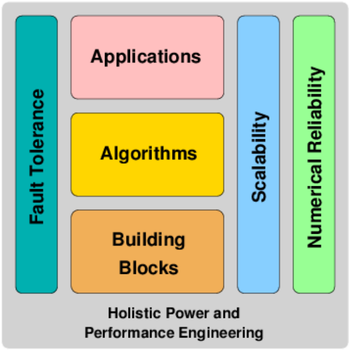
ESSEX II – Annual Report 2016



Gerhard Wellein	Computer Science, University Erlangen
Bruno Lang	Applied Computer Science, University Wuppertal
Achim Basermann	Simulation & SW Technology, German Aerospace
Holger Fehske	Institute for Physics, University Greifswald
Georg Hager	Erlangen Regional Computing Center
Tetsuya Sakurai	Applied Mathematics, University of Tsukuba
Kengo Nakajima	Computer Science, University of Tokyo

Garching, March 20th, 2017

ESSEX-II team

21 (accepted) publications

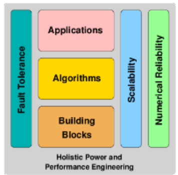
21 Talks

Full project in person meetings:

- January 21 / 22 (Berlin): Start-up Meeting
- September 19 – 22 (Erlangen): Coding week

Continous meetings / visits / video conferences

Joint PMAA 2016 minisymposium and ISC 2017 minisymposium

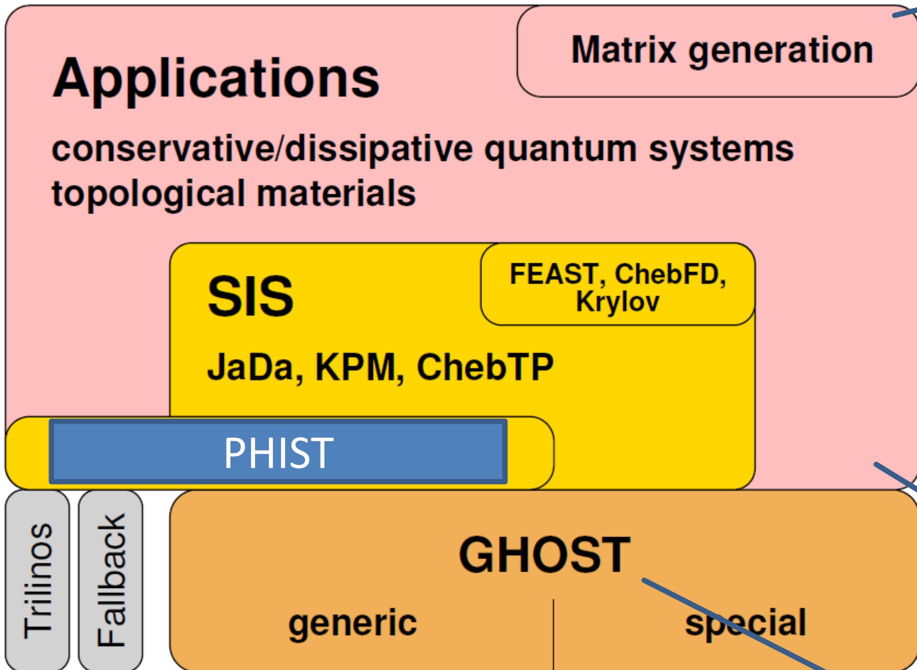


Motivated by quantum physics applications

ESSEX - I		ESSEX - II
Conservative Quantum Systems	→	Dissipative Quantum Systems
$A^+ = A$	→	$A^+ \neq A$
$Ax = \lambda x$	→	$Ax = \lambda Bx$ $A(\lambda)x = 0$
ESSR: Blueprints	→	Interoperable Library

→ Sparse eigenvalue solvers of broad applicability

ESSEX-II: Focus topics in 2016



Scalable matrix generation for a collection of quantum physics problems

Impact of enhanced / reduced accuracy

Implementation & evaluation of various preconditioning strategies

CRAFT library:
Application-level Checkpoint/Restart & Automatic Fault Tolerance (MPI-ULFM)

