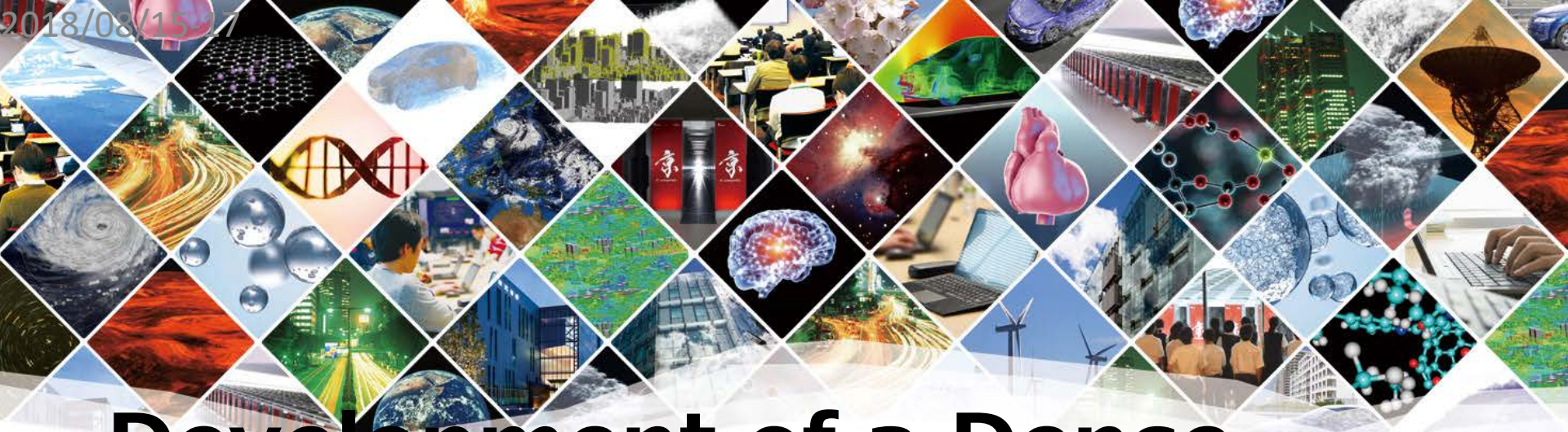


Development of a Dense Eigenvalue solver for Exa-scale Systems

Toshiyuki IMAMURA

RIKEN Center for Computational Science

MolSSI Workshop / ELSI Conference, the Graduate,

Richmond VA, USA, 15-17 August 2018

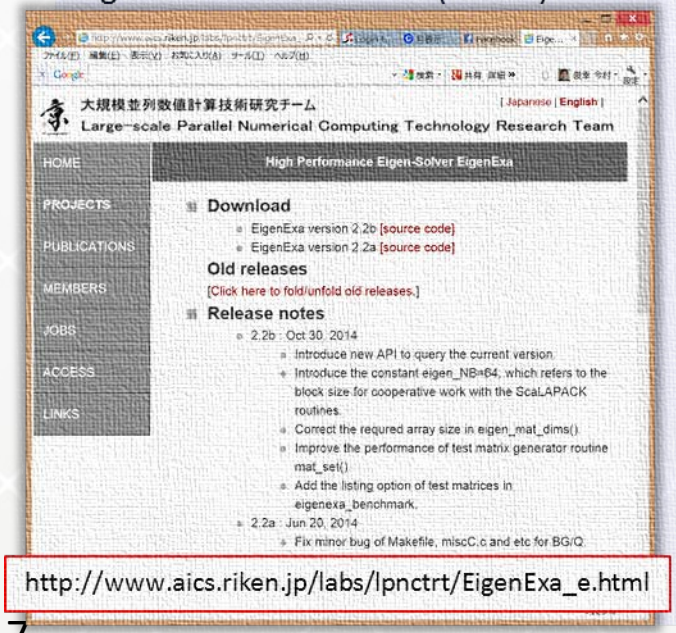
Computer simulations create the future

Agenda

1. **Quick Overview of Project**
   - Past and Present of EigenExa
   - Diagonalization of a 1million x 1million matrix on K computer
2. **The latest updates**
3. **Future direction**
4. **Summary**

Prof. Sakurai Team 'H4ES'

EigenExa:: dense solver (RIKEN)



http://www.aics.riken.jp/labs/lpnctrt/EigenExa_e.html

Z-Pares:: sparse solver (U.Tsukuba)

http://zpares.cs.tsukuba.ac.jp/

H4ES (2011-2016, granted by CREST JST)

Computer simulations create the future

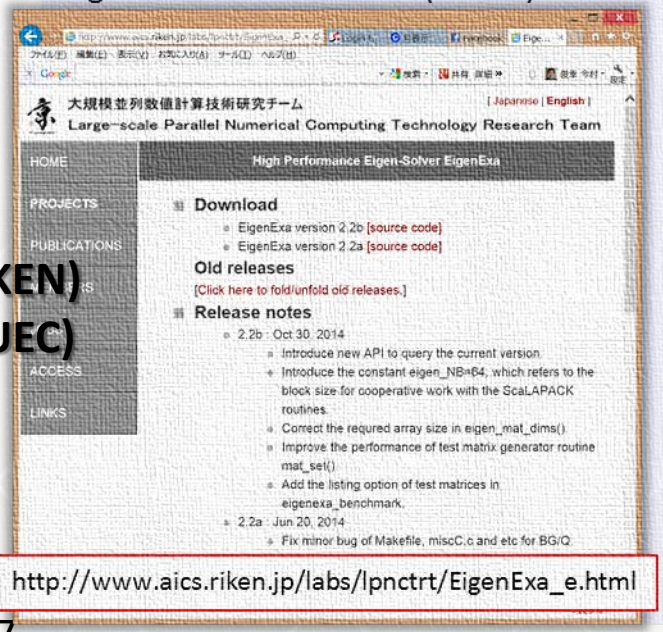
1. **Eigenvalue problem is of significance in many scientific and engineering fields**

$$Ax = \lambda Bx$$

1. **Sparse and dense solver must be tightly cooperating!**
2. **Not only theory but computer science and applications!**

Prof. Sakurai Team 'H4ES'

EigenExa:: dense solver (RIKEN)



http://www.aics.riken.jp/labs/lpnctrt/EigenExa_e.html

Z-Pares:: sparse solver (U.Tsukuba)

http://zpares.cs.tsukuba.ac.jp/

Dense:
Imamura(RIKEN)
Yamamoto(UEC)

Sparse & LS:
Sakurai(Tsukuba)
Zhang(Nagoya)

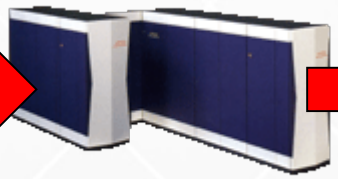
Application:
Hoshi(Tottori)
Kuramashi(Tsukuba)

