MAX PLANCK INSTITUTE
FOR DYNAMICS OF COMPLEX
TECHNICAL SYSTEMS
MAGDEBURG

CSC
COMPUTATIONAL METHODS IN
SYSTEMS AND CONTROL THEORY

Opportunities for ELPA to Accelerate the Solution of the Bethe-Salpeter Eigenvalue Problem

Peter Benner, Andreas Marek, Carolin Penke

August 16, 2018

ELSI Workshop 2018

Partners:

MPCDF Max Planck Computing & Data Facility
Rechenzentrum Garching der Max-Planck-Gesellschaft

The Problem

The Bethe-Salpeter Eigenvalue Problem

Find eigenvalues, right and left eigenvectors for

$$H_{BS}x = \lambda x \text{ with}$$

$$H_{BS} = \begin{bmatrix} A & B \\ -\bar{B} & -\bar{A} \end{bmatrix} = \begin{bmatrix} A & B \\ -B^H & -A^T \end{bmatrix},$$

$$A = A^H, \quad B = B^T \in \mathbb{C}^{n \times n} \text{ are } \textbf{dense}.$$

The Problem

The Bethe-Salpeter Eigenvalue Problem

Find eigenvalues, right and left eigenvectors for

$$H_{BS}x = \lambda x \text{ with}$$

$$H_{BS} = \begin{bmatrix} A & B \\ -\bar{B} & -\bar{A} \end{bmatrix} = \begin{bmatrix} A & B \\ -B^H & -A^T \end{bmatrix},$$

$$A = A^H, \quad B = B^T \in \mathbb{C}^{n \times n} \text{ are } \textbf{dense}.$$

- Comes up in quantum chemical simulations.
- n is proportional to n_e^2 where n_e is the number of electrons in the system.
  - $\rightarrow$ n can become very large ($> 50,000$)

The Problem

The Bethe-Salpeter Eigenvalue Problem

Find eigenvalues, right and left eigenvectors for

$$H_{BS}x = \lambda x \text{ with}$$

$$H_{BS} = \begin{bmatrix} A & B \\ -\bar{B} & -\bar{A} \end{bmatrix} = \begin{bmatrix} A & B \\ -B^H & -A^T \end{bmatrix},$$

$$A = A^H, \quad B = B^T \in \mathbb{C}^{n \times n} \text{ are } \textbf{dense}.$$

- Comes up in quantum chemical simulations.
- n is proportional to n_e^2 where n_e is the number of electrons in the system.
  - $\rightarrow$ n can become very large ($> 50,000$)

$\rightarrow$ Parallel and scalable algorithms running on supercomputers necessary.

Applications in Quantum Chemistry

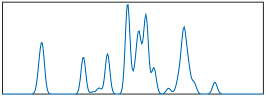
Eigenvalues λ_i, right eigenvectors x_i and left eigenvectors y_i are used to compute

- Spectral Density

$$\phi(\omega) = \frac{1}{2n} \sum_{j=1}^{2n} \delta(\omega - \lambda_j),$$

- Optical Absorption Spectrum

$$\epsilon^+(\omega) = \sum_{j=1}^{n} \frac{(d_r^H x_j)(y_j^H d_l)}{y_j^H x_j} \delta(\omega - \lambda_j).$$



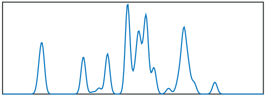
Eigenvalues λ_i, right eigenvectors x_i and left eigenvectors y_i are used to compute

- Spectral Density

$$\phi(\omega) = \frac{1}{2n}\sum_{j=1}^{2n}\delta(\omega - \lambda_j),$$

- Optical Absorption Spectrum

$$\epsilon^+(\omega) = \sum_{j=1}^{n}\frac{(d_r^H x_j)(y_j^H d_l)}{y_j^H x_j}\delta(\omega - \lambda_j).$$

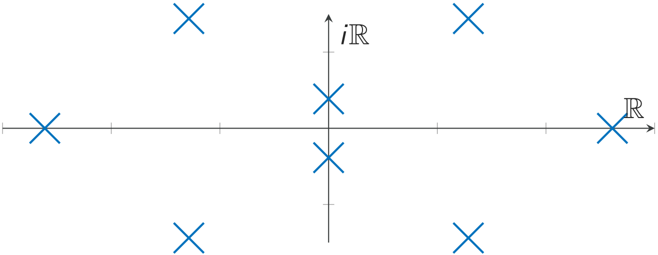


→ We are interested in all (or most) eigenpairs.

