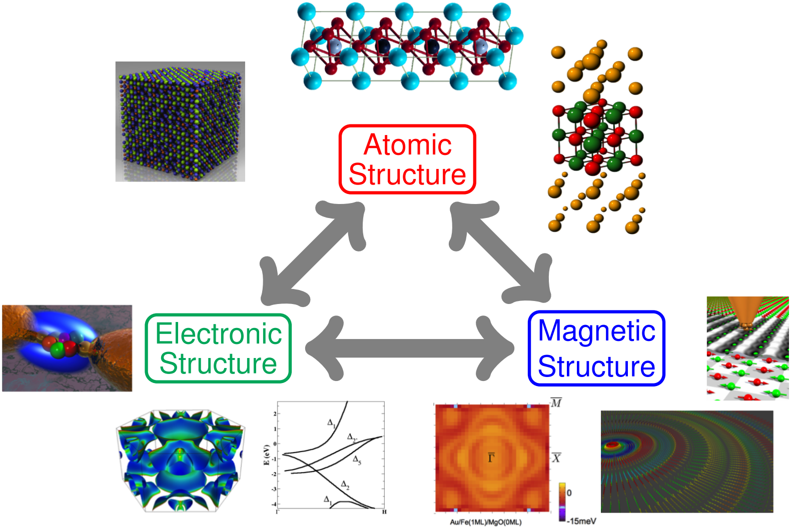
Mitglied der Helmholtz-Gemeinschaft

JÜLICH FORSCHUNGSZENTRUM

Enabling large scale LAPW DFT calculations by a scalable iterative eigensolver

CSE15, Salt Lake City. March 17th | **E. Di Napoli**, D. Wortmann, and M. Berljafa

Typical Applications



Outline

The FLAPW method

Sequences of correlated eigenproblems

The algorithm: Chebyshev Accelerated Subspace Iteration (CHASE)

CHASE parallelization and numerical tests

Outline

The FLAPW method

Sequences of correlated eigenproblems

The algorithm: Chebyshev Accelerated Subspace Iteration (CHASE)

CHASE parallelization and numerical tests

Density Functional Theory (DFT)

1. $\Phi(x_1; s_1, x_2; s_2, \ldots, x_n; s_n) \Longrightarrow \Lambda_{i,a}\phi_a(x_i; s_i)$
2. **density of states** $n(\mathbf{r}) = \sum_a f_a \left|\phi_a(\mathbf{r})\right|^2$
3. In the Schrödinger equation the exact Coulomb interaction is substituted with an effective potential $V_0(\mathbf{r}) = V_{\mathrm{I}}(\mathbf{r}) + V_{\mathrm{H}}(\mathbf{r}) + V_{\mathrm{xc}}(\mathbf{r})$

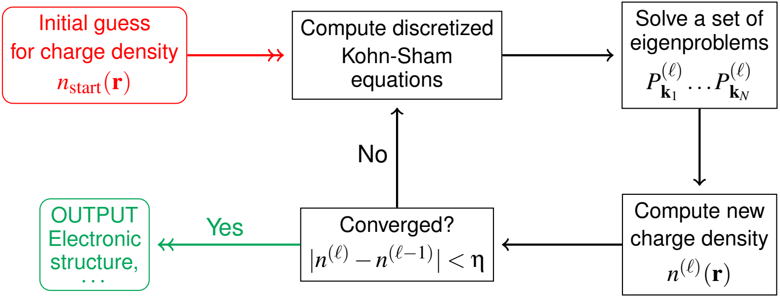
Hohenberg-Kohn theorem

- $\exists$ one-to-one correspondence $n(\mathbf{r}) \leftrightarrow V_0(\mathbf{r}) \quad \Longrightarrow \quad V_0(\mathbf{r}) = V_0(\mathbf{r})[n]$
- $\exists!$ a functional $E[n]$: $\quad E_0 = \min_n E[n]$

