Domain Decomposition-based contour integration eigenvalue solvers

Vassilis Kalantzis
joint work with
Yousef Saad

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

SIAM ALA 2015, 10-30-2015

Acknowledgments

- Joint work with Eric Polizzi and James Kestyn (UM Amherst).
- Special thanks to Minnesota Supercomputing Institute for allowing access to its supercomputers.
- Work supported by NSF and DOE (DE-SC0008877).

University of Minnesota

Supercomputing Institute

Contents

Introduction

The sparse symmetric eigenvalue problem

$$Ax = \lambda x$$

where $A \in \mathbb{R}^{n\times n}$, $x \in \mathbb{R}^n$ and $\lambda \in \mathbb{R}$. A pair (λ, x) is called an *eigenpair* of A.

Focus in this talk

Find all (λ, x) pairs inside the interval $[\alpha, \beta]$ where $\alpha,\ \beta \in \mathbb{R}$ and $\lambda_1 \leq \alpha,\ \beta \leq \lambda_n$.

Typical approach

Project A on a low-dimensional subspace by

$$V^\top AVy = \theta V^\top Vy, \quad \tilde{x} = Vy.$$

- V: Krylov, (Generalized-Jacobi)-Davidson, contour integration,...

Contour integration (CINT)

$$V := \mathcal{P}\hat{V} = \frac{1}{2i\pi}\int_{\Gamma}(\zeta I - A)^{-1}d\zeta\ \hat{V} \equiv XX^{\top}\hat{V},$$

with $\Gamma \to$ complex contour with endpoints $[\alpha, \beta]$.

- V is an exact invariant subspace

Numerical approximation

$$\mathcal{P}\hat{V} \approx \tilde{\mathcal{P}}\hat{V} = \sum_{j=1}^{n_c}\omega_j(\zeta_j I - A)^{-1}\hat{V}, \quad \rho(z) = \sum_{j=1}^{n_c}\frac{\omega_j}{\zeta_j - z}$$

with (weight,pole)$\equiv (\omega_j, \zeta_j),\ j = 1, \ldots, n_c$.

- Trapezoidal, Midpoint, Gauss-Legendre,...
- Zolotarev, Least-Squares,...
- FEAST (Polizzi), Sakurai-Sugiura (SS), SS-CIRR,.....

Main characteristics of CINT

- Can be seen as a (rational) filtering technique
- Different levels of parallelism
- Eigenvalue problem $\rightarrow$ Linear systems with multiple right-hand sides

In this talk

- We study contour integration from a Domain Decomposition (DD) point-of-view
- Two ideas:
  - Use DD to derive CINT schemes
  - Use DD to accelerate FEAST or other CINT-based method
- We target parallel architectures

Contents

Partitioning of the domain (Metis, Scotch,...)

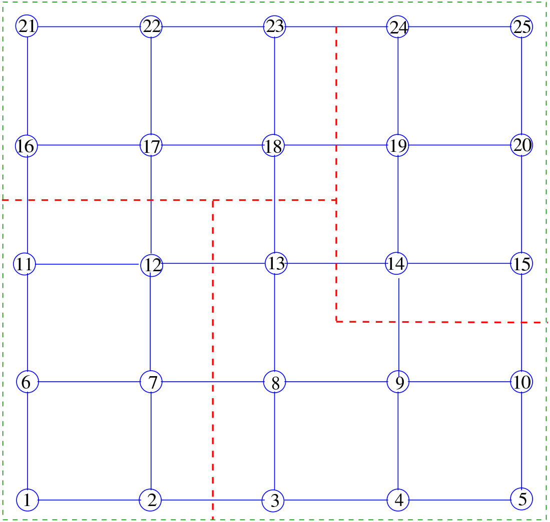
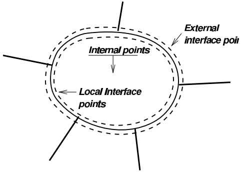
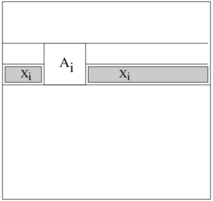


Figure: An edge-separator (vertex-based partitioning)

