UNIVERSITY OF MINNESOTA TWIN CITIES

The EVSL package for symmetric eigenvalue problems

Yousef Saad

Department of Computer Science and Engineering

University of Minnesota

15th Copper Mountain Conference
Mar. 28, 2018

First:

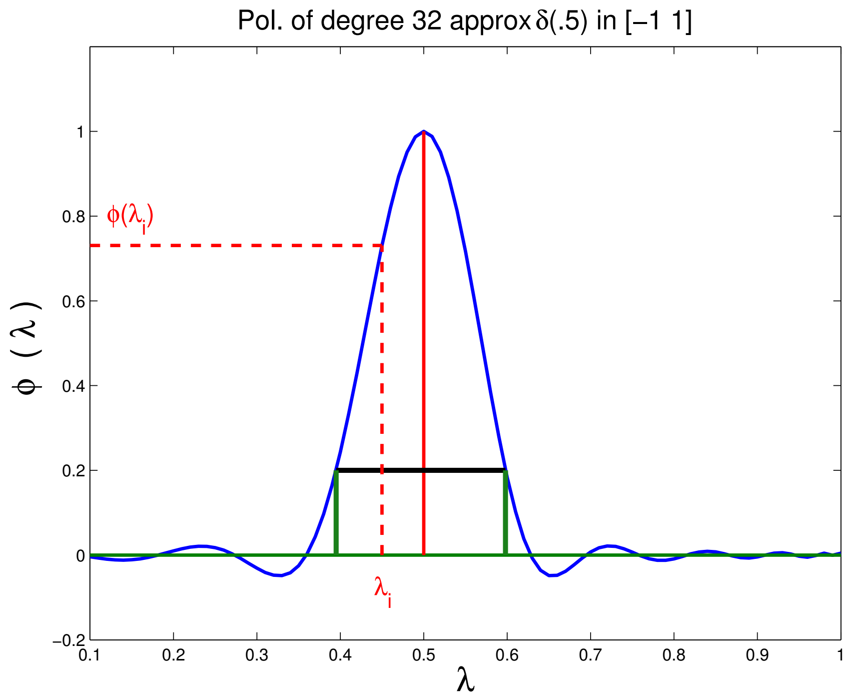
- **Joint work with Ruipeng Li, Yuanzhe Xi, and Luke Erlandson**
- **Application side: collaboration with Jia Shi, Maarten V. de Hoop (Rice)**
- **Support: NSF**

“Spectrum Slicing”

- Context: very large number of eigenvalues to be computed
- Goal: compute spectrum by slices by applying filtering
- Apply Lanczos or Subspace iteration to problem:

$$\phi(A)u = \mu u$$

$\phi(t) \equiv$ a polynomial or rational function that enhances wanted eigenvalues

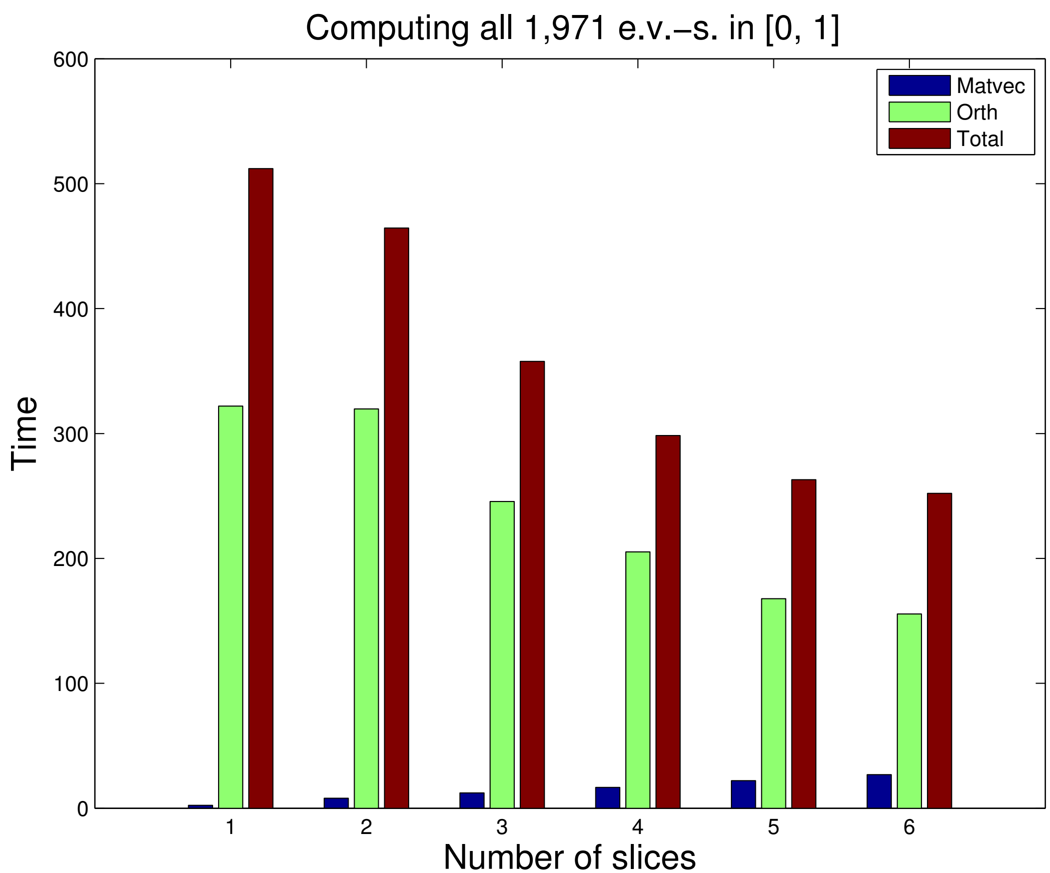


Rationale. Eigenvectors on both ends of wanted spectrum need not be orthogonalized against each other :

