UNIVERSITY OF MINNESOTA TWIN CITIES

Polynomial and rational filtering for eigenvalue problems and the EVSL project

Yousef Saad

Department of Computer Science and Engineering

University of Minnesota

PASC17 – Lugano
June 27, 2017

Large eigenvalue problems in applications

➤ Challenge in eigenvalue problems: extract large number of eigenvalues & vectors of very large matrices (quantum physics/ chemistry, ...) - often in the middle of spectrum.

➤ Example: *Excited states* involve transitions $\longrightarrow$ much more complex computations than for DFT (ground states)

➤ Large matrices, *many* eigen-pairs to compute

Illustration:

'Hamiltonian of size $n \sim 10^6$ *get 10% of bands'*

Soving large interior eigenvalue problems

Three broad approaches:

1. Shift-invert (real shifts)
2. Polynomial filtering
3. Rational filtering (Cauchy, + others).

Issues with shift-and invert (and related approaches)

- Issue 1: factorization may be too expensive
  - Can use iterative methods?
- Issue 2: Iterative techniques often fail –
  - Reason: Highly indefinite problems.

- First Alternative: 'Spectrum slicing' with Polynomial filtering

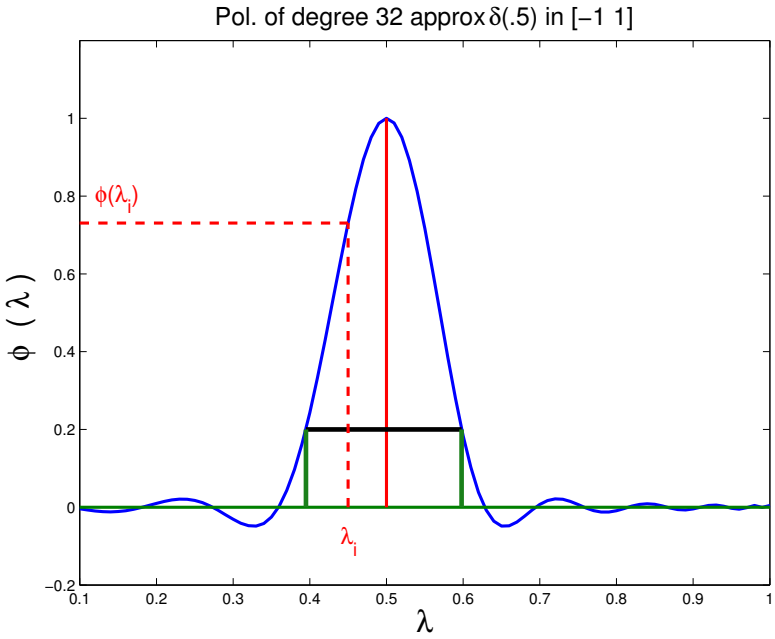
Polynomial filtering

- Apply Lanczos or Subspace iteration to: $M = \rho(A)$ where $\rho(t)$ is a polynomial
- Each *matvec* $y = Av$ is replaced by $y = \rho(A)v$.
- Eigenvalues in high part of filter will be computed first.
- Old (forgotten) idea. But new context is *very* favorable

What polynomials?

- LS approximations to δ-Dirac functions

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



- W'll express everything in the interval $[-1, 1]$

Theory

The Chebyshev expansion of δ_γ is

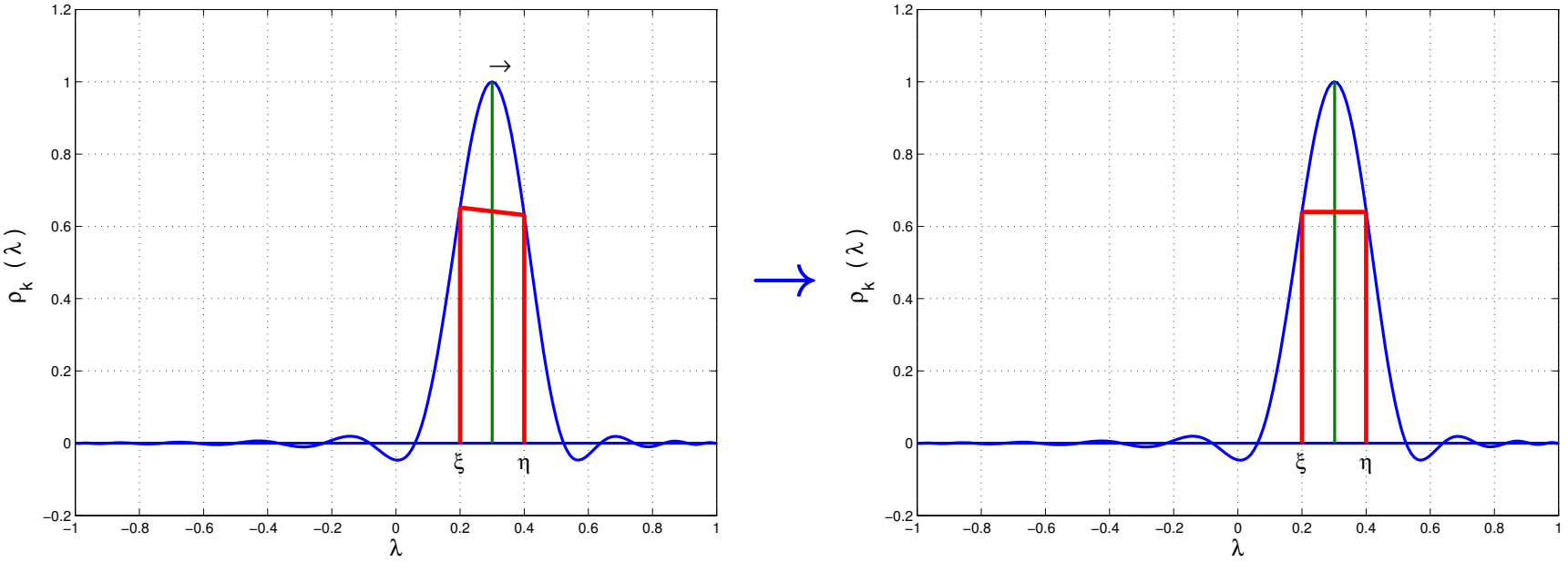
$$\rho_k(t) = \sum_{j=0}^{k} \mu_j T_j(t) \quad \text{with} \quad \mu_j = \begin{cases} \frac{1}{2} & j = 0 \\ \cos(j \cos^{-1}(\gamma)) & j > 0 \end{cases}$$

➤ Recall: The delta Dirac function is not a function – we can't properly approximate it in least-squares sense. However:

Proposition Let $\hat{\rho}_k(t)$ be the polynomial that minimizes $\|r(t)\|_w$ over all polynomials r of degree $\leq k$, such that $r(\gamma) = 1$, where $\|.\|_w$ represents the Chebyshev L^2-norm. Then $\hat{\rho}_k(t) = \rho_k(t)/\rho_k(\gamma)$.