Properties of (definite) Bethe-Salpeter EVP

- Due to the special structure eigenvalues appear in quadruples $(\lambda, -\lambda, \bar{\lambda}, -\bar{\lambda})$.

[Benner, Fassbender, Yang '14/'18]



Properties of (definite) Bethe-Salpeter EVP

- Due to the special structure eigenvalues appear in quadruples $(\lambda, -\lambda, \bar{\lambda}, -\bar{\lambda})$.

  [Benner, Fassbender, Yang '14/'18]

- H_{BS} is called definite if $\begin{bmatrix} I_n & 0 \\ 0 & -I_n \end{bmatrix} H_{BS} = \begin{bmatrix} A & B \\ \bar{B} & \bar{A} \end{bmatrix} > 0$, holds in many physical settings. Here eigenvalues come in real pairs and it holds

Theorem [Shao, da Jornada, Yang, Deslippe, Louie '15]

There exist $X_1, X_2 \in \mathbb{C}^{n \times n}$, and $\lambda_1, \ldots, \lambda_n \in \mathbb{R}_+$, $\Lambda_+ = diag\{\lambda_1, \ldots, \lambda_n\}$, s.t.

$$H_{BS} X = X \begin{bmatrix} \Lambda_+ & \\ & -\Lambda_+ \end{bmatrix}, \quad Y^H H_{BS} = \begin{bmatrix} \Lambda_+ & \\ & -\Lambda_+ \end{bmatrix} Y^H, \quad Y^H X = I_{2n}$$

$$\text{where} \quad X = \begin{bmatrix} X_1 & \bar{X}_2 \\ X_2 & \bar{X}_1 \end{bmatrix}, \quad Y = \begin{bmatrix} X_1 & -\bar{X}_2 \\ -X_2 & \bar{X}_1 \end{bmatrix}.$$

A Structure-Preserving Method

[Shao, da Jornada, Yang, Deslippe, Louie '15]

A direct method is implemented in BSEPACK[1]. The structure-preserving acquisition of all eigenpairs in high precision relies on a connection to Hamiltonian matrices.

Theorem

Let $Q = \frac{1}{\sqrt{2}} \begin{bmatrix} I_n & -iI_n \\ I_n & iI_n \end{bmatrix}$, then

$$Q^H \begin{bmatrix} A & B \\ -\bar{B} & -\bar{A} \end{bmatrix} Q = i \begin{bmatrix} Im(A+B) & -Re(A-B) \\ Re(A+B) & Im(A-B) \end{bmatrix} =: iH,$$

where H is real Hamiltonian, i.e. $JH = (JH)^T$ with $J = \begin{bmatrix} 0 & I \\ -I & 0 \end{bmatrix}$.

[1] https://sites.google.com/a/lbl.gov/bsepack/

The resulting algorithm

Algorithm 1 Algorithm for the complex Bethe-Salpeter eigenvalue problem

Require: $A = A^H, B = B^T \in \mathbb{C}^{n \times n}$, s.t. $\begin{bmatrix} I_n & 0 \\ 0 & -I_n \end{bmatrix} H_{BS} = \begin{bmatrix} A & B \\ \bar{B} & \bar{A} \end{bmatrix} > 0$

Ensure: $X_1, X_2 \in \mathbb{C}^{n \times n}$ and $\Lambda_+ = diag\{\lambda_1, \ldots, \lambda_n\}$ satisfying $H \begin{bmatrix} X_1 \\ X_2 \end{bmatrix} = \begin{bmatrix} X_1 \\ X_2 \end{bmatrix} \Lambda_+$.

