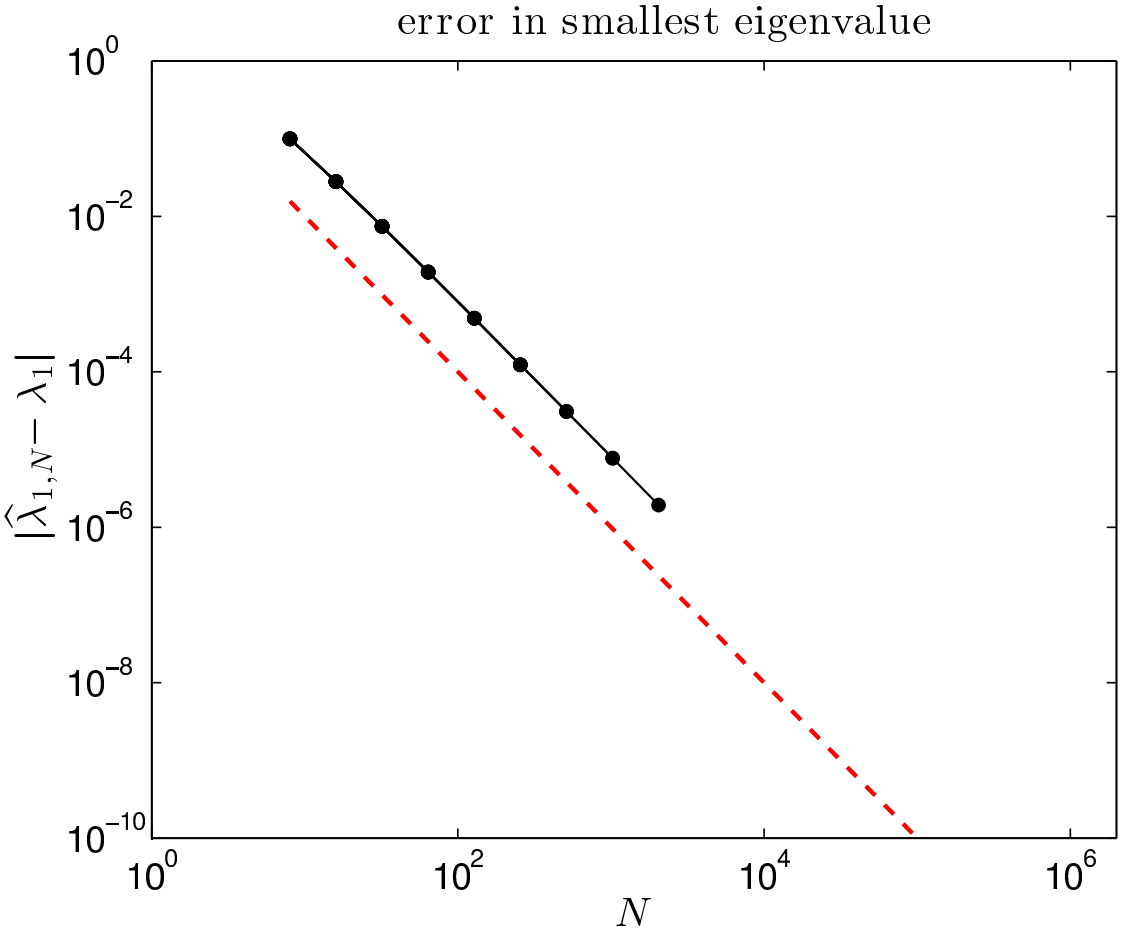


Standard Eigensolvers and Relative Accuracy

Use ARPACK (via `eigs` in MATLAB) to compute the smallest eigenvalue

$$\lambda_{1,N} = \frac{2}{h^2}(1 - \cos(\pi h)) = \pi^2 + \mathcal{O}(h^2).$$

Denote the eigenvalue `eigs` computes as $\widehat{\lambda}_{1,N}$.

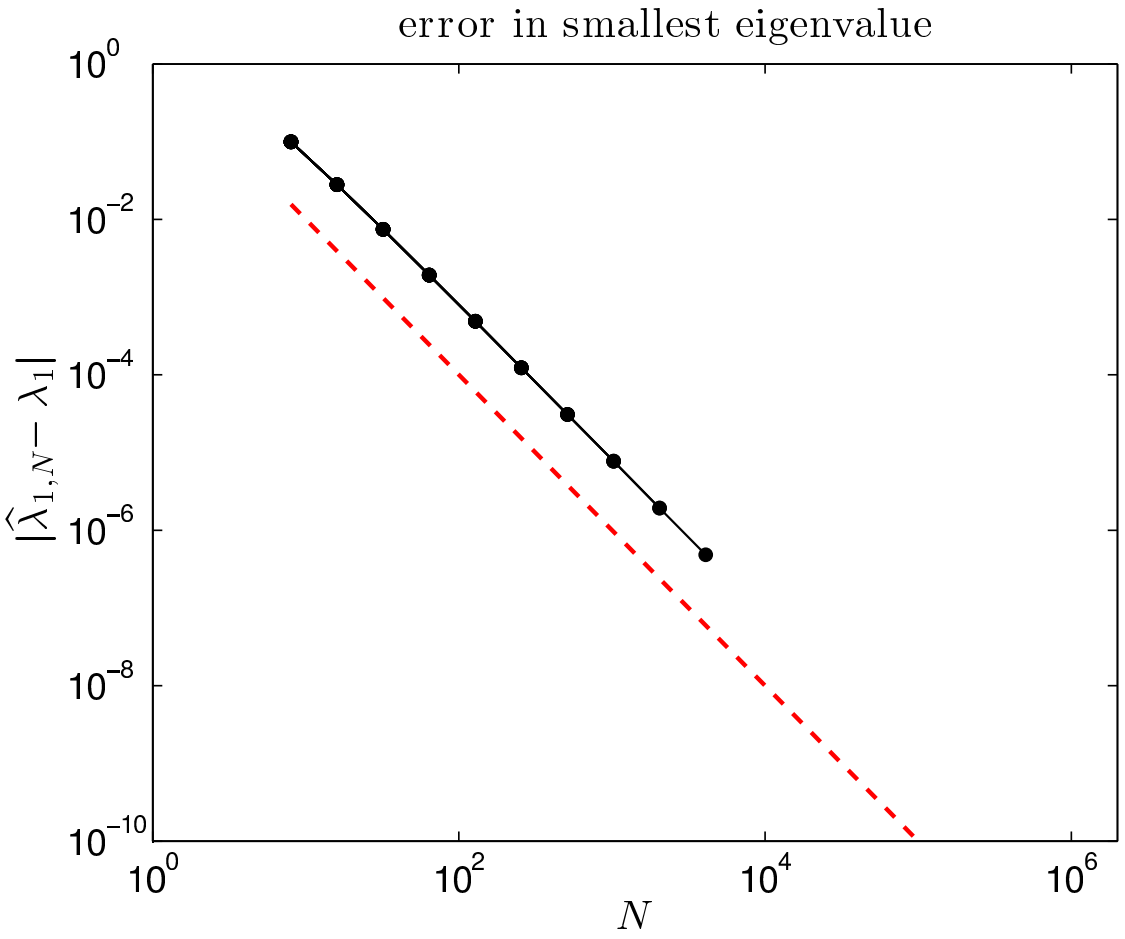


Standard Eigensolvers and Relative Accuracy

Use ARPACK (via `eigs` in MATLAB) to compute the smallest eigenvalue

$$\lambda_{1,N} = \frac{2}{h^2}(1 - \cos(\pi h)) = \pi^2 + \mathcal{O}(h^2).$$

Denote the eigenvalue `eigs` computes as $\widehat{\lambda}_{1,N}$.