A few technical details. Issue # one: 'balance the filter'

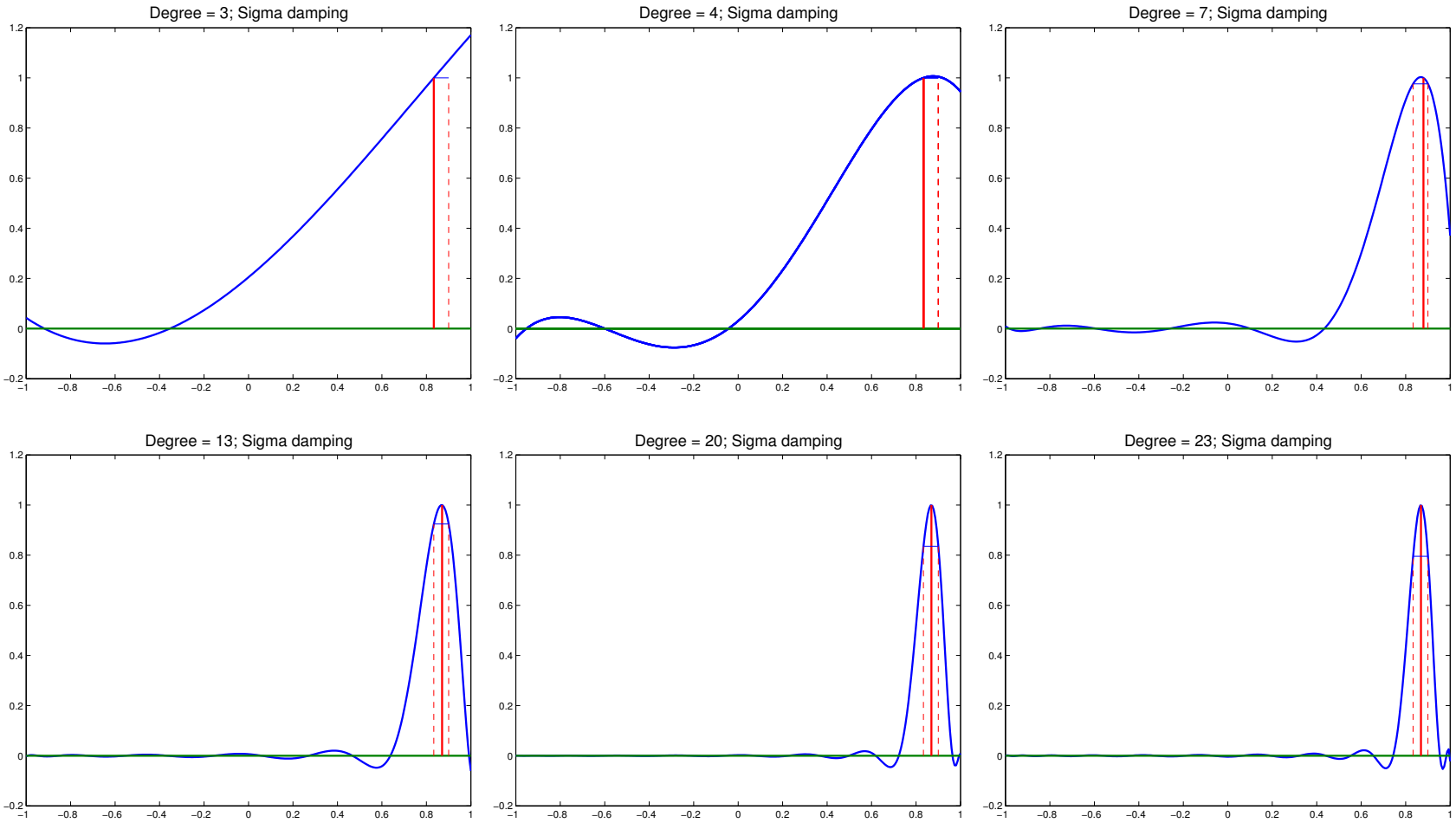
- To facilitate the selection of *'wanted'* eigenvalues [Select λ's such that $\rho(\lambda) >$ bar] we need to ...
- ... find γ so that $\rho(\xi) == \rho(\eta)$



Procedure: Solve the equation $\rho_\gamma(\xi) - \rho_\gamma(\eta) = 0$ with respect to γ, accurately. Use Newton or eigenvalue formulation.

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..]



Polynomial filtered Lanczos

- Use the Lanczos algorithm with A replaced by $p_k(A)$, where $p_k(t) =$ polynomial of degree k
- Idea not new (and not too popular in the past)

What is new?

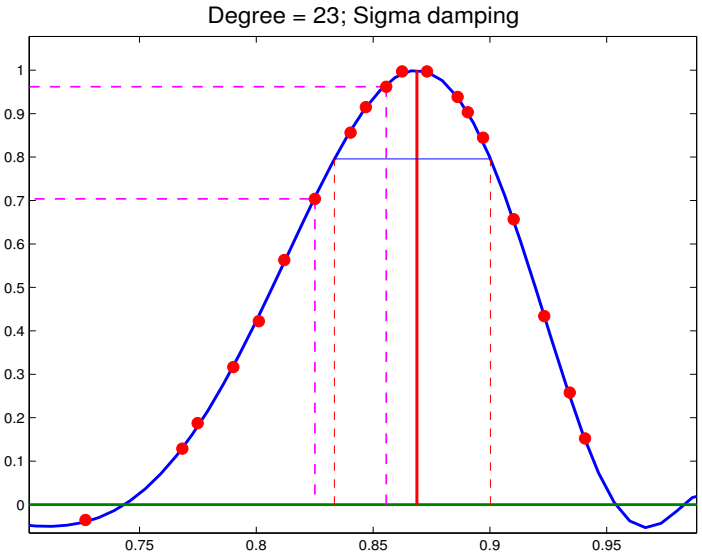
1. Very large problems;
2. (tens of) Thousands of eigenvalues;
3. Parallelism.

