Eigenproblems and Eigensolvers in Density Functional Theory

Broadly defined “reverse engineering” applied to simulations in materials science

Edoardo Di Napoli

Jülich Supercomputing Center - Institute for Advanced Simulation
Forschungszentrum Jülich GmbH

Aachen Institute for Advanced Study in Computational Engineering Science
RWTH Aachen University

JSC, February the 2nd 2012
### The science-driven approach

Mathematical Model $\Rightarrow$ Algorithmic Structure $\Rightarrow$ Simulations $\Rightarrow$ Physics

1. **Math $\Rightarrow$ Algorithm** – Often a “literal” translation of the mathematical model into a combination of algorithmic tasks usually designed in abstraction from other considerations (kernel opt., tasks harmonization, etc.);

2. **Algorithm $\Rightarrow$ Sim** – The simulation is an end product, the successful implementation of an algorithm into a series of machine accessible operations resulting in the computation of meaningful physical quantities;

3. **Math $\Rightarrow$ Sim** – The mathematical model and the simulation are considered as practically disjoint.
Reverse Simulation
A different approach

Mathematical model
Algorithmic structure

FLAPW Paradigm

Mathematical Model
Algorithmic Structure

Feedback from analysis of correlated eigenproblems \( \{P_i\} \)

An Example: Sequences of eigenproblems in Density Functional Theory
1. DFT and the generalized eigenvalue problem

2. Sequences of Eigenproblems

3. Correlation

4. Exploiting the correlation \textsc{Algorithm} \Leftarrow \textsc{Sim}

5. Iterative solver on multi-threaded architectures
Outline

1. DFT and the generalized eigenvalue problem
2. Sequences of Eigenproblems
3. Correlation
4. Exploiting the correlation ALGORITHM ← SIM
5. Iterative solver on multi-threaded architectures
Outline

1. DFT and the generalized eigenvalue problem
2. Sequences of Eigenproblems
3. Correlation
4. Exploiting the correlation ALGORITHM ← SIM
5. Iterative solver on multi-threaded architectures
Outline

1. DFT and the generalized eigenvalue problem
2. Sequences of Eigenproblems
3. Correlation
4. Exploiting the correlation \textsc{Algorithm} $\leftarrow$ \textsc{Sim}
5. Iterative solver on multi-threaded architectures
Outline

1. DFT and the generalized eigenvalue problem
2. Sequences of Eigenproblems
3. Correlation
4. Exploiting the correlation ALGORITHM ⇐ SIM
5. Iterative solver on multi-threaded architectures
Quantum Mechanics and its ingredients

- \( H = -\frac{1}{2} \sum_{i=1}^{n} \nabla_i^2 - \sum_{i=1}^{n} \sum_{\nu} \frac{Z_{\nu}}{|x_i - a_{\nu}|} + \sum_{i<j} \frac{1}{|x_i - x_j|} \)

- \( \Phi(x_1; s_1, x_2; s_2, \ldots, x_n; s_n) \)

\( \Phi : \left( \mathbb{R}^3 \times \{ \pm \frac{1}{2} \} \right)^n \longrightarrow \mathbb{R} \) high-dimensional anti-symmetric function – describes the orbitals of atoms and molecules. In the *Born-Oppenheimer* approximation, it is the solution of the

**Electronic Schrödinger Equation**

\[
H \Phi(x_1; s_1, x_2; s_2, \ldots, x_n; s_n) = E \Phi(x_1; s_1, x_2; s_2, \ldots, x_n; s_n)
\]
Basic ingredients

- Hartree-Fock (H-F) or mean-field theory approximation (electrons in the valence band see the nucleus with a screened charge)
- Solution for ground state $E_0$ (state with lowest energy)

1. $\Phi(x_1; s_1, x_2; s_2, \ldots, x_n; s_n) \approx \Lambda_{i,v} \phi_v(x_i; s_i) \implies \text{density of states } n(r) = \sum_i |\phi_i(r)|^2$

2. In the Schrödinger equation the exact inter-electronic interaction $\sum_{i<j} \frac{1}{|x_i-x_j|}$ is substituted with an effective potential $V_0(r)$