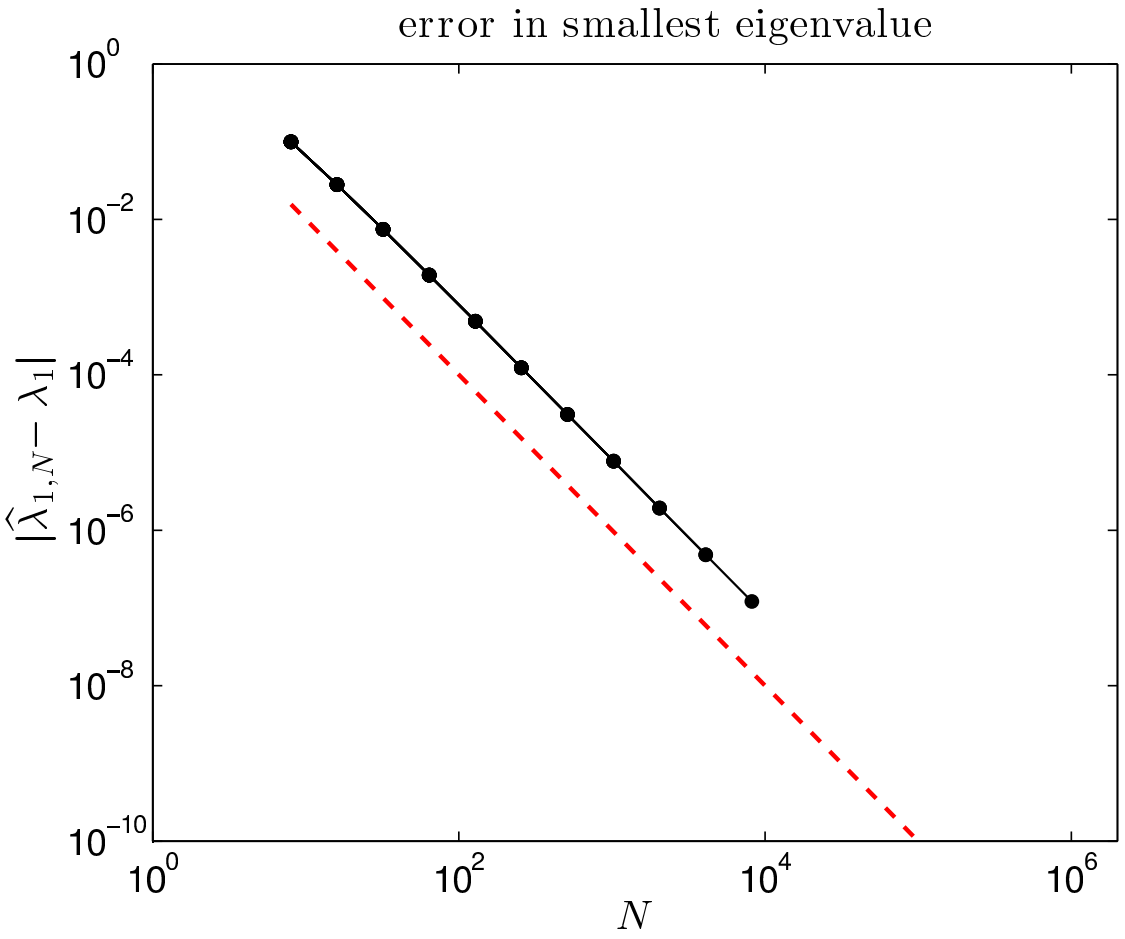


Standard Eigensolvers and Relative Accuracy

Use ARPACK (via `eigs` in MATLAB) to compute the smallest eigenvalue

$$\lambda_{1,N} = \frac{2}{h^2}(1 - \cos(\pi h)) = \pi^2 + \mathcal{O}(h^2).$$

Denote the eigenvalue `eigs` computes as $\widehat{\lambda}_{1,N}$.

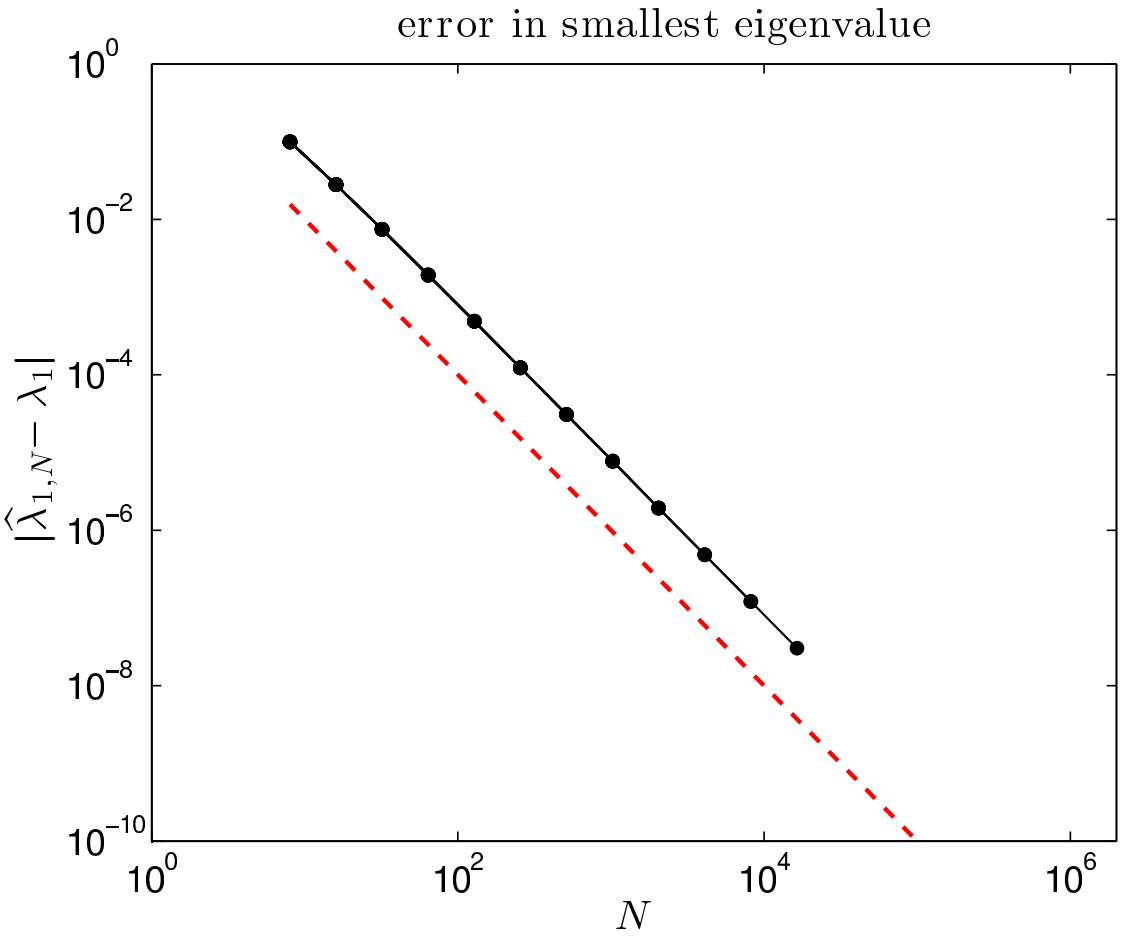


Standard Eigensolvers and Relative Accuracy

Use ARPACK (via `eigs` in MATLAB) to compute the smallest eigenvalue

$$\lambda_{1,N} = \frac{2}{h^2}(1 - \cos(\pi h)) = \pi^2 + \mathcal{O}(h^2).$$

Denote the eigenvalue `eigs` computes as $\widehat{\lambda}_{1,N}$.