The local viewpoint – assume M partitions

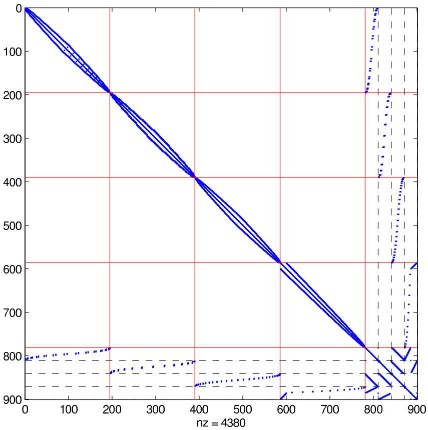




Stack interior variables $u_1, u_2, \ldots, u_P$ into u, then interface variables y,

$$\begin{pmatrix} B_1 & & & & E_1 \\ & B_2 & & & E_2 \\ & & \ddots & & \vdots \\ & & & B_M & E_M \\ E_1^\top & E_2^\top & \cdots & E_M^\top & C \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \\ \vdots \\ u_M \\ y \end{pmatrix} = \lambda \begin{pmatrix} u_1 \\ u_2 \\ \vdots \\ u_M \\ y \end{pmatrix}$$

Pictorially:



Write as

$$A = \begin{pmatrix} B & E \\ E^\top & C \end{pmatrix}.$$

Contents

Expressing $(A - \zeta I)^{-1}$ in DD

Let $\zeta \in \mathbb{C}$ and recall that

$$A = \begin{pmatrix} B & E \\ E^\top & C \end{pmatrix}.$$

Expressing $(A - \zeta I)^{-1}$ in DD

Let $\zeta \in \mathbb{C}$ and recall that

$$A = \begin{pmatrix} B & E \\ E^\top & C \end{pmatrix}.$$

Then

$$(A - \zeta I)^{-1} = \begin{pmatrix} (B - \zeta I)^{-1} + F(\zeta)S(\zeta)^{-1}F(\zeta)^\top & -F(\zeta)S(\zeta)^{-1} \\ -S(\zeta)^{-1}F(\zeta)^\top & S(\zeta)^{-1} \end{pmatrix},$$

where

$$F(\zeta) = (B - \zeta I)^{-1}E$$
$$S(\zeta) = C - \zeta I - E^T(B - \zeta I)^{-1}E.$$

Spectral projectors and DD

As previously,

$$(A-\zeta I)^{-1} = \begin{pmatrix} (B-\zeta I)^{-1} + F(\zeta)S(\zeta)^{-1}F(\zeta)^{\top} & -F(\zeta)S(\zeta)^{-1} \\ -S(\zeta)^{-1}F(\zeta)^{\top} & S(\zeta)^{-1} \end{pmatrix},$$

Then,

$$\mathcal{P}_{DD} = \frac{-1}{2i\pi}\int_{\Gamma}(A-\zeta I)^{-1}d\zeta \equiv \begin{pmatrix} \mathcal{H} & -\mathcal{W} \\ -\mathcal{W}^{\top} & \mathcal{G} \end{pmatrix}$$

Spectral projectors and DD

As previously,

$$(A-\zeta I)^{-1} = \begin{pmatrix} (B-\zeta I)^{-1} + F(\zeta)S(\zeta)^{-1}F(\zeta)^\top & -F(\zeta)S(\zeta)^{-1} \\ -S(\zeta)^{-1}F(\zeta)^\top & S(\zeta)^{-1} \end{pmatrix},$$

Then,

$$\mathcal{P}_{DD} = \frac{-1}{2i\pi}\int_\Gamma (A-\zeta I)^{-1} d\zeta \equiv \begin{pmatrix} \mathcal{H} & -\mathcal{W} \\ -\mathcal{W}^\top & \mathcal{G} \end{pmatrix}$$

$$\begin{cases} \mathcal{H} = \frac{-1}{2i\pi}\int_\Gamma [(B-\zeta I)^{-1} + F(\zeta)S(\zeta)^{-1}F(\zeta)^\top] d\zeta \\ \mathcal{G} = \frac{-1}{2i\pi}\int_\Gamma S(\zeta)^{-1} d\zeta \\ \mathcal{W} = \frac{-1}{2i\pi}\int_\Gamma F(\zeta)S(\zeta)^{-1} d\zeta. \end{cases}$$

