Domain decomposition algorithms for the solution of sparse symmetric generalized eigenvalue problems

Vassilis Kalantzis

Computer Science and Engineering Department
University of Minnesota - Twin Cities, USA

Thesis defense (Supervisor: Yousef Saad)
9-6-2018

University of Minnesota

Supercomputing Institute

Acknowledgments

- To my committee members.
- The computational results featured in this thesis defense were performed using resources of the University of Minnesota Supercomputing Institute.
- CSE department, Gerondelis foundation, NSF, DOE.

Contents

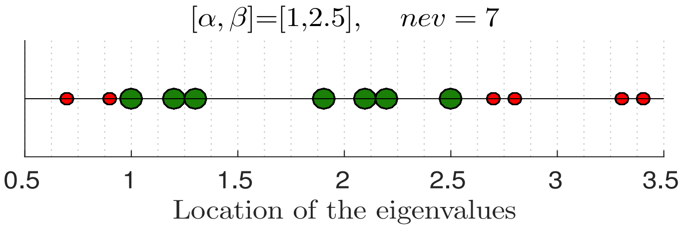
The algebraic generalized eigenvalue problem

The symmetric generalized eigenvalue problem is formally defined as

$$Ax = \lambda Mx.$$

Matrices A and M are assumed sparse and symmetric, while M is also SPD.

- The pencil (A, M) has n eigenpairs which we will denote by $\left(\lambda_i, x^{(i)}\right)$, $i = 1, \ldots, n$.
- We are only interested in computing those eigenpairs $\left(\lambda_i, x^{(i)}\right)$ for which $\lambda_i \in [\alpha, \beta]$.
- We will denote the number of eigenvalues which satisfy the above property by 'nev'.



Why care about solving large-scale eigenvalue problems?

Applications:

1. Frequency response analysis over a frequency range $\Omega = [\omega_\alpha, \omega_\beta]$ $((A - \omega M)z = f)$.
2. Density functional theory (Kohn-Sham equation).
3. Data analysis (spectral clustering, link prediction, recommender systems, centrality scores).

Why care about solving large-scale eigenvalue problems?

Applications:

1. Frequency response analysis over a frequency range $\Omega = [\omega_\alpha, \omega_\beta]$ ($(A - \omega M)z = f$).
2. Density functional theory (Kohn-Sham equation).
3. Data analysis (spectral clustering, link prediction, recommender systems, centrality scores).

Current trends/reality:

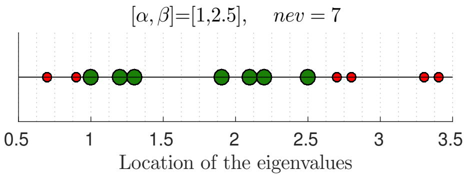
1. Problems get bigger; both in (a) size, and (b) number of eigenvalues-eigenvectors sought.
2. Orthogonalization is a well-known limitation; both in terms of (a) synchronization, and (b) FLOPS.
3. We need distributed memory implementations (easier said than done).

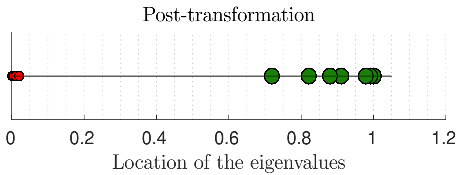
Landscape of large-scale eigenvalue solvers

- Projection schemes like Lanczos or Subspace Iteration are good at computing extremal eigenvalues (eigenvectors).
- If $[\alpha, \beta]$ is in the interior of the spectrum $\rightarrow$ spectral transformation.

Landscape of large-scale eigenvalue solvers

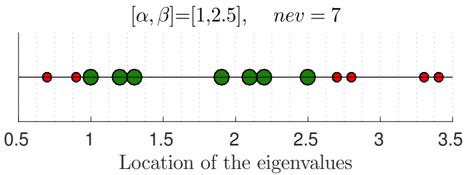
- Projection schemes like Lanczos or Subspace Iteration are good at computing extremal eigenvalues (eigenvectors).
- If $[\alpha, \beta]$ is in the interior of the spectrum $\rightarrow$ spectral transformation.

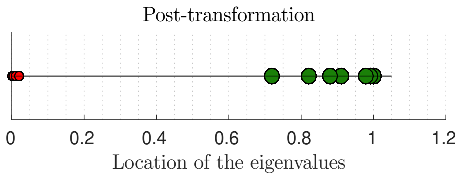




Landscape of large-scale eigenvalue solvers

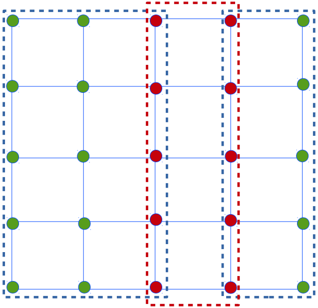
- Projection schemes like Lanczos or Subspace Iteration are good at computing extremal eigenvalues (eigenvectors).
- If $[\alpha, \beta]$ is in the interior of the spectrum $\rightarrow$ spectral transformation.



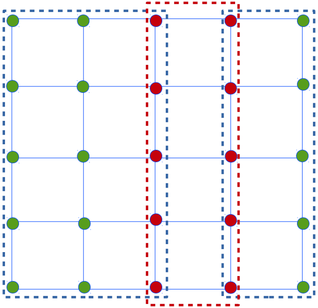


- Transformation: either (a) polynomial, or (b) rational.
- Each one has its own advantages/disadvantages.
- Spectral transformations form the main idea behind many state-of-the-art libraries (e.g., EVSL, FEAST).

About this thesis: domain decomposition eigenvalue solvers (I)



About this thesis: domain decomposition eigenvalue solvers (I)



DD decouples the original eigenvalue problem into two parts:

- The first part considers (only) interface (red) variables.
- The second part considers (only) interior (green) variables. (communication-free parallelism)

About this thesis: domain decomposition eigenvalue solvers (II)

This thesis is concerned with the development of domain decomposition algorithms for the solution of sparse symmetric generalized eigenvalue problems.

About this thesis: domain decomposition eigenvalue solvers (II)

This thesis is concerned with the development of domain decomposition algorithms for the solution of sparse symmetric generalized eigenvalue problems.

- We propose several numerical techniques to solve the interior/interface problems.
  - polynomial/rational filtering (*nev* is large).
  - root-finding (*nev* is small).
- The algorithms proposed in this thesis are implemented in multi-core/many-core architectures.

About this thesis: domain decomposition eigenvalue solvers (II)

This thesis is concerned with the development of domain decomposition algorithms for the solution of sparse symmetric generalized eigenvalue problems.

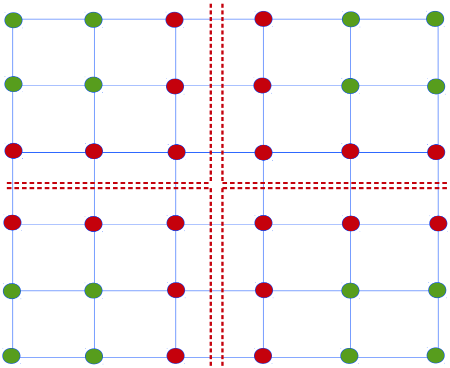
- We propose several numerical techniques to solve the interior/interface problems.
  - polynomial/rational filtering (*nev* is large).
  - root-finding (*nev* is small).
- The algorithms proposed in this thesis are implemented in multi-core/many-core architectures.
- Overall, this thesis contributes algorithms which:
  - Reduce orthogonalization costs.
  - Take advantage of current high-end computers.
  - (Potentially) Converge faster than techniques such as shift-and-invert Krylov or the Rayleigh Quotient Iteration.
- On the other hand, the proposed techniques typically deliver eigenpair approximations which typically are less accurate (is this always an issue?).