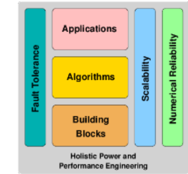
Reference paper
TSMM@GPU kernels
P100/KNL porting / optimization

GHOST: General Hybrid Optimized Sparse Toolkit
PHIST: Pipelined Hybrid Parallel Iterative Solver Toolkit

ESSEX project webpage: http://blogs.fau.de/essex/

Solvers & Hardware efficiency

Hardware efficient (linear) solvers
Kaczmark solver (used in BEAST-C)

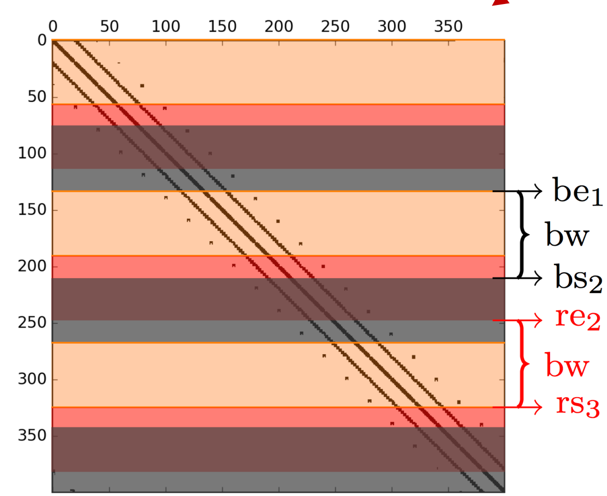


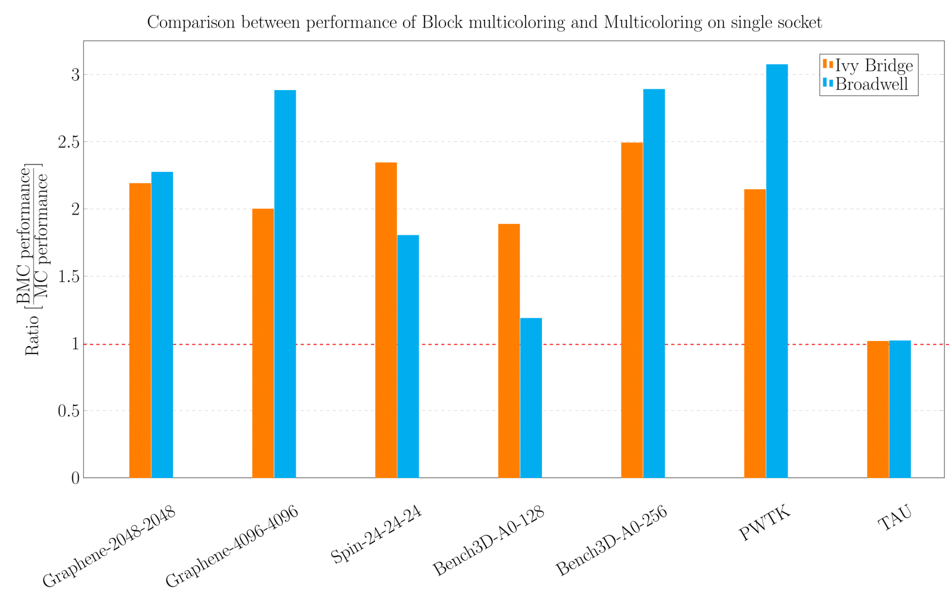
for $row = 0 : nrows$ **do**
  $scale = b[row]$
  $norm = 0$
  for $idx = rowptr[row] : rowptr[row+1]$ **do**
    $scale -= val[idx] * x[col[idx]]$
    $norm += val[idx] * val[idx]$
  end for
  $scale* = omega/norm$
  for $idx = rowptr[row] : rowptr[row+1]$ **do**
    $x[col[idx]] += scale * val[idx]$
  end for
end for

$$x^{k+1} = x^k + \omega * \frac{(b_i - < A_i, x^k >)}{\|A_i\|^2} * A_i^T$$

Parallelization

- Standard: D2-multicoloring (distance: 2) (D2-MC)
- **New: block-multicoloring (BMC)**
  (+ D2-MC, where needed)