The high-dimensional Schrödinger equation translates into a set of coupled non-linear low-dimensional self-consistent Kohn-Sham (KS) equation

$$\forall\, a \quad \text{solve} \quad \hat{H}_{\mathrm{KS}}\phi_a(\mathbf{r}) = \left(-\frac{\hbar^2}{2m}\nabla^2 + V_0(\mathbf{r})\right)\phi_a(\mathbf{r}) = \varepsilon_a\phi_a(\mathbf{r})$$

DFT self-consistent field cycle



Zoo of methods

LDA
GGA
LDA + U
Hybrid functionals
GW-approximation

Plane waves
Localized basis set
Real space grids
Green functions

$$\left(-\frac{\hbar^2}{2m}\nabla^2 + V_0(\mathbf{r})\right)\phi_a(\mathbf{r}) = \varepsilon_a \phi_a(\mathbf{r})$$

Finite differences
Non-relaticistic eqs.
Scalar-relativistic approx,
Spin-orbit coupling
Dirac equation

All-electron
Pseudo-potential
Shape approximations
Full-potential
Spin polarized calculations

Introduction to FLAPW

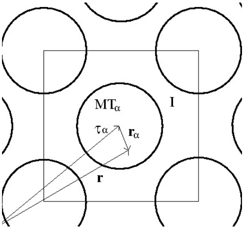
LAPW basis set

$$\psi_{\mathbf{k},\nu}(\mathbf{r}) = \sum_{|\mathbf{G}+\mathbf{k}|\leq \mathbf{G}_{max}} c^{\mathbf{G}}_{\mathbf{k},\nu} \phi_{\mathbf{G}}(\mathbf{k},\mathbf{r})$$

k Bloch vector

ν band index

$$\phi_{\mathbf{G}}(\mathbf{k},\mathbf{r}) = \begin{cases} e^{i(\mathbf{k}+\mathbf{G})\mathbf{r}} & \text{Interstitial (I)} \\ \sum_{\ell,m} \left[a^{\alpha,\mathbf{G}}_{\ell m}(\mathbf{k}) u^{\alpha}_{\ell}(r) + b^{\alpha,\mathbf{G}}_{\ell m}(\mathbf{k}) \dot{u}^{\alpha}_{\ell}(r) \right] Y_{\ell m}(\hat{\mathbf{r}}_{\alpha}) & \text{Muffin Tin} \end{cases}$$



boundary conditions

Continuity of wavefunction and its derivative at MT boundary

$\Downarrow$

$a^{\alpha,\mathbf{G}}_{\ell m}(\mathbf{k})$ and $b^{\alpha,\mathbf{G}}_{\ell m}(\mathbf{k})$

Where does the CPU time go?

H and S	Eigensolver	Charge	CPU time	PE
50 %	13 %	33%	28 min.	1
27 %	20 %	44 %	36 min.	12
33 %	50 %	17 %	10 min.	30
23 %	61 %	11 %	12 min.	40

Where does the CPU time go?

H and S	Eigensolver	Charge	CPU time	PE
50 %	13 %	33%	28 min.	1
27 %	20 %	44 %	36 min.	12
33 %	50 %	17 %	10 min.	30
23 %	61 %	11 %	12 min.	40