Contents

Reordering equations/unknowns ($p \geq 2$ subdomains)

$$A = \begin{pmatrix} B_1 & & & & E_1 \\ & B_2 & & & E_2 \\ & & \ddots & & \vdots \\ & & & B_p & E_p \\ E_1^T & E_2^T & \cdots & E_p^T & C \end{pmatrix},$$

$$M = \begin{pmatrix} M_B^{(1)} & & & & M_E^{(1)} \\ & M_B^{(2)} & & & M_E^{(2)} \\ & & \ddots & & \vdots \\ & & & M_B^{(p)} & M_E^{(p)} \\ \left(M_E^{(1)}\right)^T & \left(M_E^{(2)}\right)^T & \cdots & \left(M_E^{(p)}\right)^T & M_C \end{pmatrix}.$$



Reordering equations/unknowns ($p \geq 2$ subdomains)

$$A = \begin{pmatrix} B_1 & & & & E_1 \\ & B_2 & & & E_2 \\ & & \ddots & & \vdots \\ & & & B_p & E_p \\ E_1^T & E_2^T & \cdots & E_p^T & C \end{pmatrix},$$

$$M = \begin{pmatrix} M_B^{(1)} & & & & M_E^{(1)} \\ & M_B^{(2)} & & & M_E^{(2)} \\ & & \ddots & & \vdots \\ & & & M_B^{(p)} & M_E^{(p)} \\ \left(M_E^{(1)}\right)^T & \left(M_E^{(2)}\right)^T & \cdots & \left(M_E^{(p)}\right)^T & M_C \end{pmatrix}.$$

Notation: write as

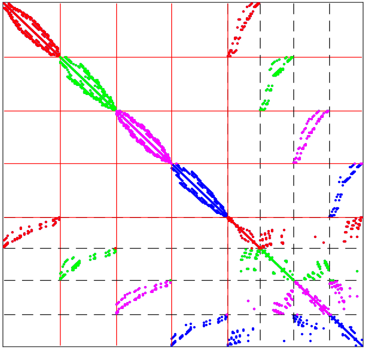
$$A = \begin{pmatrix} B & E \\ E^T & C \end{pmatrix}, M = \begin{pmatrix} M_B & M_E \\ M_E^T & M_C \end{pmatrix},$$

$$x^{(i)} = \begin{pmatrix} u^{(i)} \\ y^{(i)} \end{pmatrix} = \begin{pmatrix} u_1^{(i)} \\ \vdots \\ u_p^{(i)} \\ y_1^{(i)} \\ \vdots \\ y_p^{(i)} \end{pmatrix}.$$

An example of the sparsity pattern of A and M for $p = 4$

$$A = \begin{pmatrix} B & E \\ E^T & C \end{pmatrix} = \begin{pmatrix} B_1 & & & & E_1 \\ & B_2 & & & E_2 \\ & & \ddots & & \vdots \\ & & & B_p & E_p \\ E_1^T & E_2^T & \cdots & E_p^T & C \end{pmatrix}$$

Sparsity pattern of matrix $|A| + |M|$



Invariant subspaces from a Schur complement viewpoint

$$(A - \lambda_i M)x^{(i)} = \begin{pmatrix} B - \lambda_i M_B & E - \lambda_i M_E \\ E^T - \lambda_i M_E^T & C - \lambda_i M_C \end{pmatrix} \begin{pmatrix} u^{(i)} \\ y^{(i)} \end{pmatrix} = 0.$$

Invariant subspaces from a Schur complement viewpoint

$$(A-\lambda_i M)x^{(i)} = \begin{pmatrix} B-\lambda_i M_B & E-\lambda_i M_E \\ E^T-\lambda_i M_E^T & C-\lambda_i M_C \end{pmatrix} \begin{pmatrix} u^{(i)} \\ y^{(i)} \end{pmatrix} = 0.$$

Eliminating $u^{(i)}$ from the first block of rows gives:

Invariant subspaces from a Schur complement viewpoint

$$(A-\lambda_i M)x^{(i)} = \begin{pmatrix} B-\lambda_i M_B & E-\lambda_i M_E \\ E^T-\lambda_i M_E^T & C-\lambda_i M_C \end{pmatrix} \begin{pmatrix} u^{(i)} \\ y^{(i)} \end{pmatrix} = 0.$$

Eliminating $u^{(i)}$ from the first block of rows gives:

$$\left[C-\lambda_i M_C - \underbrace{(E-\lambda_i M_E)^T (B-\lambda_i M_B)^{-1} (E-\lambda_i M_E)}_{\text{block-diagonal}} \right] y^{(i)} = 0,$$

Invariant subspaces from a Schur complement viewpoint

$$(A - \lambda_i M)x^{(i)} = \begin{pmatrix} B - \lambda_i M_B & E - \lambda_i M_E \\ E^T - \lambda_i M_E^T & C - \lambda_i M_C \end{pmatrix} \begin{pmatrix} u^{(i)} \\ y^{(i)} \end{pmatrix} = 0.$$

Eliminating $u^{(i)}$ from the first block of rows gives:

$$\left[C - \lambda_i M_C - \underbrace{(E - \lambda_i M_E)^T (B - \lambda_i M_B)^{-1} (E - \lambda_i M_E)}_{\text{block-diagonal}} \right] y^{(i)} = 0,$$

$$\mathrm{u}^{(i)} = -\underbrace{(B - \lambda_i M_B)^{-1} (E - \lambda_i M_E)}_{\text{block-diagonal}} y^{(i)}.$$

Invariant subspaces from a Schur complement viewpoint

$$(A-\lambda_i M)x^{(i)} = \begin{pmatrix} B-\lambda_i M_B & E-\lambda_i M_E \\ E^T-\lambda_i M_E^T & C-\lambda_i M_C \end{pmatrix} \begin{pmatrix} u^{(i)} \\ y^{(i)} \end{pmatrix} = 0.$$

Eliminating $u^{(i)}$ from the first block of rows gives:

$$\left[C-\lambda_i M_C - \underbrace{(E-\lambda_i M_E)^T (B-\lambda_i M_B)^{-1} (E-\lambda_i M_E)}_{\text{block-diagonal}} \right] y^{(i)} = 0,$$

$$\mathrm{u}^{(i)} = -\underbrace{(B-\lambda_i M_B)^{-1}(E-\lambda_i M_E)}_{\text{block-diagonal}} y^{(i)}.$$

To compute the eigenpairs $\left(\lambda_i, x^{(i)}\right)_{i=1,\ldots,nev}$

Perform a Rayleigh-Ritz projection onto $\mathcal{Z} = \mathcal{U} \oplus \mathcal{Y}$:

$$\mathcal{Y} = \operatorname{span}\left\{y^{(i)}\right\}_{i=1,\ldots,nev},$$

$$\mathcal{U} = \operatorname{span}\left\{-(B-\lambda_i M_B)^{-1}(E-\lambda_i M_E)y^{(i)}\right\}_{i=1,\ldots,nev}.$$

Contents

Rational filtering

- We consider the following rational filter

$$\rho(\zeta) = \sum_{\ell=1}^{2N_c} \frac{\omega_\ell}{\zeta - \zeta_\ell} \approx \underbrace{\frac{1}{2\pi i} \int_{\Gamma_{[\alpha,\beta]}} \frac{1}{\nu - \zeta} d\nu}_{I_{[\alpha,\beta]}(\zeta)}$$

Rational filtering

- We consider the following rational filter

$$\rho(\zeta) = \sum_{\ell=1}^{2N_c} \frac{\omega_\ell}{\zeta - \zeta_\ell} \approx \underbrace{\frac{1}{2\pi i} \int_{\Gamma_{[\alpha,\beta]}} \frac{1}{\nu - \zeta} d\nu}_{I_{[\alpha,\beta]}(\zeta)}$$

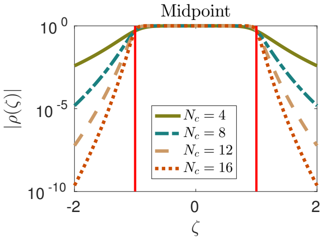
$$= 2\Re e \left\{ \sum_{\ell=1}^{N_c} \frac{\omega_\ell}{\zeta - \zeta_\ell} \right\}.$$