➤ Idea: Get the spectrum by ‘slices’ or ’windows’ [e.g., a few hundreds or thousands of pairs at a time]

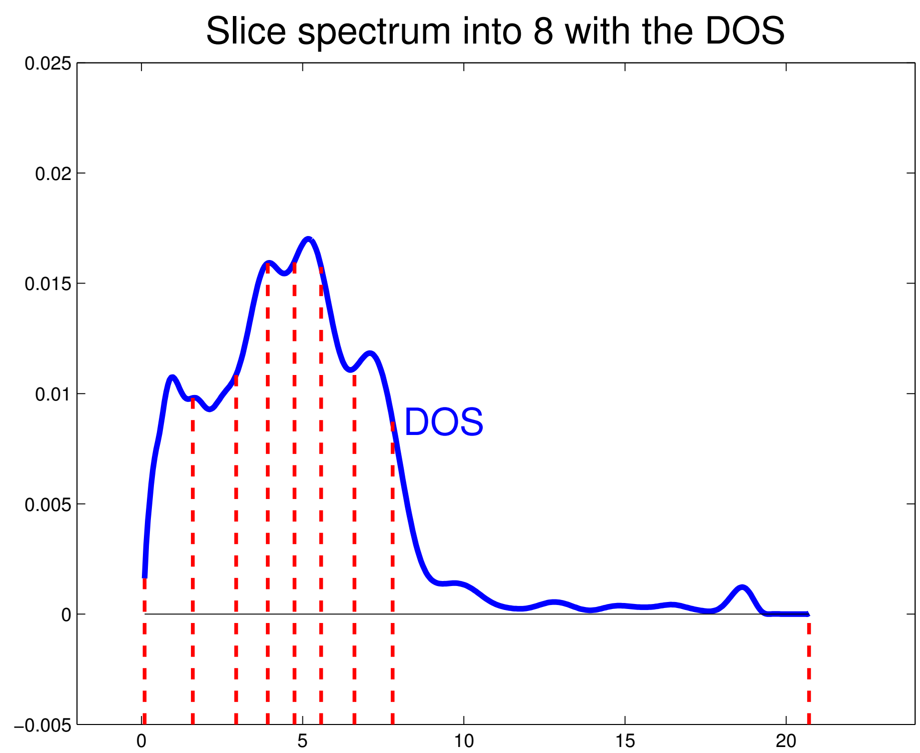
➤ Note: Orthogonalization cost can be very high if we do not slice the spectrum

Illustration: All eigenvalues in [0, 1] of a 49^3 Laplacean



Note: This is a small pb. in a scalar environment. Effect likely much more pronounced in a fully parallel case.

How do I slice my spectrum?



Answer: Use the spectral density, aka, 'Density Of States' (DOS)