Solving the generalized eigenvalue problem

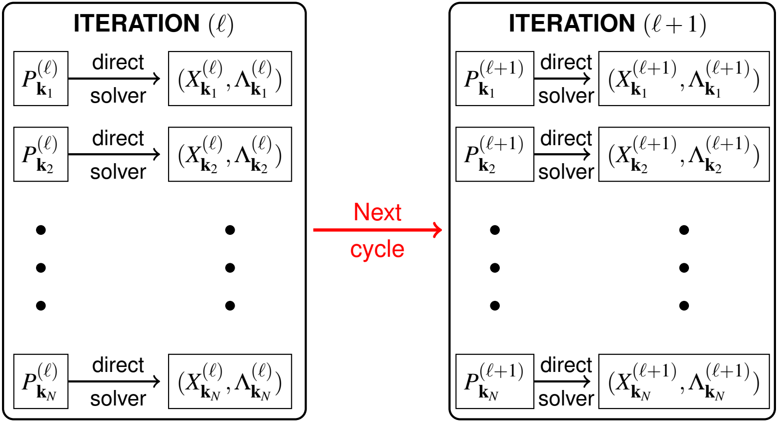
1. every $P_{\mathbf{k}}^{(\ell)}$: $A_{\mathbf{k}}^{(\ell)} c_{\mathbf{k}} = B_{\mathbf{k}}^{(\ell)} \lambda c_{\mathbf{k}}$ is a generalized eigenvalue problem;
2. A and B are **DENSE** and hermitian (B is positive definite);
3. required: lower $2 \div 10$ % of eigenpairs;
4. momentum vector index: $\mathbf{k} = 1 : 10 \div 100$;
5. iteration cycle index: $\ell = 1 : 20 \div 50$.

Outline

- The FLAPW method
- Sequences of correlated eigenproblems
- The algorithm: Chebyshev Accelerated Subspace Iteration (CHASE)
- CHASE parallelization and numerical tests

Sequences of Eigenproblems

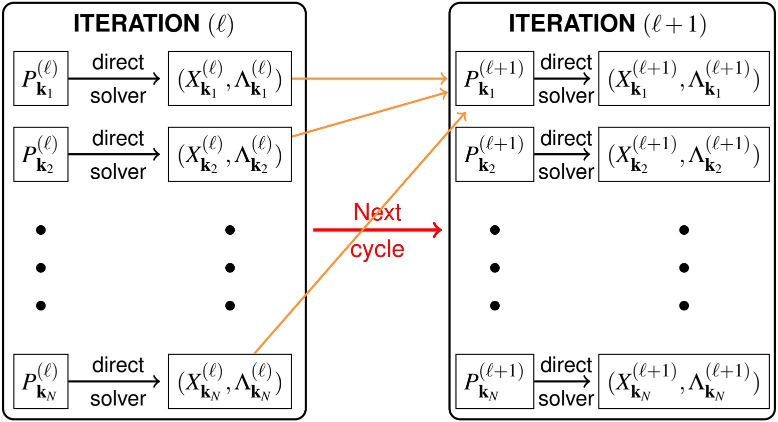
Adjacent iteration cycles



$X \equiv \{x_1, \ldots, x_n\}$ $\Lambda \equiv \mathrm{diag}(\lambda_1, \ldots, \lambda_n)$

Sequences of Eigenproblems

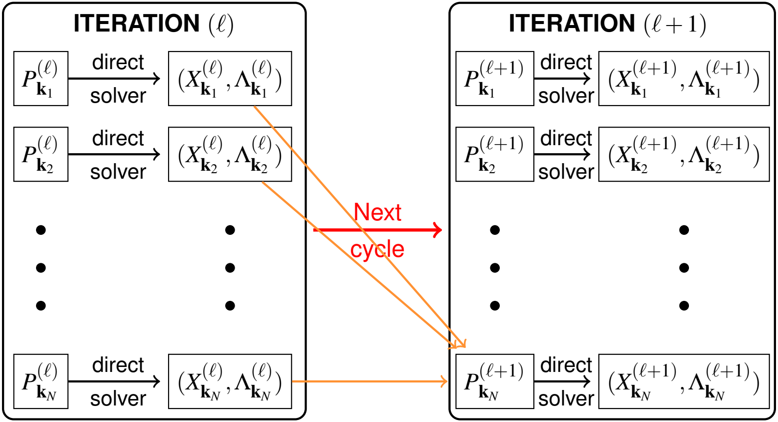
Adjacent iteration cycles



$X \equiv \{x_1, \ldots, x_n\}$ $\qquad$ $\Lambda \equiv \mathrm{diag}(\lambda_1, \ldots, \lambda_n)$