Rational filtering

- We consider the following rational filter

$$\rho(\zeta) = \sum_{\ell=1}^{2N_c} \frac{\omega_\ell}{\zeta - \zeta_\ell} \approx \underbrace{\frac{1}{2\pi i}\int_{\Gamma_{[\alpha,\beta]}} \frac{1}{\nu - \zeta} d\nu}_{I_{[\alpha,\beta]}(\zeta)}$$

$$= 2\Re e\left\{\sum_{\ell=1}^{N_c} \frac{\omega_\ell}{\zeta - \zeta_\ell}\right\}.$$

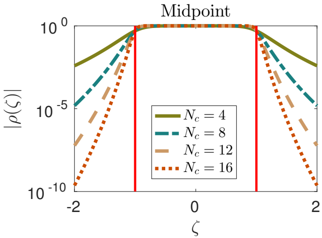


Rational filtering

- We consider the following rational filter

$$\rho(\zeta) = \sum_{\ell=1}^{2N_c} \frac{\omega_\ell}{\zeta - \zeta_\ell} \approx \underbrace{\frac{1}{2\pi i} \int_{\Gamma_{[\alpha,\beta]}} \frac{1}{\nu - \zeta} d\nu}_{I_{[\alpha,\beta]}(\zeta)}$$

$$= 2\Re e \left\{ \sum_{\ell=1}^{N_c} \frac{\omega_\ell}{\zeta - \zeta_\ell} \right\}.$$



- It is possible to apply $\rho(.)$ to (A, M):

$$\rho(M^{-1}A) = 2\Re e \left\{ \sum_{\ell=1}^{N_c} \omega_\ell (A - \zeta_\ell M)^{-1} M \right\}.$$

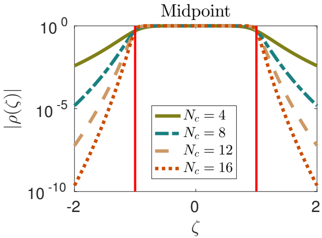
- Examples: FEAST (Subspace Iteration), Sakurai-Sugiura (Moments-based).
- Krylov projection schemes are also possible (RF-KRYLOV).

Rational filtering

- We consider the following rational filter

$$\rho(\zeta) = \sum_{\ell=1}^{2N_c} \frac{\omega_\ell}{\zeta - \zeta_\ell} \approx \underbrace{\frac{1}{2\pi i} \int_{\Gamma_{[\alpha,\beta]}} \frac{1}{\nu - \zeta} d\nu}_{I_{[\alpha,\beta]}(\zeta)}$$

$$= 2\Re e \left\{ \sum_{\ell=1}^{N_c} \frac{\omega_\ell}{\zeta - \zeta_\ell} \right\}.$$



- It is possible to apply $\rho(.)$ to (A, M):

$$\rho(M^{-1}A) = 2\Re e \left\{ \sum_{\ell=1}^{N_c} \omega_\ell (A - \zeta_\ell M)^{-1} M \right\}.$$

  - Examples: FEAST (Subspace Iteration), Sakurai-Sugiura (Moments-based).
  - Krylov projection schemes are also possible (RF-KRYLOV).
- Our idea: Decouple application of $\rho(\zeta)$ to interior/interface variables.
- Potential advantages:
  1. Reduced use of complex arithmetic.
  2. Orthonormalization of shorter vectors (interface variables).
  3. Faster convergence.

Summary of the proposed technique

- Our goal is to construct a subspace $\mathcal{Z} = \mathcal{U} \oplus \mathcal{Y}$ to perform a Rayleigh-Ritz projection onto.
- Recall that, ideally,

$$\mathcal{Y} = \mathrm{span}\left\{y^{(i)}\right\}_{i=1,\ldots,nev},$$

$$\mathcal{U} = \mathrm{span}\left\{-(B-\lambda_i M_B)^{-1}(E-\lambda_i M_E)y^{(i)}\right\}_{i=1,\ldots,nev}.$$

Summary of the proposed technique

- Our goal is to construct a subspace $\mathcal{Z} = \mathcal{U} \oplus \mathcal{Y}$ to perform a Rayleigh-Ritz projection onto.
- Recall that, ideally,

$$\mathcal{Y} = \mathrm{span}\left\{y^{(i)}\right\}_{i=1,\ldots,nev},$$
$$\mathcal{U} = \mathrm{span}\left\{-(B - \lambda_i M_B)^{-1}(E - \lambda_i M_E)y^{(i)}\right\}_{i=1,\ldots,nev}.$$

The technique proposed in this section:

1. Constructs $\mathcal{Y}$ by applying the rational filter $\rho(\zeta)$ to the interface region (Schur complement matrices).
2. Uses the above subspace to construct $\mathcal{U}$. This step is performed in real arithmetic and is embarrassingly parallel.

How to approximate $\mathrm{span}\left\{y^{(1)},\ldots,y^{(nev)}\right\}$ (I)

Let $\zeta \in \mathbb{C}$ and define

$$B_\zeta = B - \zeta M_B, \quad E_\zeta = E - \zeta M_E, \quad C_\zeta = C - \zeta M_C,$$
$$S(\zeta) = C_\zeta - E_\zeta^T B_\zeta^{-1} E_\zeta.$$

How to approximate $\operatorname{span}\{y^{(1)}, \ldots, y^{(nev)}\}$ (I)

Let $\zeta \in \mathbb{C}$ and define

$$B_\zeta = B - \zeta M_B, \quad E_\zeta = E - \zeta M_E, \quad C_\zeta = C - \zeta M_C,$$
$$S(\zeta) = C_\zeta - E_\zeta^T B_\zeta^{-1} E_\zeta.$$

Then,

$$(A - \zeta M)^{-1} = \begin{pmatrix} B_\zeta^{-1} + B_\zeta^{-1} E_\zeta S(\zeta)^{-1} E_\zeta^T B_\zeta^{-1} & -B_\zeta^{-1} E_\zeta S(\zeta)^{-1} \\ -S(\zeta)^{-1} E_\zeta^T B_\zeta^{-1} & S(\zeta)^{-1} \end{pmatrix}.$$

How to approximate $\mathrm{span}\left\{y^{(1)},\ldots,y^{(nev)}\right\}$ (I)

Let $\zeta \in \mathbb{C}$ and define

$$B_\zeta = B - \zeta M_B, \quad E_\zeta = E - \zeta M_E, \quad C_\zeta = C - \zeta M_C,$$
$$S(\zeta) = C_\zeta - E_\zeta^T B_\zeta^{-1} E_\zeta.$$

Then,

$$(A - \zeta M)^{-1} = \begin{pmatrix} B_\zeta^{-1} + B_\zeta^{-1} E_\zeta S(\zeta)^{-1} E_\zeta^T B_\zeta^{-1} & -B_\zeta^{-1} E_\zeta S(\zeta)^{-1} \\ -S(\zeta)^{-1} E_\zeta^T B_\zeta^{-1} & S(\zeta)^{-1} \end{pmatrix}.$$

The matrix inverse $(A - \zeta M)^{-1}$ can be also written as:

$$(A - \zeta M)^{-1} = \sum_{i=1}^{n} \frac{1}{\lambda_i - \zeta} x^{(i)} \left(x^{(i)}\right)^T = \sum_{i=1}^{n} \frac{1}{\lambda_i - \zeta} \begin{bmatrix} u^{(i)} \left(u^{(i)}\right)^T & u^{(i)} \left(y^{(i)}\right)^T \\ y^{(i)} \left(u^{(i)}\right)^T & y^{(i)} \left(y^{(i)}\right)^T \end{bmatrix}.$$

How to approximate span $\left\{y^{(1)},\ldots,y^{(nev)}\right\}$ (II)

Recall that

$$\rho(M^{-1}A) = 2\Re e\left\{\sum_{\ell=1}^{N_c}\omega_\ell(A-\zeta_\ell M)^{-1}M\right\}.$$

Combining alltogether we get:

$$\rho(M^{-1}A) = 2\Re e\left\{\sum_{\ell=1}^{N_c}\omega_\ell\begin{bmatrix} B_{\zeta_\ell}^{-1}+B_{\zeta_\ell}^{-1}E_{\zeta_\ell}S(\zeta_\ell)^{-1}E_{\zeta_\ell}^TB_{\zeta_\ell}^{-1} & -B_{\zeta_\ell}^{-1}E_{\zeta_\ell}S(\zeta_\ell)^{-1} \\ -S(\zeta_\ell)^{-1}E_{\zeta_\ell}^TB_{\zeta_\ell}^{-1} & \boxed{S(\zeta_\ell)^{-1}}\end{bmatrix}\right\}M$$