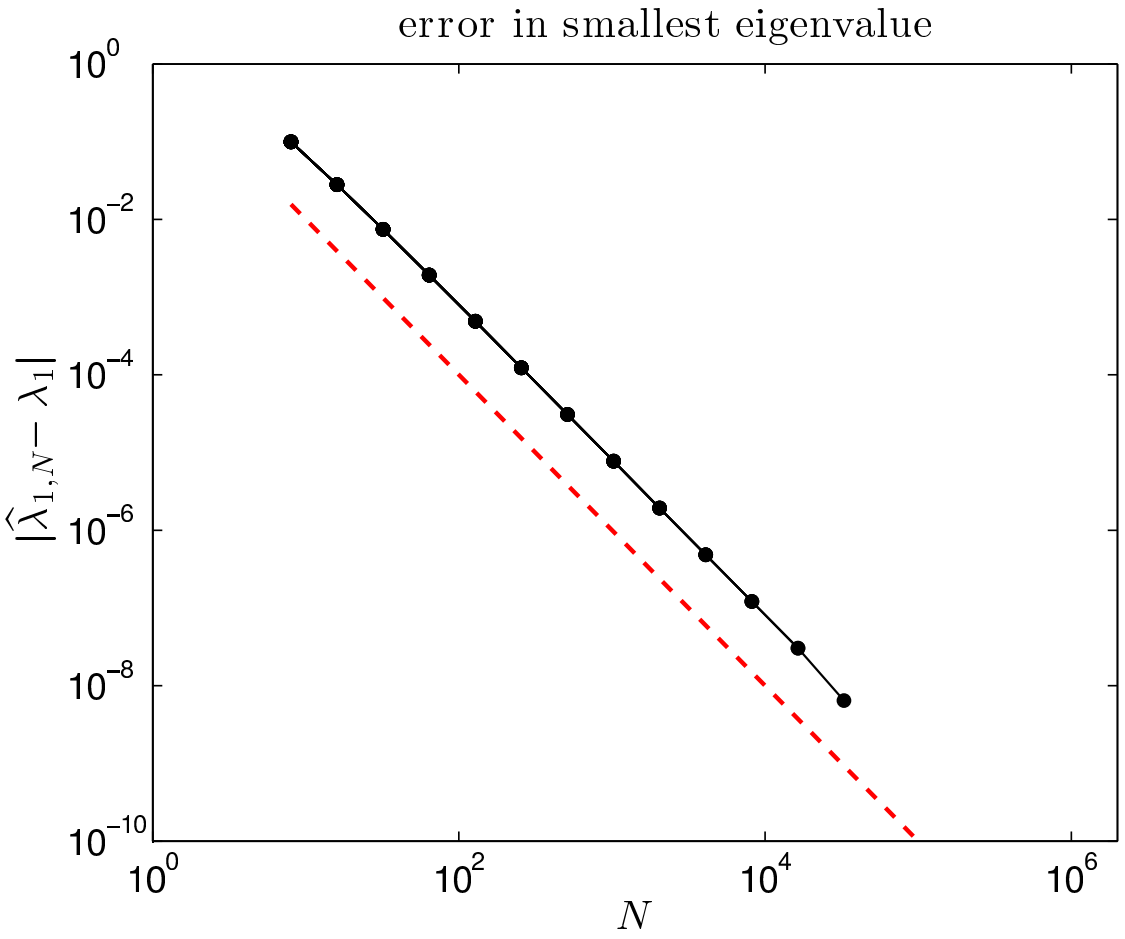


Standard Eigensolvers and Relative Accuracy

Use ARPACK (via `eigs` in MATLAB) to compute the smallest eigenvalue

$$\lambda_{1,N} = \frac{2}{h^2}(1 - \cos(\pi h)) = \pi^2 + \mathcal{O}(h^2).$$

Denote the eigenvalue `eigs` computes as $\widehat{\lambda}_{1,N}$.

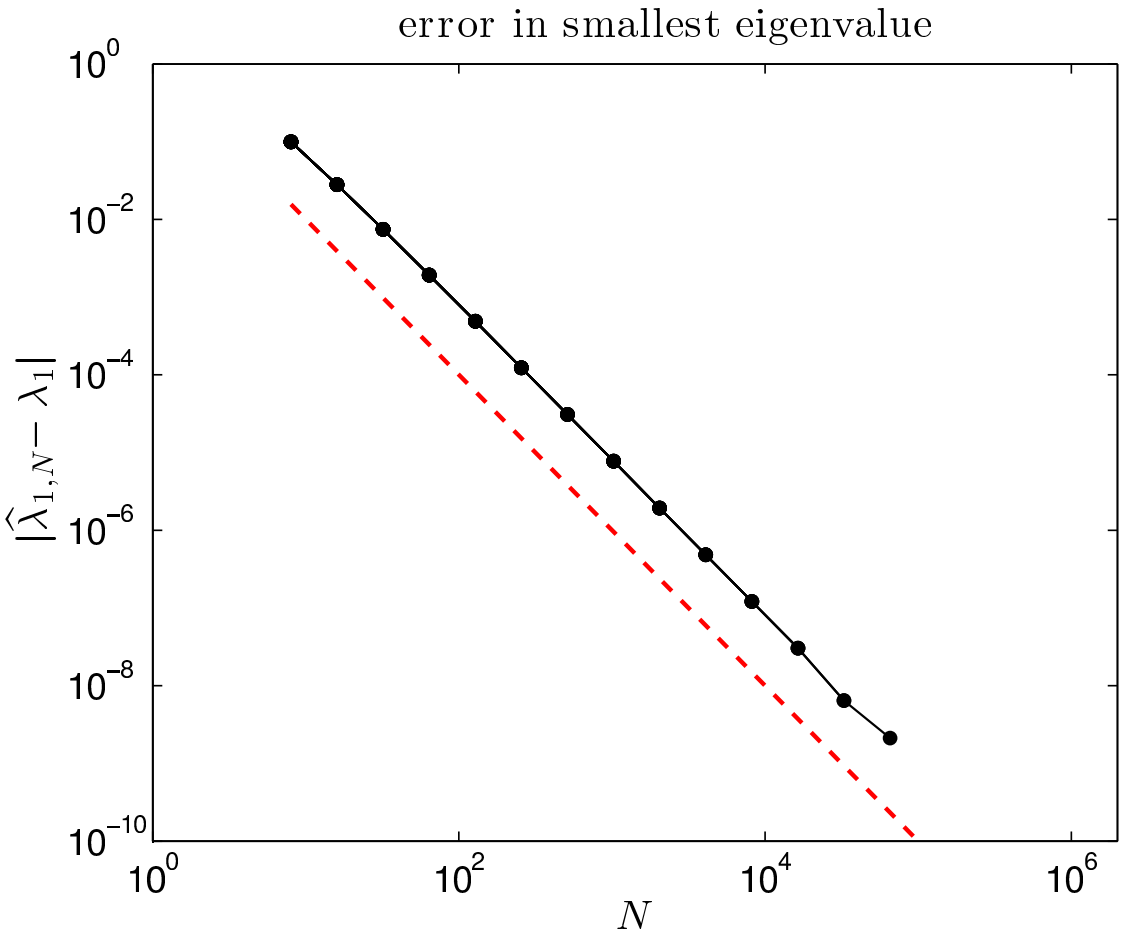


Standard Eigensolvers and Relative Accuracy

Use ARPACK (via `eigs` in MATLAB) to compute the smallest eigenvalue

$$\lambda_{1,N} = \frac{2}{h^2}(1 - \cos(\pi h)) = \pi^2 + \mathcal{O}(h^2).$$

Denote the eigenvalue `eigs` computes as $\widehat{\lambda}_{1,N}$.

