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
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 $\rightarrow$ 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’
Solving 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 Sub-space 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 $\delta$-Dirac functions

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

- W’ll express everything in the interval $[-1, 1]$
The Chebyshev expansion of $\delta_\gamma$ 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 $\lambda$’s such that $\rho(\lambda) \geq \bar{\text{bar}}$] we need to ...

➢ ... find $\gamma$ so that $\rho(\xi) = \rho(\eta)$

Procedure: Solve the equation $\rho_\gamma(\xi) - \rho_\gamma(\eta) = 0$ with respect to $\gamma$, 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 $p_k(t) = \text{polynomial of degree } k$

- 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 analysts, 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: $5 \times 10^{16} = 50$ 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 $\rho(\lambda_i) \geq \bar{\rho}$
- Ignore if $\rho(\lambda_i) < \bar{\rho}$
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

Locked vectors
Thick restart vectors
Lanczos 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

- 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:

$$
\tilde{B} = S^{-1}BS^{-1} \quad S = \text{diag}(B)^{1/2}
$$
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 $K_{uu}$, $M_{uu}$ from Suite Sparse collection.

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

\[ B^{-1/2} A B^{-1/2} u = \lambda u \]

- 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

<table>
<thead>
<tr>
<th>Matrix</th>
<th>n</th>
<th>$\sim$ nnz</th>
<th>$[a, b]$</th>
<th>$[\xi, \eta]$</th>
<th>$\nu_{[\xi, \eta]}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ge$<em>{87}$H$</em>{76}$</td>
<td>112,985</td>
<td>7.9M</td>
<td>$[-1.21, 32.76]$</td>
<td>$[-0.64, -0.0053]$</td>
<td>212</td>
</tr>
<tr>
<td>Ge$<em>{99}$H$</em>{100}$</td>
<td>112,985</td>
<td>8.5M</td>
<td>$[-1.22, 32.70]$</td>
<td>$[-0.65, -0.0096]$</td>
<td>250</td>
</tr>
<tr>
<td>Si$<em>{41}$Ge$</em>{41}$H$_{72}$</td>
<td>185,639</td>
<td>15.0M</td>
<td>$[-1.12, 49.82]$</td>
<td>$[-0.64, -0.0028]$</td>
<td>218</td>
</tr>
<tr>
<td>Si$<em>{87}$H$</em>{76}$</td>
<td>240,369</td>
<td>10.6M</td>
<td>$[-1.19, 43.07]$</td>
<td>$[-0.66, -0.0300]$</td>
<td>213</td>
</tr>
<tr>
<td>Ga$<em>{41}$As$</em>{41}$H$_{72}$</td>
<td>268,096</td>
<td>18.5M</td>
<td>$[-1.25, 1301]$</td>
<td>$[-0.64, -0.0000]$</td>
<td>201</td>
</tr>
</tbody>
</table>
**Results:** (Thick-Restart Lanczos)

<table>
<thead>
<tr>
<th>Matrix</th>
<th>deg</th>
<th>iter</th>
<th>matvec</th>
<th>CPU time (sec)</th>
<th>residual</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>matvec</td>
<td>total</td>
</tr>
<tr>
<td>Ge\textsubscript{87}H\textsubscript{76}</td>
<td>26</td>
<td>1431</td>
<td>37482</td>
<td>282.70</td>
<td>395.91</td>
</tr>
<tr>
<td>Ge\textsubscript{99}H\textsubscript{100}</td>
<td>26</td>
<td>1615</td>
<td>42330</td>
<td>338.76</td>
<td>488.91</td>
</tr>
<tr>
<td>Si\textsubscript{41}Ge\textsubscript{41}H\textsubscript{72}</td>
<td>35</td>
<td>1420</td>
<td>50032</td>
<td>702.32</td>
<td>891.98</td>
</tr>
<tr>
<td>Si\textsubscript{87}H\textsubscript{76}</td>
<td>30</td>
<td>1427</td>
<td>43095</td>
<td>468.48</td>
<td>699.90</td>
</tr>
<tr>
<td>Ga\textsubscript{41}As\textsubscript{41}H\textsubscript{72}</td>
<td>202</td>
<td>2334</td>
<td>471669</td>
<td>8179.51</td>
<td>9190.46</td>
</tr>
</tbody>
</table>

 век

Demo with Si10H16 \([n = 17,077, \text{nnz}(A) = 446,500]\)
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:

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

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

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. \( P \to \tilde{P} \)
- 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 $\alpha = 10$, $\beta = 0.1$

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

- Can select poles far away from real axis → 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.
**SOFTWARE**

**EVSL** 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.

http://www-users.cs.umn.edu/~saad/software/
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]
- $Ax = \lambda Bx$
- Fortran (03) interface.
- Spectrum slicing by Lanczos and KPM
- Efficient Spectrum slicing for $Ax = \lambda Bx$ (no solves with $B$).

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:
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](http://www.cs.umn.edu/~saad/software/EVSL)