How to approximate $\mathrm{span}\left\{y^{(1)},\ldots,y^{(nev)}\right\}$ (II)

Recall that

$$\rho(M^{-1}A) = 2\Re e\left\{\sum_{\ell=1}^{N_c}\omega_\ell(A-\zeta_\ell M)^{-1}M\right\}.$$

Combining alltogether we get:

$$\rho(M^{-1}A) = 2\Re e\left\{\sum_{\ell=1}^{N_c}\omega_\ell\begin{bmatrix} B_{\zeta_\ell}^{-1}+B_{\zeta_\ell}^{-1}E_{\zeta_\ell}S(\zeta_\ell)^{-1}E_{\zeta_\ell}^TB_{\zeta_\ell}^{-1} & -B_{\zeta_\ell}^{-1}E_{\zeta_\ell}S(\zeta_\ell)^{-1} \\ -S(\zeta_\ell)^{-1}E_{\zeta_\ell}^TB_{\zeta_\ell}^{-1} & \boxed{S(\zeta_\ell)^{-1}}\end{bmatrix}\right\}M$$

$$= \sum_{i=1}^{n}\rho(\lambda_i)\begin{bmatrix} u^{(i)}\left(u^{(i)}\right)^T & u^{(i)}\left(y^{(i)}\right)^T \\ y^{(i)}\left(u^{(i)}\right)^T & \boxed{y^{(i)}\left(y^{(i)}\right)^T}\end{bmatrix}M. \quad \left(\rho(\lambda_i) = 2\Re e\left\{\sum_{\ell=1}^{N_c}\frac{\omega_\ell}{\lambda_i-\zeta_\ell}\right\}\right)$$

How to approximate $\mathrm{span}\left\{y^{(1)},\ldots,y^{(nev)}\right\}$ (III)

Equating blocks leads to:

$$2\Re e\left\{\sum_{\ell=1}^{N_c}\omega_\ell S(\zeta_\ell)^{-1}\right\}=\sum_{i=1}^{n}\rho(\lambda_i)y^{(i)}\left(y^{(i)}\right)^T.$$

Since $\rho(\lambda_1),\ldots,\rho(\lambda_{nev})\neq 0$:

$$\mathrm{span}\left\{y^{(1)},\ldots,y^{(nev)}\right\}\subseteq\mathrm{range}\left(2\Re e\left\{\sum_{\ell=1}^{N_c}\omega_\ell S(\zeta_\ell)^{-1}\right\}\right).$$

How to approximate $\mathrm{span}\left\{y^{(1)},\ldots,y^{(nev)}\right\}$ (III)

Equating blocks leads to:

$$2\Re e\left\{\sum_{\ell=1}^{N_c}\omega_\ell S(\zeta_\ell)^{-1}\right\}=\sum_{i=1}^{n}\rho(\lambda_i)y^{(i)}\left(y^{(i)}\right)^T.$$

Since $\rho(\lambda_1),\ldots,\rho(\lambda_{nev})\neq 0$:

$$\mathrm{span}\left\{y^{(1)},\ldots,y^{(nev)}\right\}\subseteq \mathrm{range}\left(2\Re e\left\{\sum_{\ell=1}^{N_c}\omega_\ell S(\zeta_\ell)^{-1}\right\}\right).$$

Capture $\mathrm{range}\left(\Re e\left\{\sum_{\ell=1}^{N_c}\omega_\ell S(\zeta_\ell)^{-1}\right\}\right)$ by a Krylov projection scheme.

How to approximate $\operatorname{span}\left\{y^{(1)}, \ldots, y^{(nev)}\right\}$ (IV)

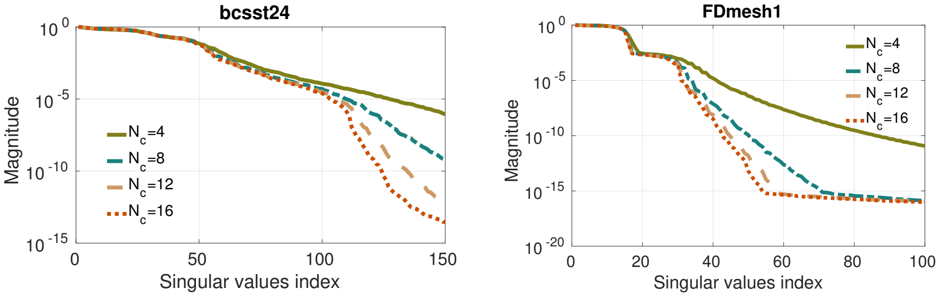


Figure: Leading singular values of $2\Re e\left\{\sum_{\ell=1}^{N_c} \omega_\ell S(\zeta_\ell)^{-1}\right\} = \sum_{i=1}^{n} \rho(\lambda_i) y^{(i)} \left(y^{(i)}\right)^T, \quad ([\alpha, \beta] = [\lambda_1, \lambda_{100}])$.

How to approximate $\mathrm{span}\left\{y^{(1)},\ldots,y^{(nev)}\right\}$ (IV)

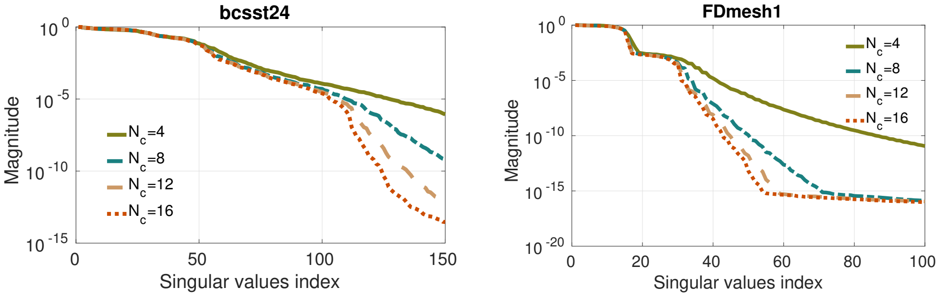


Figure: Leading singular values of $2\Re e\left\{\sum_{\ell=1}^{N_c}\omega_\ell S(\zeta_\ell)^{-1}\right\}=\sum_{i=1}^{n}\rho(\lambda_i)y^{(i)}\left(y^{(i)}\right)^T,\quad([\alpha,\beta]=[\lambda_1,\lambda_{100}])$.

What if $\mathrm{rank}\left(\left[y^{(1)},\ldots,y^{(nev)}\right]\right)<nev$?

Finalizing the proposed scheme (RF-DDES)

- Ideally, $\mathcal{U} = \left\{ u^{(1)}, \ldots, u^{(nev)} \right\}$, where

$$
\begin{aligned}
u^{(i)} &= -B_{\lambda_i}^{-1} E_{\lambda_i} y^{(i)} \\
&= -\left(B_{\lambda_i}^{-1} E_\sigma + (\lambda_i - \sigma) B_{\lambda_i}^{-1} M_E \right) y^{(i)}.
\end{aligned}
$$

Finalizing the proposed scheme (RF-DDES)

- Ideally, $\mathcal{U} = \left\{ u^{(1)}, \ldots, u^{(nev)} \right\}$, where

$$\begin{aligned} u^{(i)} &= -B_{\lambda_i}^{-1} E_{\lambda_i} y^{(i)} \\ &= -\left(B_{\lambda_i}^{-1} E_\sigma + (\lambda_i - \sigma) B_{\lambda_i}^{-1} M_E \right) y^{(i)}. \end{aligned}$$

- Set

$$B_{\lambda_i}^{-1} \approx B_\sigma^{-1} \sum_{k=0}^{\psi-1} (\lambda_i - \sigma) M_B B_\sigma^{-1}.$$

Finalizing the proposed scheme (RF-DDES)

- Ideally, $\mathcal{U} = \left\{ u^{(1)}, \ldots, u^{(nev)} \right\}$, where

$$
\begin{aligned}
u^{(i)} &= -B_{\lambda_i}^{-1} E_{\lambda_i} y^{(i)} \\
&= -\left(B_{\lambda_i}^{-1} E_\sigma + (\lambda_i - \sigma) B_{\lambda_i}^{-1} M_E \right) y^{(i)}.
\end{aligned}
$$

- Set

$$
B_{\lambda_i}^{-1} \approx B_\sigma^{-1} \sum_{k=0}^{\psi-1} (\lambda_i - \sigma) M_B B_\sigma^{-1}.
$$

- We finally set $\mathcal{U} = \mathrm{span}([V, U_1, U_2])$ where

$$
U_1 = -\left[B_\sigma^{-1} E_\sigma Y, \ldots, (B_\sigma^{-1} M_B)^{\psi-1} B_\sigma^{-1} E_\sigma Y \right],
$$

$$
U_2 = \left[B_\sigma^{-1} M_E Y, \ldots, (B_\sigma^{-1} M_B)^{\psi-1} B_\sigma^{-1} M_E Y \right],
$$

- V includes the eigenvectors associated with the $nev_B p$ smallest eigenvalues of (B_σ, M_B).