History of the dense solver is long

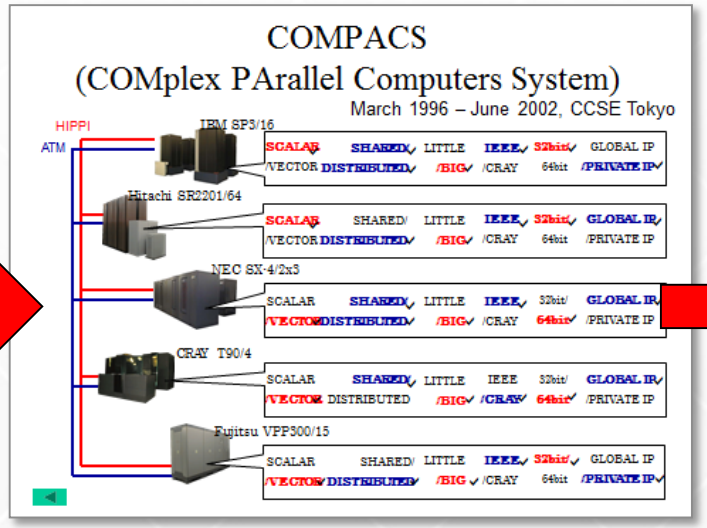
Computer simulations create the future

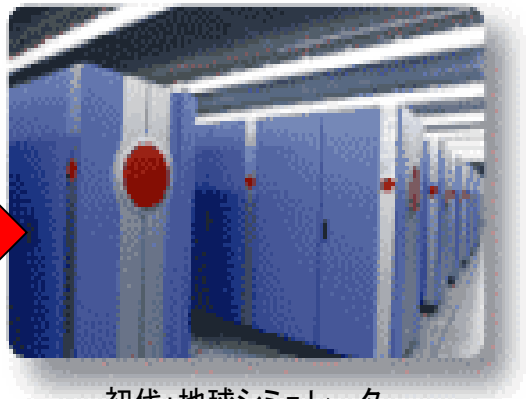
- **From VPP to the K computer**

ADENART256



VPP500@Kyoto-u

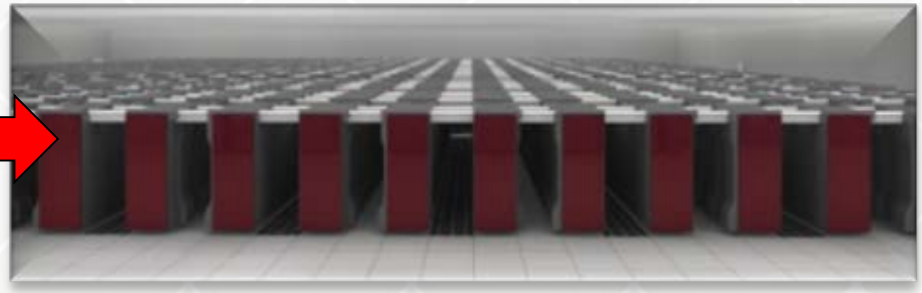




初代：地球シミュレータ
Earth Simulator 1

T2K Open supercom @ U-Tokyo

RICC @ RIKEN



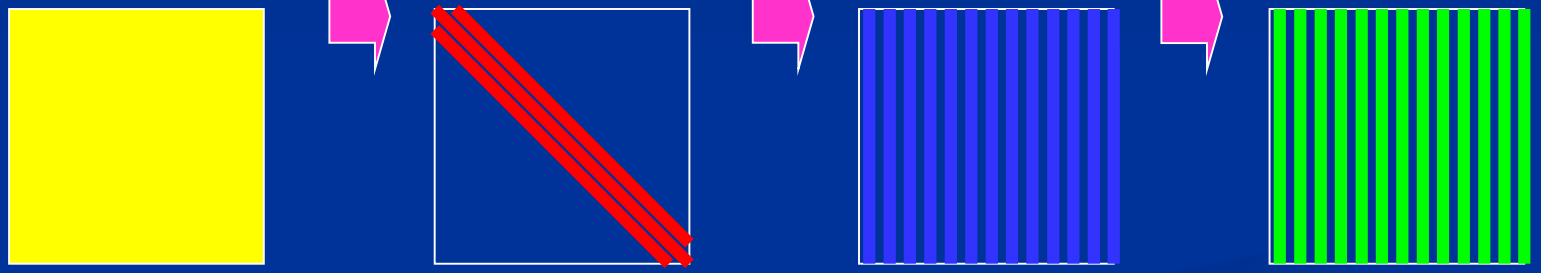
京コンピュータ @AICS.RIKEN
The K Computer

3, “Full Diagonalization” on ES

S Yamada, T Imamura, T Kano, M Machida, High-performance computing for exact numerical approaches to quantum many-body problems on the earth simulator, SC06

Method:

Householder $O(N^3)$ → Divide&Conqure $O(N^2)$~$O(N^3)$ → Back Transform $O(N^3)$



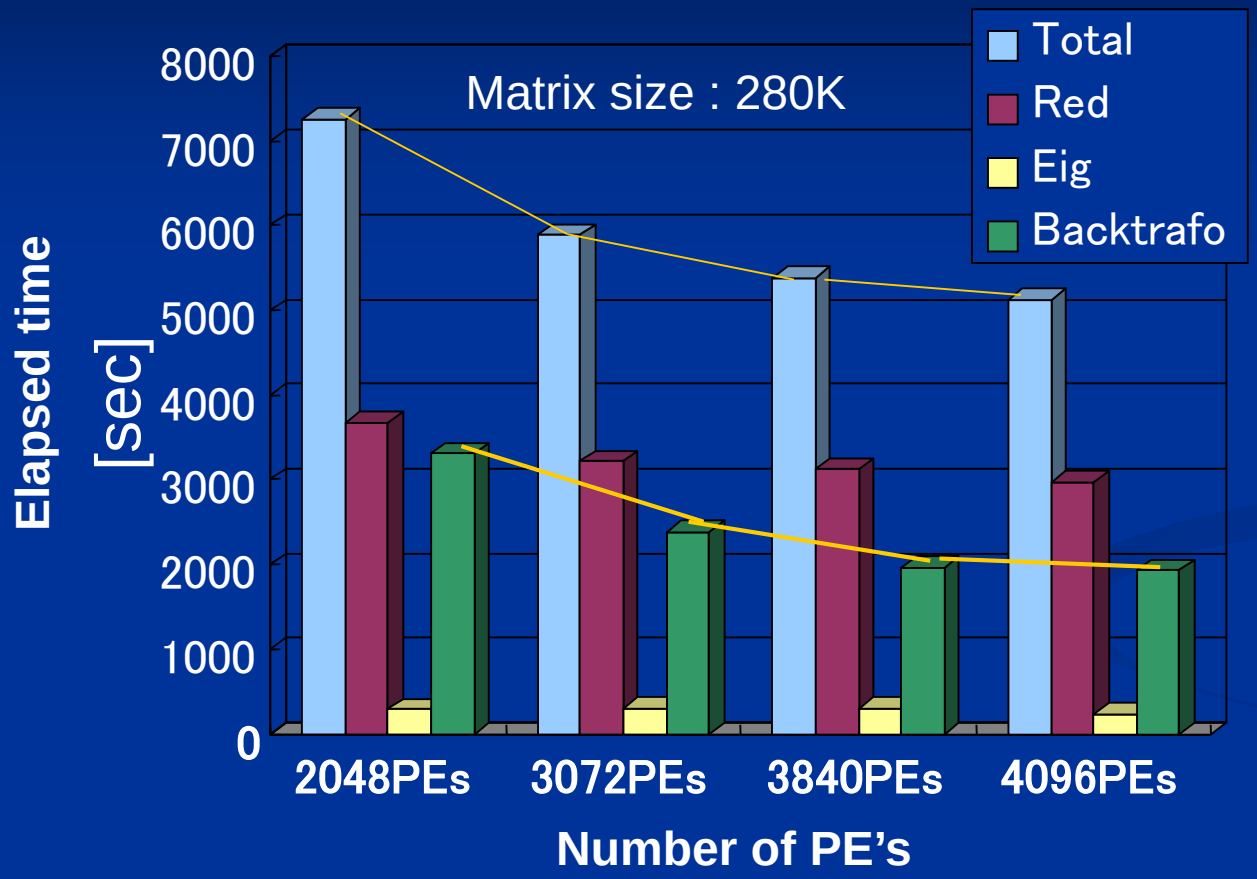
Dense matrix → Tridiagonal → Eigenpairs → Eigenpairs