3. One-to-one correspondence $n(r) \iff V_0(r)$

4. $\exists!$ a functional $E[n] : E_0 = \min_n E[n]$

Shift in focus

Many orbital functions $\phi_i \implies \text{One } n(r)$
In practice the high-dimensional Schrödinger equation translates to a non-linear low-dimensional self-consistent equation (Kohn-Sham) whose solution is searched in an iterative fashion

\[ V(\mathbf{r}) \rightarrow V(\mathbf{r})[n] \]

\[ \left( -\frac{\hbar^2}{2m} \nabla^2 + V(\mathbf{r}) \right) \phi_i(\mathbf{r}) = \varepsilon_i \phi_i(\mathbf{r}) \]

\[ n(\mathbf{r}) = \sum_i |\phi_i(\mathbf{r})|^2 \]

**One theory -> many methods**

Several can be the schemes through which DFT can be practically realized. Each one depends on how \( V(\mathbf{r})[n] \) and the \( \phi_i(\mathbf{r}) \) are computed. We concentrate on one of them, namely the Full Potential Augmented Plane Wave method (FLAPW)
Generalized eigenvalue problems \[ Ax = \lambda B x \]

In the FLAPW framework the Kohn-Sham equations translate to generalized eigenvalue equations.

\[ \phi_i(r) \rightarrow \phi_{k,\nu}(r) = \sum_{|G+k| \leq K_{max}} c^G_{k,\nu} \psi_G(k, r) \]

Inserting the LAPW expansion in the equations

\[ \psi^*_G(k, r) \sum_{G'} \hat{H}_{KS} c^G_{k,\nu} \psi_{G'}(k, r) = \lambda_{kv} \psi^*_G(k, r) \sum_{G'} c^G_{k,\nu} \psi_{G'}(k, r), \]

defining the matrix entries for the left and right hand side respectively as Hamiltonian \[ A(k) \] and overlap matrices \[ B(k) \]

\[ \{A(k), B(k)\} = \sum_{\alpha} \int \psi^*_G(k, r) \{\hat{H}_{KS}, \hat{1}\} \psi_{G'}(k, r) \]

and remembering that the basis set is over-complete \((\Rightarrow B \neq \text{diag})\)

\[ \sum_{G'} A_{GG'}(k) c^{G'}_{kv} = \lambda_{kv} \sum_{G'} B_{GG'}(k) c^{G'}_{kv} \]
FLAPW self-consistent cycle

Spherical $V_0(r)$ \[ n_0(r) \text{ init.} \]

Calculation of full-potential $V(r)[n]$

Determining the Lapw wavefunctions $\psi_G(k,r)$

Matrices generation

$A_k = \langle \psi(k) | H | \psi'(k) \rangle$

$B_k = \langle \psi(k) | S | \psi'(k) \rangle$

Generalized eigenproblems solution

$A_k x = \lambda B_k x$

Convergence check $n(r)$ mixing \[ n(r) \text{ for next cycle} \]

Calculation of new charge density $n(r)$

3-D stars, 2-D stars lattice harmonics

$\psi^G_{lm}$ and $b^G_{lm}$ coeff.

Pseudo-charge method

Fermi energy calculation

Eigenpairs selection

Fermi energy calculation

Eigenproblems: distribution and setup
A (mostly) converging process
The traditional approach

**Initialization:** At every iteration of the cycle all numerical quantities are entirely re-computed

- A new set of basis functions is assembled at each iteration cycle (Math ⇐ Sim)
- The entries of the matrices $A$ and $B$ are re-initialized at each iteration cycle (Algorithm ⇐ Sim)

**Eigenproblems:** Each eigenproblem $P^{(i)} : A^{(i)}x = \lambda B^{(i)}x$ at iteration $i$ is solved in total independence from the eigenproblem $P^{(i-1)}$ of the previous iteration (Algorithm ⇐ Sim)

**Convergence:** Starting with an electron density close enough to the one minimizing the energy $E_0$ it is likely to reach convergence within few tens of iterations. Unfortunately there is no general theorem establishing the converging conditions

- # of steps is still uncertain. They depend on the material and the initial guess (area of convergence) (Math ⇐ Algorithm)
- $|n'(\mathbf{r}) - n(\mathbf{r})|$ undergo relatively small decreasing oscillations (Math ⇐ Algorithm & Sim)
Reverse simulation: an **Algorithm** ↔ **Simulation** case

**Sequences of eigenproblems**

- Consider the set of generalized eigenproblems $P^{(1)} \ldots P^{(i)} P^{(i+1)} \ldots P^{(N)} \neq (P)^N$
- Could this sequence $\{P^{(i)}\}$ of eigenproblems evolve following a convergence pattern in line with the convergence of $n(r)$?