- Combine with spectrum slicing
- Main attraction: reduce cost of orthogonalization

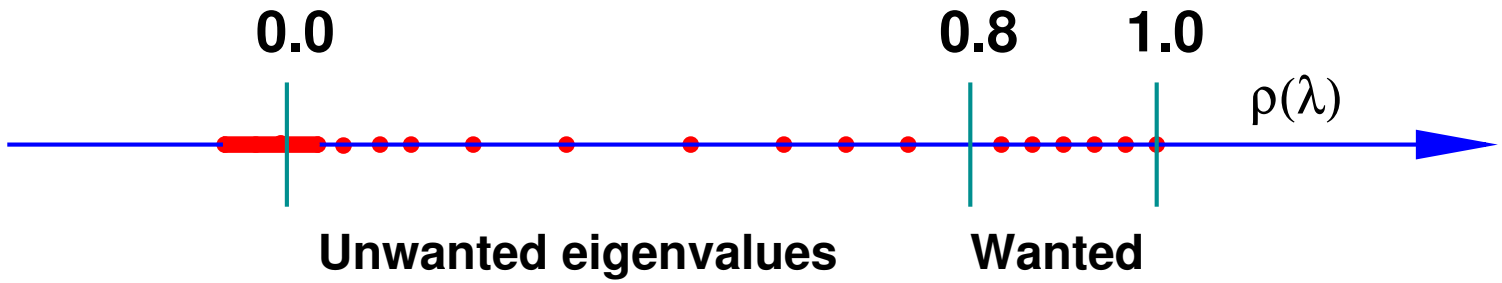
*Hypothetical scenario: large A, *many* wanted eigenpairs*

- Assume A has size $10M$
- ... and you want to compute 50,000 eigenvalues/vectors (huge for numerical analysits, not for physicists) ...
- ... in the lower part of the spectrum - or the middle.
- By (any) standard method you will need to orthogonalize at least 50K vectors of size 10M. Then:
  - Space needed: $\approx 4 \times 10^{12}$ b = 4TB *just for the basis*
  - Orthogonalization cost: 5×10^{16} = 50 PetaOPS.
  - At step k, each orthogonalization step costs $\approx 4kn$
  - This is $\approx 200,000n$ for k close to $50,000$.

Polynomial filtered Lanczos: No-Restart version

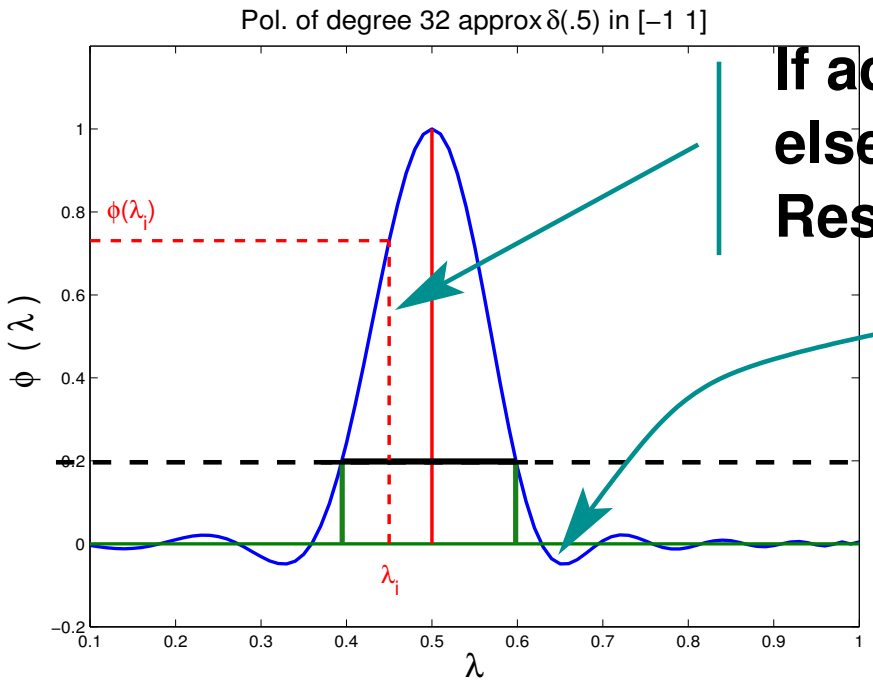


- Use Lanczos with full reorthogonalization on $\rho(A)$. Eigenvalues of $\rho(A)$: $\rho(\lambda_i)$
- Accept if $\rho(\lambda_i) \geq$ `bar`
- Ignore if $\rho(\lambda_i) <$ `bar`



Polynomial filtered Lanczos: Thick-Restart version

- PolFilt Thick-Restart Lanczos in a picture:



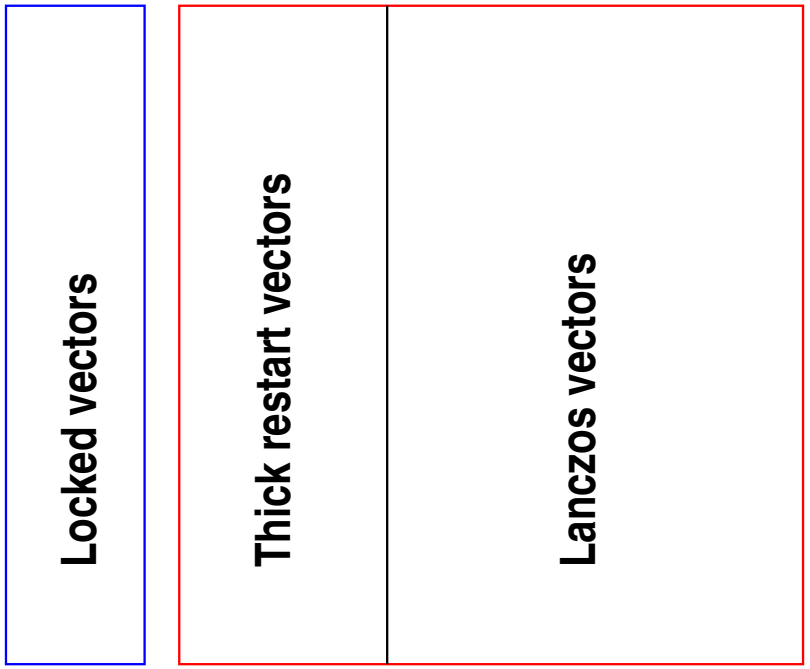
If accurate then lock else add to Thick Restart set.