Householder:	Self-development	Block Algorithm Loop unrolling 2-D distribution
Divide & Conqure:	Porting ScaLAPACK	
Back Transform:	Self-development	

The best combination in terms of stability, vectorization, and parallelization

3, “Full Diagonalization” on ES

Test of PE No. scalability:



Matrix dim.: 280,000
Memory:2 T Bytes
No. of PE’s:
2048～4096

Reduction:
Householder transformation

Eig:
Divide & Conquer

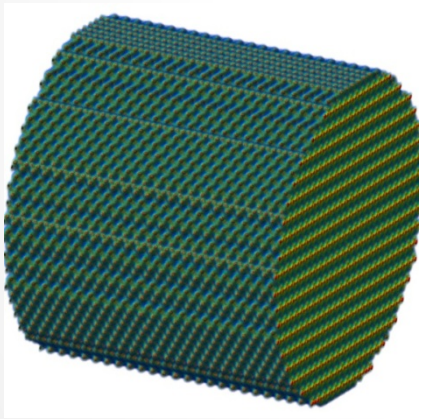
Backtrafo:
Back transformation

Total scalability is OK !

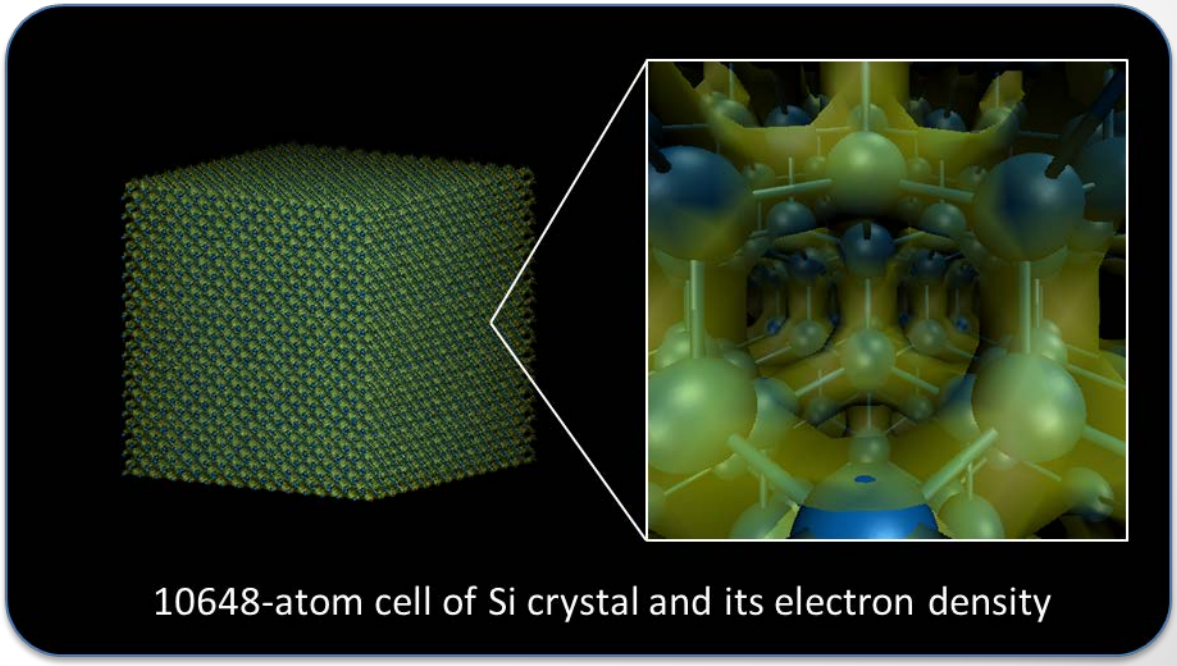
Especially, scalability of the back transformation is good !

2.1 Review on peta-scale eigenvalue solvers

- For grand challenge applications like a nano-material simulation, they demand a high-performance eigensolver strongly.



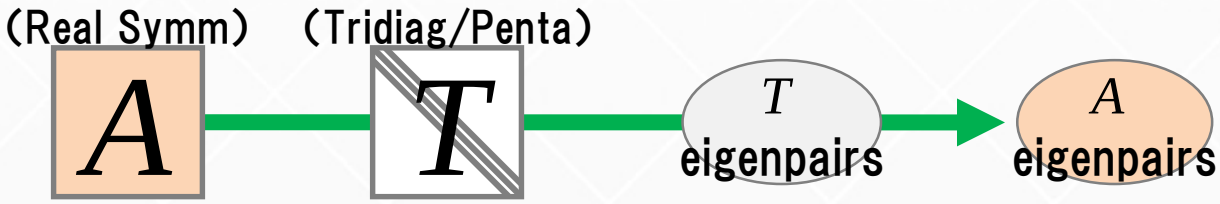
Silicon nanowire



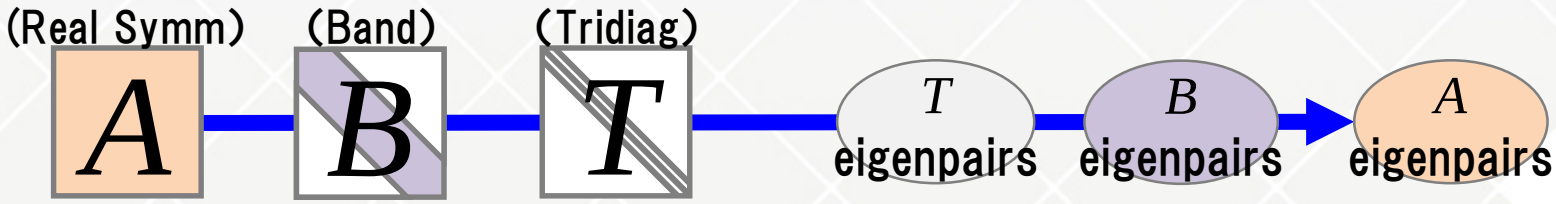
Y.Hasegawa, et al @AICS.RIKEN, GBP winner, SC2011

Two major algorithms for dense SEVP

- **1-stage algorithm (EigenExa, ELPA1, Wilkinson512/Kevd)**
  - Standard numerical scheme
  - Householder-tridiagonalization has memory-bound bottleneck



- **2-stage algorithm (ELPA2, PLASMA, MAGMA)**
  - Transform to Banded-matrix = Less Byte/Flop operation by GEMM
  - Needed 2-stage back-transformation, but slows unfortunately. However, a big advantage when a few eigenmodes are computed



Divide and Conquer method for a banded matrix

Computer simulations create the future

(Λ, X) = Divide and Conquer(matrix B)

Decompose B into B_1 and B_2 with k-perturbations as

$$B = B_1 \oplus B_2 + \alpha_1 c_1 c_1^t + \alpha_2 c_2 c_2^t + \ldots \alpha_b c_b c_b^t$$

(Λ1 , X1) = Divide and Conquer (B1);
(Λ2 , X2) = Divide and Conquer (B2);

→ $\Lambda^{(0)} \quad X^{(0)}$

$$B^{(1)} = \Lambda^{(0)} + \alpha_1 d_1 d_1^t, \; d_1 = {X^{(0)}}^t c_1$$

Solve eigenvalues and vectors of B^{(1)} via secular eq.

$$f(\lambda) = 1 + \alpha_1 \sum_{i=1}^{n} \frac{{d_1}_i^2}{\lambda_i - \lambda} = 0 \rightarrow \Lambda^{(1)}$$