**Investigating evidence of evolution**

- **(a)** Eigenvectors transformation from one iteration to the next
- **(b)** Distance between successive Hamiltonian matrices $A_k$
Sequences of eigenproblems

- Consider the set of generalized eigenproblems \( P^{(1)} \ldots P^{(i)} \) \( P^{(i+1)} \ldots P^{(N)} \neq (P)^N \)

- Could this sequence \( \{P^{(i)}\} \) of eigenproblems evolve following a convergence pattern in line with the convergence of \( n(r) \)?

Investigating evidence of evolution

(a) Eigenvectors transformation from one iteration to the next

(b) Distance between successive Hamiltonian matrices \( A_k \)
Eigenvectors properties

\[ \sum_{G'} [A_{GG'}(k) - \lambda_{k\nu} B_{GG'}(k)] c^G_{k\nu} = 0 \]

**Reminder:** the eigenvectors \( x \), solving for each problem \( P \), are vectors of coefficients \( c^G_{k\nu} \) expressing orbital wave functions \( \phi_{k,\nu}(r) \) as a linear combination of basis wave functions \( \psi_G(k, r) \)

**Observations:**

1. eigenproblems across \( k \) index have different size and we consider them independent from each other (for the moment)
2. eigenvectors are seemingly uncorrelated across iterations \( i \)
3. \( k = 1:10-100 \); \( i = 1:20-50 \)

**Actions:**

- study the evolutions of the angles b/w eigenvectors of successive iterations
- develop a method that establishes systematically a one-to-one correspondence b/w eigenvectors
- collect data on eigenvectors deviation angles
Eigenvectors evolution
fixed $\lambda$

Evolution of subspace angle for eigenvector 7 and all 15 k-points

Iterations (2 $\rightarrow$ 27)

Angle b/w eigenvectors of adjacent iterations

Fe$_5 \ell$
Eigenvectors evolution
fixed $k$

**Evolution of subspace angle for eigenvectors of $k$-point 11 and all eigs**

Angle b/w eigenvectors of adjacent iterations

Iterations (2 $\rightarrow$ 27)

Fe$_5$ℓ
Angles decrease monotonically with some oscillation

Majority of angles are small after the first few iterations

“Universal” behavior

This property could not be extracted a-priori from the mathematical model. It results from a systematic analysis of the simulation. It is inferred from the numerical properties of the simulation. It is the consequence of applying the broadly defined concept of “reverse engineering” to the end product of a computational process.
Hamiltonian entries
differential variation

$\text{CaFe}_2\text{As}_2$

$|A^{(4)} - A^{(3)}| / \delta^{(3)} > 0.10$ – Iteration 4
The variation of most of the entries is concentrated below values one order of magnitude smaller than the maximal variation.

There is no apparent evolution in the amount of matrix entries that varies.

Generally speaking the variations are one or more order of magnitude smaller than the matrix entries.
Correlation and its exploitation

- There is a strong correlation between eigenvectors of successive eigenproblems $x^{(i-1)}$ and $x^{(i)}$
- Hamiltonian entries reveal patterns in agreement with the eigenvector correlation
- This “reverse simulation” approach validates the importance of looking at the set of eigenproblems as a sequence $\{P^{(i)}\}$

**Algorithm ⇔ Sim**

The stage is favorable to an iterative solver where the eigenvectors of $P^{(i-1)}$ are fed to the solve $P^{(i)}$

**Two stage investigation:**

1. Can we speed-up iterative solvers (Implicit Restarted Arnoldi, Krylov-Schur, Subspace Iteration, etc.)?
2. Are iterative solvers for dense problems a possible alternative to direct methods?
Speeding-up the algorithm
A case study: CaFe$_2$As$_2$

**Molecule:** an example of high-T superconducting material
**Matrix size:** 2628
**Num of fixed-point iterations:** 32
**Num of k-points:** 27

**B ill-conditioned**

B is in general almost singular. Ex:
\[
\text{size}(A) = 50 \rightarrow \kappa(A) \approx 10^4 \quad \text{size}(A) = 500 \rightarrow \kappa(A) \approx 10^7
\]

We used the standard form for the problem
\[
Ax = \lambda Bx \quad \rightarrow \quad A'y = \lambda y \quad \text{with} \quad A' = L^{-1}AL^{-T} \quad \text{and} \quad y = L^T x
\]