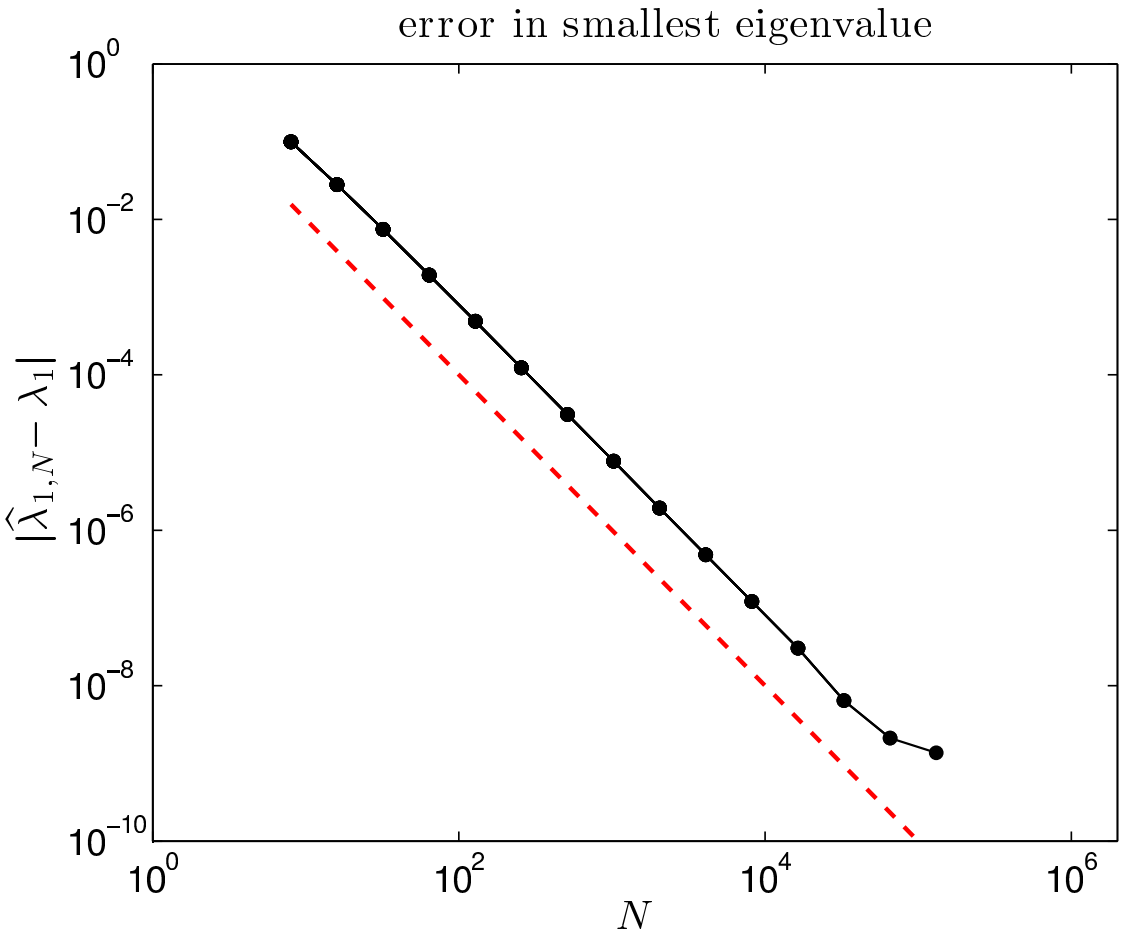


Standard Eigensolvers and Relative Accuracy

Use ARPACK (via `eigs` in MATLAB) to compute the smallest eigenvalue

$$\lambda_{1,N} = \frac{2}{h^2}(1 - \cos(\pi h)) = \pi^2 + \mathcal{O}(h^2).$$

Denote the eigenvalue `eigs` computes as $\widehat{\lambda}_{1,N}$.

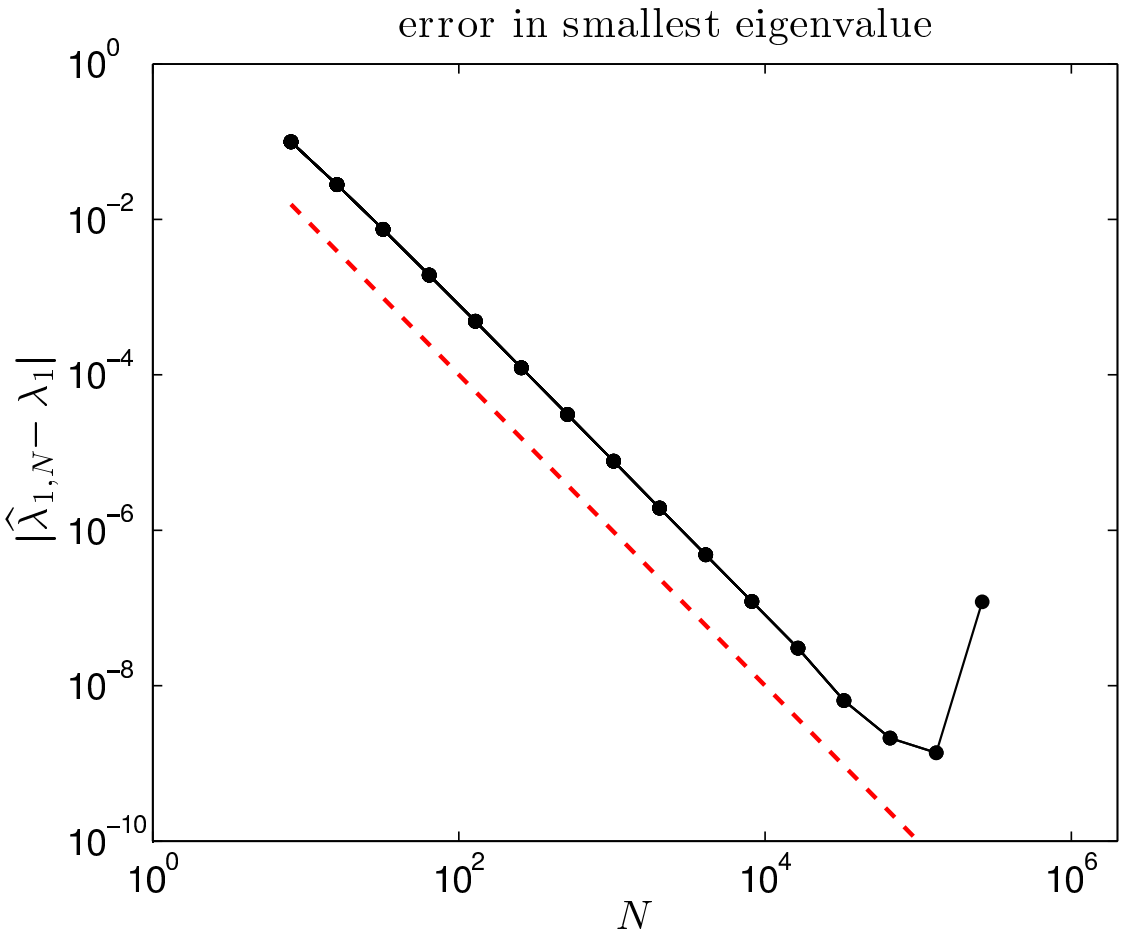


Standard Eigensolvers and Relative Accuracy

Use ARPACK (via `eigs` in MATLAB) to compute the smallest eigenvalue

$$\lambda_{1,N} = \frac{2}{h^2}(1 - \cos(\pi h)) = \pi^2 + \mathcal{O}(h^2).$$

Denote the eigenvalue `eigs` computes as $\widehat{\lambda}_{1,N}$.