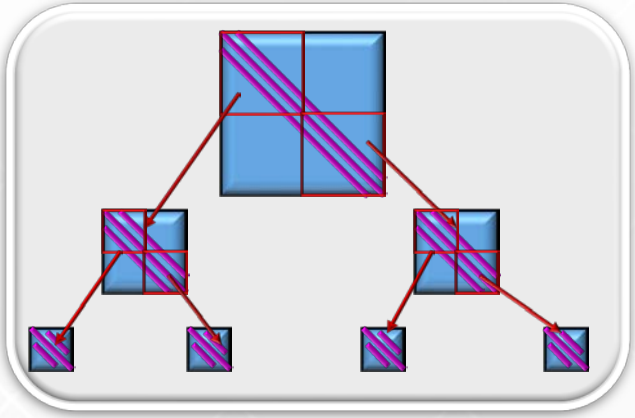
$$x = (X_1, X_2)\mathrm{Normal}((\Lambda^{(0)} - \lambda I)^{-1} d_1) \rightarrow X^{(1)}$$

Do i=2,…

$$B^{(i)} = \Lambda^{(i-1)} + \alpha_i d_i d_i^t, \; d_i = \prod {X^{(i-1)}}^t c_i$$

$$\Lambda = \Lambda^{(k)}$$
$$X = \prod X^{(k)}$$

Rough estimation: the cost $\mathrm{O}((2\mathrm{b}-1)\mathrm{N}^3)$, it increases 2k-1 times compared with k=1. Usually, New 1-step scheme shows better performance than old-1-step or 2-step.

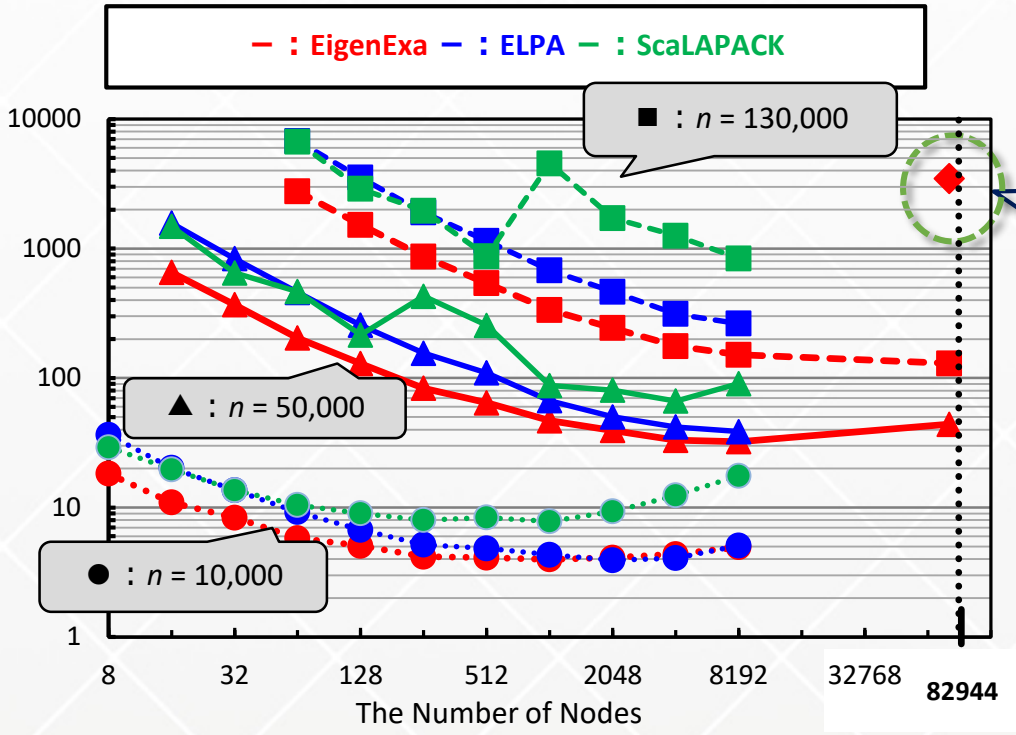


*This type of DC algorithm was discussed by Arbenz, Gander, Golub, Gansterner, et al. in early 1990's.
*We implemented a parallel version adopting a robust technique to modify orthogonality of eigenvectors.

EigenExa: World Largest Dense Eigenvalue Computation

Computer simulations create the future

- **We have successfully done a world largest-scale dense eigenvalue benchmark (one million dimension) by EigenExa taking advantage of the overall nodes (82,944 processors) of K computer in 3,464 seconds. Our EigenExa achieves 1.7 PFLOPS (16% of the K computer's peak performance).**
- **Feasibility and Reliability for the algorithm and the library are confirmed, especially assumed on a post-K system.**



◆ : n = 1,000,000

EigenExa solves a world largest-scale problem.
(1.7 PFLOPS, 16% of K computer's theoretical peak performance)

$$\max_i \|A\boldsymbol{v}_i - \lambda_i \boldsymbol{v}_i\|_2 / \|A\|_F = 3.1 \times 10^{-13}$$
$$\|V^\top V - I\|_F = 2.1 \times 10^{-10}$$

- ✓ n is the dimension of problems.
- ✓ 1 MPI process * 8 threads per node.
- ✓ Test matrices are randomly generated.

Specification of K computer
- **Peak performance: 10.6 PFLOPS**
- **Num. of Nodes: 82,944**
- **Performance/node: 128 GFLOPS (One octa-core SPARC 64 VIIIfx)**
- **Network: Tofu interconnect (6D mesh-torus)**

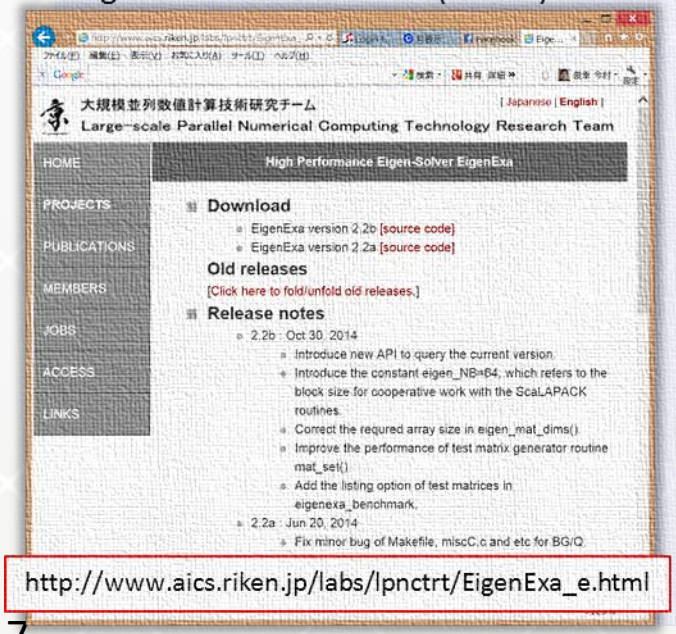
Related performance report is
T.Fukaya, TI. "Performance evaluation of the EigenExa eigensolver on Oakleaf-FX: tridiagonalization versus pentadiagonalization", PDSEC2015

Agenda

1. **Quick Overview of Project**
   - Past and Present of EigenExa
   - Diagonalization of a 1million x 1million matrix on K computer
2. **The latest updates**
3. **Future direction**
4. **Summary**

Prof. Sakurai Team 'H4ES'