Finalizing the proposed scheme (RF-DDES)

- Ideally, $\mathcal{U} = \left\{u^{(1)}, \ldots, u^{(nev)}\right\}$, where

$$\begin{aligned} u^{(i)} &= -B_{\lambda_i}^{-1} E_{\lambda_i} y^{(i)} \\ &= -\left(B_{\lambda_i}^{-1} E_\sigma + (\lambda_i - \sigma) B_{\lambda_i}^{-1} M_E\right) y^{(i)}. \end{aligned}$$

- Set

$$B_{\lambda_i}^{-1} \approx B_\sigma^{-1} \sum_{k=0}^{\psi-1} (\lambda_i - \sigma) M_B B_\sigma^{-1}.$$

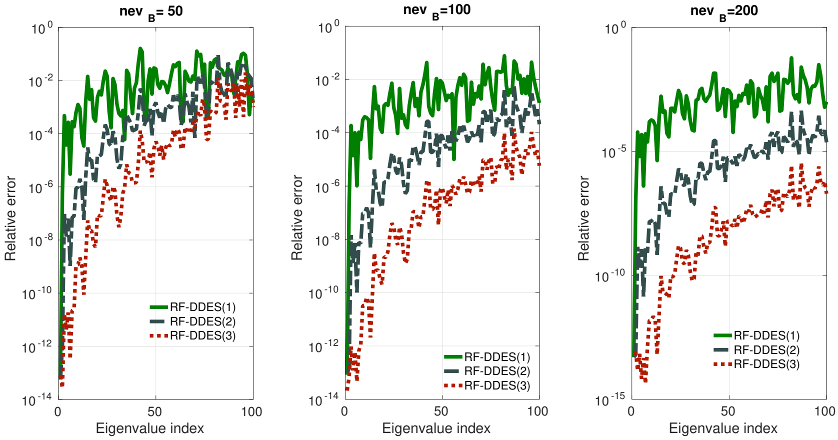
- We finally set $\mathcal{U} = \mathrm{span}([V, U_1, U_2])$ where

$$U_1 = -\left[B_\sigma^{-1} E_\sigma Y, \ldots, (B_\sigma^{-1} M_B)^{\psi-1} B_\sigma^{-1} E_\sigma Y\right],$$

$$U_2 = \left[B_\sigma^{-1} M_E Y, \ldots, (B_\sigma^{-1} M_B)^{\psi-1} B_\sigma^{-1} M_E Y\right],$$

- V includes the eigenvectors associated with the $nev_B p$ smallest eigenvalues of (B_σ, M_B).
- $\left\|u^{(i)} - \hat{u}^{(i)}\right\|_{M_B} \leq \max_{\ell \geq (nev_B p)+1} O\left(\frac{(\lambda_i - \sigma)^{\psi+1}}{(\delta_\ell - \lambda_i)(\delta_\ell - \sigma)^{\psi}}\right).$

Approximation of the *nev* = 100 algebraically smallest eigenvalues of pencil qa8fk/qa8fm



A comparison of RF-KRYLOV and RF-DDES (I)

Table: Wall-clock times of RF-KRYLOV and RF-DDES using $\tau = 2,\ 4,\ 8,\ 16$ and $\tau = 32$ computational cores. RFD(2) and RFD(4) denote RF-DDES with $p = 2$ and $p = 4$ subdomains, respectively.

	$nev = 100$			$nev = 200$			$nev = 300$		
Matrix	RFK	RFD(2)	RFD(4)	RFK	RFD(2)	RFD(4)	RFK	RFD(2)	RFD(4)
shipsec8($\tau = 2$)	114	195	-	195	207	-	279	213	-
($\tau = 4$)	76	129	93	123	133	103	168	139	107
($\tau = 8$)	65	74	56	90	75	62	127	79	68
($\tau = 16$)	40	51	36	66	55	41	92	57	45
($\tau = 32$)	40	36	28	62	41	30	75	43	34
boneS01($\tau = 2$)	94	292	-	194	356	-	260	424	-
($\tau = 4$)	68	182	162	131	230	213	179	277	260
($\tau = 8$)	49	115	113	94	148	152	121	180	187
($\tau = 16$)	44	86	82	80	112	109	93	137	132
($\tau = 32$)	51	66	60	74	86	71	89	105	79

A comparison of RF-KRYLOV and RF-DDES (II)

Table: Wall-clock times of RF-KRYLOV and RF-DDES using $\tau = 2,\ 4,\ 8,\ 16$ and $\tau = 32$ computational cores. RFD(2) and RFD(4) denote RF-DDES with $p = 2$ and $p = 4$ subdomains, respectively.

	$nev = 100$			$nev = 200$			$nev = 300$		
Matrix	RFK	RFD(2)	RFD(4)	RFK	RFD(2)	RFD(4)	RFK	RFD(2)	RFD(4)
FDmesh2($\tau = 2$)	241	85	-	480	99	-	731	116	-
($\tau = 4$)	159	34	63	305	37	78	473	43	85
($\tau = 8$)	126	22	23	228	24	27	358	27	31
($\tau = 16$)	89	16	15	171	17	18	256	20	21
($\tau = 32$)	51	12	12	94	13	14	138	15	20
FDmesh3($\tau = 2$)	1021	446	-	2062	502	-	3328	564	-
($\tau = 4$)	718	201	281	1281	217	338	1844	237	362
($\tau = 8$)	423	119	111	825	132	126	1250	143	141
($\tau = 16$)	355	70	66	684	77	81	1038	88	93
($\tau = 32$)	177	47	49	343	51	58	706	62	82

Amount of time spent on orthonormalization

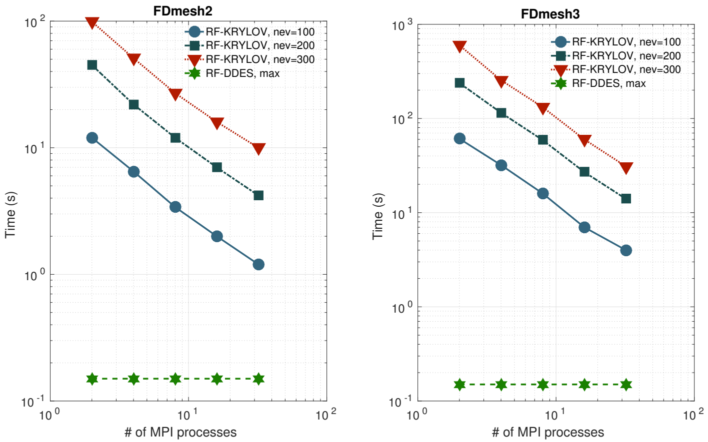


Figure: Left: “FDmesh2” ($n = 250,000$). Right: “FDmesh3” ($n = 1,000,000$).

