ELSI: A Unified Software Interface to Solve or Circumvent the Kohn-Sham Eigenvalue Problem

Victor Yu\textsuperscript{1}, Fabiano Corsetti\textsuperscript{2}, William Huhn\textsuperscript{1}, Björn Lange\textsuperscript{1}, Lin Lin\textsuperscript{3,4}, Jianfeng Lu\textsuperscript{5}, Álvaro Vázquez-Mayagoitia\textsuperscript{6}, Chao Yang\textsuperscript{7}, and Volker Blum\textsuperscript{1}

\textsuperscript{1} Department of Mechanical Engineering and Materials Science, Duke University
\textsuperscript{2} Department of Materials, Imperial College London
\textsuperscript{3} Department of Mathematics, UC Berkeley
\textsuperscript{4} Lawrence Berkeley National Laboratory
\textsuperscript{5} Department of Mathematics, Duke University
\textsuperscript{6} Argonne National Laboratory
The “Cubic Wall” in Kohn-Sham Density-Functional Theory

DFT-PBE calculations

System: Supercell models of graphene monolayer (4,050 ~ 7,200 atoms)

Basis set: 14 basis functions per atom

Problem size: 56,700 ~ 100,800

* All benchmark matrices from FHI-aims (all-electron, NAO basis)

The “Cubic Wall” in Kohn-Sham Density-Functional Theory

DFT-PBE calculations

System: Supercell models of graphene monolayer (4,050 ~ 7,200 atoms)

Basis set: 14 basis functions per atom

Problem size: 56,700 ~ 100,800

* All benchmark matrices from FHI-aims (all-electron, NAO basis)

**Scalability limit**: $O(N^3)$ solution of the Kohn-Sham eigenvalue problem

![Graph showing the time for one SCF iteration vs. the number of atoms. The graph includes lines for Eigenvalue problem, All other steps, and Total. The x-axis represents the number of atoms, and the y-axis represents the time in seconds. The graph compares the time on Edison (Intel Ivy Bridge) with 80 nodes and 24 CPU cores per node.](image)
**Scalability limit:** $O(N^3)$

Kohn-Sham eigenvalue problem

Non-orthogonal basis:

$$Hc = \lambda Sc$$

Orthogonal basis:

$$Hc = \lambda c$$

The goal is to update the electron density based on either the “*orbitals*” or the “*density matrix*”

---

**Graph:**

- **X-axis:** Number of atoms
- **Y-axis:** Time
- **Data:**
  - Edison (Intel Ivy Bridge)
    - 80 nodes x 24 CPU cores per node
  - $O(N^3)$
The “Cubic Wall” in Kohn-Sham Density-Functional Theory

**Scalability limit**: $O(N^3)$ solution of the Kohn-Sham eigenvalue problem

The *same* problem must be solved by all electronic structure codes based on KS-DFT, regardless of the underlying numerical and physical choices.
**Solutions to Kohn-Sham Eigenvalue Problem**

**ELPA**: Replacement of ScaLAPACK eigensolver with increased *speed* and *scalability*

http://elpa.mpcdf.mpg.de

**Benchmark**: Álvaro Vázquez-Mayagoitia, ANL

Theta (Intel Knights Landing)
9.65 petaFLOPS system @ ANL

![Graph showing time for ELPA 2-stage solver vs. number of MPI ranks]

Matrix size: 1048576
Solutions to Kohn-Sham Eigenvalue Problem

**ELPA**: Replacement of ScaLAPACK eigensolver with increased *speed* and *scalability*

http://elpa.mpcdf.mpg.de

**Direct, dense eigensolver** is efficient for small-to-medium-sized problems, even in O(N) DFT implementations

Benchmark: Álvaro Vázquez-Mayagoitia, ANL

Theta (Intel Knights Landing)
9.65 petaFLOPS system @ ANL
**Solutions to Kohn-Sham Eigenvalue Problem**