EigenExa:: dense solver (RIKEN)



http://www.aics.riken.jp/labs/lpnctrt/EigenExa_e.html

Z-Pares:: sparse solver (U.Tsukuba)

http://zpares.cs.tsukuba.ac.jp/

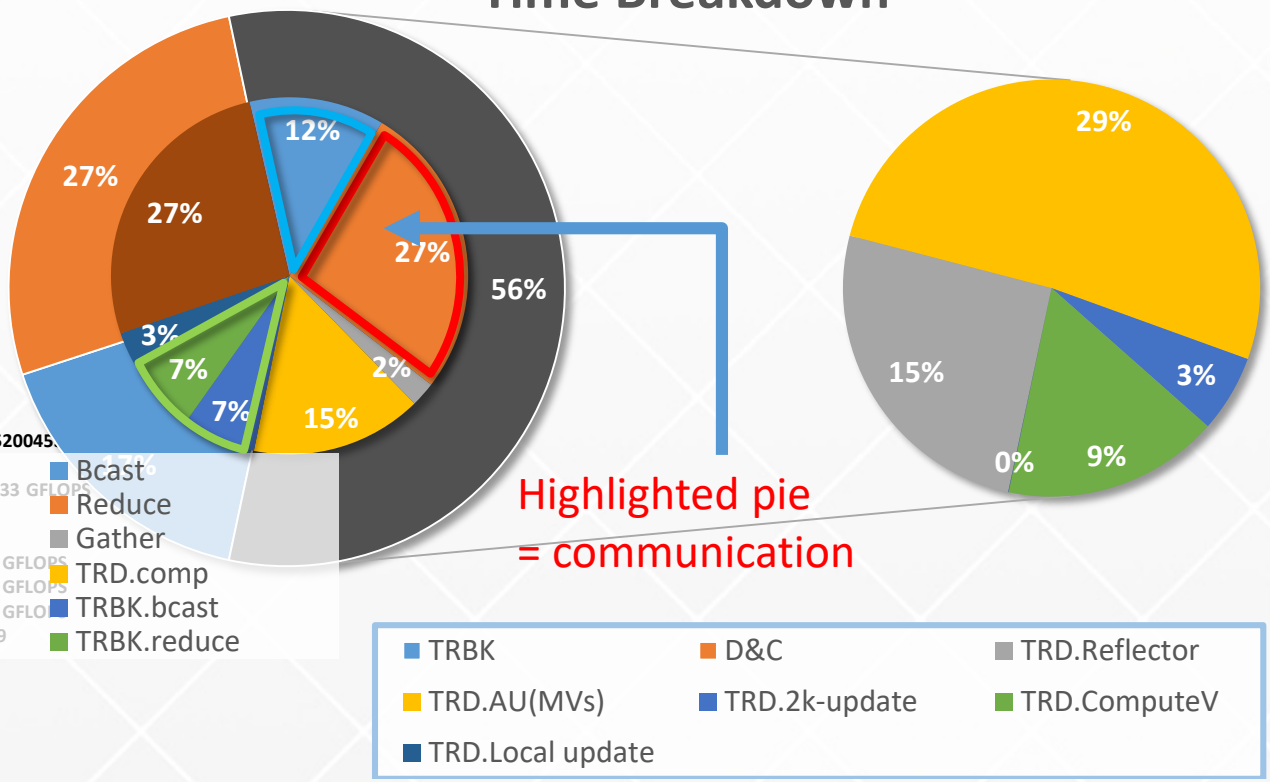
Bottleneck is network

Computer simulations create the future

```
NUM.OF.PROCESS= 82944 ( 288 288 )
NUM.OF.THREADS= 8
calc (u,beta)     503.0970594882965
mat-vec (Au)      1007.285000801086 661845.1244051798
2update (A-uv-vu) 117.4089198112488
5678160.294281102
calc v            0.000000000000000
v=v-(UV+VU)u      328.3385872840881
UV post reduction 0.6406571865081787
COMM_STAT
  BCAST  ::  424.3022489547729
  REDUCE ::  928.1299135684967
  REDIST ::  0.000000000000000
  GATHER ::  78.28400993347168
TRD-BLK 1000000 1968.435860157013
677356.7583893638 GFLOPS
TRD-BLK-INFO 1000000   48
before PDSTEDC 0.1448299884796143
PDSTEDC 905.2210271358490
MY-REDIST1 1.544256925582886
MY-REDIST2 14.75343394279480
RERE1 4.861211776733398E-02
COMM_STAT
  BCAST  ::  4.860305786132812E-02
  REDUCE ::  2.155399322509766E-02
  REDIST ::  0.000000000000000
  GATHER ::  0.000000000000000
PDGEMM 532.6731402873993 5417097.56520045
GFLOPS
D&C 921.8044028282166 3130319.58
TRBAK= 573.9026420116425
COMM= 533.7601048946381
  573.9026420116425 3484911.6445
  182.3303561210632 5484550.2486
  152.0370917320251 6577342.3353
  0.1022961139678955 7.37965488
COMM_STAT
  BCAST  ::  229.3666801452637
  REDUCE ::  234.4477448463440
  REDIST ::  0.000000000000000
  GATHER ::  0.000000000000000
TRBAKWY 573.9029450416565
TRDBAK 1000000 573.9216639995575 3484796.141101135
GFLOPS
Total 3464.162075996399 1795203.448396145 GFLOPS
Matrix dimension =  1000000
Internally required memory =  480502032  [Byte]
Elapsed time =  3464.187163788010  [sec]
```

The World Largest Dense Eigenvalue Computation

Time Breakdown



CA for EigenExa

Computer simulations create the future

- **Communication Avoiding for Householder transformation unlike CAQR**
  - Blocking technique, increasing locality by data replication, and exchange the operation order.
  - Introducing an extended form of vector ‘A’.
  - Computing Au and u^Tu, simultaneously.

TI. “Communication-Avoiding approaches of dense Eigenvalue / SVD problems”, PMAA18
TI. “Parallel dense eigenvalue solver and SVD solver for post-petascale computing systems “, PMAA16
TI, etc, “CAHTR: Communication-Avoiding Householder Tridiagonalization”, ParCo15

Principles : Distributive Law && Exchange order && Introducing the correction terms && Combine couples of collective ops.

- naive

$$s = \mathrm{sign}(\|u\|, -(u,e))$$
$$u := u - se$$
$$v = Au$$
$$[C_U; C_V] = [U^T; V^T]u$$
$$v := v - (UC_V + VC_U)$$
$$f = (u,v)$$
$$v := v - afu$$

$$\begin{bmatrix} v & d \\ s & t \end{bmatrix} = \begin{bmatrix} A & u \\ u^t & 0 \end{bmatrix} \begin{bmatrix} u & e \\ 0 & 0 \end{bmatrix}$$
$$s = \mathrm{sign}(\sqrt{s}, -t)$$
$$[u,v] := [u,v] - s[e,d]$$
$$[C_U; C_V; g] = [U^T; V^T; v^T]u$$
$$v := v - (UC_V + VC_U)$$
$$f = g - 2C_U^T C_V$$
$$v := v - afu$$