Sequences of Eigenproblems

Adjacent iteration cycles

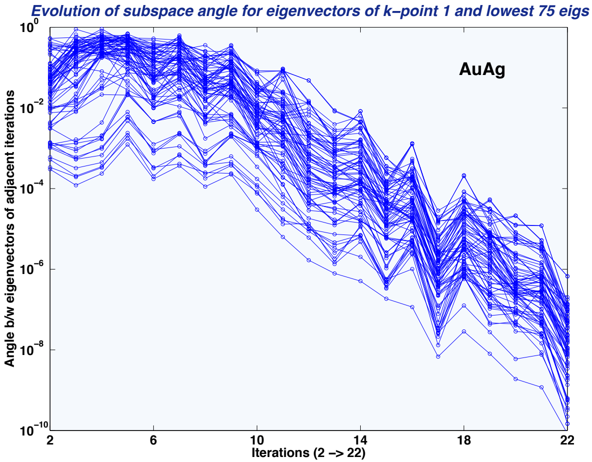


$X \equiv \{x_1, \dots, x_n\}$ $\qquad$ $\Lambda \equiv \mathrm{diag}(\lambda_1, \dots, \lambda_n)$

Angles evolution

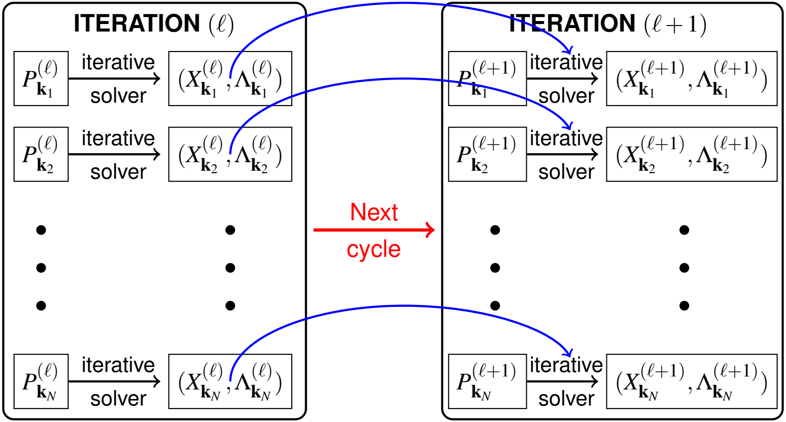
An example

Example: a metallic compound at fixed $\mathbf{k}$



An alternative solving strategy

Adjacent cycles



$X \equiv \{x_1, \ldots, x_n\}$ $\Lambda \equiv \mathrm{diag}(\lambda_1, \ldots, \lambda_n)$

Outline

The FLAPW method

Sequences of correlated eigenproblems

The algorithm: Chebyshev Accelerated Subspace Iteration (CHASE)

CHASE parallelization and numerical tests

Chebyshev Filtered Subspace Iteration method

Properties and algorithm evolution

Iterative solver musts

- input: the full set of multiple starting vectors $Z_0 \equiv X^{(\ell-1)}_{\mathbf{k}_i}(:, 1:\mathtt{NEV})$;
- needed: it can efficiently use dense linear algebra kernels (i.e. `xGEMM`);
- needed: it avoids stalling when facing small clusters of eigenvalues;

Chebyshev Subspace Iteration

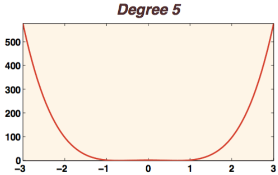
- Firstly introduced in [Rutishauser 1969]
- A version (called CheFSI) tailored to electronic structure computation in [Zhou, Saad, Tiago and Chelikowski 2006] for sparse eigenvalue problems.
- Our ChASE : 1) is tailored for dense eigenproblem sequences, 2) introduces a locking mechanism, 3) contains a refining inner loop, and 4) optimizes the polynomial degree.