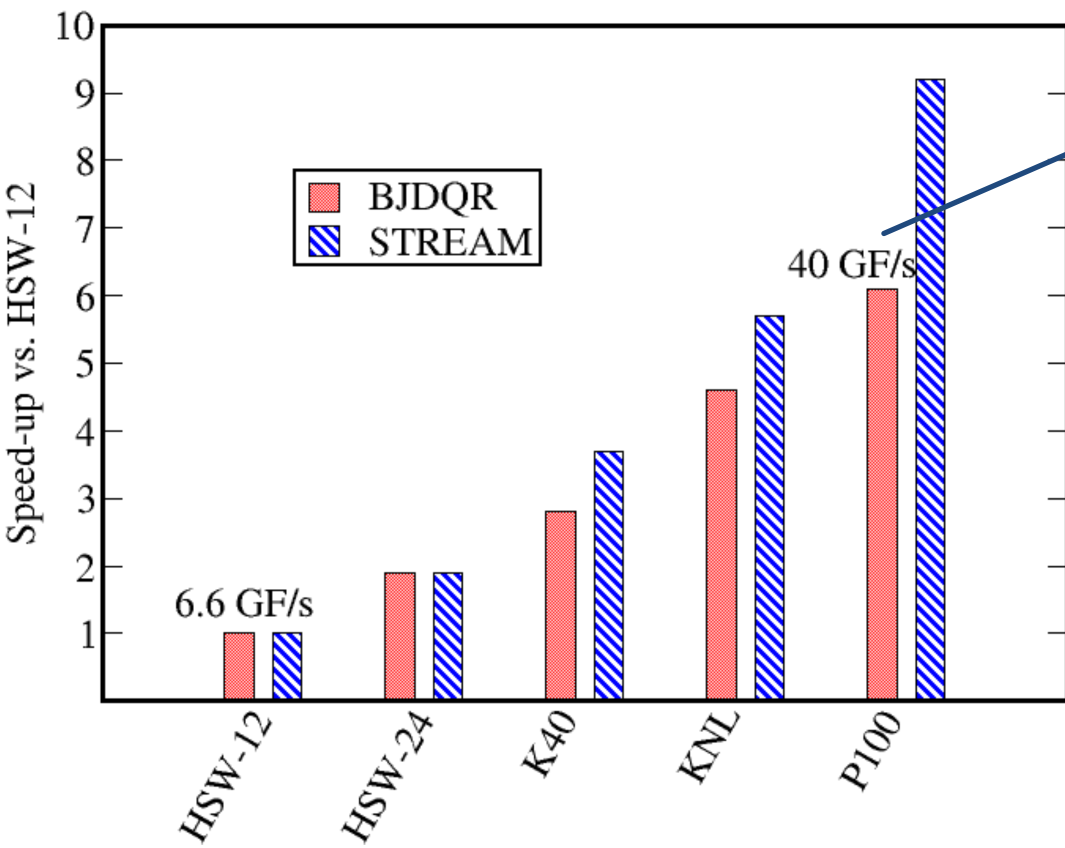




C.L. Alappat, Implementation and Performance Engineering of the Kaczmarz Method for Parallel Systems, Master Thesis