1: Construct $M = \begin{bmatrix} Re(A+B) & Im(A-B) \\ -Im(A+B) & Re(A-B) \end{bmatrix}$

2: Compute the Cholesky factorization $M = LL^T$

3: Construct $W = L^T \begin{bmatrix} 0 & I_n \\ -I_n & 0 \end{bmatrix} L$

4: Compute the spectral decomposition $-iW = \begin{bmatrix} Z_+ & Z_- \end{bmatrix} diag\{\Lambda_+, -\Lambda_+\} \begin{bmatrix} Z_+ & Z_- \end{bmatrix}^H$.

5: Set $\begin{bmatrix} X_1 \\ X_2 \end{bmatrix} = \begin{bmatrix} I_n & 0 \\ 0 & -I_n \end{bmatrix} QLZ_+\Lambda_+^{-1/2}$

The resulting algorithm

Algorithm 1 Algorithm for the complex Bethe-Salpeter eigenvalue problem

Require: $A = A^H, B = B^T \in \mathbb{C}^{n\times n}$, s.t. $\begin{bmatrix} I_n & 0 \\ 0 & -I_n \end{bmatrix} H_{BS} = \begin{bmatrix} A & B \\ \bar{B} & \bar{A} \end{bmatrix} > 0$

Ensure: $X_1, X_2 \in \mathbb{C}^{n\times n}$ and $\Lambda_+ = diag\{\lambda_1, \ldots, \lambda_n\}$ satisfying $H \begin{bmatrix} X_1 \\ X_2 \end{bmatrix} = \begin{bmatrix} X_1 \\ X_2 \end{bmatrix} \Lambda_+$.

1: Construct $M = \begin{bmatrix} Re(A+B) & Im(A-B) \\ -Im(A+B) & Re(A-B) \end{bmatrix}$

2: Compute the Cholesky factorization $M = LL^T$

3: Construct $W = L^T \begin{bmatrix} 0 & I_n \\ -I_n & 0 \end{bmatrix} L$

4: Compute the spectral decomposition $-iW = \begin{bmatrix} Z_+ & Z_- \end{bmatrix} diag\{\Lambda_+, -\Lambda_+\} \begin{bmatrix} Z_+ & Z_- \end{bmatrix}^H$.

5: Set $\begin{bmatrix} X_1 \\ X_2 \end{bmatrix} = \begin{bmatrix} I_n & 0 \\ 0 & -I_n \end{bmatrix} QLZ_+\Lambda_+^{-1/2}$

- Main workload: Solve a strictly imaginary Hermitian eigenvalue problem.

The resulting algorithm

Algorithm 1 Algorithm for the complex Bethe-Salpeter eigenvalue problem

Require: $A = A^H, B = B^T \in \mathbb{C}^{n\times n}$, s.t. $\begin{bmatrix} I_n & 0 \\ 0 & -I_n \end{bmatrix} H_{BS} = \begin{bmatrix} A & B \\ \bar{B} & \bar{A} \end{bmatrix} > 0$

Ensure: $X_1, X_2 \in \mathbb{C}^{n\times n}$ and $\Lambda_+ = diag\{\lambda_1, \ldots, \lambda_n\}$ satisfying $H \begin{bmatrix} X_1 \\ X_2 \end{bmatrix} = \begin{bmatrix} X_1 \\ X_2 \end{bmatrix} \Lambda_+$.

1: Construct $M = \begin{bmatrix} Re(A+B) & Im(A-B) \\ -Im(A+B) & Re(A-B) \end{bmatrix}$

2: Compute the Cholesky factorization $M = LL^T$

3: Construct $W = L^T \begin{bmatrix} 0 & I_n \\ -I_n & 0 \end{bmatrix} L$

4: Compute the spectral decomposition $-iW = \begin{bmatrix} Z_+ & Z_- \end{bmatrix} diag\{\Lambda_+, -\Lambda_+\} \begin{bmatrix} Z_+ & Z_- \end{bmatrix}^H$.

5: Set $\begin{bmatrix} X_1 \\ X_2 \end{bmatrix} = \begin{bmatrix} I_n & 0 \\ 0 & -I_n \end{bmatrix} QLZ_+\Lambda_+^{-1/2}$

- Main workload: Solve a strictly imaginary Hermitian eigenvalue problem.

→ Imanginary part is skew-symmetric.