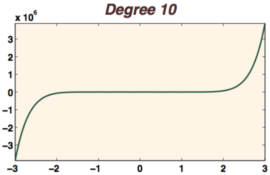
The core of the algorithm: Chebyshev filter

Chebyshev polynomials

A generic vector $v = \sum_{i=1}^{n} s_i x_i$ is very quickly aligned in the direction of the eigenvector corresponding to the extremal eigenvalue λ_1

$$
\begin{aligned}
v^m = p_m(A)v &= \sum_{i=1}^{n} s_i p_m(A) x_i = \sum_{i=1}^{n} s_i p_m(\lambda_i) x_i \\
&= s_1 x_1 + \sum_{i=2}^{n} s_i \frac{C_m(\frac{\lambda_i - c}{e})}{C_m(\frac{\lambda_1 - c}{e})} x_i \sim \boxed{s_1 x_1}
\end{aligned}
$$





The core of the algorithm: Chebyshev filter

In practice

Three-terms recurrence relation

$$C_{m+1}(t) = 2xC_m(t) - C_{m-1}(t); \qquad m \in \mathbb{N}, \quad C_0(t) = 1, \quad C_1(t) = x$$

$$Z_m \doteq p_m(\tilde{H})\, Z_0 \quad \text{with} \quad \tilde{H} = H - cI_n$$

FOR: $i = 1 \rightarrow \text{DEG} - 1$

$$Z_{i+1} \leftarrow 2\frac{\sigma_{i+1}}{e}\, \tilde{H} \times Z_i - \sigma_{i+1}\sigma_i Z_{i-1}$$

`xGEMM`

END FOR.

Polynomial degree optimization

Convergence ratio and residuals

Definition

The **convergence ratio** for the eigenvector x_i corresponding to eigenvalue $\lambda_i \notin [\alpha, \beta]$ is defined as

$$\tau(\lambda_i) = |\rho_i|^{-1} = \min_{\pm} \left| \frac{\lambda_i - c}{e} \pm \sqrt{\left(\frac{\lambda_i - c}{e}\right)^2 - 1} \right|.$$

The further away λ_i is from the interval $[\alpha, \beta]$ the smaller is $|\rho_i|^{-1}$ and the faster the convergence to x_i is.

For a set of input vectors $V = \{v_1, v_2, \ldots, v_{\text{nev}}\}$

Residuals are a function of m and $|\rho|$

$$\text{Res}(v_i^m) \sim \quad \text{Const} \times \left| \frac{1}{\rho_i} \right|^m \qquad 1 \leq i \leq k.$$

$$\text{Res}(v_i^{m+m_0}) \approx \quad \text{Res}(v_i^{m_0}) \left| \frac{1}{\rho_i} \right|^m \quad \Rightarrow \quad m_i \geq \ln \left| \frac{\text{TOL}}{\text{Res}(v_i^{m_0})} \right| / \ln \|\rho_i\|$$

ChASE pseudocode (optimized)

1. *Chebyshev filter.* Initial filter $W \longleftarrow Z_0$. with DEG$= m_0$.
2. Re-orthogonalize $W = QR$ & compute the Rayleigh quotient $G = Q^\dagger HQ$.
3. Solve the reduced problem $GY = Y\Lambda$ and compute the approximate Ritz pairs $(\Lambda, W \leftarrow QY)$ and store their residuals $\mathrm{Res}(w_i)$.

ChASE pseudocode (optimized)

1. *Chebyshev filter.* Initial filter $W \longleftarrow Z_0$. with DEG$= m_0$.
2. Re-orthogonalize $W = QR$ & compute the Rayleigh quotient $G = Q^\dagger HQ$.
3. Solve the reduced problem $GY = Y\Lambda$ and compute the approximate Ritz pairs $(\Lambda, W \leftarrow QY)$ and store their residuals $\mathrm{Res}(w_i)$.