Contents

Summary of the proposed technique

- Our goal is to construct a subspace $\mathcal{Z} = \mathcal{U} \oplus \mathcal{Y}$ to perform a Rayleigh-Ritz projection onto.
- Recall that, ideally,

$$\mathcal{Y} = \text{span}\left\{y^{(i)}\right\}_{i=1,\ldots,nev},$$
$$\mathcal{U} = \text{span}\left\{u^{(i)} \equiv -(B - \lambda_i M_B)^{-1}(E - \lambda_i M_E)y^{(i)}\right\}_{i=1,\ldots,nev}.$$

Summary of the proposed technique

- Our goal is to construct a subspace $\mathcal{Z} = \mathcal{U} \oplus \mathcal{Y}$ to perform a Rayleigh-Ritz projection onto.
- Recall that, ideally,

$$\mathcal{Y} = \mathrm{span}\left\{y^{(i)}\right\}_{i=1,\ldots,nev},$$
$$\mathcal{U} = \mathrm{span}\left\{u^{(i)} \equiv -(B - \lambda_i M_B)^{-1}(E - \lambda_i M_E)y^{(i)}\right\}_{i=1,\ldots,nev}.$$

The technique proposed in this section:

1. Builds both $\mathcal{Y}$ and $\mathcal{U}$ by using the rational filter $\rho(\zeta)$ (but without using Krylov).
2. Considers more than one levels of distributed memory parallelism.
3. Considers the use of preconditioned iterative linear system solvers.

DD-PP and rational filtering (I)

Recall that

$$(\zeta M - A)^{-1} = \begin{pmatrix} -\left[B_\zeta^{-1} + B_\zeta^{-1} E_\zeta S(\zeta)^{-1} E_\zeta^H B_\zeta^{-1}\right] & B_\zeta^{-1} E_\zeta S(\zeta)^{-1} \\ S(\zeta)^{-1} E_\zeta^T B_\zeta^{-1} & -S(\zeta)^{-1} \end{pmatrix}, \quad x^{(i)} = \begin{pmatrix} u^{(i)} \\ y^{(i)} \end{pmatrix}.$$

DD-PP and rational filtering (I)

Recall that

$$(\zeta M - A)^{-1} = \begin{pmatrix} -\left[B_\zeta^{-1} + B_\zeta^{-1} E_\zeta S(\zeta)^{-1} E_\zeta^H B_\zeta^{-1}\right] & B_\zeta^{-1} E_\zeta S(\zeta)^{-1} \\ S(\zeta)^{-1} E_\zeta^T B_\zeta^{-1} & -S(\zeta)^{-1} \end{pmatrix}, \quad x^{(i)} = \begin{pmatrix} u^{(i)} \\ y^{(i)} \end{pmatrix}.$$

Again, we have

$$\rho(M^{-1}A) = 2\Re e \left\{ \sum_{\ell=1}^{N_c} \omega_\ell \begin{bmatrix} -\left[B_{\zeta_\ell}^{-1} + B_{\zeta_\ell}^{-1} E_{\zeta_\ell} S(\zeta_\ell)^{-1} E_{\zeta_\ell}^T B_{\zeta_\ell}^{-1}\right] & \boxed{B_{\zeta_\ell}^{-1} E_{\zeta_\ell} S(\zeta_\ell)^{-1}} \\ S(\zeta_\ell)^{-1} E_{\zeta_\ell}^T B_{\zeta_\ell}^{-1} & \boxed{-S(\zeta_\ell)^{-1}} \end{bmatrix} \right\} M$$

DD-PP and rational filtering (I)

Recall that

$$(\zeta M - A)^{-1} = \begin{pmatrix} -\left[B_\zeta^{-1} + B_\zeta^{-1} E_\zeta S(\zeta)^{-1} E_\zeta^H B_\zeta^{-1}\right] & B_\zeta^{-1} E_\zeta S(\zeta)^{-1} \\ S(\zeta)^{-1} E_\zeta^T B_\zeta^{-1} & -S(\zeta)^{-1} \end{pmatrix}, \quad x^{(i)} = \begin{pmatrix} u^{(i)} \\ y^{(i)} \end{pmatrix}.$$

Again, we have

$$\rho(M^{-1}A) = 2\Re e \left\{ \sum_{\ell=1}^{N_c} \omega_\ell \begin{bmatrix} -\left[B_{\zeta_\ell}^{-1} + B_{\zeta_\ell}^{-1} E_{\zeta_\ell} S(\zeta_\ell)^{-1} E_{\zeta_\ell}^T B_{\zeta_\ell}^{-1}\right] & \boxed{B_{\zeta_\ell}^{-1} E_{\zeta_\ell} S(\zeta_\ell)^{-1}} \\ S(\zeta_\ell)^{-1} E_{\zeta_\ell}^T B_{\zeta_\ell}^{-1} & \boxed{-S(\zeta_\ell)^{-1}} \end{bmatrix} \right\} M$$

$$= \sum_{i=1}^{n} \rho(\lambda_i) \begin{bmatrix} u^{(i)} \left(u^{(i)}\right)^T & \boxed{u^{(i)} \left(y^{(i)}\right)^T} \\ y^{(i)} \left(u^{(i)}\right)^T & \boxed{y^{(i)} \left(y^{(i)}\right)^T} \end{bmatrix} M. \quad \left(\rho(\lambda_i) = 2\Re e \left\{ \sum_{\ell=1}^{N_c} \frac{\omega_\ell}{\zeta_\ell - \lambda_i} \right\}\right)$$

DD-PP and rational filtering (II)

Equating the (1,2) and (2,2) block gives:

$$2\Re e\left\{\sum_{\ell=1}^{N_c}\omega_\ell B_{\zeta_\ell}^{-1}E_{\zeta_\ell}S(\zeta_\ell)^{-1}\right\}=\sum_{i=1}^{n}\rho(\lambda_i)u^{(i)}\left(y^{(i)}\right)^T \quad \left(\mathrm{range}(\times)\supseteq\mathrm{span}\left\{u^{(i)}\right\}_{i=1,\ldots,nev}\right)$$

$$2\Re e\left\{\sum_{\ell=1}^{N_c}-\omega_\ell S(\zeta_\ell)^{-1}\right\}=\sum_{i=1}^{n}\rho(\lambda_i)y^{(i)}\left(y^{(i)}\right)^T \quad \left(\mathrm{range}(\times)\supseteq\mathrm{span}\left\{y^{(i)}\right\}_{i=1,\ldots,nev}\right)$$

The algorithm:

1: Set $R\in\mathbb{R}^{s\times r},\ r\geq nev$.
2: **for** $j=1$ to N_c **do**
3: $\quad W_s := S(\zeta_j)^{-1}R;\qquad Y := Y-\Re\{\omega_j W_s\}$ (*distributed*)
4: $\quad W_u := B_{\zeta_j}^{-1}E_{\zeta_j}W_s;\qquad U := U+\Re\{\omega_j W_u\}$ (*local*)
5: **end for**
6: Perform RR projection onto $\mathcal{Z}=\mathcal{U}\oplus\mathcal{Y}$, where $\mathcal{U}=\mathrm{span}(U),\ \mathcal{Y}=\mathrm{span}(Y)$.

Discussion and alternative schemes

Practical considerations of DD-PP

- DD-PP requires one linear system solution with matrices B_{ζ_j} and (S_{ζ_j}) per rhs/quadrature node.
- DD-PP can be adaptive by choosing a "thin" R and repeating.

Discussion and alternative schemes

Practical considerations of DD-PP

- DD-PP requires one linear system solution with matrices B_{ζ_j} and (S_{ζ_j}) per rhs/quadrature node.
- DD-PP can be adaptive by choosing a "thin" R and repeating.

Combining DD with FEAST $\rightarrow$ DD-FP

- Domain decomposition can be also useful in the setting of solving the linear systems of the form $(A - \zeta_j M)x = b$ in FEAST.
- Enhanced flexibility $\rightarrow$ Schur complement preconditioners.
- Compared to DD-PP above, DD-FP requires $d_i N_c r$ additional FLOPS and one more linear system solution with matrix B_{ζ_j} per rhs/quadrature node.

2D Laplacians

Table: Avg. time spent on a single quadrature node when approximating the eigenvalues $\lambda_{1001}, \ldots, \lambda_{1200}$ and associated eigenvectors by DD-PP, and speedup over DD-FP.