Implementation

- W is skew-symmetric

⇒ can be reduced to tridiagonal form (e.g via Householder transformations):

$$UWU^T = T = \begin{bmatrix} 0 & \alpha_1 & & \\ -\alpha_1 & 0 & \ddots & \\ & \ddots & \ddots & \alpha_{2n-2} \\ & & -\alpha_{2n-2} & 0 \end{bmatrix}$$

Implementation

- W is skew-symmetric

$\Rightarrow$ can be reduced to tridiagonal form (e.g via Householder transformations):

$$UWU^T = T = \begin{bmatrix} 0 & \alpha_1 & & \\ -\alpha_1 & 0 & \ddots & \\ & \ddots & \ddots & \alpha_{2n-2} \\ & & -\alpha_{2n-2} & 0 \end{bmatrix}$$

- Can be transformed to symmetric form using $D = diag\{1, i, i^2, \ldots, i^{2n}\}$

$$-iD^H \begin{bmatrix} 0 & \alpha_1 & & \\ -\alpha_1 & 0 & \ddots & \\ & \ddots & \ddots & \alpha_{2n-2} \\ & & -\alpha_{2n-2} & 0 \end{bmatrix} D = \begin{bmatrix} 0 & \alpha_1 & & \\ \alpha_1 & 0 & \ddots & \\ & \ddots & \ddots & \alpha_{2n-2} \\ & & \alpha_{2n-2} & 0 \end{bmatrix}$$

Implementation

- Reduction to tridiagonal form by editing ScaLAPACK's routine for tridiagonal reduction of symmetric matrices:
  - `PDSYTRD` $\rightarrow$ `PDSSTRD`

Implementation

- Reduction to tridiagonal form by editing ScaLAPACK's routine for tridiagonal reduction of symmetric matrices:
  - `PDSYTRD` $\rightarrow$ `PDSSTRD`
- Solve symmetric tridiagonal EVP via
  1. Bisection method (`PDSTEBZ`)
  2. Inverse Iteration (`PDSTEIN`).

Implementation

- Reduction to tridiagonal form by editing ScaLAPACK's routine for tridiagonal reduction of symmetric matrices:
  - `PDSYTRD` $\rightarrow$ `PDSSTRD`
- Solve symmetric tridiagonal EVP via
  1. Bisection method (`PDSTEBZ`)
  2. Inverse Iteration (`PDSTEIN`).
- Back transformation of eigenvectors:

$$\begin{bmatrix} X_1 \\ X_2 \end{bmatrix} = \begin{bmatrix} I_n & 0 \\ 0 & -I_n \end{bmatrix} QLUDV_+\Lambda_+^{-1/2},$$

where all matrices but $D = diag\{1, i, i^2 \dots, i^{2n}\}$ are real.

The ELPA Library

- BSEPACK is just proof-of-concept, not performance optimized.

→ Room for improvement by using better libraries!

The ELPA Project[2]

Eigenvalue So**L**vers for **P**etaflop-**A**pplications

The publicly available ELPA library provides **highly efficient** *and* **highly scalable** *direct eigensolvers for* **symmetric matrices**. *Though especially designed for use for PetaFlop/s applications solving large problem sizes on* **massively parallel supercomputers**, *ELPA eigensolvers have proven to be also very efficient for smaller matrices.*

- Mainly developed at *Max Planck Computing and Data Facility* (MPCDF) in Garching.
- Original application area: electronic structure calculations.

[2] https://elpa.mpcdf.mpg.de/

Tridiagonalisation in ELPA

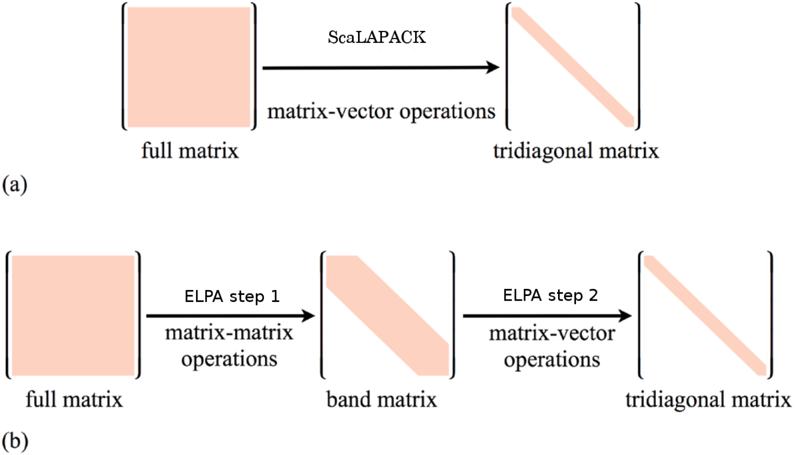


Figure: ELPA employs a two-step tridiagonalisation for symmetric or Hermitian matrices. [Marek et al. '14]