REPEAT UNTIL CONVERGENCE:

4. *Optimizer.* Compute the polynomial degrees $m_i \geq \ln\left|\frac{\mathsf{TOL}}{\mathrm{Res}(w_i)}\right| / \ln\|\rho_i\|$.
5. *Chebyshev filter.* Filter $W \longleftarrow Z_0$ with DEG$= m_i$.
6. Re-orthogonalize $W = QR$ & compute the Rayleigh quotient $G = Q^\dagger HQ$.
7. Solve the reduced problem $GY = Y\Lambda$ and compute the approximate Ritz pairs $(\Lambda, W \leftarrow QY)$.
8. *lock* the converged vectors.
9. Store the residuals $\mathrm{Res}(w_i)$ of the unconverged vectors.

END REPEAT

Outline

The FLAPW method

Sequences of correlated eigenproblems

The algorithm: Chebyshev Accelerated Subspace Iteration (CHASE)

CHASE parallelization and numerical tests

Experimental tests setup

C++ implementation of ChASE

- EleChASE – Elemental (MPI) parallelization for distributed memory
- OMPChASE – OpenMP 4.0 parallelization for shared memory
- CUChASE – CUDA parallelization for GPUs
- Interface C++/Fortran so as to call ChASE from FLEUR

Tests were performed on the JUROPA and the RWTH RZ cluster.

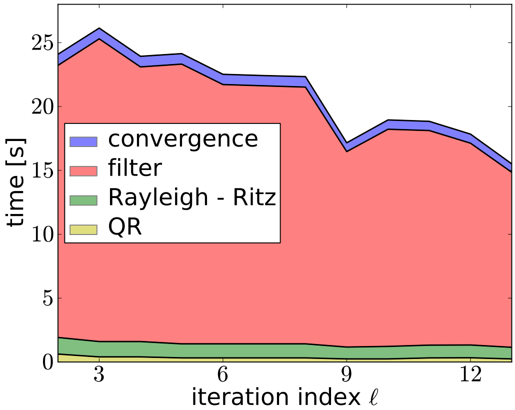
- 2 Intel Xeon 5570 (Nehalem-EP) 4d-core processors/node;
- 2 Intel Xeon E5 2670 (Sandy-Bridge) 8-core processors/node;
- NVIDIA K20m
- Xeon Phi

Matrix sizes: 2,600 $\div$ 29,500.

ChASE time profile

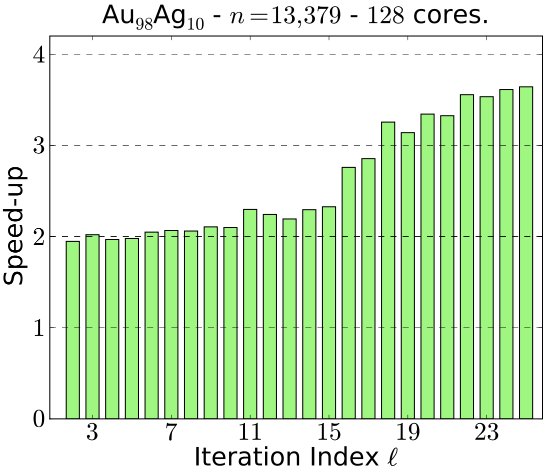
As a function of iteration cycles

Time spent in each stage of the algorithm as a function of the iteration index ℓ for a system of size $n = 9,273$.