optimal

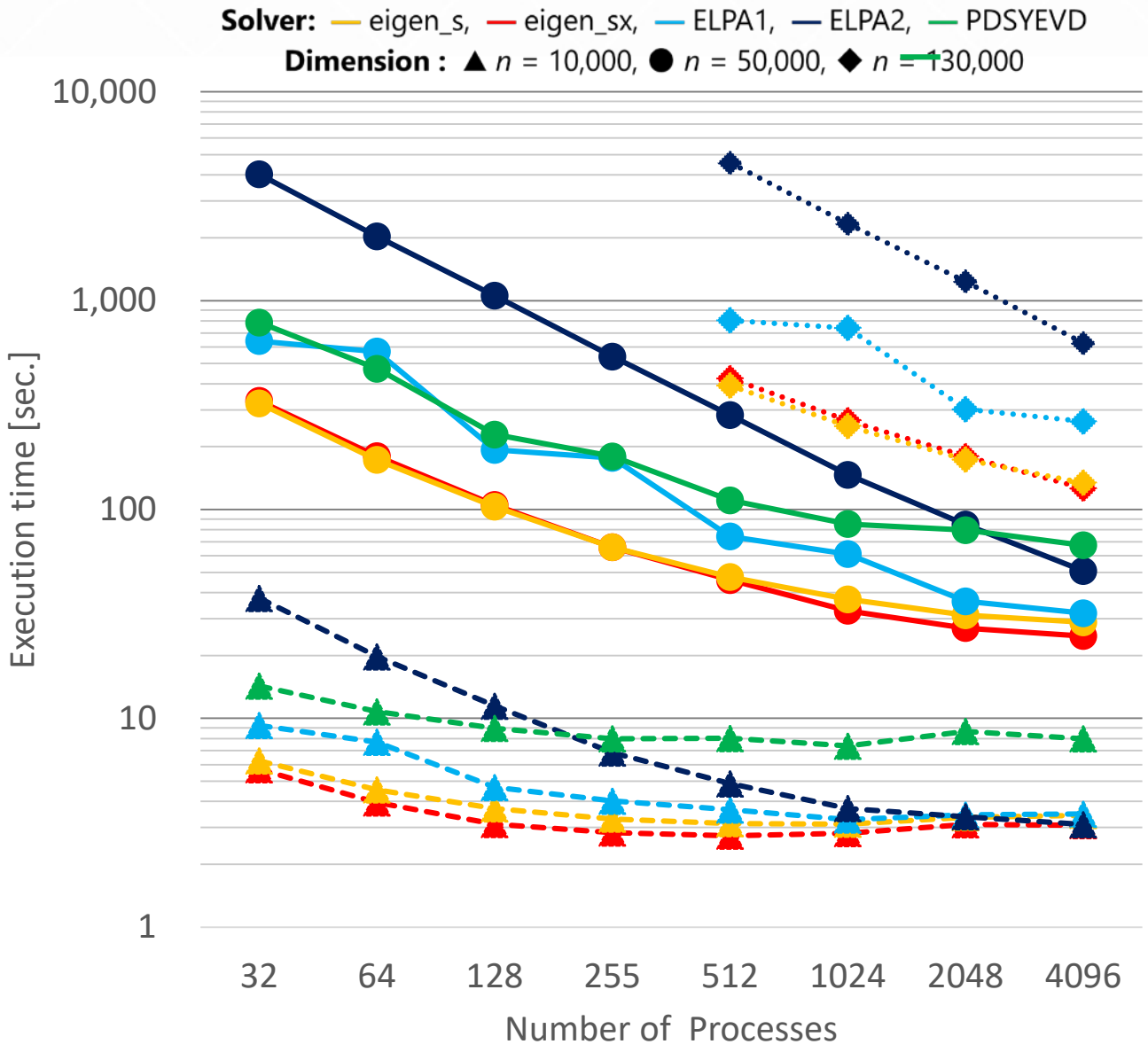
$$\left[\begin{array}{c|c} v & d \\ s & t \\ C_U & \gamma_U \\ C_V & \gamma_V \\ \hline g & v_1 \\ v_1 & a_{11} \end{array}\right] = \left[\begin{array}{c|ccc} I & 0 & 0 & 0 \\ 0 & I & 0 & 0 \\ 0 & 0 & I & 0 \\ 0 & 0 & 0 & I \\ \hline u^t & 0 & 0 & 0 \\ e^t & 0 & 0 & 0 \end{array}\right] \left[\begin{array}{c|ccc} A & u & U & V \\ \hline u^t & 0 & 0 & 0 \\ U^t & 0 & 0 & 0 \\ V^t & 0 & 0 & 0 \end{array}\right] \left[\begin{array}{c|c} u & e \\ \hline 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{array}\right]$$

$$s = \mathrm{sign}(\sqrt{s}, -t)$$
$$[u,v] := [u,v] - s[e,d]$$
$$[C_U; C_V] = [C_U; C_V] - s[\gamma_U; \gamma_V]$$
$$v := v - (UC_V + VC_U)$$
$$f = g - 2C_U^T C_V - s(2v_1 - sa_{11})$$
$$v := v - afu$$

Performance of the latest EigenExa (2.5RC)

Computer simulations create the future

- **Test configurations:**
  - K computer
  - One SPARC64 VIIIfx CPU (128 GFLOPS, 8 cores) / node
  - Tofu (6-dimensional torus internode network)
  - all eigenpairs of n x n Frank matrix
  - 1 MPI process/node, 8 threads/process
  - Fujitsu multithreaded BLAS / MPI libraries
  - ELPA version 2016
  - Without openmp for loop optimization but thread-parallelized BLAS is used.

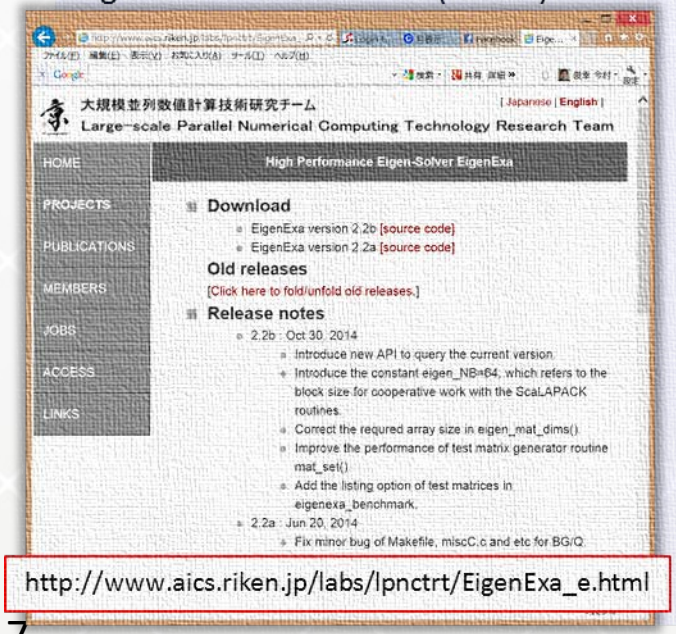


Agenda

1. **Quick Overview of Project**
   - Past and Present of EigenExa
   - Diagonalization of a 1million x 1million matrix on K computer
2. **The latest updates**
3. **Future direction**
4. **Summary**

Prof. Sakurai Team 'H4ES'

EigenExa:: dense solver (RIKEN)



http://www.aics.riken.jp/labs/lpnctrt/EigenExa_e.html

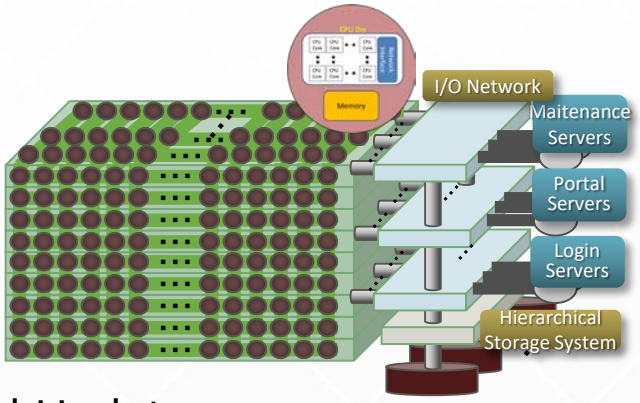
Z-Pares:: sparse solver (U.Tsukuba)

http://zpares.cs.tsukuba.ac.jp/

FLAGSHIP2020 Project

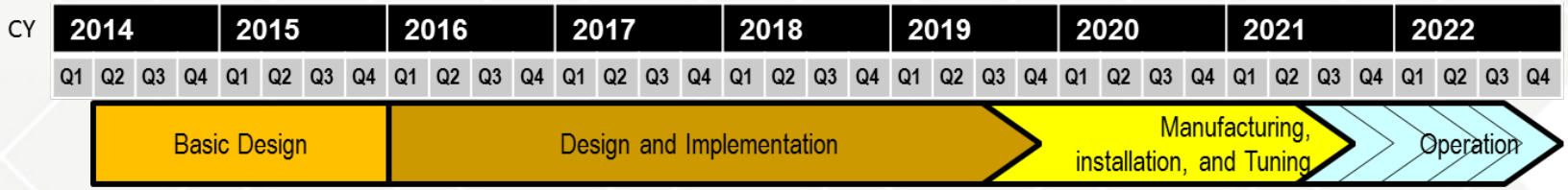
Computer simulations create the future

- Missions
  - Building the Japanese national flagship supercomputer, post K, and
  - Developing wide range of HPC applications, running on post K, in order to solve social and science issues in Japan