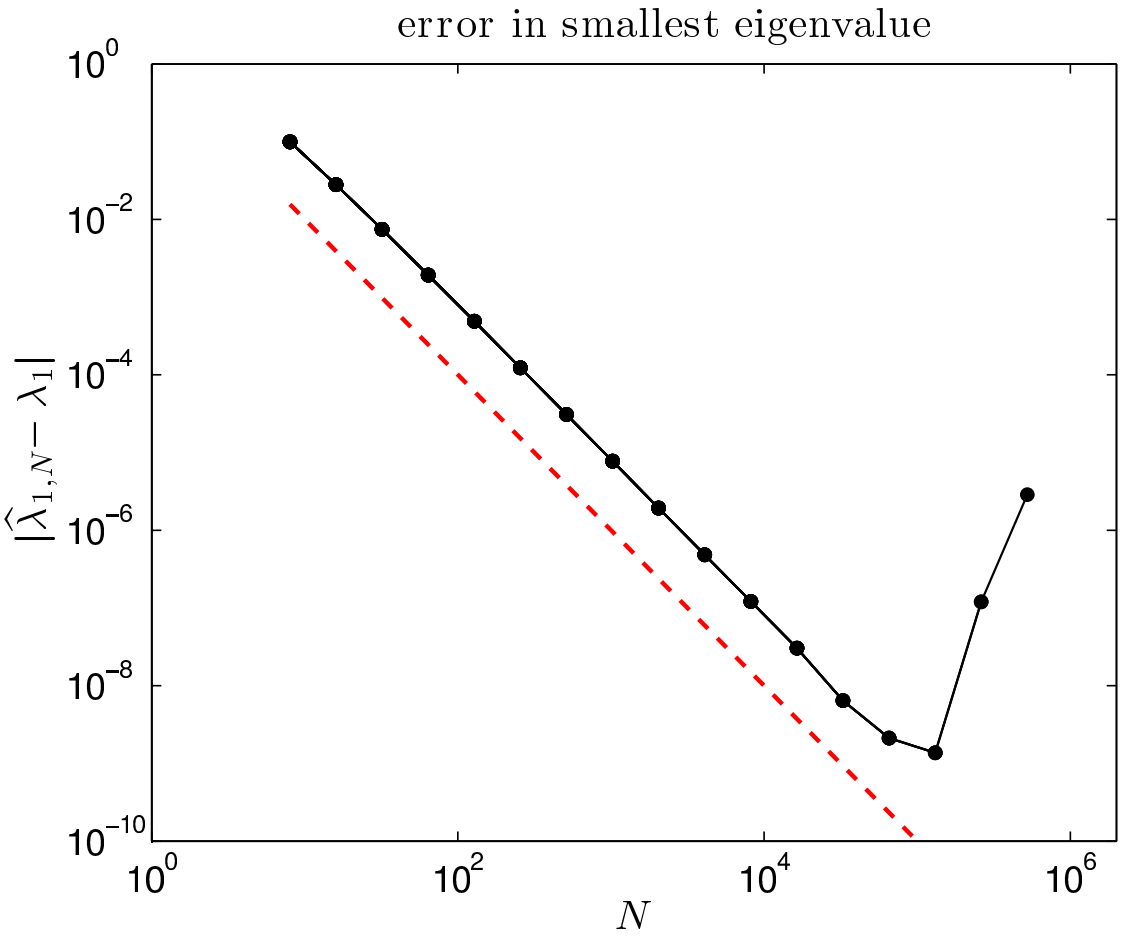


Standard Eigensolvers and Relative Accuracy

Use ARPACK (via `eigs` in MATLAB) to compute the smallest eigenvalue

$$\lambda_{1,N} = \frac{2}{h^2}(1 - \cos(\pi h)) = \pi^2 + \mathcal{O}(h^2).$$

Denote the eigenvalue `eigs` computes as $\widehat{\lambda}_{1,N}$.

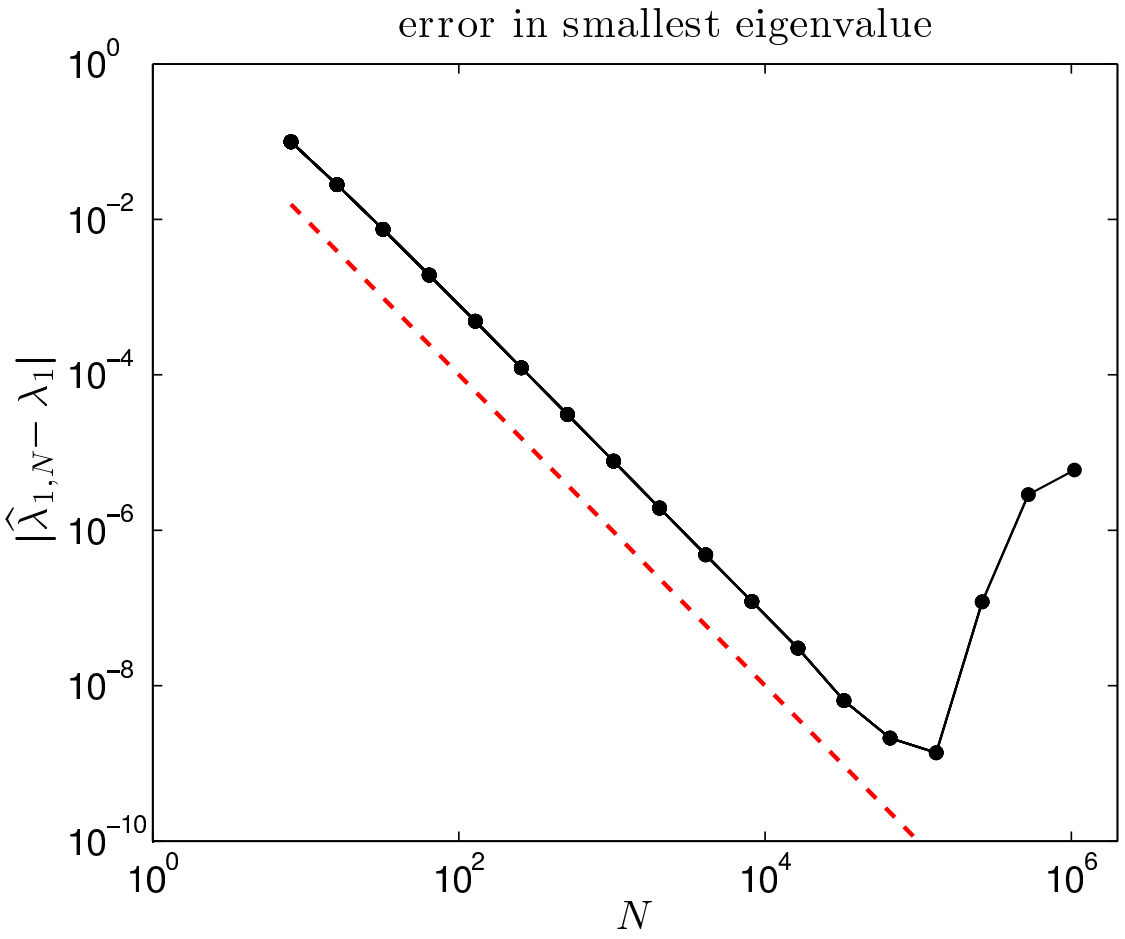


Standard Eigensolvers and Relative Accuracy

Use ARPACK (via `eigs` in MATLAB) to compute the smallest eigenvalue

$$\lambda_{1,N} = \frac{2}{h^2}(1 - \cos(\pi h)) = \pi^2 + \mathcal{O}(h^2).$$

Denote the eigenvalue `eigs` computes as $\widehat{\lambda}_{1,N}$.