**ELPA**: Replacement of ScALAPACK eigensolver with increased *speed* and *scalability*

http://elpa.mpcdf.mpg.de

**Direct, dense eigensolver** is efficient for small-to-medium-sized problems, even in O(N) DFT implementations

Stable and generally applicable to metals, semiconductors, insulators

Benchmark: Álvaro Vázquez-Mayagoitia, ANL

Theta (Intel Knights Landing)
9.65 petaFLOPS system @ ANL

Matrix size: 1048576

Time for ELPA 2-stage solver [s]

Number of MPI ranks
**Solutions to Kohn-Sham Eigenvalue Problem**

**ELPA**: Replacement of ScaLAPACK eigensolver with increased **speed** and **scalability**

http://elpa.mpcdf.mpg.de

**Direct, dense eigensolver** is efficient for small-to-medium-sized problems, even in $O(N)$ DFT implementations

Stable and generally applicable to metals, semiconductors, insulators

**$O(N^3)$ scaling bottleneck**

**Dense** memory bottleneck

 Benchmark: Álvaro Vázquez-Mayagoitia, ANL

Theta (Intel Knights Landing) 9.65 petaFLOPS system @ ANL

Matrix size: 1048576
Solutions to Kohn-Sham Eigenvalue Problem

**ELPA** (eigenvalues/vectors)

Massively parallel dense eigensolver

$O(N^3)$

Solutions to Kohn-Sham Eigenvalue Problem

**ELPA** (eigenvalues/vectors)
Massively parallel dense eigensolver
\[ O(N^3) \]

**libOMM** (density matrix)
Orbital minimization method
\[ O(N^3) \] with a reduced prefactor

Solutions to Kohn-Sham Eigenvalue Problem

**ELPA** (eigenvalues/vectors)
Massively parallel dense eigensolver
$O(N^3)$


**libOMM** (density matrix)
Orbital minimization method
$O(N^3)$ with a reduced prefactor


**PEXSI** (density matrix)
Pole expansion and selected inversion exploiting sparse linear algebra
At most $O(N^2)$

Solutions to Kohn-Sham Eigenvalue Problem

**ELPA** (eigenvalues/vectors)
Massively parallel dense eigensolver

$O(N^3)$


**libOMM** (density matrix)
Orbital minimization method

$O(N^3)$ with a reduced prefactor


**PEXSI** (density matrix)
Pole expansion and selected inversion exploiting sparse linear algebra

At most $O(N^2)$


And many others, including:

ScaLAPACK, O(N) methods, ...
Solutions to Kohn-Sham Eigenvalue Problem

- **ELPA** (eigenvalues/vectors)
  - Massively parallel eigensolver
  - $O(N^3)$

- **libOMM** (density matrix)
  - orbital minimization method
  - $O(N^3)$ with a reduced prefactor

- **PEXSI** (density matrix)
  - Pole expansion and selected inversion exploiting sparse linear algebra
  - At most $O(N^2)$

- Why not switch dynamically between different solvers?

- Including:
Solutions to Kohn-Sham Eigenvalue Problem

**ELPA** (eigenvalues/vectors)
Massively parallel dense eigensolver
$O(N^3)$

**libOMM** (density matrix)
Orbital minimization method
$O(N^3)$ with a reduced prefactor

**PEXSI** (density matrix)
Pole expansion and selected inversion exploiting sparse linear algebra
At most $O(N^2)$

---

**Why not switch dynamically between different solvers?**
-- Unfortunately not an easy task, due to:

- Different **matrix storage strategies**
- Different **Application Programming Interfaces**
- Different **programming languages**

---

Solutions to Kohn-Sham Eigenvalue Problem

**ELPA** (eigenvalues/vectors)
Massively parallel dense eigensolver $O(N^3)$

**libOMM** (density matrix)
Orbital minimization method $O(N^3)$ with a reduced prefactor

**PEXSI**
Pole expansion and selected inversion exploiting sparse linear algebra
At most $O(N^2)$

§ Why not switch dynamically between different solvers?
-- Unfortunately not an easy task, due to:

- Different matrix storage strategies
- Different Application Programming Interfaces
- Different programming languages

§ ELSI is designed to simplify the access to KS solvers

---


wy29@duke.edu
ELSI: Connection between DFT Codes and KS Solutions

KS-DFT Codes

Create Hamiltonian and overlap

Solvers

- ELPA
- libOMM
- PEXSI
...