	$p = 8$		$p = 16$		$p = 32$	
	DD-PP	(x)DD-FP	DD-PP	(x)DD-FP	DD-PP	(x)DD-FP
$n = 500^2$						
$r = nev + 10$	9.45	**1.45**	6.77	**1.31**	5.25	**1.20**
$r = 3nev/2 + 10$	13.5	**1.47**	9.65	**1.32**	7.59	**1.19**
$r = 2nev + 10$	18.1	**1.44**	12.9	**1.32**	10.0	**1.23**
$n = 1000^2$						
$r = nev + 10$	41.8	**1.51**	25.3	**1.41**	17.9	**1.29**
$r = 3nev/2 + 10$	59.7	**1.59**	36.0	**1.40**	25.5	**1.30**
$r = 2nev + 10$	79.1	**1.62**	68.1	**1.47**	34.1	**1.28**
$n = 1500^2$						
$r = nev + 10$	100.8	**1.41**	65.2	**1.38**	39.9	**1.10**
$r = 3nev/2 + 10$	144.2	**1.39**	93.1	**1.40**	57.6	**1.12**
$r = 2nev + 10$	192.7	**1.49**	124.5	**1.44**	76.0	**1.11**

Experiments with a 3D Laplacian ($n = 150^3$)

	Its		$p \times \tau = 64$		$p \times \tau = 128$		$p \times \tau = 256$	
	$N_c = 1$	$N_c = 2$	FEAST	DD-FP	FEAST	DD-FP	FEAST	DD-FP
$[\alpha, \beta] \equiv [\lambda_{101}, \lambda_{120}]$								
$r = 50$	8	5	1,607.2	324.8	841.4	240.0	685.0	217.9
$r = 100$	6	4	2,073.9	473.1	1,092.1	353.2	875.2	313.8
$r = 39$	8	5	1,420.6	265.5	741.6	194.8	609.1	166.8
$[\alpha, \beta] \equiv [\lambda_{501}, \lambda_{520}]$								
$r = 50$	9	5	1,723.9	1,029.1	904.3	777.4	732.5	685.9
$r = 100$	5	4	1,840.5	1,140.3	966.9	862.3	780.0	781.2
$r = 39$	9	5	1,492.9	808.9	780.4	609.5	638.5	541.6
$[\alpha, \beta] \equiv [\lambda_{501}, \lambda_{600}]$								
$r = 200$	12	5	6,013.9	13,373.4	3,185.8	10,195.8	2,510.2	9,447.2
$r = 400$	7	3	6,950.2	15,596.3	3,664.1	11,892.2	2,876.2	10,564.3
$r = 166$	13	5	5,220.4	11,954.5	2,759.9	9,114.0	2,178.1	8,444.6

DD-FP: Schur complement linear systems were solver by preconditioned GMRES with dual thresholding, while we kept the number of MPI processes fixed to 32, each process using τ threads.

2D MPI grid for the 150^3 Laplacian (FEAST: $64 \times \kappa$, DD-FP: $32 \times \kappa$)

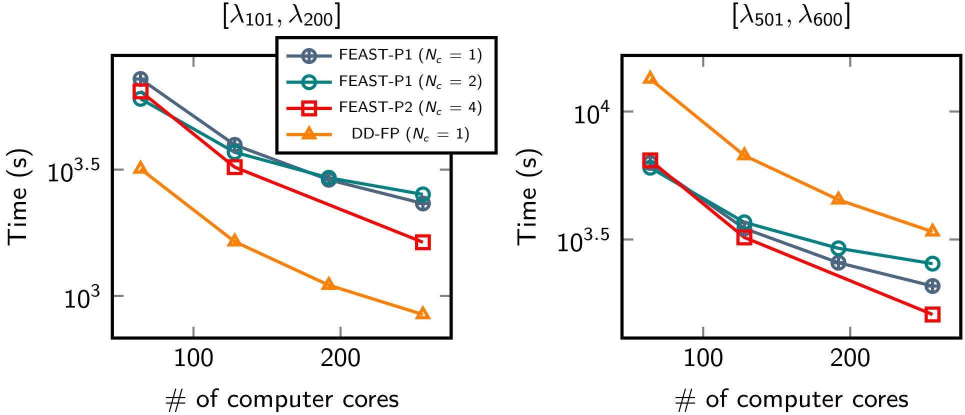


Figure: Distribute rhs (P1) or quadrature nodes (P2); $[\alpha, \beta] \equiv [\lambda_{501}, \lambda_{600}]$, $r = 200$.

Contents

Eigenvalue problems in DFT

- The all-electron Schrödinger equation is replaced by a one-electron Schrödinger equation with an effective potential → nonlinear "eigenvector" problem (Kohn-Sham)

$$\left[-\frac{\nabla^2}{2} + V_{ion}(r) + V_H(\rho(r), r) + V_{XC}(\rho(r), r)\right] \Psi_i(r) = E_i \Psi_i(r),$$

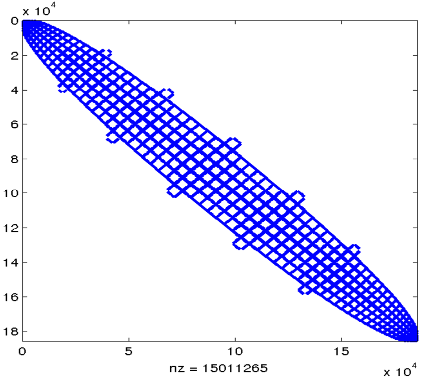
- The key quantity is the charge-density:

$$\rho(r) = 2 \sum_{i=1}^{n_{occ}} |\Psi_i(r)|^2,$$

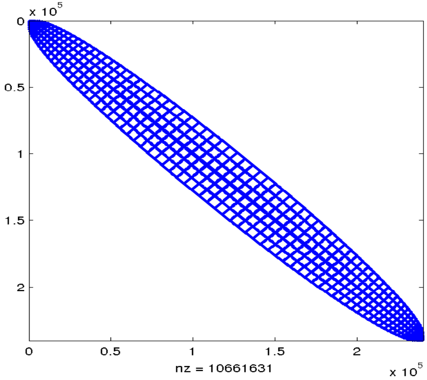
  where n_{occ} is the number of occupied states (for most systems of interest this is half the number of valence electrons).

- Self-consistent iteration (repeat until convergence): compute $\rho(r)$, update V_H, V_{XC}, compute $\Psi_1(r), \ldots, \Psi_{n_{occ}}(r)$.

Sparsity pattern of the PARSEC matrices



$Si_{41}Ge_{41}H_{72}$



$Si_{87}H_{76}$

Polynomial filtering

- **Idea:** Apply Lanczos to $p(A)$ instead of A
- $p(A)$ is a matrix polynomial
- **Goal:** Amplify/damp wanted/unwanted portion of the spectrum

Polynomial filtering

- **Idea:** Apply Lanczos to $p(A)$ instead of A
- $p(A)$ is a matrix polynomial
- **Goal:** Amplify/damp wanted/unwanted portion of the spectrum

Chebyshev series approximation ($\zeta \in [\alpha, \beta] \subseteq [-1, 1]$)

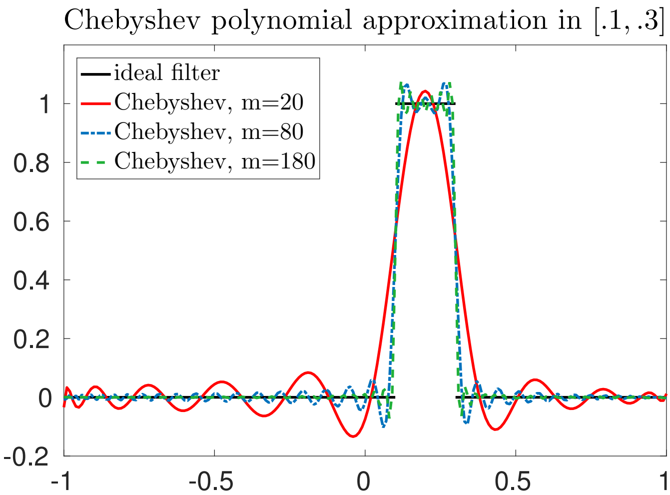
$$\phi(\zeta) = \sum_{i=0}^{\infty} b_i T_i(\zeta),$$

where (for given α and β),

$$b_i = \begin{cases} \left(\arccos(\alpha) - \arccos(\beta)\right)/\pi, & i = 0, \\ 2\left(\sin\left(i\arccos(\alpha)\right) - \sin\left(i\arccos(\beta)\right)\right)/i\pi, & i > 0. \end{cases}$$

In practice we fix m and truncate: $p(\zeta) = \sum_{i=0}^{m} b_i T_i(\zeta)$.