Reject

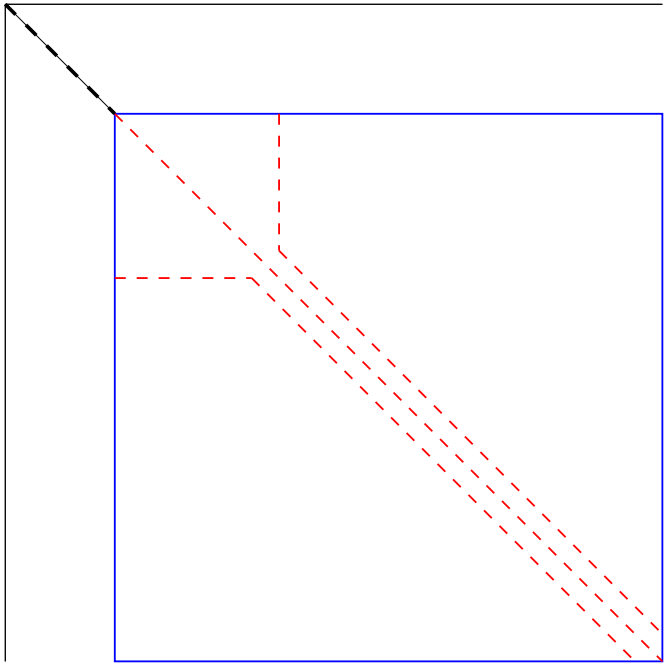
- Due to locking, no more candidates will show up in wanted area after some point → Stop.

TR Lanczos: The 3 types of basis vectors

Basis vectors



Matrix representation

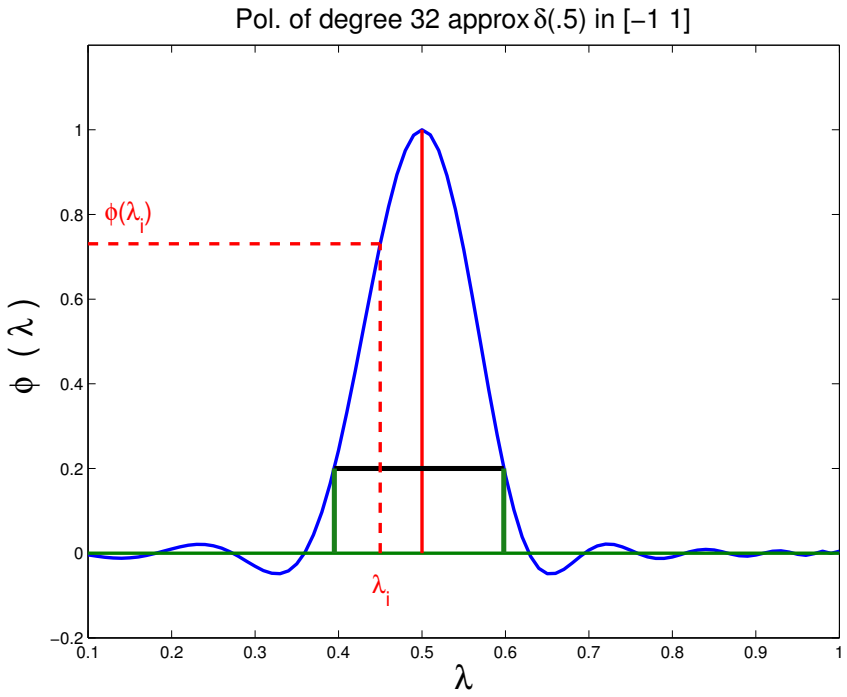


"Spectrum Slicing"

- Situation: 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

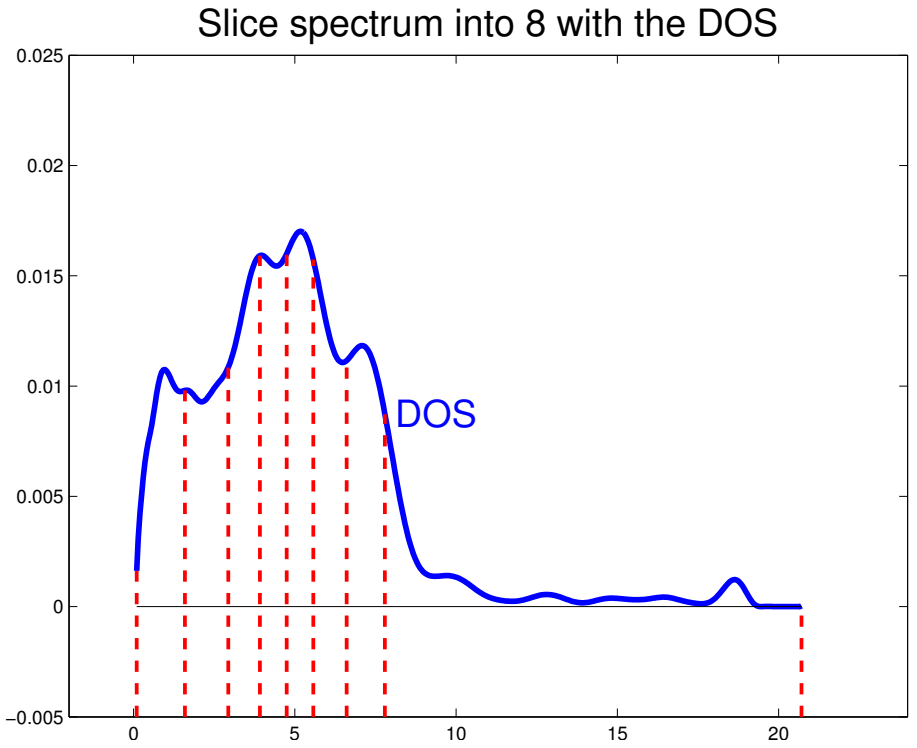


Rationale. Eigenvectors on both ends of wanted spectrum need not be orthogonalized against each other :

- Idea: Get the spectrum by ‘slices’ or ’windows’ [e.g., a few hundreds or thousands of pairs at a time]
- Can use polynomial or rational filters

How do I slice my spectrum?