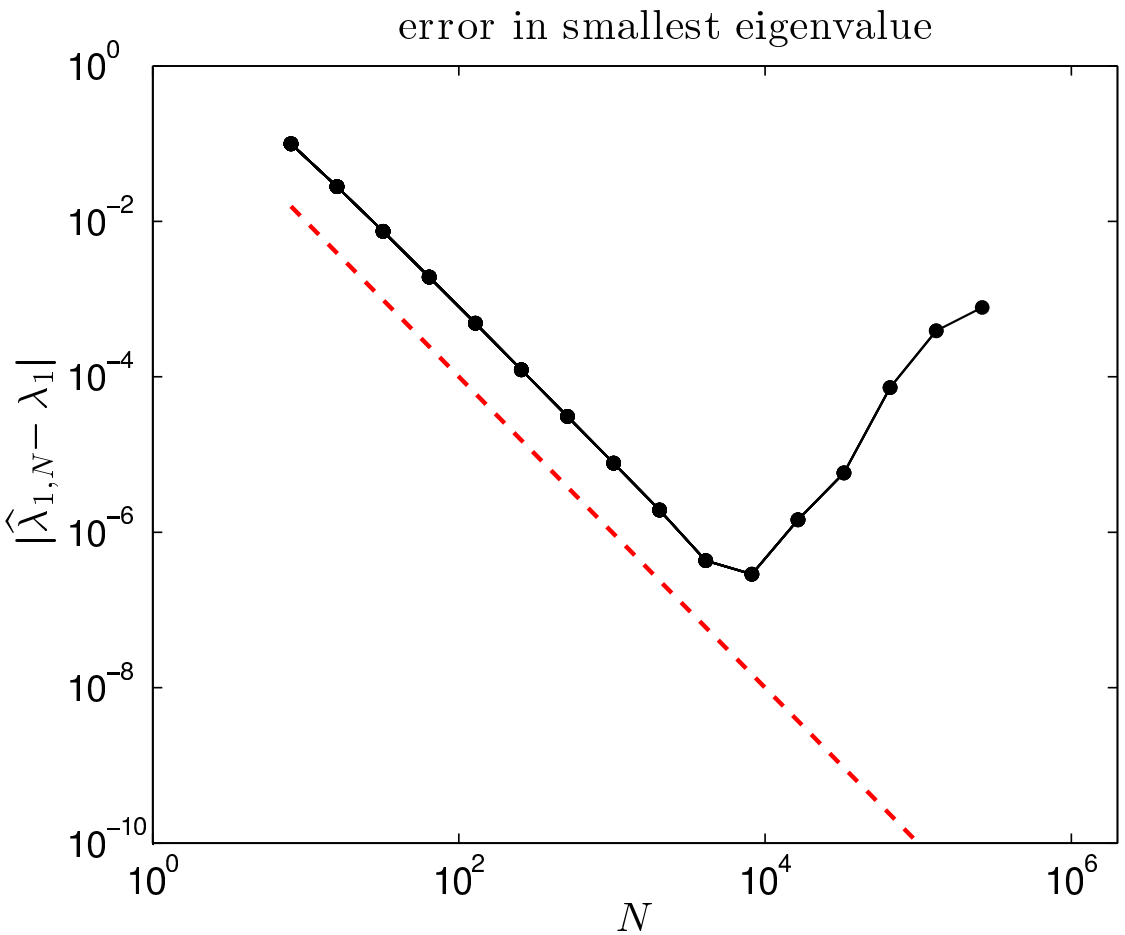


Standard Eigensolvers and Relative Accuracy

Repeat the same experiment, using `eig` (a dense eigensolver from LAPACK) to compute the smallest magnitude eigenvalue

$$\lambda_{1,N} = \frac{2}{h^2}(1 - \cos(\pi h)) = \pi^2 + \mathcal{O}(h^2).$$

Now denote this computed eigenvalue as $\widehat{\lambda}_{1,N}$.



Standard Eigensolvers and Relative Accuracy

Where do these errors come from? Standard (sparse and dense) eigensolvers only compute eigenvalues to a *high absolute accuracy*. We have no guarantee that the eigenvalues have *high relative accuracy*.

More precisely, standard dense symmetric eigensolvers yield approximations $\widehat{\lambda}_{1,N}$ to $\lambda_{1,N}$ that obey

$$|\lambda_{1,N} - \widehat{\lambda}_{1,N}| \leq c_N \, \varepsilon_{\text{mach}} \|\mathbf{A}_N\|,$$

where $\varepsilon_{\text{mach}}$ is *machine epsilon* (2.2×10^{-16} for double precision), and c_N is a modest N-dependent constant.

Standard Eigensolvers and Relative Accuracy

Where do these errors come from? Standard (sparse and dense) eigensolvers only compute eigenvalues to a *high absolute accuracy*. We have no guarantee that the eigenvalues have *high relative accuracy*.

More precisely, standard dense symmetric eigensolvers yield approximations $\widehat{\lambda}_{1,N}$ to $\lambda_{1,N}$ that obey

$$|\lambda_{1,N} - \widehat{\lambda}_{1,N}| \leq c_N \, \varepsilon_{\text{mach}} \|\mathbf{A}_N\|,$$

where $\varepsilon_{\text{mach}}$ is *machine epsilon* (2.2×10^{-16} for double precision), and c_N is a modest N-dependent constant. For our problem,

$$\|\mathbf{A}_N\| = \lambda_{N,N} = \frac{4}{h^2} - \pi^2 + \mathcal{O}(h^2).$$

(We are, after all, approximating an unbounded differential operator.)

Standard Eigensolvers and Relative Accuracy

Where do these errors come from? Standard (sparse and dense) eigensolvers only compute eigenvalues to a *high absolute accuracy*. We have no guarantee that the eigenvalues have *high relative accuracy*.

More precisely, standard dense symmetric eigensolvers yield approximations $\widehat{\lambda}_{1,N}$ to $\lambda_{1,N}$ that obey

$$|\lambda_{1,N} - \widehat{\lambda}_{1,N}| \leq c_N \, \varepsilon_{\text{mach}} \|\mathbf{A}_N\|,$$

where $\varepsilon_{\text{mach}}$ is *machine epsilon* (2.2×10^{-16} for double precision), and c_N is a modest N-dependent constant. For our problem,

$$\|\mathbf{A}_N\| = \lambda_{N,N} = \frac{4}{h^2} - \pi^2 + \mathcal{O}(h^2).$$

(We are, after all, approximating an unbounded differential operator.)

Meanwhile, the convergence theory for the discretization gives

$$|\lambda_1 - \lambda_{1,N}| = \frac{1}{12}\pi^4 h^2 + \mathcal{O}(h^4).$$

Standard Eigensolvers and Relative Accuracy

Collecting these observations

$$|\lambda_{1,N} - \widehat{\lambda}_{1,N}| \leq c_N \, \varepsilon_{\text{mach}} \|\mathbf{A}_N\|$$

$$\|\mathbf{A}_N\| = \lambda_{N,N} = \frac{2}{h^2}(1 - \cos(\pi N h)) = \frac{4}{h^2} - \pi^2 + \mathcal{O}(h^2)$$

$$|\lambda_1 - \lambda_{1,N}| = \frac{1}{12}\pi^4 h^2 + \mathcal{O}(h^4)$$

we find an explanation for the poor convergence.

The error between the *computed* $\widehat{\lambda}_{1,N}$ and the *desired* λ_1 is bounded by:

$$|\lambda_1 - \widehat{\lambda}_{1,N}| \quad \leq \quad |\lambda_1 - \lambda_{1,N}| + |\lambda_{1,N} - \widehat{\lambda}_{1,N}|$$

Standard Eigensolvers and Relative Accuracy

Collecting these observations

$$|\lambda_{1,N} - \widehat{\lambda}_{1,N}| \leq c_N \varepsilon_{\text{mach}} \|\mathbf{A}_N\|$$

$$\|\mathbf{A}_N\| = \lambda_{N,N} = \frac{2}{h^2}(1 - \cos(\pi N h)) = \frac{4}{h^2} - \pi^2 + \mathcal{O}(h^2)$$

$$|\lambda_1 - \lambda_{1,N}| = \frac{1}{12}\pi^4 h^2 + \mathcal{O}(h^4)$$

we find an explanation for the poor convergence.

The error between the *computed* $\widehat{\lambda}_{1,N}$ and the *desired* λ_1 is bounded by:

$$\begin{aligned} |\lambda_1 - \widehat{\lambda}_{1,N}| &\leq |\lambda_1 - \lambda_{1,N}| + |\lambda_{1,N} - \widehat{\lambda}_{1,N}| \\ &\leq \left(\frac{1}{12}\pi^4 h^2 + \mathcal{O}(h^4)\right) + c_N \varepsilon_{\text{mach}} \left(\frac{4}{h^2} - \pi^2 + \mathcal{O}(h^2)\right) \end{aligned}$$

Standard Eigensolvers and Relative Accuracy

Collecting these observations

$$|\lambda_{1,N} - \widehat{\lambda}_{1,N}| \leq c_N \varepsilon_{\text{mach}} \|\mathbf{A}_N\|$$

$$\|\mathbf{A}_N\| = \lambda_{N,N} = \frac{2}{h^2}(1 - \cos(\pi N h)) = \frac{4}{h^2} - \pi^2 + \mathcal{O}(h^2)$$

$$|\lambda_1 - \lambda_{1,N}| = \frac{1}{12}\pi^4 h^2 + \mathcal{O}(h^4)$$

we find an explanation for the poor convergence.