ELPA vs. ScaLAPACK

- BLAS lvl. 3 routines for redution to banded form.
  → More data locality, less communication, highly efficient GEMM routines!
- Carefully crafted communication patterns in reduction to tridiagonal form and eigenvector back transformation.
- Solution of Tridiagonal Systems: Divide-and-Conquer instead of Bisection Method and Inverse Iteration.
- OpenMP and GPU support.

Better Performance and **Scalability**!

Using ELPA to enhance BSEPACK

Two promising points of attack for ELPA:

1. Diagonalize complex Hermitian matrix $-iW = Z\Lambda Z^H$ using ELPA
   + Main portion of the workload could benefit from performance and scalability of ELPA.
   – Uses complex arithmetic, while BSEPACK mainly work on real data.

2. Extend the ELPA algorithm for skew-symmetric matrices, just as ScaLAPACK was extended in BSEPACK.
   + Easy from mathematical point of view
   – Major revision of ELPA software stack necessary.

Using ELPA to enhance BSEPACK

Two promising points of attack for ELPA:

1. Diagonalize complex Hermitian matrix $-iW = Z\Lambda Z^H$ using ELPA
   - $+$ Main portion of the workload could benefit from performance and scalability of ELPA.
   - $-$ Uses complex arithmetic, while BSEPACK mainly work on real data.

2. Extend the ELPA algorithm for skew-symmetric matrices, just as ScaLAPACK was extended in BSEPACK.
   - $+$ Easy from mathematical point of view
   - $-$ Major revision of ELPA software stack necessary.

Numerical Test Setup

System DRACO

- Located at Max Planck Computing and Data Facility
- HPC Extension to HPC System HYDRA
- 880 nodes, Intel 'Haswell' Xeon E5-2698, 32 cores @ 2.3 GHz
- 128 GB main memory per node
- Interconnect: fast InfiniBand FDR14 network

http://www.mpcdf.mpg.de/services/computing/draco/about-the-system

- $A \in \mathbb{C}^{n \times n}$ and $B \in \mathbb{C}^{n \times n}$ where initialized with random complex values.
- Diagonal of A positive and scaled up, s.t. $\begin{bmatrix} A & B \\ \bar{B} & \bar{A} \end{bmatrix}$ is positive definite (Gershgorin Circle Theorem).
- To work on a matrix on a distributed memory machine, it is divided into many subblocks of a certain block sizes $n_b \times n_b$.

- $A \in \mathbb{C}^{n \times n}$ and $B \in \mathbb{C}^{n \times n}$ where initialized with random complex values.
- Diagonal of A positive and scaled up, s.t. $\begin{bmatrix} A & B \\ \bar{B} & \bar{A} \end{bmatrix}$ is positive definite (Gershgorin Circle Theorem).
- To work on a matrix on a distributed memory machine, it is divided into many subblocks of a certain block sizes $n_b \times n_b$.

Numerical Tests:

1. Test impact of block size on performance for $n = 25,000$.
2. Test strong scalability for medium sized matrices $n = 25,000$.
3. Compare runtimes for larger matrices up to $n = 75,000$.