Answer: Use the DOS.



- We must have:

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

What about matrix pencils?

- DOS for generalized eigenvalue problems

$$Ax = \lambda Bx$$

- Assume: A is symmetric and B is SPD.
- In principle: can just apply methods to $B^{-1}Ax = \lambda x$, using B - inner products.
- Requires factoring B. Too expensive [Think 3D Pbs]

⋆ *Observe:* B is usually very *strongly* diagonally dominant.

- Especially true after Left+Right Diag. scaling :

$$\tilde{B} = S^{-1}BS^{-1} \qquad S = diag(B)^{1/2}$$

General observation for FEM mass matrices [See, e.g., Wathen'87, Wathen Rees '08]:
* Conforming tetrahedral (P1) elements in 3D $\rightarrow \kappa(\tilde{B}) \leq 5$
* Rectangular bilinear (Q1) elements in 2D $\rightarrow \kappa(\tilde{B}) \leq 9$.

Example: Matrix pair *Kuu, Muu* from Suite Sparse collection.

➤ Matrices A and B have dimension $n = 7,102$. $\texttt{nnz}(A) = 340,200$ $\texttt{nnz}(B) = 170,134$.

➤ After scaling by diagonals to have diag. entries equal to one, all eigenvalues of B are in interval

$$[0.6254,\ 1.5899]$$

Approximation theory to the rescue.

⋆ *Idea:* Compute the DOS for the standard problem

$$B^{-1/2}AB^{-1/2}u = \lambda u$$

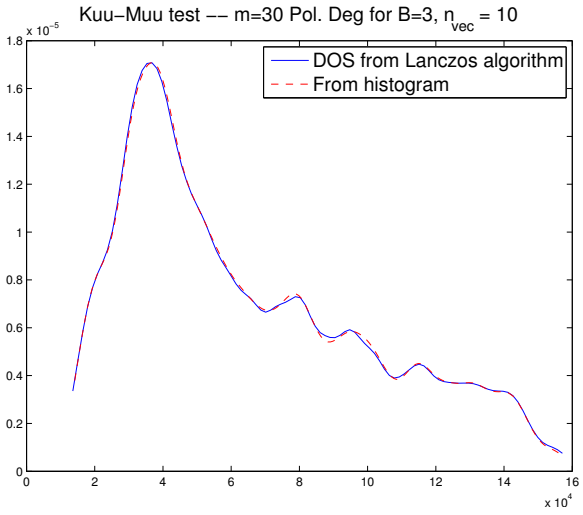
- Use a very low degree polynomial to approximate $B^{-1/2}$.
- We use Chebyshev expansions.
- Degree k determined automatically by enforcing

$$\|t^{-1/2} - p_k(t)\|_\infty < tol$$

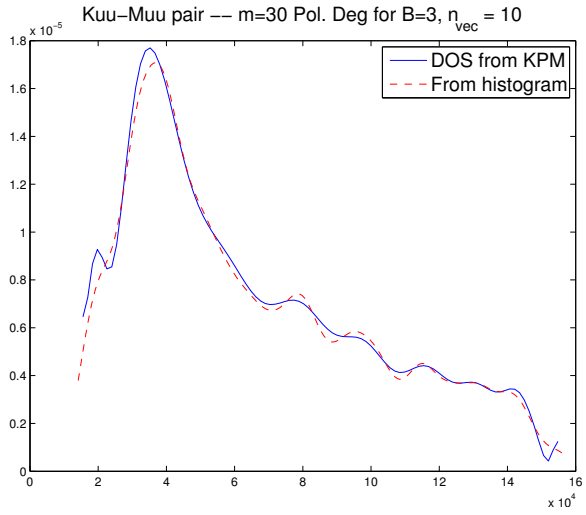
- Theoretical results establish convergence that is exponential with respect to degree.

Example: Results for Kuu-Muu example

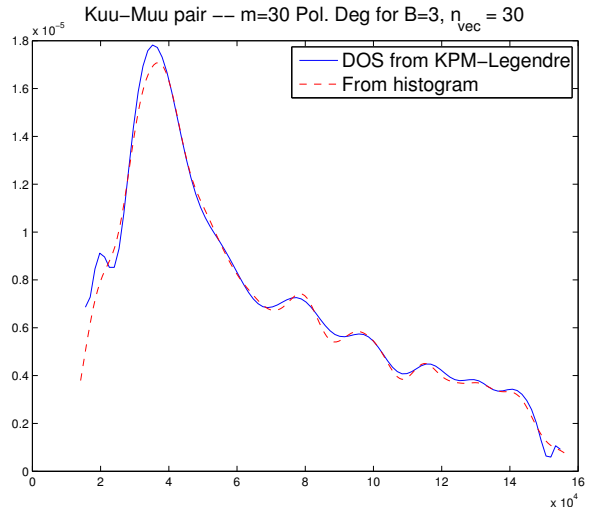
- Using polynomials of degree 3 (!) to approximate $B^{-1/2}$
- Krylov subspace of dim. 30 (== deg. of polynomial in KPM)
- 10 Sample vectors used



Lanczos



KPM-Chebyshev



KPM-Legendre

Experiments:Hamiltonian matrices from PARSEC

Matrix	n	$\sim$ nnz	$[a, b]$	$[\xi, \eta]$	$\nu_{[\xi,\eta]}$
$Ge_{87}H_{76}$	$112,985$	$7.9M$	$[-1.21, 32.76]$	$[-0.64, -0.0053]$	212
$Ge_{99}H_{100}$	$112,985$	$8.5M$	$[-1.22, 32.70]$	$[-0.65, -0.0096]$	250
$Si_{41}Ge_{41}H_{72}$	$185,639$	$15.0M$	$[-1.12, 49.82]$	$[-0.64, -0.0028]$	218
$Si_{87}H_{76}$	$240,369$	$10.6M$	$[-1.19, 43.07]$	$[-0.66, -0.0300]$	213
$Ga_{41}As_{41}H_{72}$	$268,096$	$18.5M$	$[-1.25, 1301]$	$[-0.64, -0.0000]$	201