Chebyshev polynomial approximation (cont.)



Block Lanczos

The filtered Lanzos procedure iteratively constructs an orthonormal basis for the Krylov subspace generated by $p(A)$ and Q:

$$\mathcal{K}_k(p(A), Q) = \mathrm{span}\{Q, p(A)Q, \ldots, p(A)^{k-1}Q\}.$$

Now let $Q_k \in \mathbb{R}^{n \times rk}$: the matrix whose columns are generated by $k-1$ steps of the block Lanczos.

Block Lanczos

The filtered Lanzos procedure iteratively constructs an orthonormal basis for the Krylov subspace generated by $p(A)$ and Q:

$$\mathcal{K}_k(p(A), Q) = \mathrm{span}\{Q, p(A)Q, \ldots, p(A)^{k-1}Q\}.$$

Now let $Q_k \in \mathbb{R}^{n \times rk}$: the matrix whose columns are generated by $k-1$ steps of the block Lanczos. Then, block Lanczos generates:

$$p(A)Q_k = Q_{k+1}\tilde{T}_k,$$

where

$$\tilde{T}_k = \begin{bmatrix} T_k \\ S_k E_k^T \end{bmatrix}, \quad T_k = \begin{bmatrix} D_1 & S_1^T & & & \\ S_1 & D_2 & S_2^T & & \\ & S_2 & D_3 & \ddots & \\ & & \ddots & \ddots & S_{k-1}^T \\ & & & S_{k-1} & D_k \end{bmatrix}.$$

PARSEC test matrices

Matrix	n	nnz/n	$[\lambda_1, \lambda_n]$	$[\alpha, \beta]$	nev
Ge87H76	$112,985$	69.9	$[1.21402, 32.7641]$	$[-0.645, -0.0053]$	212
Ge99H100	$112,985$	74.8	$[1.22642, 32.7031]$	$[-0.650, -0.0096]$	250
Si41Ge41H72	$185,639$	80.9	$[1.21358, 49.8185]$	$[-0.640, -0.0028]$	218
Si87H76	$240,369$	44.4	$[1.19638, 43.0746]$	$[-0.660, -0.3300]$	107
Ga41As41H72	$268,096$	69.0	$[1.25019, 1300.93]$	$[-0.640, 0.0000]$	201

- The Hamiltonians are real, symmetric, with (severely) clustered eigenvalues.
- We profile and compare the GPU implementation against FILTLAN, a multi-threaded CPU implementation of a polynomial filtering Lanczos procedure.

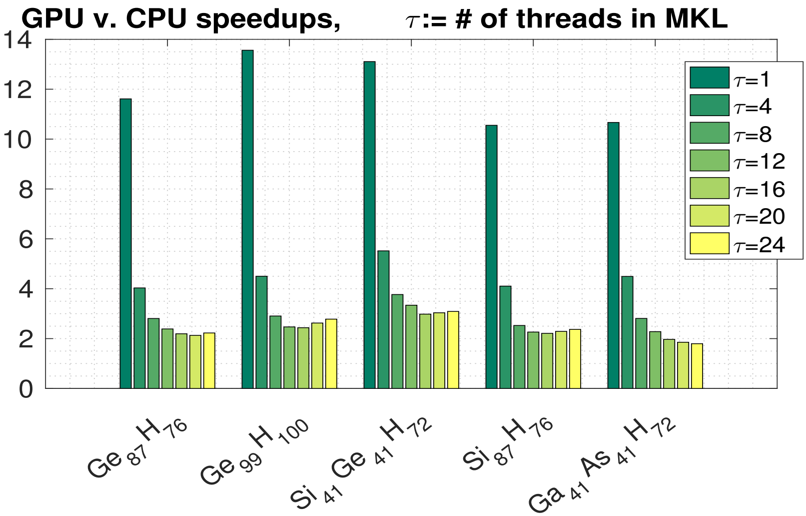
Performance on a K40 GPU

Matrix	interval	nev	m	iters	MV	time
			50	210	$31,500$	31
			100	180	$54,000$	40
Ge87H76	$[-0.645, -0.0053]$	212	150	150	$67,500$	44
			50	210	$31,500$	32
			100	180	$54,000$	41
Ge99H100	$[-0.650, -0.0096]$	250	150	180	$81,000$	56
			50	210	$31,500$	56
			100	180	$54,000$	73
Si41Ge41H72	$[-0.640, -0.0028]$	218	150	180	$81,000$	99
			50	150	$22,500$	38
			100	90	$27,000$	35
Si87H76	$[-0.660, -0.3300]$	107	150	120	$54,000$	63
			200	180	$144,000$	225
			300	180	$162,000$	236
Ga41As41H72	$[-0.640, 0.0000]$	201	400	180	$216,000$	306

Performance on a K40 GPU (cont.)

Matrix	m	iters	PREPROC	ORTH	MV
	50	210	7%	22%	52%
	100	180	5%	13%	71%
`Ge87H76`	150	150	5%	9%	80%
	50	210	7%	21%	53%
	100	180	5%	13%	71%
`Ge99H100`	150	180	4%	10%	79%
	50	210	10%	19%	55%
	100	180	8%	12%	72%
`Si41Ge41H72`	150	180	6%	9%	80%
	50	150	11%	22%	54%
	100	90	12%	12%	70%
`Si87H76`	150	120	7%	10%	78%
	200	240	4%	8%	82%
	300	180	4%	5%	88%
`Ga41As41H72`	400	180	3%	4%	91%

A comparison between GPU and CPU



Summary and future work

Summary:

- Domain decomposition techniques reduce orthogonalization costs.
- The main bottleneck lies on the solution of the interface eigenvalue problem.
- This dissertation suggested two main approaches:
  - Rational filtering.
  - Newton-based iterations (not discussed in this talk).
- Numerical experiments on distributed memory architectures were performed.
- A GPU implementation of a polynomial filtering approach was also discussed.

Future work:

- Rational filtering DD for non-symmetric systems.
- Multilevel preconditioners for complex linear systems.
- Rayleigh-Ritz subspaces augmented by eigenvector derivatives.

List of publications (see http://www-users.cs.umn.edu/kalan019)

- V. Kalantzis, Y. Xi, and Y. Saad, "*Beyond Automated MultiLevel Substructuring: Domain Decomposition with Rational Filtering*", **SIAM Journal on Scientific Computing**, 2018.
- V. Kalantzis, J. Kestyn, E. Polizzi, and Y. Saad, "*Domain Decomposition Approaches for Accelerating Contour Integration Eigenvalue Solvers for Symmetric Eigenvalue Problems*", **Numerical Linear Algebra with Applications**, 2018.
- J. L. Aurentz, V. Kalantzis, and Y. Saad, "*Cucheb : A GPU Implementation of the Filtered Lanczos Procedure*", **Computer Physics Communications**, 2017.
- J. Kestyn, V. Kalantzis, E. Polizzi, and Y. Saad, "*PFEAST : A High Performance Sparse Eigenvalue Solver Using Distributed – Memory Linear Solvers*", **ACM/IEEE Supercomputing Conference**, 2016.
- V. Kalantzis, R. Li, and Y. Saad, "*Spectral Schur Complement Techniques for Symmetric Eigenvalue Problems*", **Electronic Transactions in Numerical Analysis**, 2016.
- V. Kalantzis and Y. Saad, "*Spectral Schur Complement Techniques for Symmetric Generalized Eigenvalue Problems*", **Preprint**.