Compute eigenvalues, eigenvectors and density matrix
ELSI: Connection between DFT Codes and KS Solutions

KS-DFT Codes

Create Hamiltonian and overlap

ELSI

Matrix Conversion

Solvers

Compute eigenvalues, eigenvectors and density matrix

ELPA
libOMM
PEXSI
...
**ELSI: Connection between DFT Codes and KS Solutions**

**KS-DFT Codes**
- \( \ldots, \phi, \rho \)
- DM
- parameters
- \( S \)
- \( H \)

**Create Hamiltonian and overlap**
- Convert data format and distribution automatically

**ELSI**
- Matrix Conversion

**Solvers**
- ELPA
- libOMM
- PEXSI
- \( \ldots \)

**Compute eigenvalues, eigenvectors and density matrix**
ELSI: Connection between DFT Codes and KS Solutions

KS-DFT Codes

Create Hamiltonian and overlap

- Convert data format and distribution automatically
- Analyze the problem:
  Basis set size, sparsity, property of overlap matrix, requested output, desired accuracy, available computational resource (ongoing)

ELSI

Matrix Conversion

Solvers

Compute eigenvalues, eigenvectors and density matrix

ELPA
libOMM
PEXSI

...
**ELSI: Connection between DFT Codes and KS Solutions**

**KS-DFT Codes**

```
... \rightarrow \phi \rightarrow DM \rightarrow parameters \rightarrow S \rightarrow H
```

- Convert data format and distribution automatically
- Analyze the problem:
  - Basis set size, sparsity, property of overlap matrix, requested output, desired accuracy, available computational resource (ongoing)
- Suggest a solver with reasonable settings (future)

**ELSI**

```
\phi \rightarrow DM \rightarrow problem
```

**Solvers**

- ELPA
- libOMM
- PEXSI
- ...

```
Create Hamiltonian and overlap
Compute eigenvalues, eigenvectors and density matrix
```
ELSI: Connection between DFT Codes and KS Solutions

What can we do with ELSI?

- Convert data format and distribution automatically
- Analyze the problem: basis set size, sparsity, property of overlap matrix, requested output, desired accuracy, available computational resource (ongoing)
- Suggest a solver with reasonable settings (future)

What can we do with ELSI?

- Compute eigenvalues, eigenvectors and density matrix

KS-DFT Codes

Create Hamiltonian and overlap

ELSI

Solvers

ELPA
libOMM
PEXSI
...
Investigate the World Beyond the “Cubic Wall”

DFT-PBE calculations

System: Supercell models of graphene monolayer (800 ~ 7,200 atoms)

Basis set: 14 basis functions per atom

Problem size: 44,800 ~ 100,800
DFT-PBE calculations

System: Supercell models of graphene monolayer (800 ~ 7,200 atoms)

Basis set: 14 basis functions per atom

Problem size: 44,800 ~ 100,800
Investigate the World Beyond the “Cubic Wall”

DFT-PBE calculations

System: Supercell models of graphene monolayer (800 ~ 7,200 atoms)

Basis set: 14 basis functions per atom

Problem size: 44,800 ~ 100,800

Time for KS solution [s]

Number of atoms

Edison (Intel Ivy Bridge)
80 nodes x 24 CPU cores per node

ELPA (occupied states + some empty states)
libOMM
DFT-PBE calculations

System: Supercell models of graphene monolayer (800 ~ 7,200 atoms)

Basis set: 14 basis functions per atom

Problem size: 44,800 ~ 100,800

EDD (Intel Ivy Bridge)
80 nodes x 24 CPU cores per node
Investigate the World Beyond the “Cubic Wall”

DFT-PBE calculations

System: Supercell models of graphene monolayer (800 ~ 7,200 atoms)