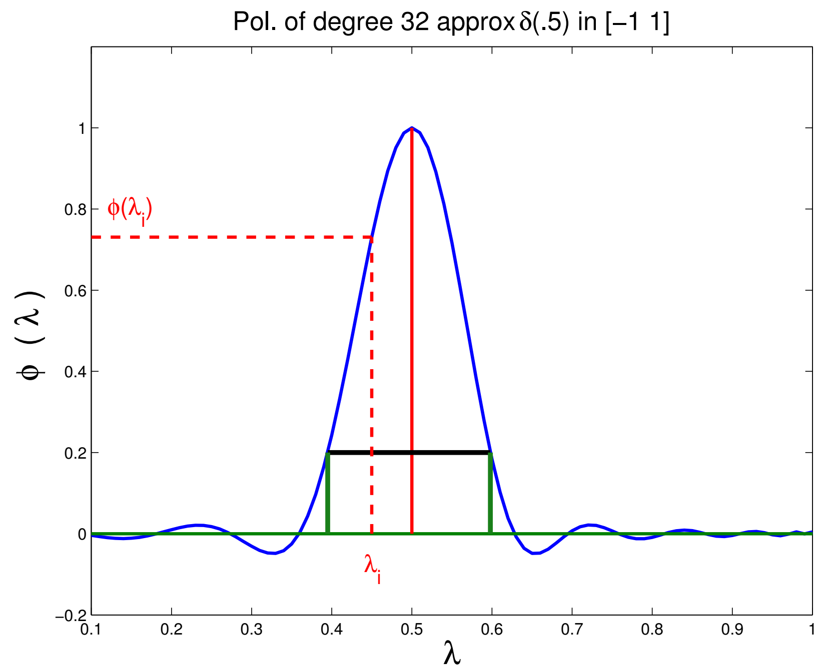
➤ DOS inexpensive to compute

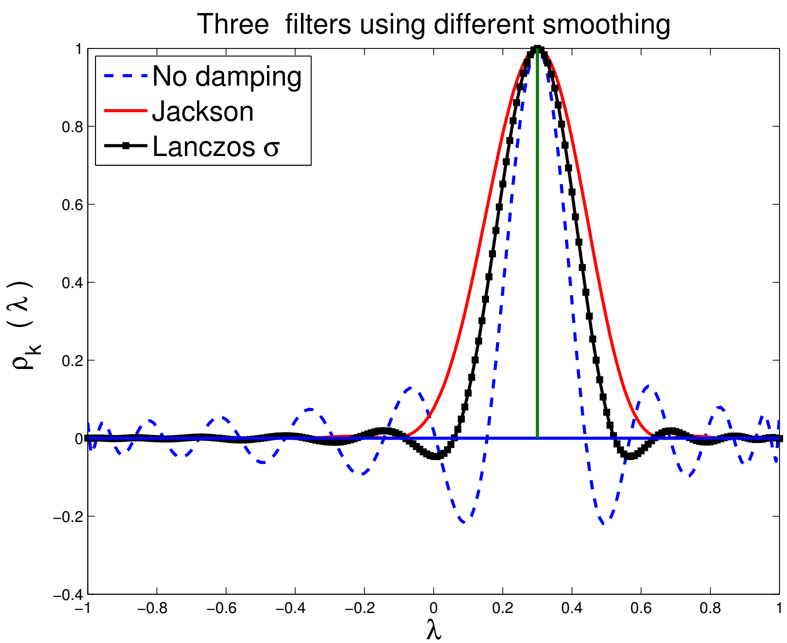
➤ We must have:

$$\int_{t_i}^{t_{i+1}} \phi(t)dt = \frac{1}{n_{slices}} \int_a^b \phi(t)dt$$

Polynomial filtering: The δ-Dirac function approach

➤ Obtain the LS approximation to the $\delta-$ Dirac function – Centered at some point (TBD) inside interval. $\longrightarrow$





$\longleftarrow$ Damping: Jackson, Lanczos σ damping, or none.

'The soul of a new filter' – A few technical details

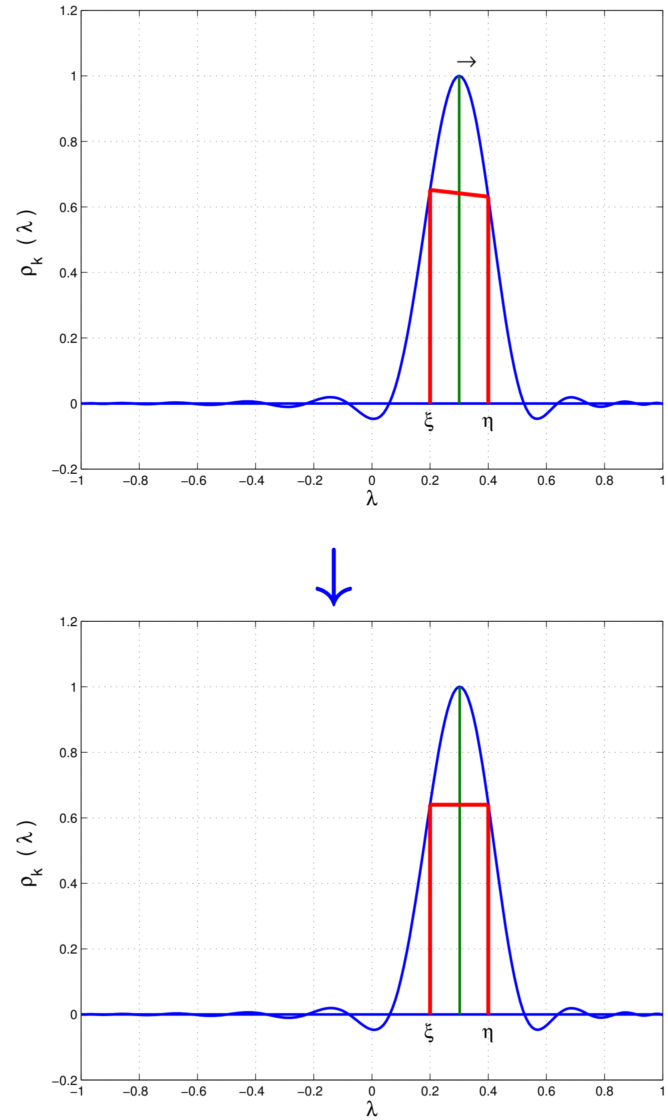
Issue # one: 'balance the filter'

➤ To facilitate the selection of *'wanted'* eigenvalues [Select $\boldsymbol{\lambda}$'s such that $\rho(\boldsymbol{\lambda}) >$ bar] we need to ...