- Project organization
  - Post K Computer development
    - RIKEN AICS is in charge of development
    - Fujitsu is vendor partner.
    - International collaborations: DOE, JLESC, ..
  - Applications
    - The government selected 9 social & scientific priority issues and their R&D organizations.

- Status and Update
  - "Basic Design" was finalized and now in "Design and Implementation" phase.
    - Now, we are working on detail evaluation by simulators and compilers
  - We have decided to choose ARM v8 with SVE as ISA for post-K manycore processor.
  - Some delay of delivery will be expected.

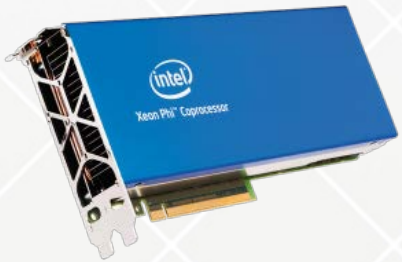


Target Architecture in near future

Computer simulations create the future

- **We also have two branched projects from EigenExa on the K computer architecture**
  - GPU (single):
    - Eigen-G = Experimental code on a single node + a single GPU environment
    - ASPEN.K2 = Automatic-tuning GPU BLAS kernels, especially, SYMV kernel
  - Intel Xeon Phi
    - Divide and conquer algorithm for GEVP focused on a pair of banded matrices

TI, etc, "Eigen-G: GPU-based eigenvalue solver for real-symmetric dense matrices", PPAM2013, LNCS8384
TI, etc. "High Performance SYMV Kernel on a Fermi-core GPU", VECPAR 2012, LNCS 7851,
TI, etc. "Automatic-tuning for CUDA-BLAS kernels by Multi-stage d-Spline Pruning Strategy", @^2HPSC 2014



Y.Hirota, etc, "Parallel Divide-and-Conquer Algorithm for Solving Tridiagonal Eigenvalue Problems on Manycore Systems", PPAM2017, LNCS10777
Y.Hirota, etc. "Acceleration of Divide and Conquer Method for Generalized Eigenvalue Problems of Banded Matrices on Manycore Architectures, PMAA14.

For Very small kernel

- **Kevd: Developed by Mr. Kudo (was a Ph.D student)**
  - For acceleration of a block Jacobi iteration
  - Wilkinson's overlap technique (SYR2(K)+SYMV)
  - 4x4 micro memory tiling to make use of cache line maximum so called SOSOA
  - SIMD intrinsic

- **Extension**
  - 8x4 or 8x8 tiling
  - AVX2, AVX-512 for KNL

  ARM-SVE for post-K

```
! Wilkinson's technique
! Blue: DSYMV
! Red : DSYR2
do i=1,m
  ui=u(i), vi=v(i), yi=y(i), xi=x(i)
  do j=1,i-1
    aij = a(j,i)
    aij += u(j)*vi + v(j)*ui
    y(j) += aij * xi
    yi += aij * x(j)
    a(j,i) = aij
  enddo
enddo
```

Kevd on K and KNL

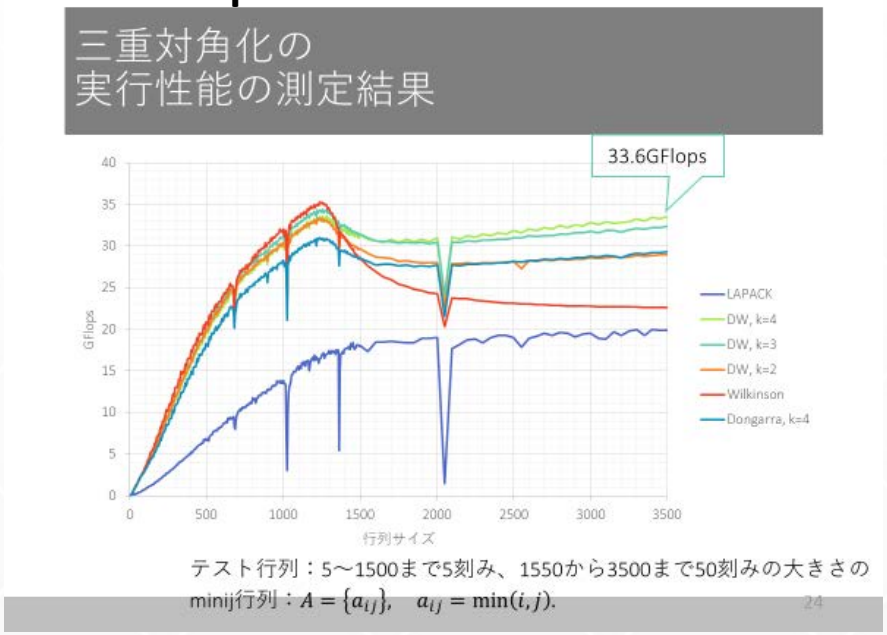
- **Benchmark and Performance comparison**
  - K: SPARC64 VIII+fx (by Mr.Kudo and Prof.Yamamoto)
  - x86 (Xeon)
  - KNL

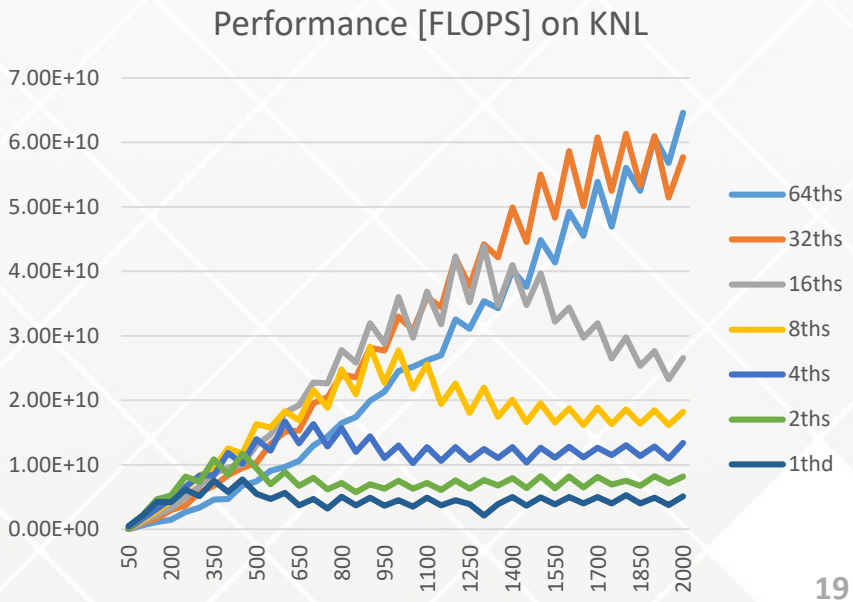
- **Wilkinson512 on NKL**
  - -O3 -xHost –qopenmp
  - -lmkl_intel_lp64 -lmkl_intel_thread -lmkl_core -liomp5 -lpthread -lm -ldl