Basis set: 14 basis functions per atom

Problem size: 44,800 ~ 100,800

Time for KS solution [s]

- ELPA (occupied states + some empty states)
- libOMM
- PEXSI

Edison (Intel Ivy Bridge)
80 nodes x 24 CPU cores per node

Number of atoms

800 1800 3200 5000 7200

$10^1$ $10^2$
DFT-PBE calculations

System: Supercell models of graphene monolayer (800 ~ 7,200 atoms)

Basis set: 14 basis functions per atom

Problem size: 44,800 ~ 100,800

Crossover: ~ 3,000 atoms
Investigate the World Beyond the “Cubic Wall”

DFT-PBE calculations

System: Supercell models of graphene monolayer (800 ~ 7,200 atoms)

Basis set: 14 basis functions per atom

Problem size: 44,800 ~ 100,800

Crossover: ~ 3,000 atoms
Speed-up at 7,200 atoms: 2.06x

Time for KS solution [s]

Edison (Intel Ivy Bridge)
80 nodes x 24 CPU cores per node

2.06x
Combine the Strengths of ELSI components

DFT-PBE calculations

System: Polymerized heptazine 4-layer model (288 atoms)

Basis set: 12.5 ~ 83.8 basis functions per atom

Problem size: 3,600 ~ 24,144
Combine the Strengths of ELSI components

In the initial implementation, libOMM took long time to the solution

![Graph showing the comparison between ELPA and libOMM for the time needed to solve the Kohn-Sham equations on the Theta (Intel Knight Landing) supercomputer. The x-axis represents the number of basis functions, and the y-axis represents the time in seconds. The graph shows a clear advantage for ELPA over libOMM, with ELPA scaling more efficiently as the number of basis functions increases.]
In the initial implementation, libOMM took long time to the solution

Bottleneck: ScaLAPACK subroutine \texttt{pdsygst} (reduces a generalized eigenproblem to the standard form)
**Combine the Strengths of ELSI components**

In the initial implementation, libOMM took long time to the solution

Bottleneck: ScaLAPACK subroutine *pdsygst* (reduces a generalized eigenproblem to the standard form)

A significant speed-up achieved by using *ELPA* counterpart subroutines to perform the transformation
Philosophy of the ELSI Interface

*Easy to implement*

Designed to be flexible for rapid integration into a variety of KS-DFT codes
Philosophy of the ELSI Interface

*Easy to implement*

Designed to be flexible for rapid integration into a variety of KS-DFT codes

Original interfaces to use
ELPA, libOMM, PEXSI libraries

- elpa_get_communicators
- elpa_solve_evp_1stage
- elpa_solve_evp_2stage
- elpa_cholesky
- ms_scalapack_setup_omm
- pexsi_plan_init
- pexsi_load_hs_matrices
- pexsi_dft_driver
Philosophy of the ELSI Interface

**Easy to implement**

Designed to be flexible for rapid integration into a variety of KS-DFT codes

Original interfaces to use
ELPA, libOMM, PEXSI libraries

- elpa_get_communicators
- elpa_solve_evp_1stage
- elpa_solve_evp_2stage
- elpa_cholesky
- ms_scalapack_setup
- omm
- pexsi_plan_init
- pexsi_load_hs_matrices
- pexsi_dft_driver

Great and easy to use independently
Philosophy of the ELSI Interface

**Easy to implement**

Designed to be flexible for rapid integration into a variety of KS-DFT codes

Original interfaces to use
ELPA, libOMM, PEXSI libraries

elpa_get_communicators
elpa_solve_evp_1stage
elpa_solve_evp_2stage
elpa_cholesky

ms_scalapack_setup
omm

pexsi_plan_init
pexsi_load_hs_matrices
pexsi_dft_driver

Great and easy to use independently

But complicated to integrate all of them together in a single DFT code
Philosophy of the ELSI Interface

**Easy to implement**

Designed to be flexible for rapid integration into a variety of KS-DFT codes

Original interfaces to use
ELPA, libOMM, PEXSI libraries

- elpa_get_communicators
- elpa_solve_evp_1stage
- elpa_solve_evp_2stage
- elpa_cholesky