Hardware efficient (eigen) solvers
Performance Portability with PHIST+GHOST

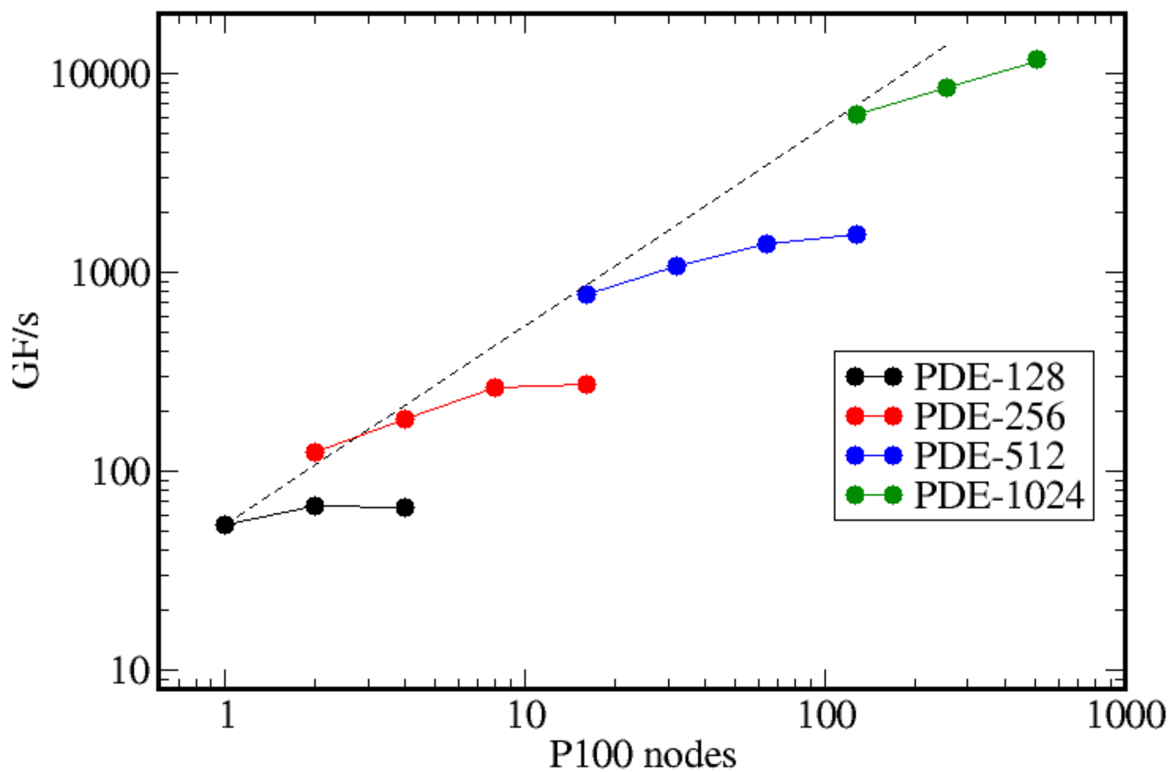
- Find 20 left-most eigenpairs of a spin-chain matrix (N ≈ 2.7M)
- Block Jacobi-Davidson QR method with MINRES ‘preconditioner’ (BJQR)
- Hardware bottleneck: main memory bandwidth(?)



Room for improvement?!

Hardware efficient (eigen) solvers
Parallel Performance Portability: PHIST+GHOST

- Find 10 left-most eigenpairs of 3D non-symmetric PDE
- Block Jacobi-Davidson QR method with GMRES ‘preconditioner’ (BJQR)
- Scalability tests on Piz Daint (with P100) (2 vectors/block - limited memory)



Solvers & Methods
numerical efficiency

Projection based eigensolvers: Mixed precision

Finding the eigenpairs (λ, x) of $Ax = Bx\lambda$ inside $I_\lambda = [\underline{\lambda}, \overline{\lambda}]$

Single precision in early iterations:
a) Save energy *b)* predict behaviors in high precision operations

: Convergence barred : Limited effect : Lasting effect

Choose m ($>$ number of desired eigenpairs) and $Y \in \mathbb{R}^{n \times m}$
while *not converged*