➤ ... find γ so that $\rho(\boldsymbol{\xi}) - \rho(\boldsymbol{\eta}) = 0$

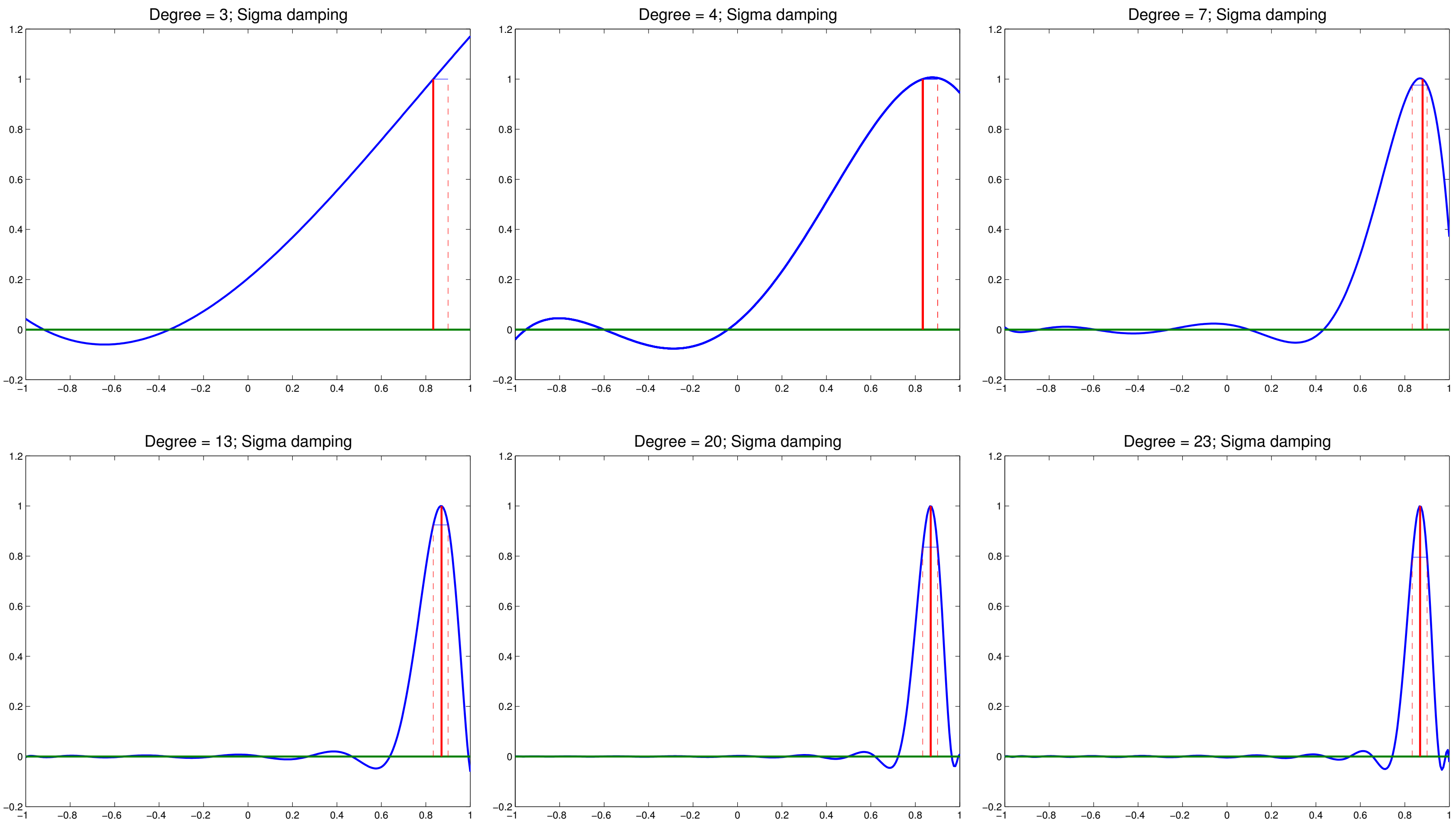
Procedure: Solve the equation $\rho_\gamma(\boldsymbol{\xi}) - \rho_\gamma(\boldsymbol{\eta}) = 0$ with respect to γ, accurately.

Use Newton scheme



Issue # two: Determine degree & polynomial (automatically)

Start low then increase degree until value (s) at the boundary (ies) become small enough - Exple for [0.833, 0.907..]



Which Projection: Lanczos,w/o restarts, Subspace iteration,..

Options:

➤ Subspace iteration: quite appealing in some applications (e.g., electronic structure): Can re-use previous subspace.

➤ Simplest: (+ most efficient) Lanczos without restarts

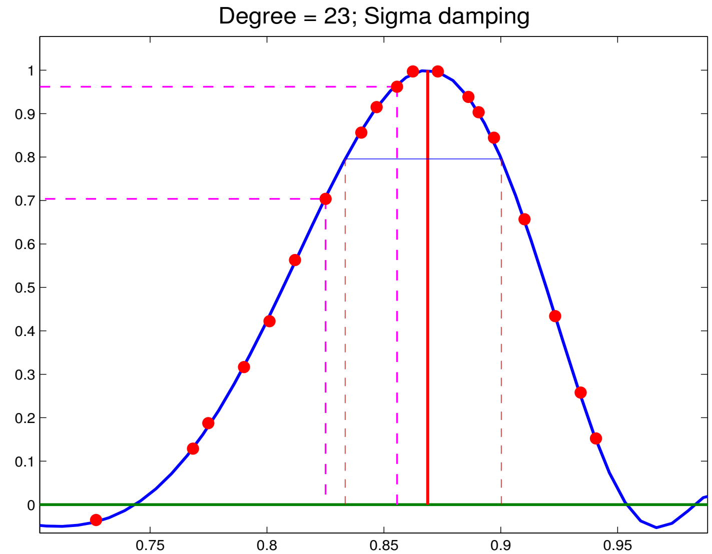
➤ Lanczos with Thick-Restarting [TR Lanczos, Stathopoulos et al ’98, Wu & Simon’00]

➤ Crucial tool in TR Lanczos: deflation (’Locking’)