**TOL:**
\[
\frac{\|Ax - \lambda x\|}{\|\lambda x\|} = 1e - 07
\]

\[Nev = \text{Number of searched eigenpairs}\]
\[Nvc = \text{size of subspace}\]
\[Nvc \geq \max[2 \times Nev, Nev + 15]\]

for random vectors
They can accept, as initial guess, only **one vector** (we chose the dominant one)

<table>
<thead>
<tr>
<th>Iter cycle</th>
<th>Nev=4, Nvc=20 IRAM random</th>
<th>Nev=4, Nvc=20 IRAM feed-in</th>
<th>Nev=10, Nvc=25 (K-S) random</th>
<th>Nev=10, Nvc=25 (K-S) feed-in</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>5.37 s</td>
<td>3.37 s</td>
<td>3.37 s</td>
<td>2.54 s</td>
</tr>
<tr>
<td>20</td>
<td>5.37 s</td>
<td>3.34 s</td>
<td>3.39 s</td>
<td>2.51 s</td>
</tr>
<tr>
<td>30</td>
<td>5.37 s</td>
<td>3.36 s</td>
<td>3.39 s</td>
<td>2.55 s</td>
</tr>
</tbody>
</table>

**Considerations:**

1. between 25% and 35% speed-up
2. Results depend on the Nev (speed up decrease as Nev increase)
3. Convergence in IRAM-random seems to be influenced by the clustering of eigenvalues
Some preliminary results

1 Feeding to traditional solvers:
   - All methods experience between 25% and 35% speed-up
   - Due to the different number of eigenvectors that can be “seeded”, the speed up for IRAM and K-S is limited to few sought after eigenpairs
   - Subspace Iteration does not have limits but it is still not competitive

2 Feeding to block solvers (on-going work):
   - Block iterative methods allow the possibility of “seeding” the eigensolver with many eigenvectors.
   - Block solvers experience the same speed-up found for standard solver.
   - The increased “feeding” capacity allows speed-ups even when a substantial fraction of the spectrum is sought after
   - Evolution of the sequence implies an increase in speed-ups towards the end of the sequence
The quest for competitiveness

State of the Art

- Iterative solvers are generally considered “not competitive” when used to solve large (> 1,000) dense eigenproblems for a consistent fraction of their eigenpairs.
- While this is a true statement when the solver is used on a single core, it may depend on a variety of factors when parallel architectures are factored in.

To address this issue we embarked in a preliminary investigation involving eigenproblems from material science and a simple example of parallel architecture.

Structure of the Investigation

Lib. – Traditional libraries (LAPACK, BLAS, ARPACK) a modern libraries (MAGMA, CUBLAS)

Arch. – A small system equipped with multi-core processors and an hybrid configuration embedding multi-core with one GPU

Eig. – Two large and dense generalized eigenproblems extracted from materials science applications (MD and DFT)
<table>
<thead>
<tr>
<th>Eigen solver stage</th>
<th>Operation</th>
<th>Traditional libraries</th>
<th>Modern libraries</th>
</tr>
</thead>
</table>
| Reduction to STD  | $B = U^T U \rightarrow U$  
$C := U^{-T} A U^{-1}$ | LAPACK  
LAPACK/BLAS | CUBLAS/MAGMA  
CUBLAS/MAGMA |
| Direct Method ($MR^3$) | $Q^T C Q = T$  
$T Z = Z \Lambda \rightarrow T, Z$  
$Y := Q Z$ | LAPACK  
LAPACK($MR^3$)  
LAPACK | CUBLAS/MAGMA  
CUBLAS/MAGMA  
LAPACK($MR^3$)  
LAPACK |
| Krylov Subspace Method (IRAM) | $z_{k+1} := C w_k$  
$z_{k+1} \rightarrow w_{k+1}$  
$T_m, V_m \rightarrow \Lambda, Y$ | BLAS  
ARPACK  
ARPACK | CUBLAS/MAGMA  
ARPACK  
ARPACK |
| Back-transform | $X := U^{-1} Y$ | BLAS | CUBLAS/MAGMA |
### Hardware – 2 Intel Xeon Quadcore E5520 (8 cores at 2.27 GHz) with 24 Gb of memory connected to an NVIDIA Tesla C2050 (Fermi) GPU (480 cores at 1.15 GHz) with 3 Gb of memory