- Construct $U :\approx P_I Y$ (polynomial, contour integral, or moments)
- Compute SVD of U, resize subspace
- Orthogonalize U /
- Solve reduced eigenproblem $A_U W = B_U W \Lambda$
  with $A_U := U^T AU$, $B_U := U^T BU$
- $X := UW$
- Orthogonalize against locked eigenpairs X, lock newly converged
  eigenpairs /

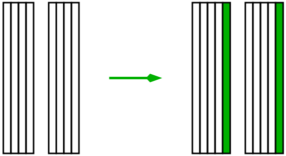
Projection based eigensolvers: Addressing SDC

Most time-consuming part in BEAST(-P/-C):
subspace projection $U = P_{A,B} Y$

&

During (polynomial or contour-based) projection:

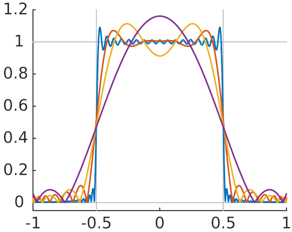
- Add checksum column(s) (linear combinations)



- $<$ 3% overhead for 1.3M top. ins. on 160 cores, check every 100 MVMs

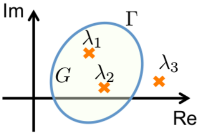
A posteriori analysis:

- Compare $\sigma_i(U^H BU)$ to max of filter function

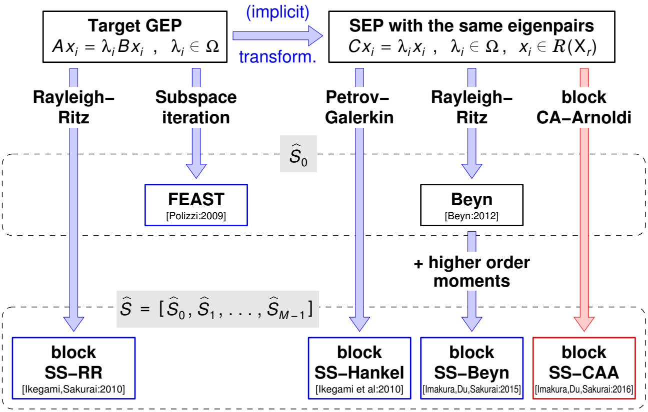


- Cheap

Block SS-CAA: A novel contour-based eigensolver



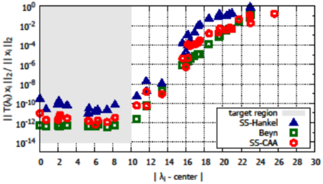
- Analyzed the relationship among typical contour integral-based eigensolvers
- Proposed novel method using communication-avoiding (CA) Arnoldi: block SS-CAA



Block SS-CAA: Results

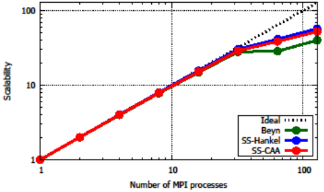
Computational accuracy:

- gen_hyper2 ($n = 1,000$) synthetic
- NLEVP: hyperbolic quadratic matrix polynomial
- MATLAB2016b



Scaling:

- railtrack2 ($n = 353,205$) model of the vibration of rail tracks
- NLEVP: quadratic eigenvalue problem
- COMA@U. of Tsukuba



Preconditioner of KS method for ill-conditioned problem

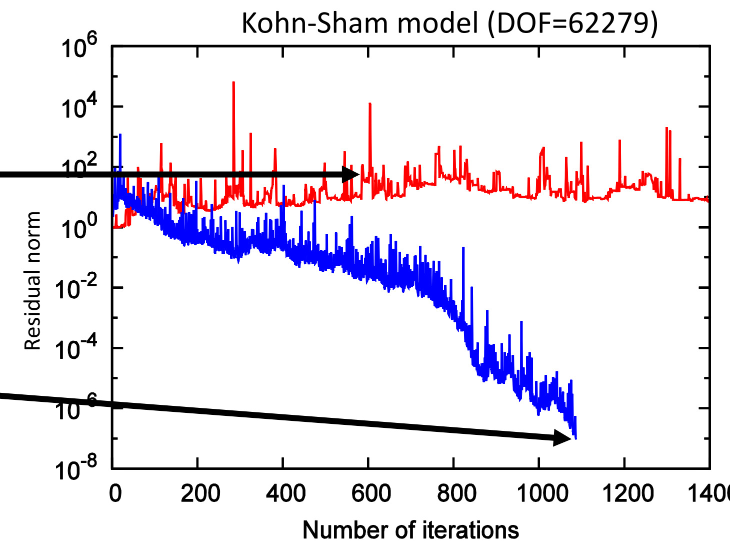
Target : Linear systems in FEAST and SSM $Ax = b$