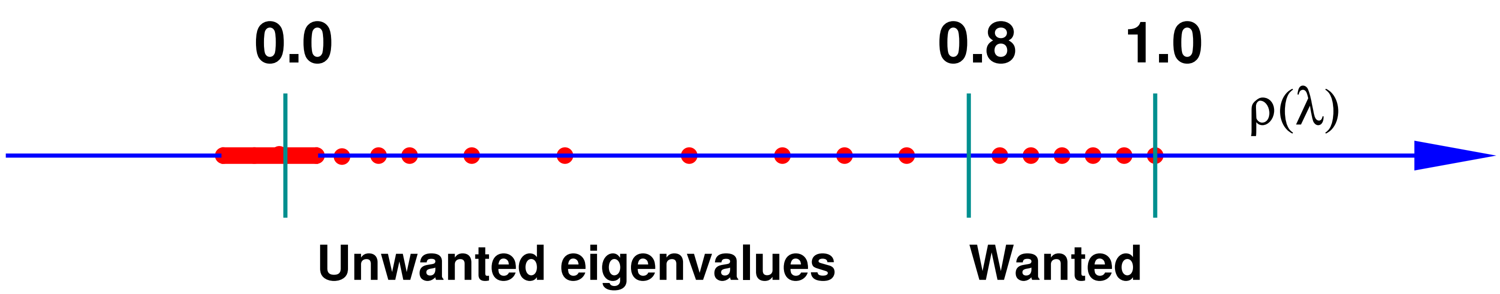
Main idea: *Keep extracting eigenvalues in interval $[\xi, \eta]$ until none are left [remember: deflation]*

➤ If filter is good: Can catch all eigenvalues in interval thanks to deflation + Lanczos.

Polynomial filtered Lanczos: No-Restart version

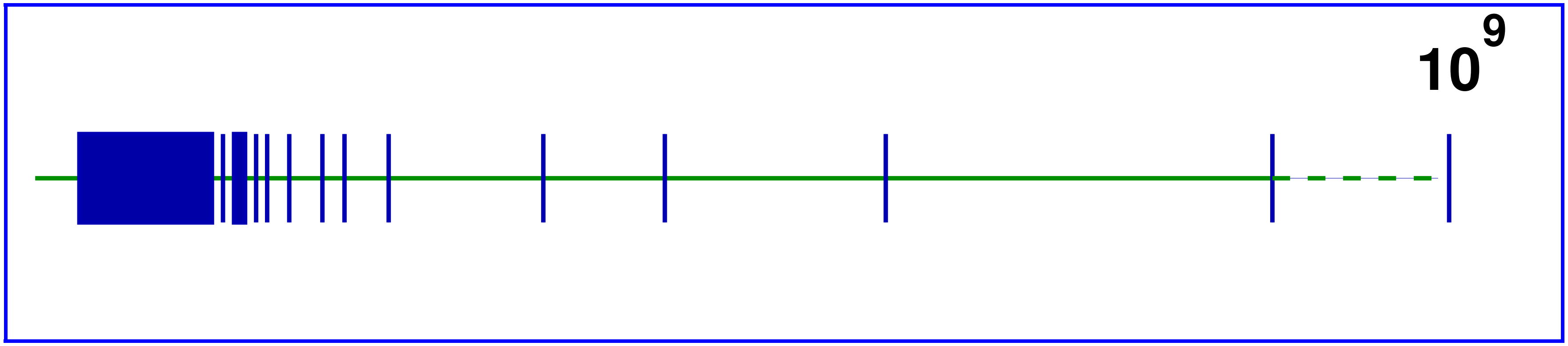


- Use Lanczos with full reorthogonalization on $\rho(A)$. Eigenvalues of $\rho(A)$: $\rho(\lambda_i)$
- Accept if $\rho(\lambda_i) \geq \texttt{bar}$
- Ignore if $\rho(\lambda_i) < \texttt{bar}$



Rational filters: Why?

➤ Consider a spectrum like this one:



➤ Polynomial filtering utterly ineffective for this case

➤ Second issue: situation when Matrix-vector products are expensive

➤ Generalized eigenvalue problems.

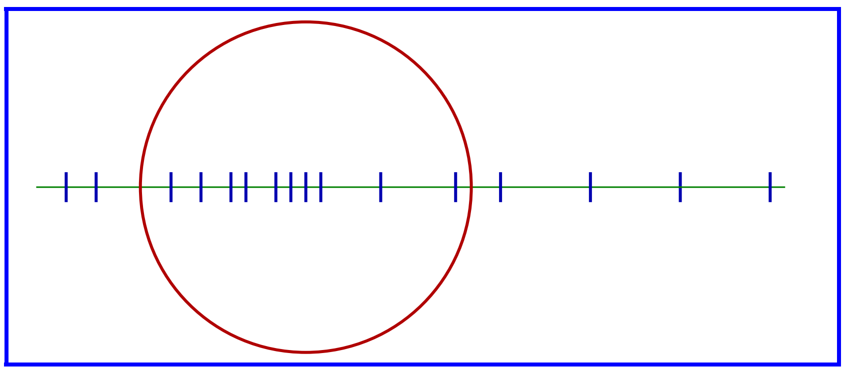
- Alternative is to use rational filters:

$$\phi(z) = \sum_j \frac{\alpha_j}{z-\sigma_j}$$

$$\phi(A) = \sum_j \alpha_j (A - \sigma_j I)^{-1}$$

$\rightarrow$ We now need to solve linear systems

- Tool: Cauchy integral representations of spectral projectors



$$P = \frac{-1}{2i\pi} \int_\Gamma (A - sI)^{-1} ds$$

- Numer. integr. $P \rightarrow \tilde{P}$
- Use Krylov or S.I. on $\tilde{P}$

- Sakurai-Sugiura approach [Krylov]

- FEAST [Subs. iter.] (E. Polizzi)

The Gauss viewpoint: Least-squares rational filters

- Given: poles $\sigma_1, \sigma_2, \cdots, \sigma_p$
- Related basis functions $\phi_j(z) = \frac{1}{z-\sigma_j}$

Find $\phi(z) = \sum_{j=1}^{p} \alpha_j \phi_j(z)$ that minimizes

$$\int_{-\infty}^{\infty} w(t)|h(t) - \phi(t)|^2 dt$$

- $h(t)$ = step function $\chi_{[-1,1]}$.
- $w(t)$= weight function.