The error between the *computed* $\widehat{\lambda}_{1,N}$ and the *desired* λ_1 is bounded by:

$$\begin{aligned} |\lambda_1 - \widehat{\lambda}_{1,N}| &\leq |\lambda_1 - \lambda_{1,N}| + |\lambda_{1,N} - \widehat{\lambda}_{1,N}| \\ &\leq \left(\frac{1}{12}\pi^4 h^2 + \mathcal{O}(h^4)\right) + c_N \varepsilon_{\text{mach}} \left(\frac{4}{h^2} - \pi^2 + \mathcal{O}(h^2)\right) \end{aligned}$$

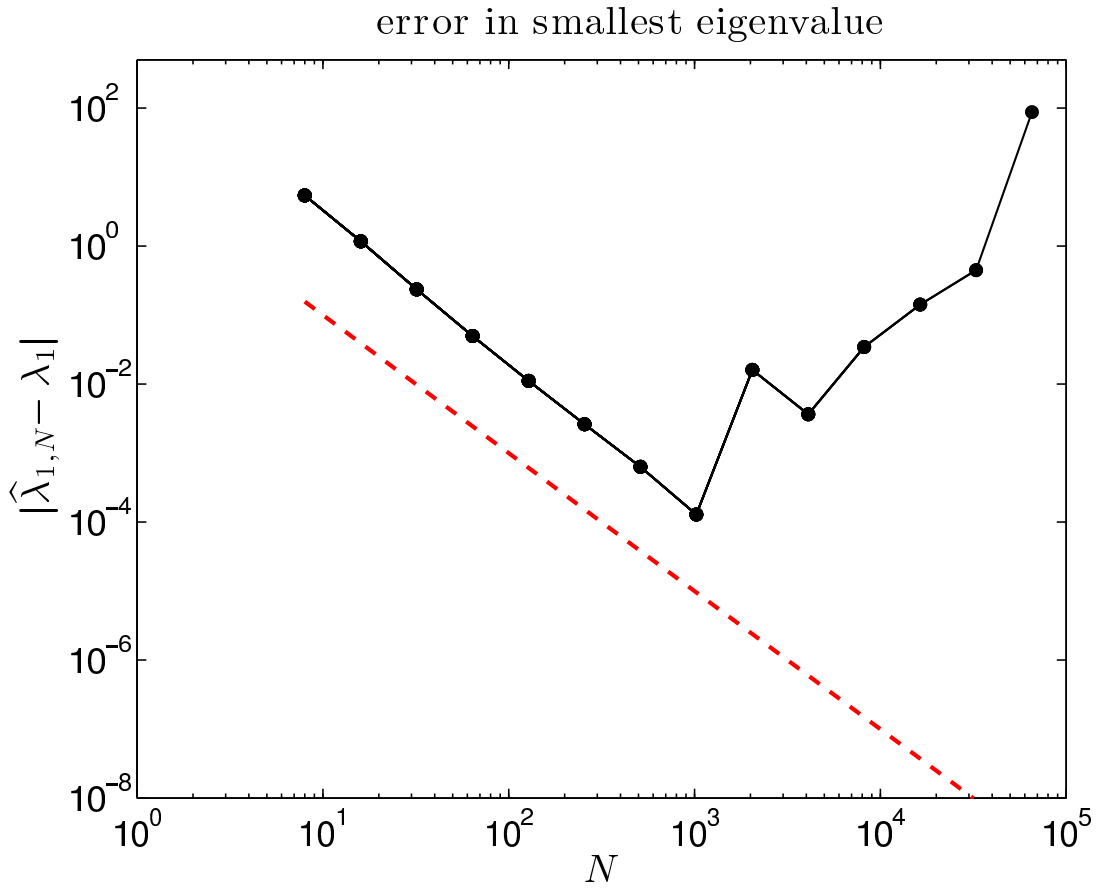
decreasing as $N \to \infty$

increasing as $N \to \infty$

Standard Eigensolvers and Relative Accuracy

The problem becomes even more acute for higher order problems. Use finite differences to discretize $Au = u''''$ on $[0, 1]$ with hinged boundary conditions $u(0) = u(1) = u''(0) = u''(1) = 0$, giving smallest eigenvalue $\lambda_1 = \pi^4$.

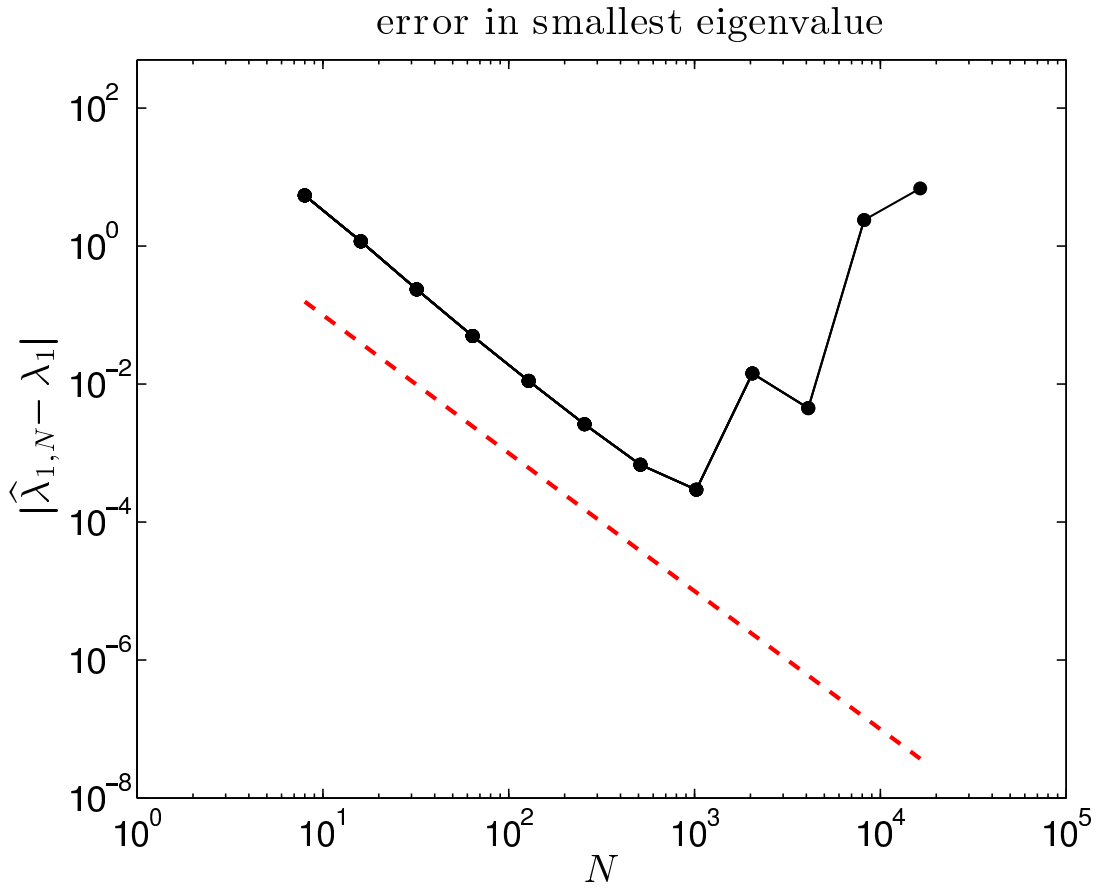
Error in eigenvalue using `eigs`:



Standard Eigensolvers and Relative Accuracy

The problem becomes even more acute for higher order problems. Use finite differences to discretize $Au = u''''$ on $[0, 1]$ with hinged boundary conditions $u(0) = u(1) = u''(0) = u''(1) = 0$, giving smallest eigenvalue $\lambda_1 = \pi^4$.

Error in eigenvalue using `eig`:



Standard Eigensolvers and Relative Accuracy

Can we work around this problem?

- What about working with $\mathbf{A}_N^{-1}$ instead?
  The error from applying $\mathbf{A}_N^{-1}$ scales like $\kappa(\mathbf{A}_N) = \|\mathbf{A}_N\| \|\mathbf{A}_N^{-1}\|$.