Model name	DOF	Non-zero entries per row
Graphene	128~1,000,000	4 or 13
Kohn-Sham	57,575~76,163	20~24 on average

▶ Properties of coefficient matrix A

- ill-conditioned ・Indefinite ・Small diagonal entries

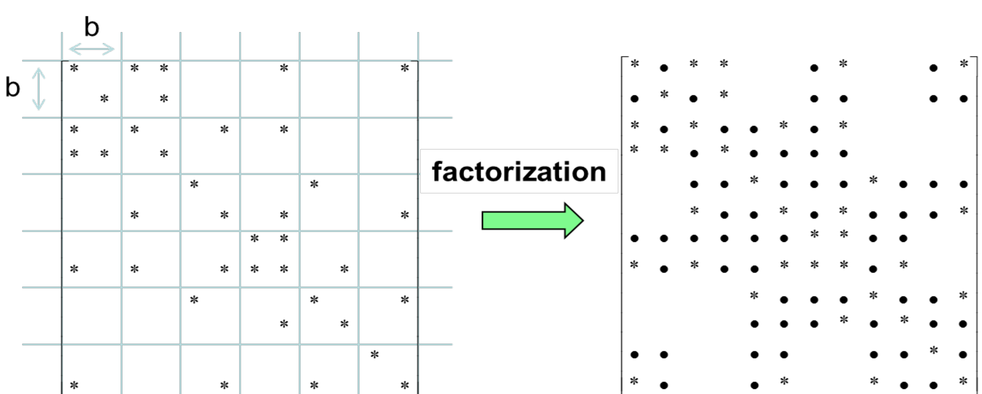
⇨ **Hard to be solved by original ICCG**