Results: (Thick-Restart Lanczos)

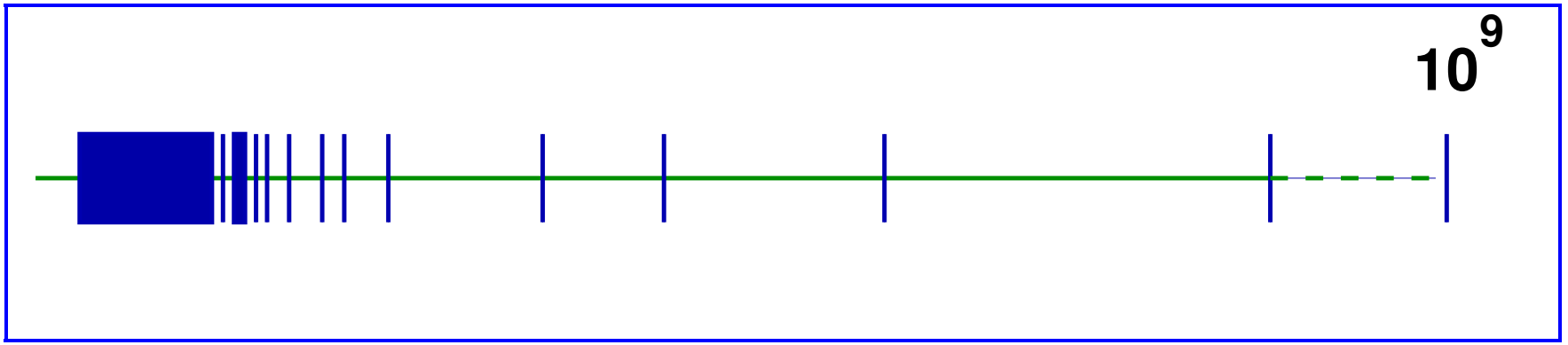
Matrix	deg	iter	matvec	CPU time (sec)		residual	
				matvec	total	max	avg
$Ge_{87}H_{76}$	26	1431	37482	282.70	395.91	9.40×10^{-09}	2.55×10^{-10}
$Ge_{99}H_{100}$	26	1615	42330	338.76	488.91	9.10×10^{-09}	2.26×10^{-10}
$Si_{41}Ge_{41}H_{72}$	35	1420	50032	702.32	891.98	3.80×10^{-09}	8.38×10^{-11}
$Si_{87}H_{76}$	30	1427	43095	468.48	699.90	7.60×10^{-09}	3.29×10^{-10}
$Ga_{41}As_{41}H_{72}$	202	2334	471669	8179.51	9190.46	4.20×10^{-12}	4.33×10^{-13}

- Demo with Si10H16 [$n = 17,077,\ nnz(A) = 446,500$]

RATIONAL FILTERS

Why use rational filters?

- Consider a spectrum like this one:



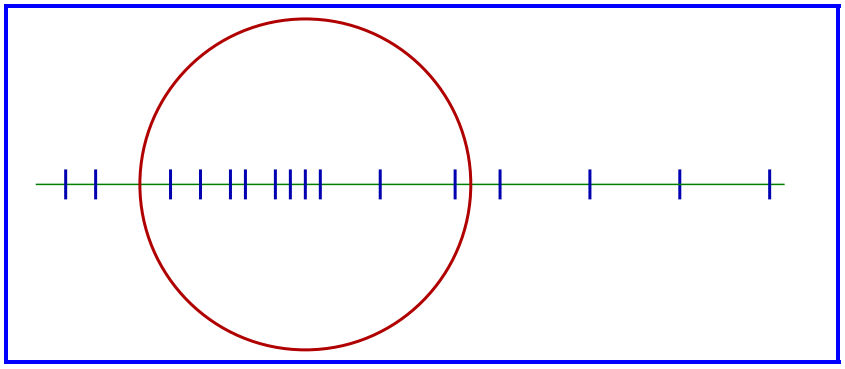
- Polynomial filtering utterly ineffective for this case
- Second issue: situation when Matrix-vector products are expensive
- Generalized eigenvalue problems.

- Alternative is to use rational filters:

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

$$\phi(A) = \sum_j \alpha_j (A - \sigma_j I)^{-1} \quad \rightarrow \quad \text{We now need to solve linear systems}$$