Extracting approximate eigenspaces

Let $\hat{V}$ be a set of mrhs to multiply $\mathcal{P}$

$$\mathcal{P}_{DD}\begin{pmatrix}\hat{V}_u\\ \hat{V}_s\end{pmatrix} = \begin{pmatrix}\mathcal{H}\hat{V}_u - \mathcal{W}\hat{V}_s\\ -\mathcal{W}^\top\hat{V}_u + \mathcal{G}\hat{V}_s\end{pmatrix} \equiv \begin{pmatrix}Z_u\\ Z_s\end{pmatrix}, \textit{with}$$

$$\begin{cases} Z_u = \dfrac{-1}{2i\pi}\int_\Gamma (B-\zeta I)^{-1}\hat{V}_u d\zeta - \dfrac{-1}{2i\pi}\int_\Gamma F(\zeta)S(\zeta)^{-1}[\hat{V}_s - F(\zeta)^\top\hat{V}_u]d\zeta \\ Z_s = \dfrac{-1}{2i\pi}\int_\Gamma S(\zeta)^{-1}[\hat{V}_s - F(\zeta)^\top\hat{V}_u]d\zeta. \end{cases}$$

Extracting approximate eigenspaces

Let $\hat{V}$ be a set of mrhs to multiply $\mathcal{P}$

$$\mathcal{P}_{DD}\begin{pmatrix}\hat{V}_u\\ \hat{V}_s\end{pmatrix}=\begin{pmatrix}\mathcal{H}\hat{V}_u-\mathcal{W}\hat{V}_s\\ -\mathcal{W}^\top\hat{V}_u+\mathcal{G}\hat{V}_s\end{pmatrix}\equiv\begin{pmatrix}Z_u\\ Z_s\end{pmatrix}, with$$

$$\begin{cases} Z_u=\frac{-1}{2i\pi}\int_\Gamma(B-\zeta I)^{-1}\hat{V}_u d\zeta-\frac{-1}{2i\pi}\int_\Gamma F(\zeta)S(\zeta)^{-1}[\hat{V}_s-F(\zeta)^\top\hat{V}_u]d\zeta\\ Z_s=\frac{-1}{2i\pi}\int_\Gamma S(\zeta)^{-1}[\hat{V}_s-F(\zeta)^\top\hat{V}_u]d\zeta.\end{cases}$$

In practice:

$$\tilde{Z}_u=\sum_{j=1}^{n_c}\omega_j(B-\zeta_j I)^{-1}\hat{V}_u-\sum_{j=1}^{n_c}\omega_j F(\zeta_j)S(\zeta)^{-1}[\hat{V}_s-F(\zeta)^\top\hat{V}_u],$$

$$\tilde{Z}_s=\sum_{j=1}^{n_c}\omega_j S(\zeta_j)^{-1}[\hat{V}_s-F(\zeta_j)^\top\hat{V}_u].$$

Pseudocode - Full projector (DD-FP)

---

for $j = 1$ to n_c **do**
  $W_u := (B - \zeta_j I)^{-1} \hat{V}_u$ (*local*)
  $W_s := \hat{V}_s - E^\top W_u$ (*local*)
  $W_s := S(\zeta_j)^{-1} W_s$; $\tilde{Z}_s := \tilde{Z}_s + \omega_j W_s$ (*distributed*)
  $W_u := W_u - (B - \zeta_j)^{-1} E W_s$; $\tilde{Z}_u := \tilde{Z}_u + \omega_j W_u$ (*local*)
end for

Practical considerations

- For each ζ_j, $j = 1, \ldots, n_c$:
  - Two solves with $B - \zeta_j I$ + One solve with $S(\zeta_j)$
- The procedure can be repeated with an updated $\hat{V}$
- "Equivalent" to FEAST tied with a DD solver

An alternative scheme

CINT along the interface unknowns

$$\mathcal{P}_{DD} = \frac{-1}{2i\pi}\int_\Gamma (A-\zeta I)^{-1} d\zeta = [\mathcal{P}_1, \mathcal{P}_2] \equiv \begin{pmatrix} * & -\mathcal{W} \\ * & \mathcal{G} \end{pmatrix},$$

$$\mathcal{G} = \frac{-1}{2i\pi}\int_\Gamma S(\zeta)^{-1} d\zeta, \qquad -\mathcal{W} = \frac{1}{2i\pi}\int_\Gamma (B-\zeta I)^{-1} E S(\zeta)^{-1} d\zeta.$$

Advantage: Does not involve the inverse of whole matrix.