For example $a = 10$, $\beta = 0.1$

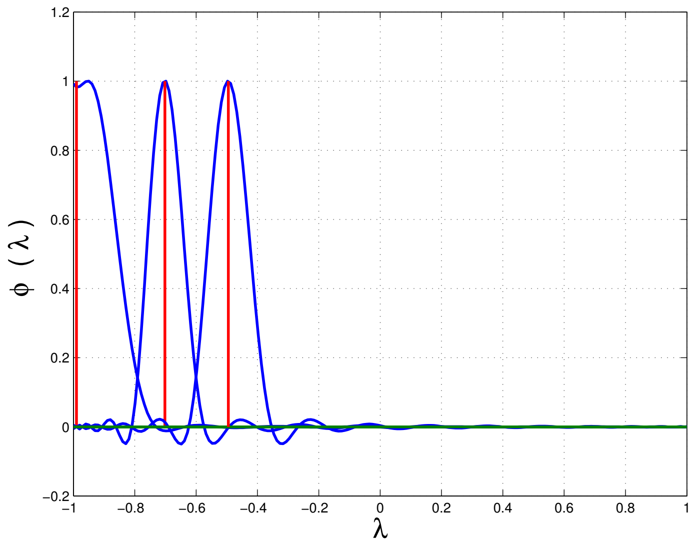
$$w(t) = \begin{cases} 0 \text{ if } & |t| > a \\ \beta \text{ if } & |t| \le 1 \\ 1 \text{ else} & \end{cases}$$

- Many advantages

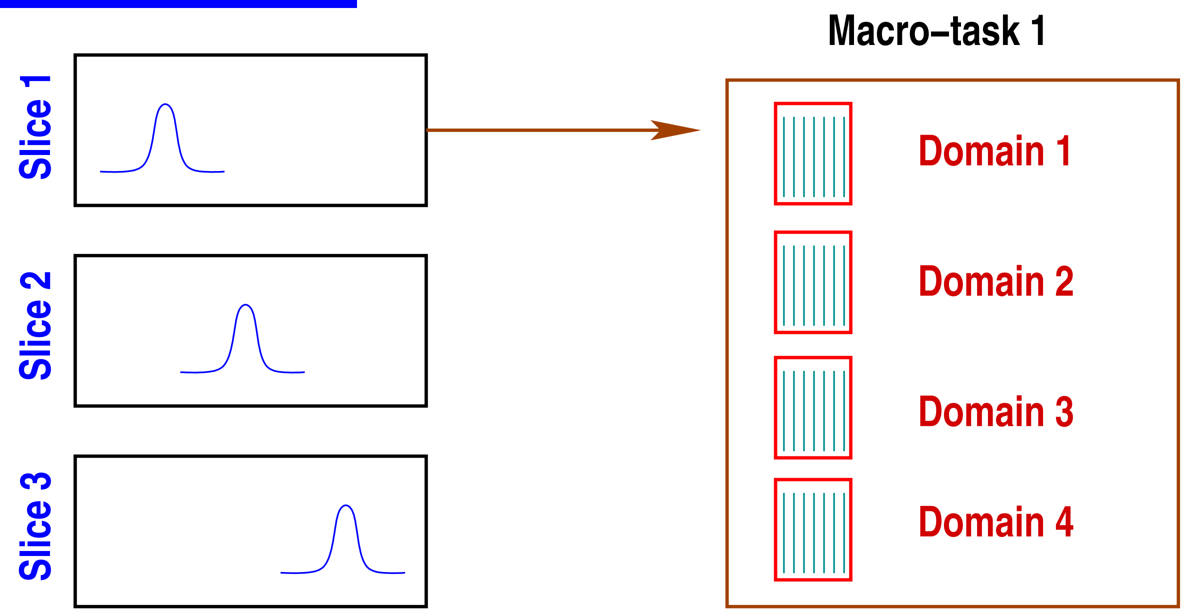
Spectrum Slicing and the EVSL project

- EVSL package now at version 1.1.x
- Uses polynomial and rational filtering: Each can be appealing in different situations.

Spectrum slicing: Invokes Kernel Polynomial Method or Lanczos quadrature to cut the overall interval containing the spectrum into small sub-intervals.