- Tool: Cauchy integral representations of spectral projectors

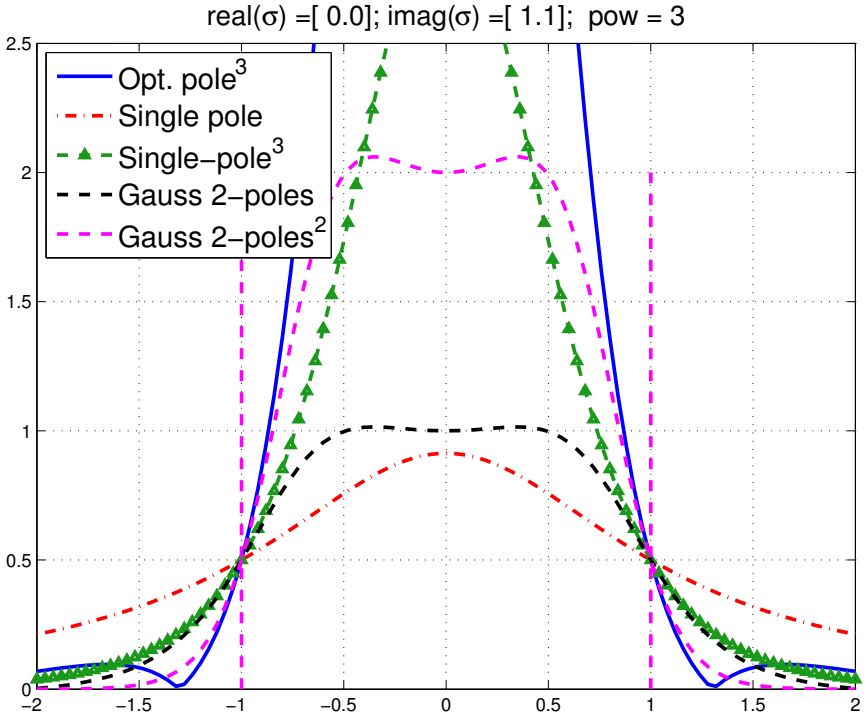


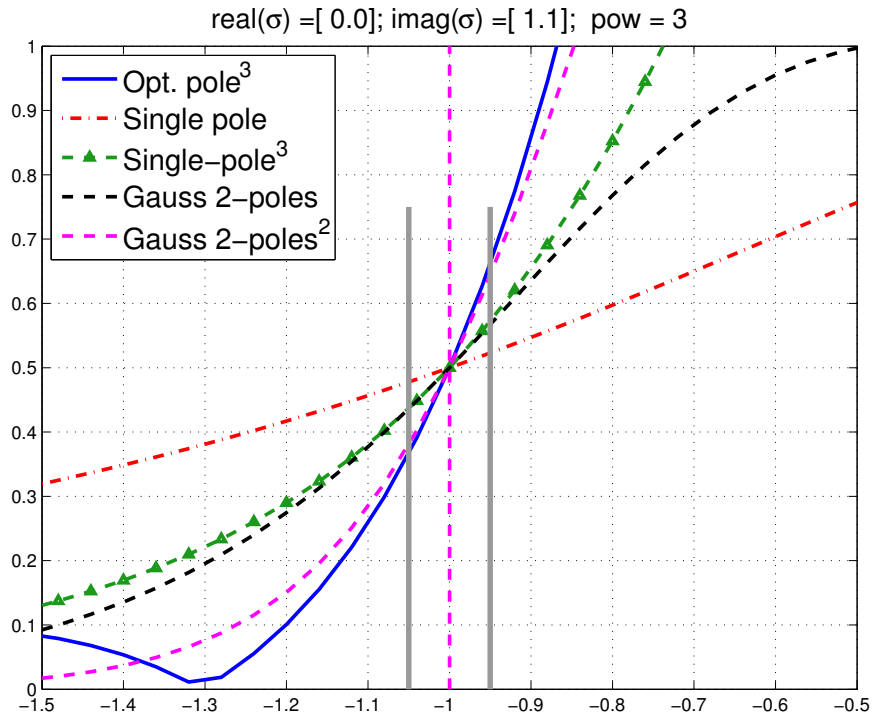
$$P = \frac{-1}{2i\pi} \int_\Gamma (A - sI)^{-1} ds$$

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

- Sakurai-Sugiura approach [Krylov]
- FEAST [Subs. iter.] (E. Polizzi)

What makes a good filter





➤ Assume subspace iteration is used with above filters. Which filter will give better convergence?

➤ Simplest and best indicator of performance of a filter is the magnitude of its derivative at -1 (or 1)

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

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

➤ Advantages:

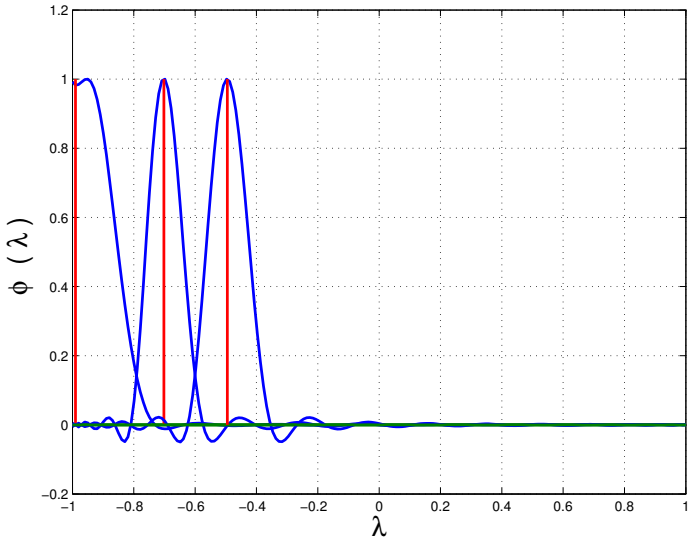
- Can select poles far away from real axis $\rightarrow$ faster iterative solvers
- Very flexible – can be adapted to many situations
- Can repeat poles (!)

➤ Implemented in EVSL.. [Interfaced to `SuiteSparse` as a solver]

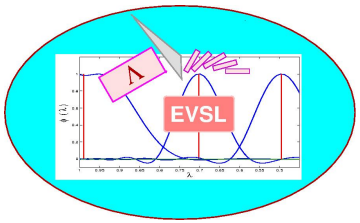
Spectrum Slicing and the EVSL project

- Newly released EVSL uses polynomial and rational filters
- Each can be appealing in different situations.

Spectrum slicing: cut the overall interval containing the spectrum into small sub-intervals and compute eigenpairs in each sub-interval independently.