API of ELSI interface

- **elsi_init**
  - ! BEGIN SCF LOOP

- **elsi_solver**
  - ! END SCF LOOP

- **elsi_finalize**
  - ms_scalapack_setup_omm
  - pexsi_plan_init
  - pexsi_load_hs_matrices
  - pexsi_dft_driver
**Philosophy of the ELSI Interface**

**Easy to implement**

Designed to be flexible for rapid integration into a variety of KS-DFT codes

**Versatile**

Includes density matrix and eigensystem formalisms on equal footing

**API of ELSI interface**

```plaintext
elsi_init
! BEGIN SCF LOOP

elsi_{dm|ev}

! END SCF LOOP

elsi_finalize
```
Philosophy of the ELSI Interface

Easy to implement

Designed to be flexible for rapid integration into a variety of KS-DFT codes

Versatile

Includes density matrix and eigensystem formalisms on equal footing

Flexible

All technical settings in solver libraries are adjustable for experienced users

API of ELSI interface

```
elsi_init
! BEGIN SCF LOOP
  elsi_customize
  elsi_{dm|ev}
! END SCF LOOP
elsi_finalize
```
Philosophy of the ELSI Interface

**Easy to implement**

Designed to be flexible for rapid integration into a variety of KS-DFT codes

**Versatile**

Includes density matrix and eigensystem formalisms on equal footing

**Flexible**

All technical settings in solver libraries are adjustable for experienced users

**Portability**

Successful compilation and execution proven on a broad list of leading supercomputers, including:
Philosophy of the ELSI Interface

**Easy to implement**

Designed to be flexible for rapid integration into a variety of KS-DFT codes

**Versatile**

Includes density matrix and eigensystem formalisms on equal footing

**Flexible**

All technical settings in solver libraries are adjustable for experienced users

**Portability**

Successful compilation and execution proven on a broad list of leading supercomputers, including:

- Titan: Cray XK7 @ ORNL
- Cori: Cray XC40/KNL @ NERSC
- Mira: IBM BlueGene/Q @ ANL
- Theta: Cray XC40/KNL @ ANL

Compilers: GNU, Intel, IBM, PGI, Cray
The ELSI Interchange: Community Website and Services

The ELSI Interchange: Community Website and Services

http://elsi-interchange.org

- Code repository (GitLab)
  Please ask us for access: elsi-team@duke.edu

- Wiki, forum, webinar, ...

- Code integration assistance
ELSI: A Unified Software Interface to Solve or Circumvent the Kohn-Sham Eigenvalue Problem

- **Simple** Easy access to *Kohn-Sham solutions* for KS-DFT codes
ELSI: A Unified Software Interface to Solve or Circumvent the Kohn-Sham Eigenvalue Problem

- **Simple** Easy access to *Kohn-Sham solutions* for KS-DFT codes
- **Portable** Supporting *all* major compiler suites
  Running on world leading *supercomputers*
**ELSI: A Unified Software Interface to Solve or Circumvent the Kohn-Sham Eigenvalue Problem**

- **Simple**  
  Easy access to *Kohn-Sham solutions* for KS-DFT codes

- **Portable**  
  Supporting *all* major compiler suites  
  Running on world leading *supercomputers*

- **Scalable**  
  (Hundreds of) *thousands* of CPU cores

---

Please give it a try!
**ELSI: A Unified Software Interface to Solve or Circumvent the Kohn-Sham Eigenvalue Problem**

- **Simple** Easy access to *Kohn-Sham solutions* for KS-DFT codes
- **Portable** Supporting *all* major compiler suites
  Running on world leading *supercomputers*
- **Scalable** (Hundreds of) *thousands* of CPU cores

Please give it a try!

**Acknowledgment**

ELSI is an NSF SI2-SSI supported software infrastructure project under Grant Number 1450280. Any opinions, findings, and conclusions or recommendations expressed here are those of the author(s) and do not necessarily reflect the views of NSF.