An alternative scheme

CINT along the interface unknowns

$$\mathcal{P}_{DD} = \frac{-1}{2i\pi}\int_\Gamma (A-\zeta I)^{-1} d\zeta = [\mathcal{P}_1, \mathcal{P}_2] \equiv \begin{pmatrix} * & -\mathcal{W} \\ * & \mathcal{G} \end{pmatrix},$$

$$\mathcal{G} = \frac{-1}{2i\pi}\int_\Gamma S(\zeta)^{-1} d\zeta, \qquad -\mathcal{W} = \frac{1}{2i\pi}\int_\Gamma (B-\zeta I)^{-1} E S(\zeta)^{-1} d\zeta.$$

Advantage: Does not involve the inverse of whole matrix.

$$\mathcal{P}_{DD} = XX^\top,\ X = \begin{pmatrix} U \\ Y \end{pmatrix} \quad \to \quad \mathcal{P}_{DD} = \begin{pmatrix} * & UY^\top \\ * & YY^\top \end{pmatrix}$$

- Just capture the range of $\mathcal{P}_2 = XY^\top \to \mathcal{P}_2 \times \texttt{randn()}$
- Also: Lanczos on $\mathcal{P}_2\mathcal{P}_2^\top$ (sequential, doubles the work)

Pseudocode - Partial projector (DD-PP)

for $j = 1$ to n_c **do**

$\quad \cancel{W_u := (B - \zeta_j I)^{-1} \hat{V}_u}$ $\qquad (local)$

$\quad \cancel{W_s := \hat{V}_s - E^\top W_u}$ $\qquad (local)$

$\quad W_s := S(\zeta_j)^{-1} \hat{V}_s; \qquad \tilde{Z}_s := \tilde{Z}_s + \omega_j W_s \qquad (distributed)$

$\quad W_u := \cancel{W_u} - (B - \zeta_j)^{-1} E W_s; \qquad \tilde{Z}_u := \tilde{Z}_u + \omega_j W_u \qquad (local)$

end for

Practical considerations

- For each $\zeta_j,\ j = 1, \ldots, n_c$:
  - One solve with $B - \zeta_j I$ + One solve with $S(\zeta_j)$
- More like a one-shot method

A more detailed analysis of DD-PP

Spectral Schur complement

- $\lambda \Leftrightarrow \det[S(\lambda)] = 0 \quad (\lambda \notin \sigma(B))$
- The eigenvector satisfies

$$x = \begin{pmatrix} -(B-\lambda I)^{-1}Ey \\ y \end{pmatrix}, \quad \textit{with} \quad y := S(\lambda)y = 0.$$

If $(B - \zeta I)^{-1}$ analytic in $[\alpha, \beta]$

$$-\mathcal{W} = \frac{1}{2i\pi}\int_\Gamma (B-\zeta I)^{-1}ES(\zeta)^{-1}d\zeta \quad \rightarrow \{(B-\lambda I)^{-1}Ey\}_\Gamma$$

$$\mathcal{G} = \frac{-1}{2i\pi}\int_\Gamma S(\zeta)^{-1}d\zeta \quad \rightarrow \{-y\}_\Gamma$$

Another idea: Solve $S(\lambda)y = 0$ directly ([VK,RLi,YS],[VanB,Kra],[Sak])

Contents

A closer look at the Schur complement

So far:

Eigenvalue problem $\rightarrow$ Linear systems with mrhs $\rightarrow$ Schur complement

From the DD framework we have

$$S(\zeta) = \begin{pmatrix} S_1(\zeta) & E_{12} & \dots & E_{1M} \\ E_{21} & S_2(\zeta) & \dots & E_{2M} \\ \vdots & & \ddots & \vdots \\ E_{M1} & E_{M2} & \dots & S_M(\zeta) \end{pmatrix},$$

where

$$S_i(\zeta) = C_i - \zeta I - E_i^T (B_i - \zeta I)^{-1} E_i, \; i = 1, \dots, M,$$

is the "local" Schur complement (complex symmetric).