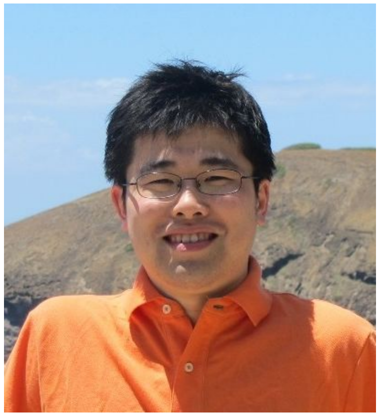


Levels of parallelism



The two main levels of parallelism in EVSL

EVSL Main Contributors (version 1.1.0+) & Support



- Ruipeng Li
LLNL

- Yuanzhe Xi
Post-doc (UMN)

- Luke Erlandson
UG Intern (UMN)

➤ Work supported by NSF (also past work: DOE)

➤ See web-site for details:

`http://www-users.cs.umn.edu/~saad/software/EVSL/`

EVSL: current status & plans

Version _1.0 Released in Sept. 2016

- Matrices in CSR format (only)
- Standard Hermitian problems (no generalized)
- Spectrum slicing with KPM (Kernel Polynomial Meth.)
- Trivial parallelism across slices with OpenMP
- Methods:
  - Non-restart Lanczos – polynomial & rational filters
  - Thick-Restart Lanczos – polynomial & rational filters
  - Subspace iteration – polynomial & rational filters

Version _1.1.x V_1.1.0 Released back in August 2017.

- general `matvec` [passed as function pointer]
- $\boldsymbol{Ax = \lambda Bx}$
- Fortran (03) interface.
- Spectrum slicing by Lanczos and KPM
- Efficient Spectrum slicing for $\boldsymbol{Ax = \lambda Bx}$ (no solves with $\boldsymbol{B}$).

Version _1.2.x pEVSL – In progress

- Fully parallel version [MPI + openMP]

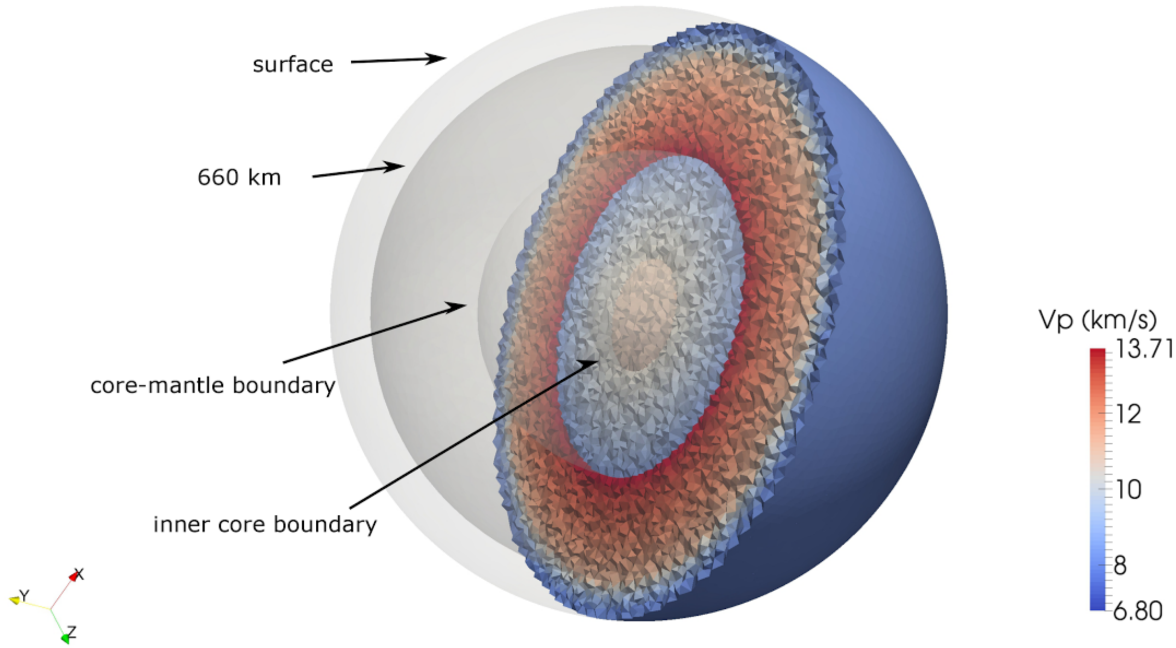
Spectrum slicing and the EVSL package

- All eigenvalues in $[0,\ 1]$ of of a 49^3 discretized Laplacian
- eigs(A,1971,'sa'): 14830.66 sec
- Solution: Use DOS to partition $[0,\ 1]$ into 5 slices
- Polynomial filtering from EVSL on Mesabi MSI, 23 threads/slice

$[a_i, a_{i+1}]$	# eigs	CPU time (sec)			max residual
		matvec	orth.	total	
$[0.00000, 0.37688]$	386	1.31	18.26	28.66	2.5×10^{-14}
$[0.37688, 0.57428]$	401	3.28	38.25	56.75	8.7×10^{-13}
$[0.57428, 0.73422]$	399	4.69	36.47	56.73	1.7×10^{-12}
$[0.73422, 0.87389]$	400	5.97	38.60	61.40	6.6×10^{-12}
$[0.87389, 1.00000]$	385	6.84	36.16	59.45	4.3×10^{-12}

➤ Grand tot. = 263 s. Time for slicing the spectrum: 1.22 sec.

Computing the Earth normal modes



- Collaborative effort: Rice-UMN:
  J. Shi, R. Li, Y. Xi, YS, and M. V. De Hoop
- FEM model leads to a generalized eigenvalue problem:

$$\begin{bmatrix} A_s & & E_{fs} \\ & 0 & A_d \\ E_{fs}^T & A_d^T & A_p \end{bmatrix} \begin{bmatrix} u^s \\ u^f \\ p^e \end{bmatrix} = \omega^2 \begin{bmatrix} M_s & & \\ & M_f & \\ & & 0 \end{bmatrix} \begin{bmatrix} u^s \\ u^f \\ p^e \end{bmatrix}$$

- Want all eigen-values/vectors inside a given interval
- Issue 1: ‘mass’ matrix has a large null space..
- Issue 2: interior eigenvalue problem
- Solution for 1: change formulation of matrix problem [eliminate p^e ...]