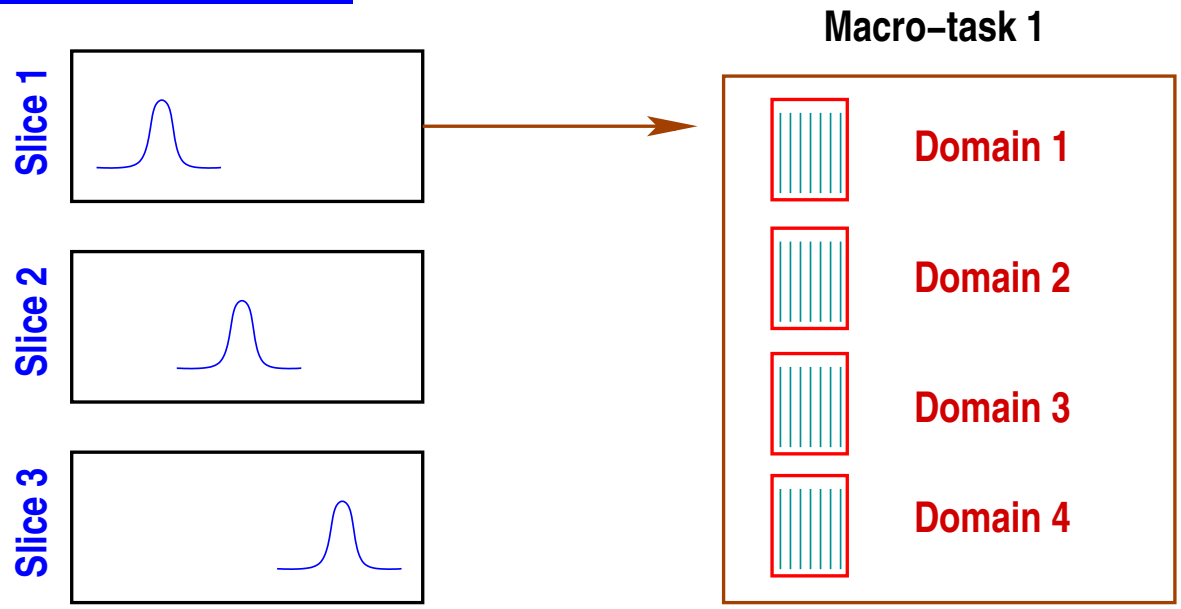
S O F T W A R E



EVSL NEW! a library of (sequential) eigensolvers based on spectrum slicing. **Version 1.0** released on [09/11/2016]
EVSL provides routines for computing eigenvalues located in a given interval, and their associated eigenvectors, of real symmetric matrices. It also provides tools for spectrum slicing, i.e., the technique of subdividing a given interval into p smaller subintervals and computing the eigenvalues in each subinterval independently. EVSL implements a polynomial filtered Lanczos algorithm (thick restart, no restart) a rational filtered Lanczos algorithm (thick restart, no restart), and a polynomial filtered subspace iteration.

ITSOL a library of (sequential) iterative solvers. **Version 2** released. [11/16/2010]
ITSOL can be viewed as an extension of the ITSOL module in the SPARSKIT package. It is written in C and aims at providing additional preconditioners for solving general sparse linear systems of equations. Preconditioners so far in this package include (1) ILUK (ILU preconditioner with level of fill) (2) ILUT (ILU preconditioner with threshold) (3) ILUC (Crout version of ILUT) (4) VBILUK (variable block preconditioner with level of fill - with automatic block detection) (5) VBILUT (variable block preconditioner with threshold - with automatic block detection) (6) ARMS (Algebraic Recursive Multilevel Solvers -- includes actually several methods - In particular the standard ARMS and the ddPQ version which uses nonsymmetric permutations).
ZITSOL a complex version of some of the methods in ITSOL is also available.

Levels of parallelism



The two main levels of parallelism in 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

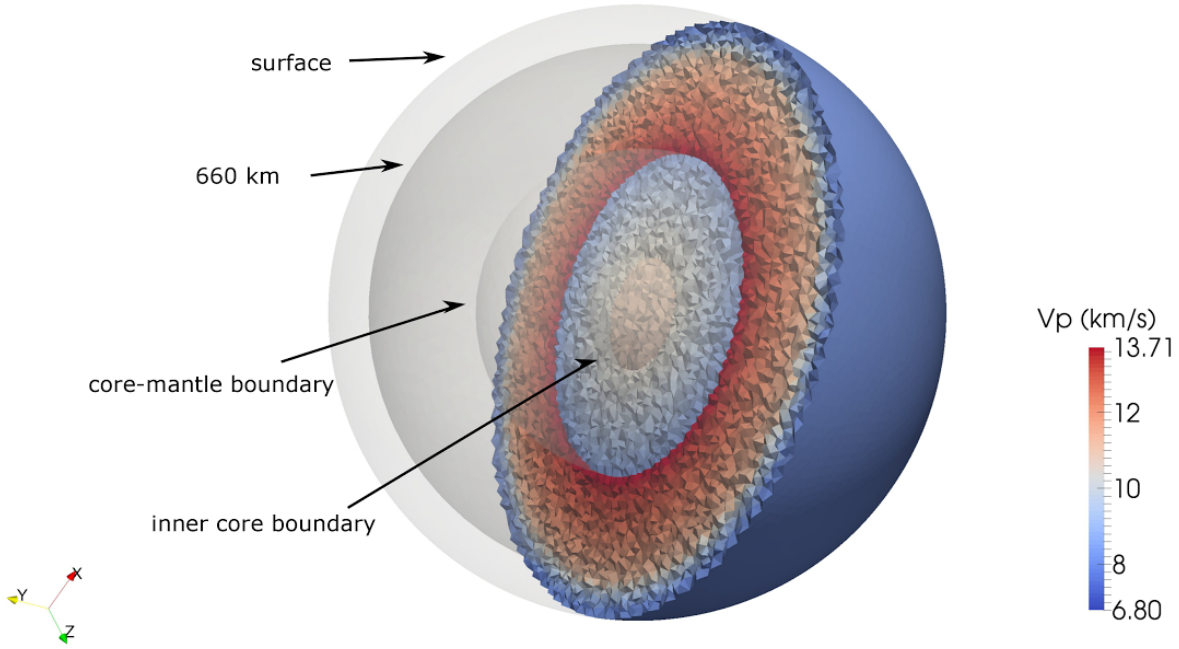
Version _1.1 Due for release end of July

- 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 Early 2018 (?)

- Fully parallel version [MPI + openMP]
- Challenge application in earth sciences [in progress]

Validation: Computing earth normal modes



- Collaboration with J. Shi & M. V. De Hoop - (Rice U.)
- FEM model leads to a generalized eigenvalue problem:

$$\begin{bmatrix} A_{CM} & & & B_{CMB} \\ & A_{IC} & & B_{ICB} \\ & & 0 & A_{dp} \\ B_{CMB}^T & B_{IMB}^T & A_{dp}^T & A_{OC} \end{bmatrix} \begin{bmatrix} u^{CM} \\ u^{IC} \\ u^{OC} \\ p \end{bmatrix} =$$

$$\omega^2 \begin{bmatrix} M_{CM} & & & \\ & M_{IC} & & \\ & & M_{OC} & \\ & & & 0 \end{bmatrix} \begin{bmatrix} u^{CM} \\ u^{IC} \\ u^{OC} \\ p \end{bmatrix}$$

- Want all eigen-values/vectors inside a given interval
- Issue: ‘mass’ matrix has a large null space..
- Solution: change formulation of matrix problem.

➤ ... Work in progress.

Conclusion

- Polynomial Filtering appealing when # of eigenpairs to be computed is large and Matvecs are cheap
- May be costly for generalized eigenvalue problems
- Will not work well for spectra with large outliers.
- Alternative: Rational filtering
- Both approaches implemented in EVSL
- Current focus of EVSL: provide as many interfaces as possible.
- `EVSL` code available here:

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