Solving linear systems with the Schur complement

Straightforward approach

- Form and factorize $S(\zeta)$
- Extremely robust but impractical for 3D problems

Alternative $\rightarrow$ Use an approximation of $S(\zeta)$

- Lots of ideas (pARMS, LORASC,...)
- Typical preconditioners implemented:
  - Block Jacobi: Use $C_i,\ C_i - \zeta I$ or $S_i(\zeta),\ i = 1, \ldots, M$
  - Global approximation: Use $C,\ C - \zeta I$ or $\approx S(\zeta)$
- Memory Vs robustness
- Important: magnitude of the imaginary part of a pole

Contents

Implementation and computing environment

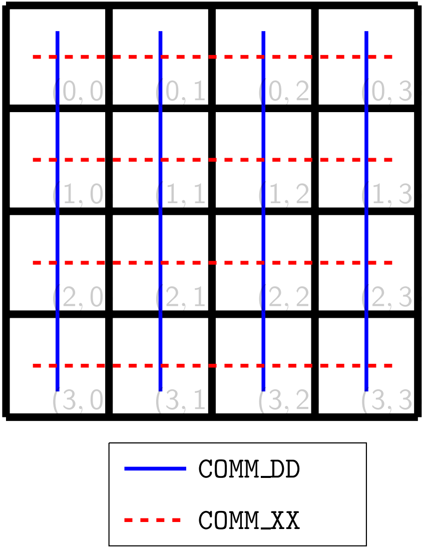
Hardware

- `ITASCA` HP Linux cluster at Minnesota Supercomputing Inst.
- 1,091 HP ProLiant BL280c G6 blade servers, each with two-socket, quad-core 2.8 GHz Intel Xeon X5560 “Nehalem EP” (24 GB per node)
- 40-gigabit QDR InfiniBand (IB) interconnect

Software

- The software was written in C++ and on top of PETSc (MPI)
- Linked to AMD, METIS, UMFPACK, MUMPS, MKL-BLAS, MKL-LAPACK
- Compiled with mpiicpc (-O3)

Parallel implementation



- Different levels of parallelism (1-D, 2-D, 3-D grids)
- `COMM_DD` is the communicator used for Domain Decomposition
- Different choices for `XX`
  - `XX` = MRHS: Parallelize the multiple right-hand sides
  - `XX` = POLES: Parallelize the number of poles
- Selection depends on the linear solver (direct or iterative)

Experimental framework

CINT + Subspace Iteration

- CINT-SI: standard "FEAST" approach
- Direct (MUMPS) or iterative (preconditioned) solver

CINT + DD

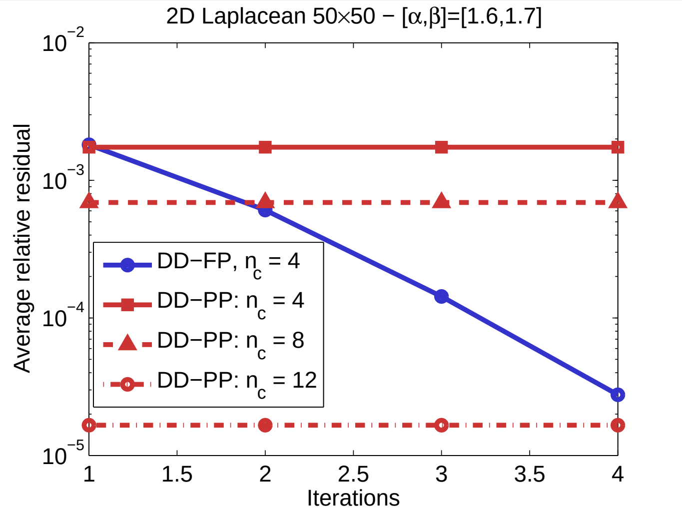
- DD-FP: implements the full projector
- DD-PP: implements the partial projector
- Schur complement: exact or approximate