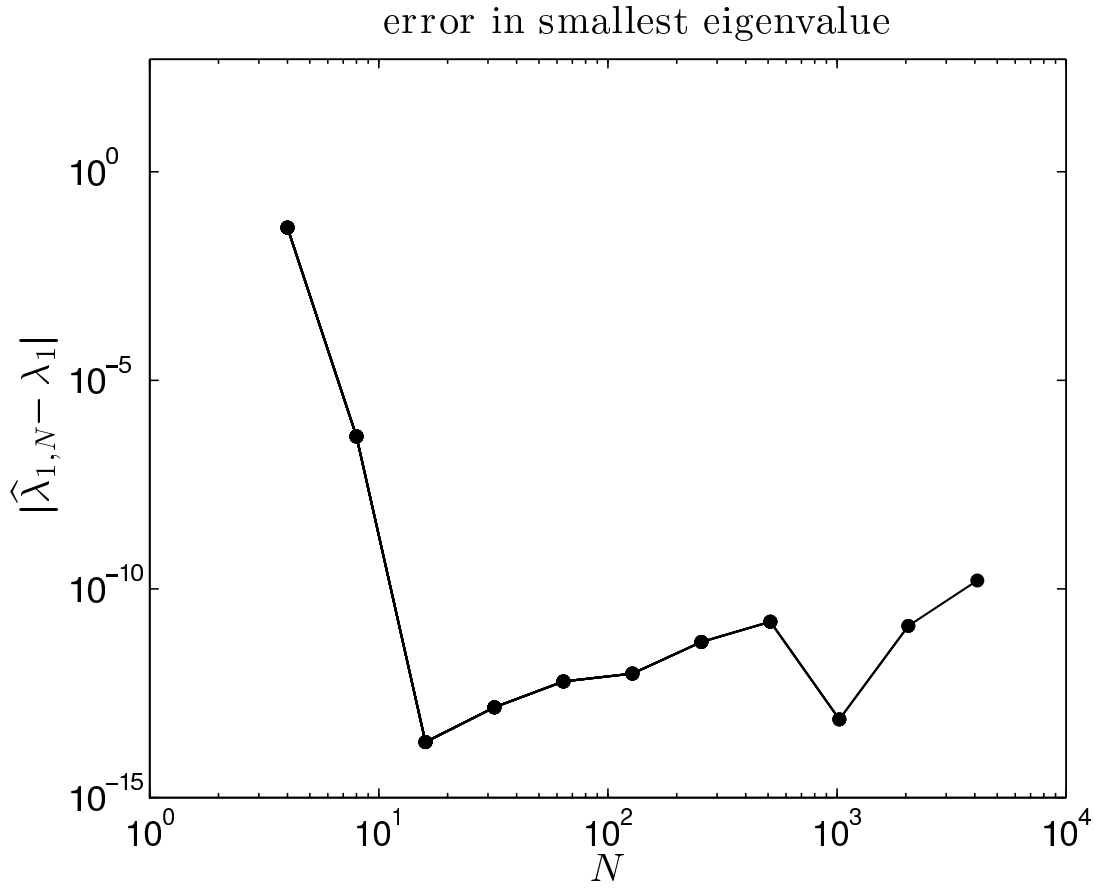
- Better algorithms exist when an accurate factorization of $\mathbf{A}$ is available.
  See, e.g., the survey by Drmač [2013] of *eigenvalue algorithms for high relative accuracy*.

- Sometimes alternative discretizations can yield better accuracy.

Standard Eigensolvers and Relative Accuracy

For example, for these simple problems we can apply Chebyshev pseudospectral collocation methods (see [Trefethen, 2000]) to obtain approximate eigenvalues that are "spectrally accurate," i.e., where $|\lambda_1 - \lambda_{1,N}| = \mathcal{O}(h^p)$ for *all powers p* as $N \to \infty$. The desired eigenvalue converges faster before $\|\mathbf{A}_N\|$ overwhelms.

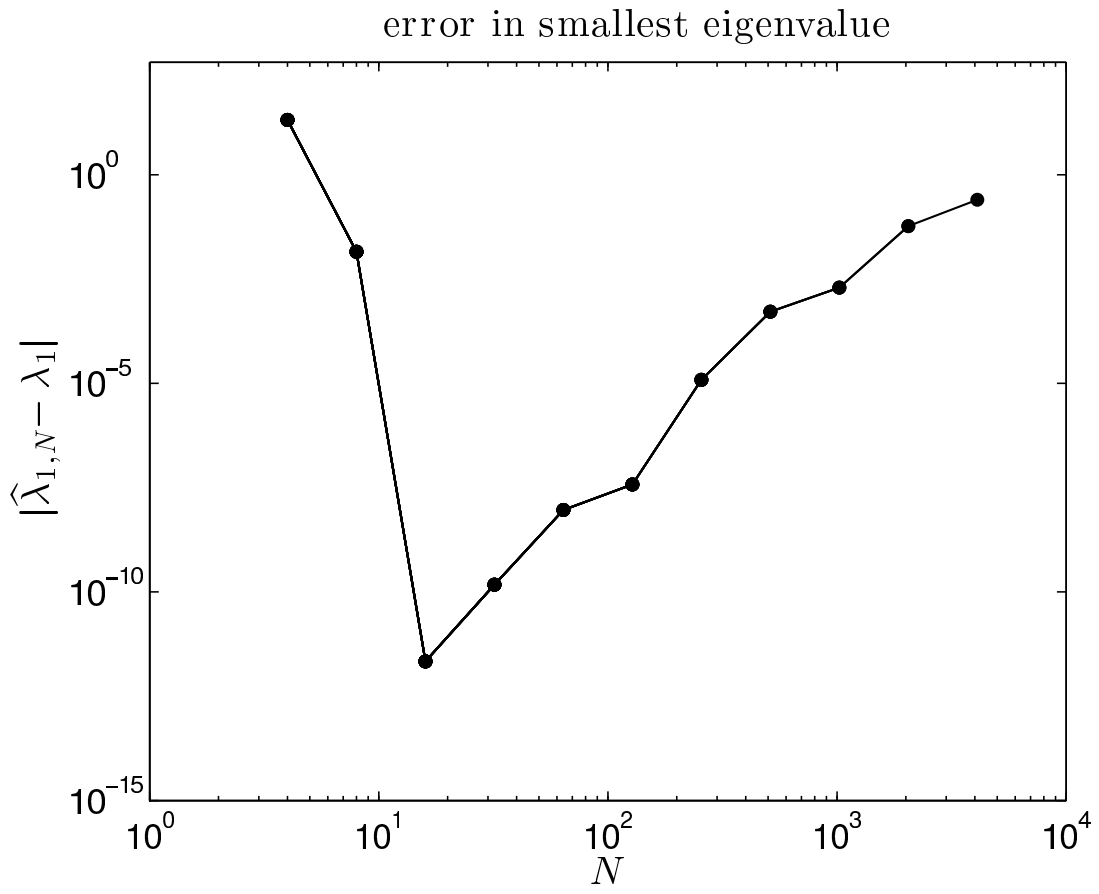
For $Au = -u''$ (with `eigs`):



Standard Eigensolvers and Relative Accuracy

For example, for these simple problems we can apply Chebyshev pseudospectral collocation methods (see [Trefethen, 2000]) to obtain approximate eigenvalues that are "spectrally accurate," i.e., where $|\lambda_1 - \lambda_{1,N}| = \mathcal{O}(h^p)$ for *all powers p* as $N \to \infty$. The desired eigenvalue converges faster before $\|\mathbf{A}_N\|$ overwhelms.

For $Au = u''''$ (with `eigs`):



Conclusions

1. DATA / MODEL
2. LINEARIZE
3. DISCRETIZE
4. PROJECT
5. COMPUTE

We have illustrated some of the challenges the follow from each of the steps in this transition from physical problem to numerically computed eigenvalue problems.

Even then, this survey has skimmed many important and interesting topics, such as Rayleigh–Ritz eigenvalue approximation, challenges of spectral approximation for non-self-adjoint problems, improved algorithms for high relative accuracy, interval arithmetic, inner products, ...

We hope this long view the approximation process can better inform the choices made at each step, so as to alleviate difficulties further down the chain.

Similarly, by better understanding the challenges that arise at the model and operator level, the numerical linear algebra community might focus energy toward problems that emerge before construction of $\mathbf{A} \in \mathbf{R}^{n \times n}$.

Thank you

Steve Cox

Jeffrey Hokanson

The Strings Lab, especially Jeffrey Bridge and Sean Hardesty.

The Eigenvalue Clinic, especially Russell Carden, Josef Sifuentes, Cosmin Ionitia, Sanda Lefteriu, and Charles Puelz.