Our Proposal

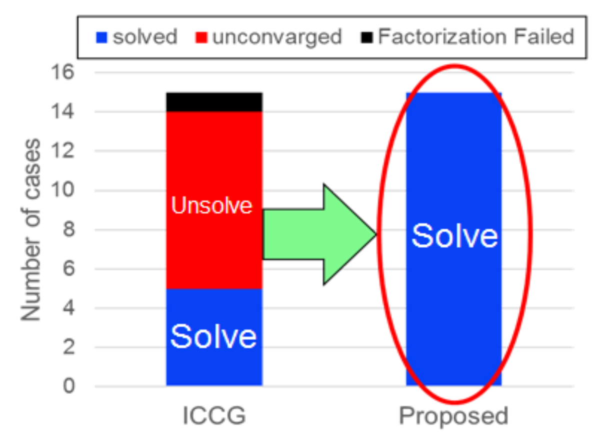
New preconditioner with regularizations

- Reg.1) Diagonal shift
- Reg.2) Blocking



Coefficient matrix A

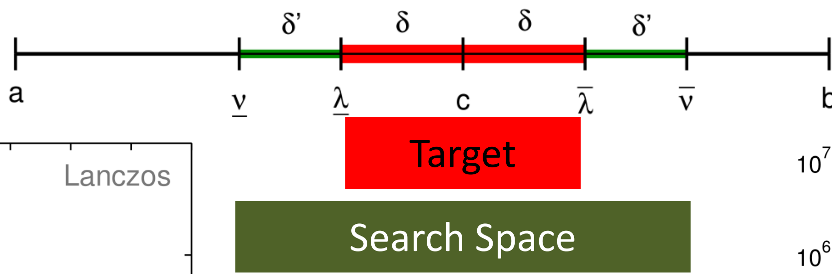
IC factorization with blocking

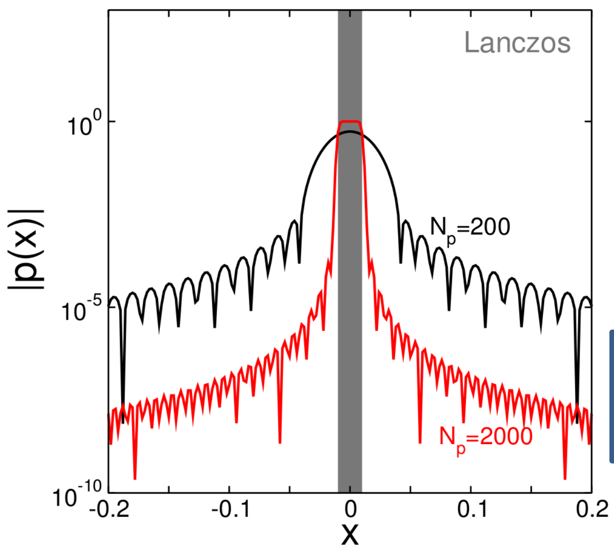


Hardware efficient solvers & applications

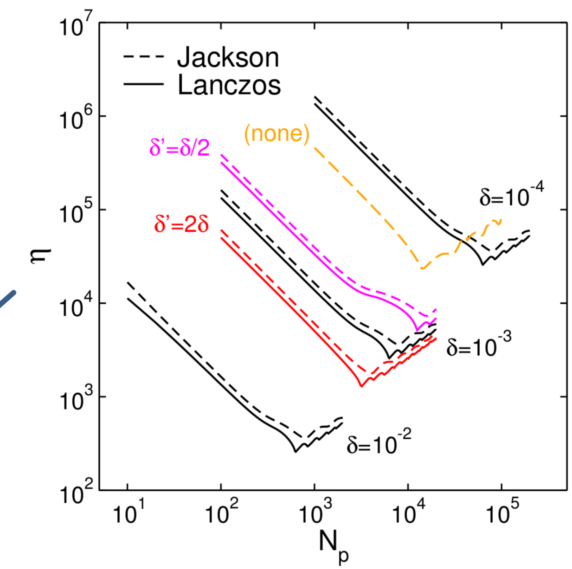
Chebyshev filter diagonalization: Algorithm & Performance (I)

Many interior eigenvalues of large sparse (Hermitian) matrices





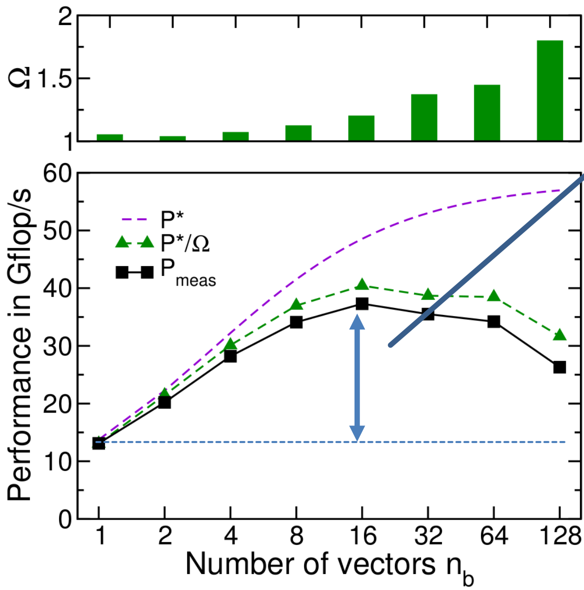
Filter quality
(lower is better)