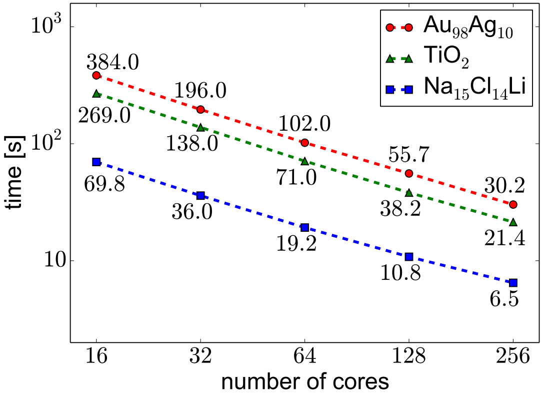
Speed-up

$$\text{Speed-up} = \frac{\text{CPU time (input random vectors)}}{\text{CPU time (input approximate eigenvectors)}}$$



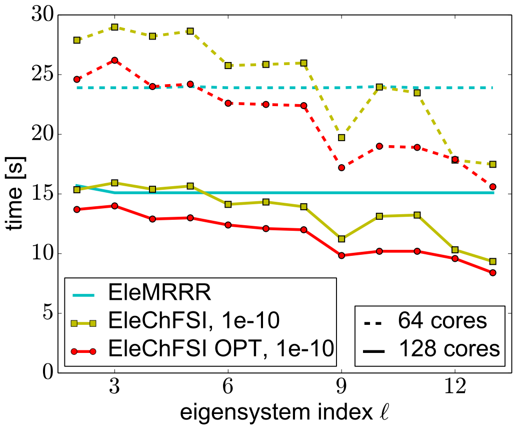
Scalability (MPI implementation)

Strong Scalability (the size of the eigenproblems are kept fixed while the number of cores is progressively increased) for EleChASE over three systems of size $n = 13,379 - 12,455 - 9,273$ respectively.



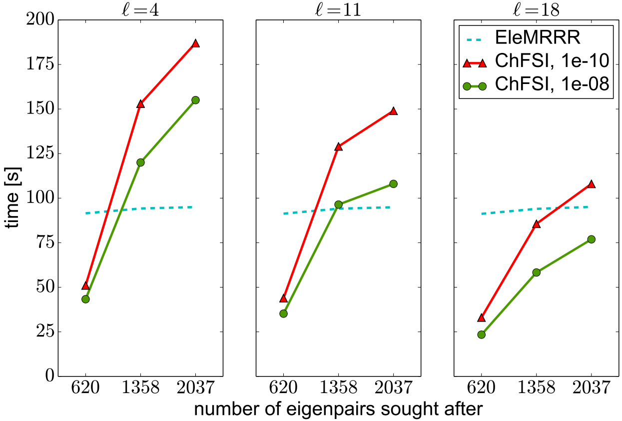
EleChASE versus direct solvers (parallel MRRR)

For the size of eigenproblems here tested the ScaLAPAK implementation of BXINV or MRRR is on par of worse than EleMRRR. For this reason a direct comparison with ScaLAPACK is not included.



EleChASE versus direct solvers (parallel MRRR)

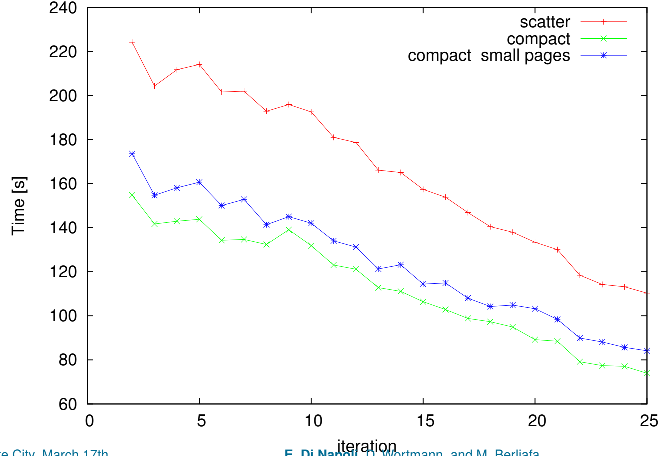
For the size of eigenproblems here tested the ScaLAPAK implementation of BXINV or MRRR is on par of worse than EleMRRR. For this reason a direct comparison with ScaLAPACK is not included.