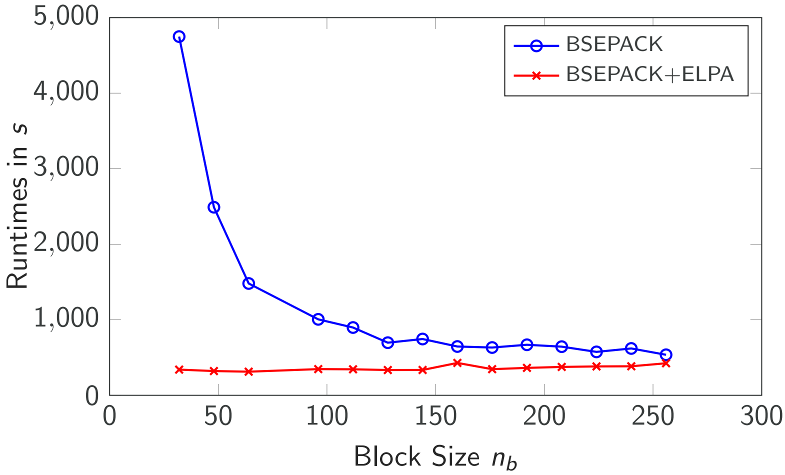


Figure: Runtimes for different block sizes at $n = 25,000$.

Scalability of BSEPACK and BSEPACK+ELPA

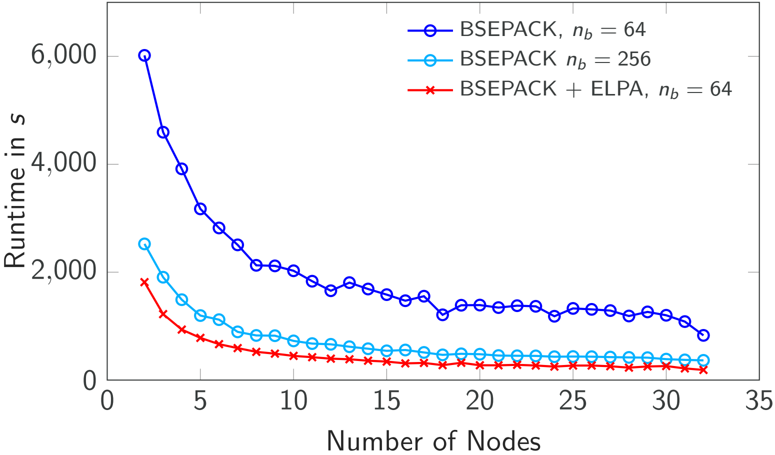


Figure: Strong Scalability for medium sized matrix ($n = 25,000$) in BSEPACK and BSEPACK enhanced with ELPA. 32 threads were used per node.

Scalability of BSEPACK and BSEPACK+ELPA

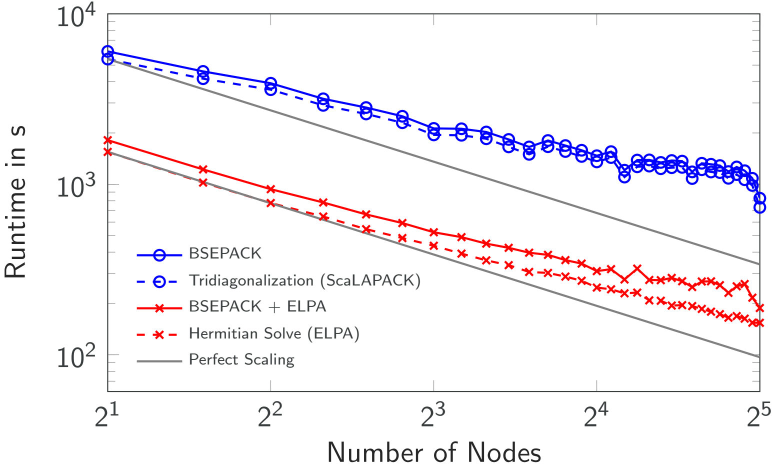


Figure: Strong Scalability for medium sized matrix ($n = 25,000$) in BSEPACK and BSEPACK enhanced with ELPA. 32 threads were used per node.

