A 3D-Parallel Interior Eigenvalue Solver

Jonas Thies\textsuperscript{1}  Martin Galgon\textsuperscript{2}  Lukas Krämer\textsuperscript{2}  Andreas Pieper\textsuperscript{3}

**ESSEX** - Equipping Sparse Solvers for the EXascale

SPPEXA workshop, Heidelberg Dec. 1-3, 2014

\textsuperscript{1}German Aerospace Center (DLR)  
Simulation and Software Technology  
jonas.thies@dlr.de

\textsuperscript{2}University of Wuppertal  
Applied Computer Science

\textsuperscript{3}University of Greifswald  
Theoretical Physics
Outline

Graphene Simulation

The FEAST Eigensolver

Linear Solver: CGMN

Parallel CGMN

Experiments

Summary and Outlook
Graphene Simulation
Graphene

Physical space: carbon atoms in 2D hexagonal mesh

Fourier space (‘reciprocal mesh’)

Tight-binding Hamiltonian

\[ H = \sum_i V_i c_i^\dagger c_i - t \sum_{\langle ij \rangle} (c_j^\dagger c_j + c_j^\dagger c_i) \]
Graphene (2)

- Analytical solution for infinite Graphene sheet
- Dirac cones: graphene between conductor and semi-conductor
Graphene modeling

- disorder
- long range stencil
- bi-layer
- gate-defined quantum dots
- spin-orbit coupling
- ...

Long range Hamiltonian:

\[ H = \sum_i V_i c_i^\dagger c_i - t \sum_{\langle ij \rangle} (c_i^\dagger c_j + c_j^\dagger c_i) - t' \sum_{\langle\langle ij \rangle\rangle} (c_i^\dagger c_j + c_j^\dagger c_i) - t'' \sum_{\langle\langle\langle ij \rangle\rangle\rangle} (c_i^\dagger c_j + c_j^\dagger c_i) \]
The FEAST Eigensolver
Graphene: eigenmodes of interest

- need many eigenvalues, $O(1000)$
- in the interior of the spectrum
- tight clusters

- eigenvalue density increases $\sim L$ for $L \times L$ graphene sheet
- rich spectrum of non-smooth modes
FEAST eigensolver (Polizzi ’09)

- Imaginary part
- Real part
- Search space

○ 100 smallest eigenvalues
■ Integration points
FEAST algorithm for $AX = BX\lambda$ (A, B symmetric)

Input: $I_\lambda := [\underline{\lambda}, \overline{\lambda}]$, an estimate $\tilde{m}$ of the number of eigenvalues in $I_\lambda$.
Output $\hat{m} \leq \tilde{m}$ eigenpairs with eigenvalue in $I_\lambda$.

Perform:

1. Choose $Y \in \mathbb{C}^{n \times \tilde{m}}$ of full rank and compute
   \[ U := \frac{1}{2\pi i} \int_C (zB - A)^{-1}BdzY, \]

2. Form $A_U := U^*AU$, $B_U := U^*BU$,

3. Solve size-$\tilde{m}$ eigenproblem $A_U\tilde{W} = B_U\tilde{W}\tilde{\Lambda}$,

4. Compute $(\tilde{\Lambda}, \tilde{X} := U \cdot \tilde{W})$,

5. If no convergence: go to Step 1 with $Y := \tilde{X}$. 
Parallelization of FEAST

Several levels:

I interval sectioning
II distribute shifts
III distribute right-hand sides
IV parallel linear solver (MPI+X)
V based on SIMD optimized kernels (spMVM, BLAS etc)
Linear Solver: CGMN
Linear systems for FEAST/graphene

Tough:

- very large ($N = 10^8 - 10^{14}$)
- complex symmetric and completely indefinite
- random numbers on and around the diagonal
- spectrum essentially continuous
- shifts get very close to the spectrum

But also nice in some ways:

- 2D mesh, very sparse ($\sim 10$ entries/row)
- many RHS/shift (block methods, recycling, ...)
An ancient row projection method

- Björck and Elfving, 1979
- CG on the ‘minimum norm’ problem, $AA^T x = b$
- preconditioned by SSOR
- efficient row-wise formulation
- extremely robust: $A$ may be singular, non-square etc.
- row scaling alleviates issue of ‘squared condition number’
Kernel operation: KACZ sweep

Interpretations:
- Kaczmarz algorithm
- SOR(\(\omega\)) on the normal equations \(AA^T x = b\)
- successive projections onto the hyperplanes defined by the rows of \(A\)

In CRS (rptr,val,col):
1: compute \( \text{nrms} = ||a_{i,:}||^2_2 \)
2: for (\(i=0; i<n; i++\)) do
   // compute \( a_{i,:} x - b_i \)
   3: scal = -b[i]
   4: for (\(j=rptr[i]; j<rptr[i+1]; j++\)) do
       scal += val[j]*x[col[j]]
   5: end for
   6: scal /= nrms[i]
   // update x
7: for (\(j=rptr[i]; j<rptr[i+1]; j++\)) do
6: x[cols[j]] -= \(\omega\) * scal * val[j]
9: end for
10: end for
Parallel CGMN
Multi-Coloring (MC) for CGMN

- requires “distance 2” coloring
- software: ColPack
  http://cscapes.cs.purdue.edu/coloringpage/software.htm
Component-Averaged Row Projection (CARP)

- Gordon & Gordon, 2005
- Kaczmarz locally
- write to halo
- exchange and average

Equivalent to Kaczmarz on a superspace of $\mathbb{R}^n$
Hybrid method: MC_CARP-CG

- global MC would require...
  - an extremely scalable coloring method
  - very well-balanced colors
  - many global sync-points (> 20 colors in our examples)

- global CARP would require...
  - huge number of MPI procs
  - increasing amount of ‘interior halo elements’
  - non-trivial implementation on GPU and Xeon Phi
  - increasing number of iterations

Idea: node-local MC with MPI-based CARP between the nodes
Experiments
Experimental setup

- Machine: Intel Xeon “Ivy Bridge”
- 10 cores/socket, 2 sockets/node
- InfiniBand between nodes

Here’s what we do:
- pick some shifts that may occur in FEAST
- handle 8 RHS at once (for good performance)
- conv tol $10^{-12}$
- solve linear systems using CGMN variants
Sequential CGMN for various shifts
Coloring vs. CARP: single socket \((1024^2 \text{ dof})\)
Scaling of Hybrid vs. CARP

Figure: Weak scaling for Graphene, $4096^2$ unknowns per node

Figure: Strong scaling and block speed-up, $8192^2$ unknowns in total
Weak scaling for a 3D benchmark

Synthetic 3D Model of Anderson localization
- uniform 3D grid
- 9-point stencil
- random numbers between -l/2 and l/2 on the diagonal (l=16.5 here)
Summary and Outlook
The (almost) final slide

- Graphene gives nice and challenging test cases for Lin. Alg.
- FEAST requires fast linear solvers for indef. systems
- row projection methods provide the necessary robustness
- algorithm that *calls for* MPI+X parallelization

Future work:
- node-level optimization, GPU and Xeon Phi
- other applications of CGMN: Helmholtz, conv. dom. flow,...
- Multigrid to resolve near kernel problem (?) (cf. recent work by I. Livshits)

**Acknowledgment:** DFG SPPEXA project ESSEX
(Equipping Sparse Solvers for the EXa-scale)
References