Offloading to Xeon Phis and GPUs

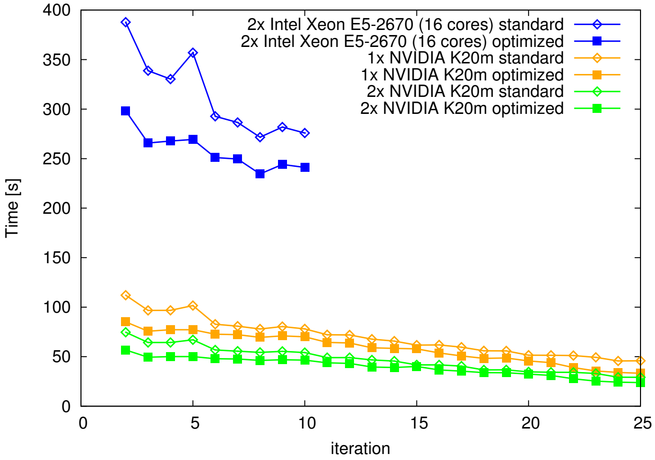
The role of affinity on Xeon Phi

`xGEMM` performs at best at 66% of peak performance (1000 GFlops)



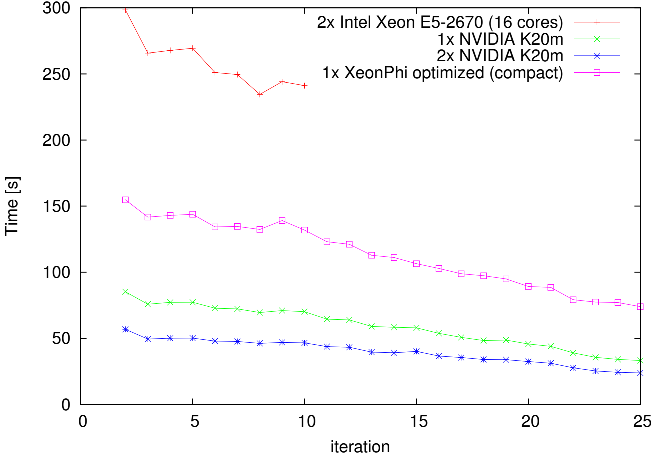
Offloading to Xeon Phis and GPUs

GPUs vs CPUs



Offloading to Xeon Phis and GPUs

multi-cores vs many-cores



Conclusions and future work

Algorithmic strategy

Sequences of "correlated" eigenproblems $\Rightarrow$ Tailored algorithms

- Exploiting the correlation of the eigenproblem sequence to speedup the solution of each $P^{(\ell)}$ is a successful strategy;
- Combining iterative methods with kernels for dense linear algebra can pay off.
- The parallelization shows great potential for scalability and parallel efficiency;
- Uncovering information can lead to further algorithmic optimizations;

ONGOING AND FUTURE WORK

1. Exploring hybrid parallelizations of the code.
2. Implement in FLEUR a mixed direct-iterative solver;

References

1 M.Berliafa, D. Wortmann and EDN
An Optimized and Scalable Eigensolver for Sequences of Eigenvalue Problems
Concurrency and Computation: Practice and Experience, 27 (2015) 905

2 EDN, and M. Berljafa
A Parallel and Scalable Iterative Solver for Sequences of Dense Eigenproblems Arising in FLAPW
Parallel Processing and Applied Mathematics, Lecture Notes in Computer Science, Vol 8385. [arXiv:1305.5120]

3 EDN, and M. Berljafa
Block Iterative Eigensolvers for Sequences of Correlated Eigenvalue Problems
Comp. Phys. Comm. 184 (2013), pp. 2478-2488, [arXiv:1206.3768].

4 EDN, P. Bientinesi, and S. Blügel,
Correlation in sequences of generalized eigenproblems arising in Density Functional Theory,
Comp. Phys. Comm. 183 (2012), pp. 1674-1682, [arXiv:1108.2594].