Runtimes for large matrices

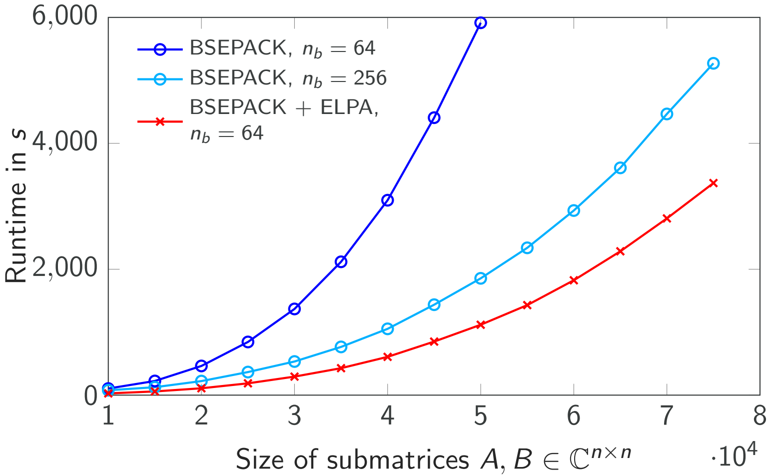


Figure: Runtimes of BSEPACK and BSEPACK+ELPA for large matrices of size $2n \times 2n$.

Runtimes for large matrices

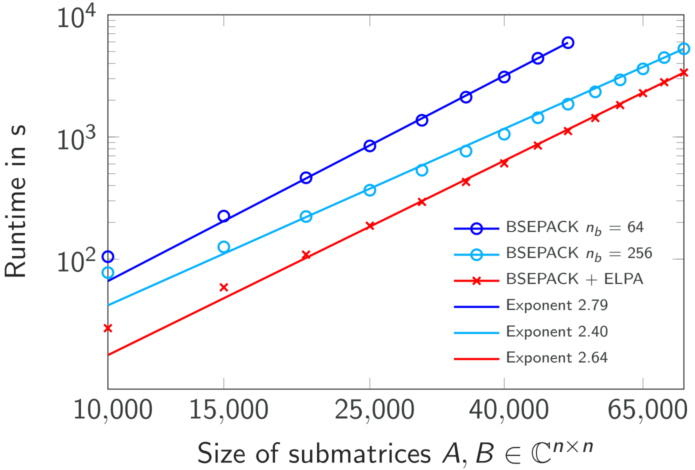


Figure: Runtimes of BSEPACK and BSEPACK+ELPA for large matrices of size $2n \times 2n$.

Using ELPA to enhance BSEPACK

Two promising points of attack for ELPA:

1. Diagonalize complex Hermitian matrix $-iW = Z\Lambda Z^H$ using ELPA
   - \+ Main portion of the workload could benefit from performance and scalability of ELPA.
   - − Uses complex arithmetic, while BSEPACK mainly work on real data.

2. Extend the ELPA algorithm for skew-symmetric matrices, just as ScaLAPACK was extended in BSEPACK.
   - \+ Easy from mathematical point of view
   - − Major revision of ELPA software stack necessary.

Stays the same:

- Computation of Householder vectors.
- Application to off-diagonal blocks.
- Tridiagonal solve.
- Most of the back transformation of eigenvectors.
- All communication and synchronizations.

Stays the same:

- Computation of Householder vectors.
- Application to off-diagonal blocks.
- Tridiagonal solve.
- Most of the back transformation of eigenvectors.
- All communication and synchronizations.

Change wherever symmetry is implicitly assumed:

- Updates on blocks including the diagonal:

$$A - (I - \tau vv^T)A(I - \tau vv^T) = A - v\underbrace{(\tau v^T A - 0.5\tau^2 v^T A v v^T)}_{u_1^T} - \underbrace{(A\tau v - 0.5\tau^2 v v^T A v)}_{u_2} v^T$$

$$A = A^T \Rightarrow u_2 = u_1, \qquad A = -A^T \Rightarrow u_2 = -u_1$$

ELPA for Skew-Symmetric Matrices

- Use skew-symmetric BLAS routines provided by BSEPACK.
  - DSYR2 $\rightarrow$ DSSR2
  - DSYMV $\rightarrow$ DSSMV
- Multiplication with $D = diag\{1, i, i^2, \ldots, i^{2n}\}$ in back transformation.

ELPA for Skew-Symmetric Matrices

- Use skew-symmetric BLAS routines provided by BSEPACK.
  - DSYR2 $\rightarrow$ DSSR2
  - DSYMV $\rightarrow$ DSSMV
- Multiplication with $D = diag\{1, i, i^2, \ldots, i^{2n}\}$ in back transformation.