➤ New formulation :

$$\underbrace{\left\{\begin{pmatrix} A_s & 0 \\ 0 & 0 \end{pmatrix} - \begin{pmatrix} E_{fs} \\ A_d \end{pmatrix} A_p^{-1} \begin{pmatrix} E_{fs}^T & A_d^T \end{pmatrix}\right\}}_{\widehat{A}} \begin{pmatrix} u^s \\ u^f \end{pmatrix} =$$

$$\omega^2 \underbrace{\begin{pmatrix} M_s & 0 \\ 0 & M_f \end{pmatrix}}_{\widehat{M}} \begin{pmatrix} u^s \\ u^f \end{pmatrix}$$

➤ Use polynomial filtering – need to solve with $\widehat{M}$ but ...

- ... severe scaling problems if direct solvers are used

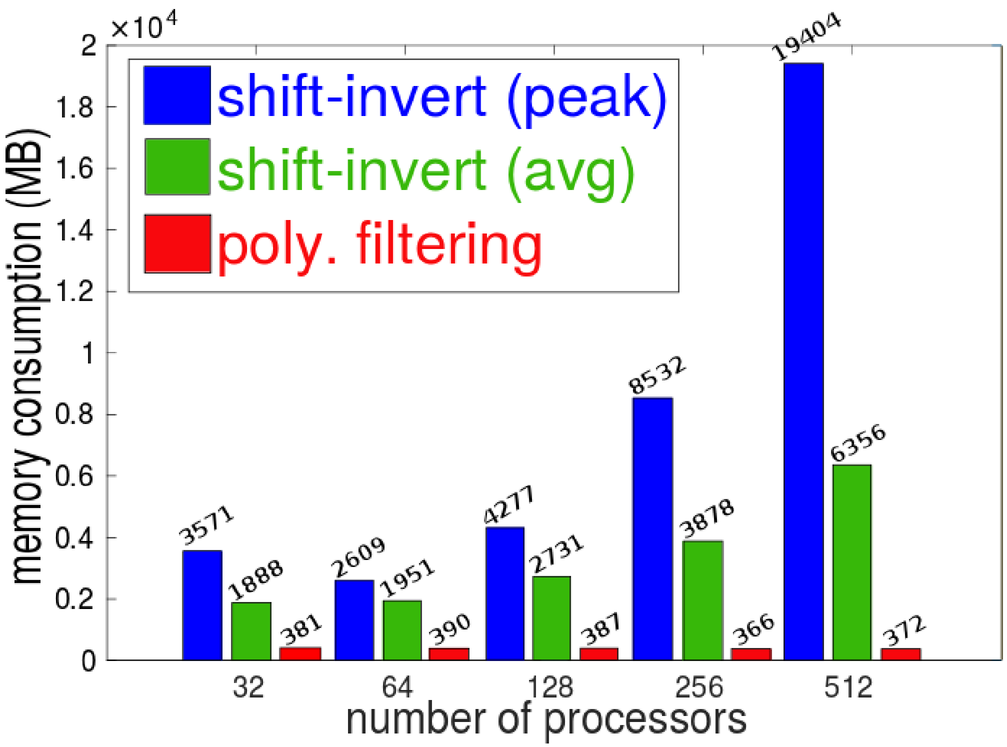
Hence:

➤ Replace action of M^{-1} by a low-deg. polynomial in M [to avoid direct solvers]

➤ Memory : parallel shift-invert and polynomial filtering

Machine: Comet, SDSC

Matrix size	# Proc.s
$\mathbf{591,303}$	32
$\mathbf{1,157,131}$	64
$\mathbf{2,425,349}$	128
$\mathbf{4,778,004}$	256
$\mathbf{9,037,671}$	512



Recent: weak calability test for different solid (Mars-like) models on TACC Stampede2

nn/np	Mat-size	Av (ms)	$\leftarrow$ Eff.	Mv (ms)	$\leftarrow$ Eff.	$M^{-1}v$ (μs)	$\leftarrow$ Eff.
2/96	1,038,084	1760	1.0	495	1.0	0.01044	1.0
4/192	2,060,190	1819	0.960	568	0.865	0.0119	0.870
8/384	3,894,783	1741	0.948	571	0.813	0.0119	0.825
16/768	7,954,392	1758	0.959	621	0.763	0.0129	0.774
32/1536	15,809,076	1660	1.009	572	0.824	0.0119	0.834
64/3072	31,138,518	1582	1.043	566	0.820	0.0117	0.837
128/6144	61,381,362	1435	1.133	546	0.838	0.0113	0.851
256/12288	120,336,519	1359	1.173	592	0.757	0.01221	0.774

Conclusion

- `EVSL` code available here: [Current version: version 1.1.1]

  `www.cs.umn.edu/~saad/software/EVSL`

- `EVSL` Also on github (development)

Plans: (1) Release fully parallel code; (2) Block versions; (3) Iterative solvers for rational filt.; (4) Nonhermitian case;

- Earth modes calculations done with fully parallel code
  $\longrightarrow$ Not quite ready for distribution

A final note: Scalability issues with parallel direct solvers ...

- ... Needed: iterative solvers for the highly indefinite case