Rough evaluation

- 2~7GFLOPS/core

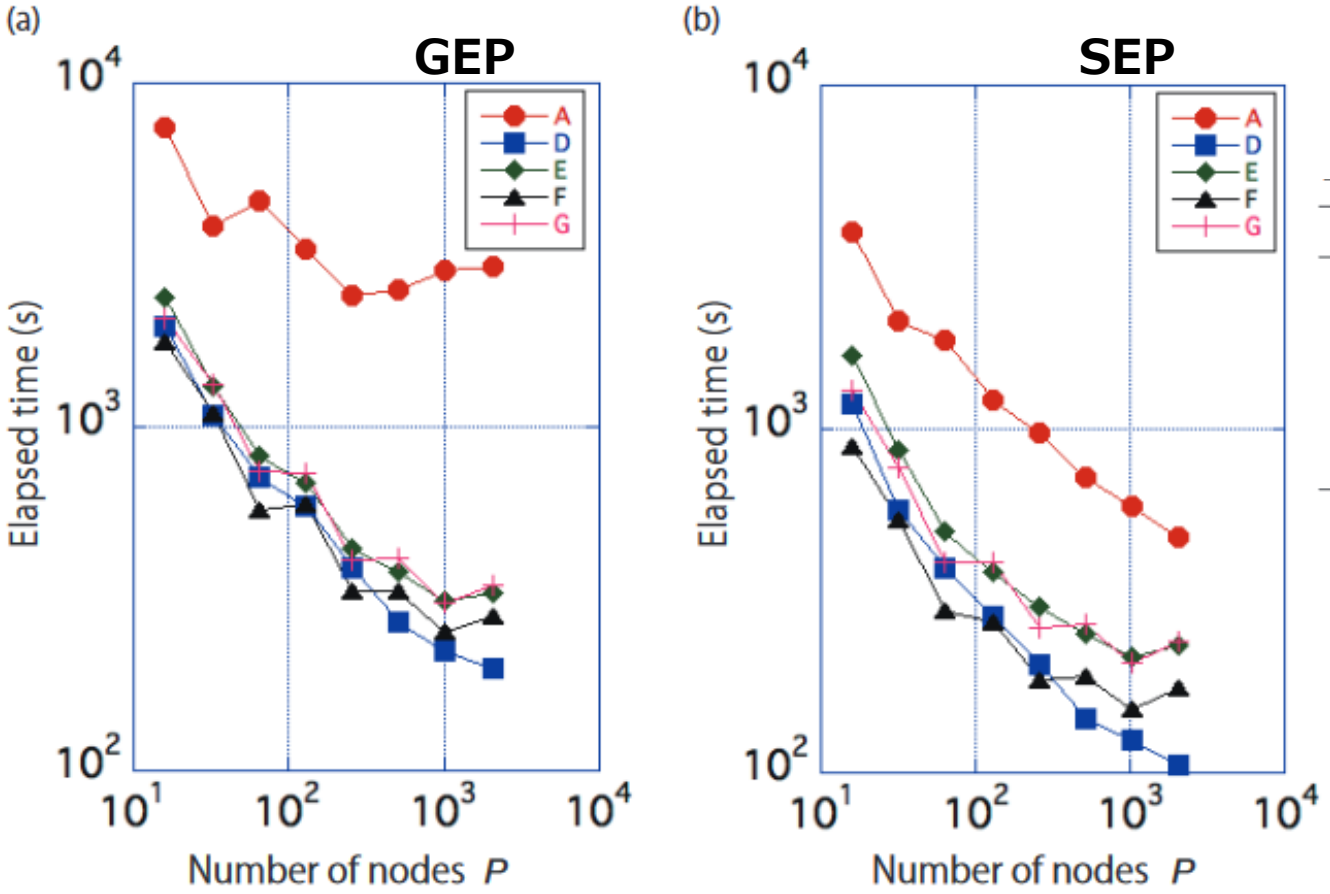
K computer





EigenKernel for GEP (Prof.Hoshi@Tottori U.)

- **ELPA 1|2(+Cholesky)+EigenExa on OFP (a KNL cluster)**



Workflow	SEP solver	Reducer
A	ScaLAPACK	ScaLAPACK
B	Eigen_sx	ScaLAPACK
C	ScaLAPACK	ELPA
D	ELPA2	ELPA
E	ELPA1	ELPA
F	Eigen_s	ELPA
G	Eigen_sx	ELPA

Fig. 3 Benchmark on Oakforest-PACS. The matrix size of the problem is $M = 90,000$. The computation was carried out with $P = 16, 32, 64, 128, 256, 512, 1024, 2048$ nodes in the workflows of A(circle), D(square), E(diamond), F(triangle), G(cross). (a) The elapsed time for the whole GEP solver. (b) The elapsed time for the reduced SEP solver.

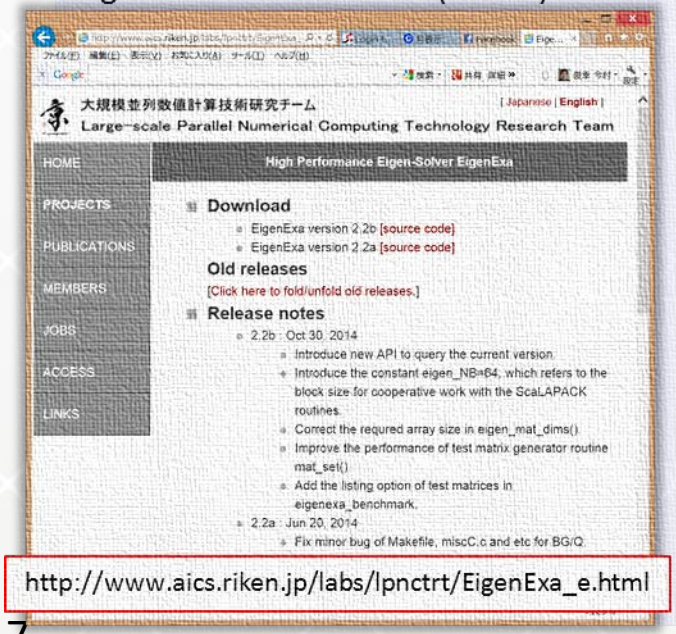
https://arxiv.org/abs/1806.00741

Agenda

1. **Quick Overview of Project**
   - Past and Present of EigenExa
   - Diagonalization of a 1million x 1million matrix on K computer
2. **The latest updates**
3. **Future direction**
4. **Summary**

Prof. Sakurai Team 'H4ES'

EigenExa:: dense solver (RIKEN)



http://www.aics.riken.jp/labs/lpnctrt/EigenExa_e.html

Z-Pares:: sparse solver (U.Tsukuba)

http://zpares.cs.tsukuba.ac.jp/

Summary of talk

- **EigenExa project (2011-2016, 2017-)**
  - The first milestone : 1million order eigenvalue computation with full nodes of K computer.
  - Second milestone : optimization of communication
  - Current version 2.4 b is available from
    - ✓ **http://www.r-ccs.riken.jp/labs/lpnctrt/en/projects/eigenexa/**
    - ✓ **http://www.r-ccs.riken.jp/labs/lpnctrt/assets/img/EigenExa-2.3c_users_manual_2014-06-24_en.pdf**

- **Future work towards post-K project also exa-scale computing**
  - Establish the CA technology for total performance of EigenExa
  - Reduced/Quadruple Precision version
  - Vector computers, other platforms

THANKS!

- **Our goal is reality on Exa-scale computing!**
  - New target architecture, ARM64+SVE
  - New topic is also concerned like Reproducibility