Preliminary results from a smaller compute server.

Bruno

- 2 Intel 'Haswell' Xeon E5-2640v3, 8 cores each, @2.6GHz
- 32 GB main memory per CPU

Runtimes for large matrices

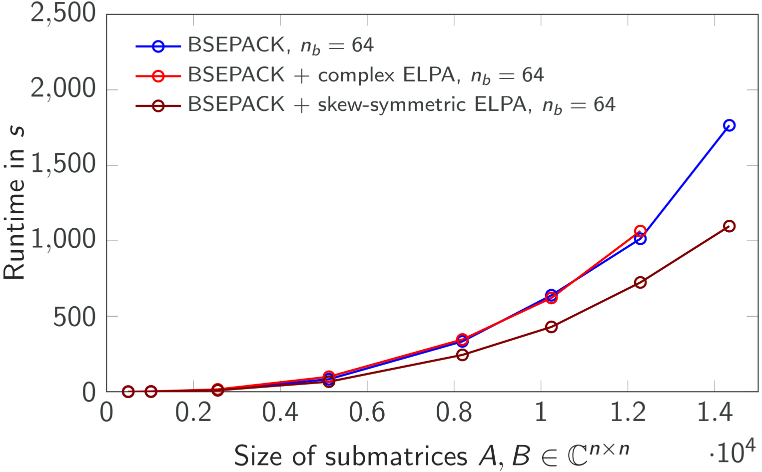


Figure: Preliminary results matrices of size $2n \times 2n$ running on 16 cores.

Easy further improvements:

- Rank-2 Update becomes easier for skew-symmetric matrices:

$$
\begin{aligned}
A-(I-\tau vv^T)A(I-\tau vv^T) = A - v(\tau v^T A - 0.5\tau^2 \underbrace{v^T A v}_{=0} v^T) \\
-(A\tau v - 0.5\tau^2 v \underbrace{v^T A v}_{=0})v^T
\end{aligned}
$$

- Eigenvector resulting from tridiagonal solve have to be multiplied by complex diagonal D before back transformation.

$\rightarrow$ Instead of applying Householder transformation directly, we apply them to identity matrix and then multiply.

- Easier to implement but more FLOPs.

Future Work

- ELPA for skew-symmetric matrices on production level (including OpenMP and GPU support)
- HPC implementations of algorithms for Hamiltonian matrices.
- Structure preserving Divide-and-Conquer scheme based on the matrix disk function / doubling algorithm [Bai, Demmel, Gu '97] and permuted Lagrangian Graph Bases [Mehrmann, Poloni '12].

Thank you for your attention!

The ELPA library can be found at
https://elpa.mpcdf.mpg.de/

M. Shao, F. H. da Jornada, C. Yang, J. Deslippe, and S. G. Louie.
Structure preserving parallel algorithms for solving the Bethe-Salpeter eigenvalue problem.
Linear Algebra and its Applications, 488(Supplement C):148 – 167, 2016.

P. Benner, H. Faßbender, and C. Yang.
Some remarks on the complex J-symmetric eigenproblem.
Preprint MPIMD/15-12, Max Planck Institute Magdeburg, July 2015.
Available from http://www.mpi-magdeburg.mpg.de/preprints/.

T. Auckenthaler, V. Blum, H.-J. Bungartz, T. Huckle, R. Johanni, L. Krmer, B. Lang, H. Lederer, and P.R. Willems.
Parallel solution of partial symmetric eigenvalue problems from electronic structure calculations.
Parallel Computing, 37(12):783 – 794, 2011.
6th International Workshop on Parallel Matrix Algorithms and Applications (PMAA'10).

Z. Bai, J. Demmel, and M. Gu.
An Inverse Free Parallel Spectral Divide and Conquer Algorithm for Nonsymmetric Eigenproblems.
76(3):279–308, 1997.

Peter Benner, Heike Fabender, and Chao Yang.
Some remarks on the complex J-symmetric eigenproblem.
Linear Algebra and its Applications, 544:407 – 442, 2018.

V. Mehrmann and F. Poloni.
Doubling algorithms with permuted Lagrangian graph bases.
SIAM J. Matrix Anal. Appl., 33(3):780–805, 2012.