Details

- # MPI processes $\rightarrow$ # cores
- Quadrature rule: Gauss-Legendre
- Eig/vle tolerance: $1e-8$

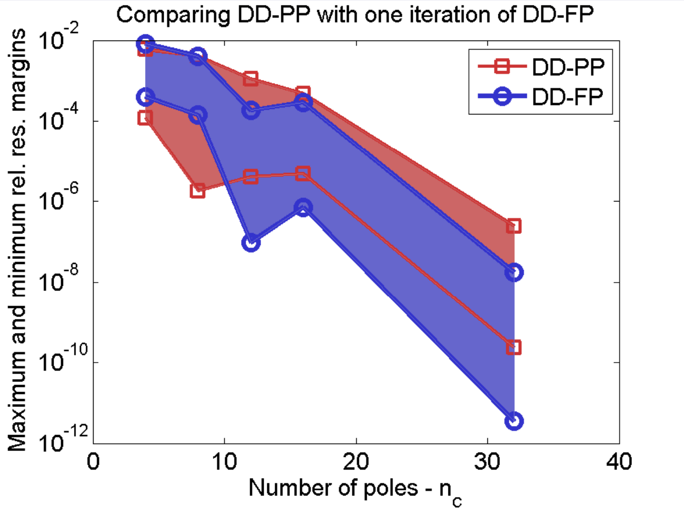
Numerical illustration

A comparison of DD-FP and DD-PP I



Numerical illustration (cont. from previous)

A comparison of DD-FP and DD-PP II



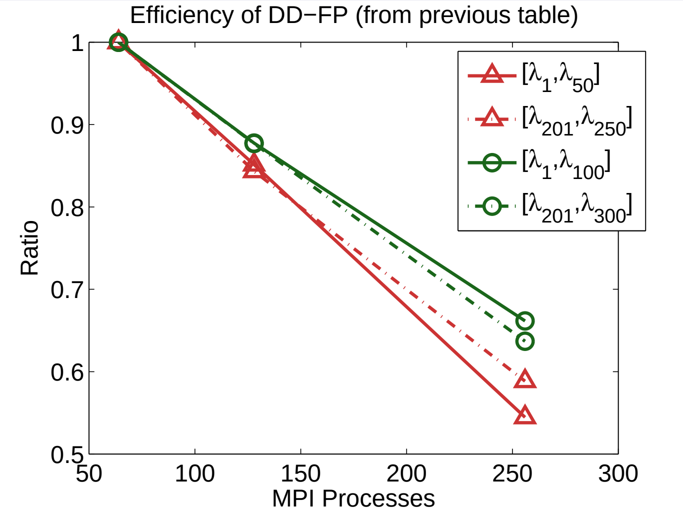
Test on a 2D 1001×1000 Laplacian

Table: Time is listed in seconds. A 2-D grid of processors was used. $S(\zeta)$ factorized explicitly. Number of poles: $n_c = 4$

		MPI 16×4		MPI 32×4		MPI 64×4	
$[\alpha, \beta]$	Its	CINT-SI	DD-FP	CINT-SI	DD-FP	CINT-SI	DD-FP
Exterior eigvls							
$[\lambda_1, \lambda_{20}]$	3	88	37	53	26	42	20
$[\lambda_1, \lambda_{50}]$	3	159	65	88	38	65	30
$[\lambda_1, \lambda_{100}]$	5	432	172	241	98	136	65
Interior eigvls							
$[\lambda_{201}, \lambda_{220}]$	3	89	37	53	26	42	20
$[\lambda_{201}, \lambda_{250}]$	4	286	113	164	67	110	48
$[\lambda_{201}, \lambda_{300}]$	4	440	214	245	122	141	84

- Exterior: Number of right-hand sides $\equiv$ number of eigvls $+ 20$
- Interior: Number of right-hand sides $\equiv 2 \times$ number of eigvls

Efficiency of DD-FP



Contents

Conclusion

In this talk

- We presented a DD-based form of contour integration
- DD can be used to:
  - Solve the linear systems in numerical integration
  - Derive DD-based contour integration methods
- The Schur complement holds a central role

Considerations

- Higher moments of the Schur complement integral
- Block Krylov subspaces
- Preconditioning of indefinite linear systems