<table>
<thead>
<tr>
<th>Operation</th>
<th>DFT Example ((n=17,243), s=448)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Trad-Dir</td>
</tr>
<tr>
<td>(B = U^T U \rightarrow U)</td>
<td>36.42</td>
</tr>
<tr>
<td>(C := U^{-T} AU^{-1})</td>
<td>140.35</td>
</tr>
<tr>
<td>(Q^T C Q = T)</td>
<td>342.01</td>
</tr>
<tr>
<td>(T Z = Z \Lambda \rightarrow T, Z)</td>
<td>4.57</td>
</tr>
<tr>
<td>(Y := Q Z)</td>
<td>7.81</td>
</tr>
<tr>
<td>(z_{k+1} := C w_k)</td>
<td>–</td>
</tr>
<tr>
<td>(z_{k+1} \rightarrow w_{k+1})</td>
<td>–</td>
</tr>
<tr>
<td>(T_m, V_m \rightarrow \Lambda, Y)</td>
<td>–</td>
</tr>
<tr>
<td>(X := U^{-1} Y)</td>
<td>2.41</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td>533.57</td>
</tr>
</tbody>
</table>
Direct vs Iterative
Execution time vs $s$ on hybrid system (GPU)

**TD:** Direct Solver (MR$^3$)

**KE:** Iterative Solver (IRAM)

---

**Experiment 1 (MD)**

<table>
<thead>
<tr>
<th>Percentage of computed eigenvalues, s/n x 100</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution time (in seconds)</td>
</tr>
<tr>
<td>TD</td>
</tr>
<tr>
<td>0.1</td>
</tr>
<tr>
<td>25</td>
</tr>
</tbody>
</table>

**Experiment 2 (DFT)**

<table>
<thead>
<tr>
<th>Percentage of computed eigenvalues, s/n x 100</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution time (in seconds)</td>
</tr>
<tr>
<td>TD</td>
</tr>
<tr>
<td>0.1</td>
</tr>
<tr>
<td>500</td>
</tr>
</tbody>
</table>

Edoardo Di Napoli (AICES/JSC)
Conclusions

Algorithm $\leftrightarrow$ Sim

- Feeding eigenvectors of $P^{(i-1)}$ speed-ups the iterative solver for $P^{(i)}$.
- Iterative methods, specifically IRAM, solving for dense medium/large eigenproblems can be competitive if a portion of eigenvalues equal or below $\sim 5\%$ (depending on the size and properties of the eigenproblem) is sought after.
- On multi-threaded architectures, when used on dense eigenproblems, iterative methods could be competitive already for medium size systems.
- When GPUs are factored in, iterative methods may be competitive even for higher portion of the spectrum.
- The algorithmic structure of FLAPW need to be re-thought to take into consideration our results $\rightarrow$ “reverse simulation” can substantially influence the computational paradigm of an application.
On going and future work

- Planning the construction of tailored iterative block solvers that have an improved convergence as well as a more effective filtering mechanism featuring collab. with Y. Zhou & Y. Saad

- Extend the study on multi-threaded architecture to larger platforms. Ultimately that’s where the improved block solvers will be tested on open to external collaborations

- Analysis on the structure of the entries of $A$ and $B$ across adjacent iteration seems to suggest an exploitation of low-rank updates for sequences of eigenproblems $(\text{ALGORITHM} \iff \text{SIM})$ a possible UROP or master project

- An ongoing study on mixed overlap matrices $\tilde{B}$ may show that there is no need to re-calculate all the basis wave functions at each new iteration cycle $(\text{MATH} \iff \text{SIM})$ on going collab. with PGI-1

- A complete study on convergence is still missing. A great deal of increase in efficiency of the simulation can be reached if the convergence process is optimized $(\text{MATH} \iff \text{SIM})$ a possible PhD thesis

- Correlation may open the road to a reduction of eigenproblem complexity through a sequence of reductions to tridiagonal form a possible UROP or master project
Acknowledgments

Thanks to...

- the audience for attending and sitting patiently
- VolkswagenStiftung for supporting this research through the fellowship "Computational Sciences"
- The AICES (RWTH Aachen University) and JSC (Forschungszentrum Jülich) for hosting
- Prof. P. Bientinesi and his group at AICES for continuous collaborative support and encouragement
- Prof. S. Blügel and his group for their feedback and contributions
References