- uses only spMVM / low synch. ⇒ very large unstructured matrices
- convergence ←→ eigenvalue density ⇒ large number of spMVMs
- clue to convergence: search vectors ≫ target vectors
  ⇒ massive vector blocking ⇒ spMMVM

Chebyshev filter diagonalization: Algorithm & Performance (II)

HSW-socket performance

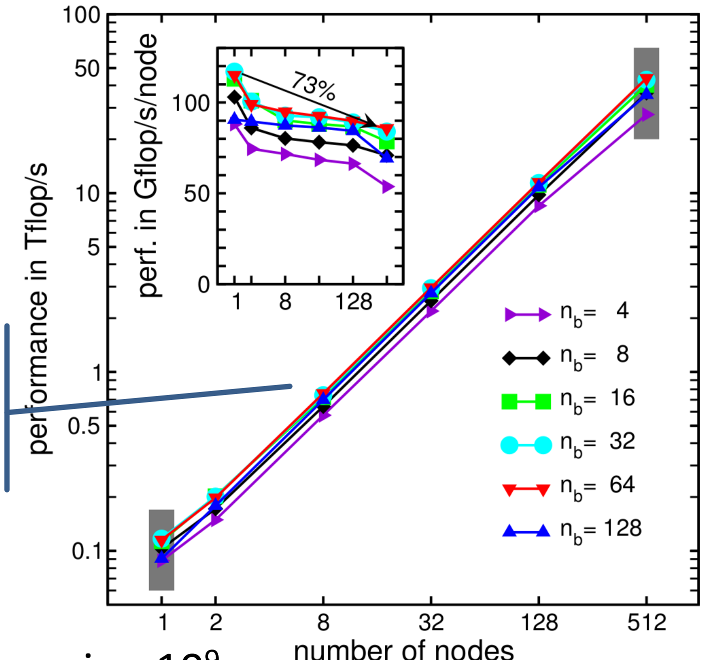


Vector blocking

3x performance gain

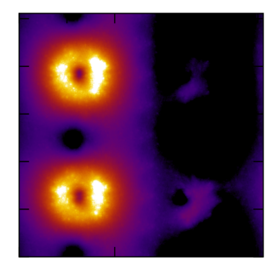
Weak scaling for topological insulator testcase

SuperMUC-2 performance



Compute 200 interior eigenpairs of a matrix of dimension 10^9
(10 hrs on 512 nodes)

Matrix	Nodes	D	$[\underline{\lambda}:\overline{\lambda}]_{rel}$	N_T	N_p	Runtime [hours]	Sust. perf. [Tflop/s]
topi1	32	6.71e7	7.27e−3	148	2159	3.2 (83%)	2.96
topi2	128		3.64e−3	148	4319	4.9 (88%)	11.5
topi3	512	1.07e9	1.82e−3	148	8639	10.1 (90%)	43.9
graphene1	128		4.84e−4	104	32463	10.8 (98%)	4.6
graphene2	512	1.07e9	2.42e−4	104	64926	16.4 (99%)	18.2



Pieper, Kreutzer, Alvermann, Galgon, Fehske, Hager, Lang, Wellein, J. Comp. Phys. 325, 226 (2016)

ESSEX@PASC2017 – 26 – 28 June, 2017

COMPUTING BULKS OF INNER EIGENPAIRS OF LARGE SPARSE MATRICES: FROM APPLICATIONS AND ALGORITHMS TO PERFORMANCE AND SOFTWARE ENGINEERING (I+II)

- **Takeo Hoshi (Tottori University, Japan)**
- **Tetsuya Sakurai (University of Tsukuba, Japan)**
- **Yousef Saad (University of Minnesota, USA)**
- **Andreas Alvermann (Universität Greifswald, Germany)**
- **Kengo Nakajima (The University of Tokyo, Japan)**
- **Hartwig Anzt (University of Tennessee, USA)**
- **Jonas Thies (German Aerospace Center, Germany)**
- **Mike Heroux (Sandia National Laboratories, USA)**

https://pasc17.pasc-conference.org/program/minisymposia/