数値計算ライブラリ開発におけるコデザインについて

今村 俊幸
理化学研究所計算科学研究機構
Toshiyuki Imamura, RIKEN AICS

第7回筑波大学計算科学研究センターシンポジウム
19-20th of October, 2015, at Tsukuba University

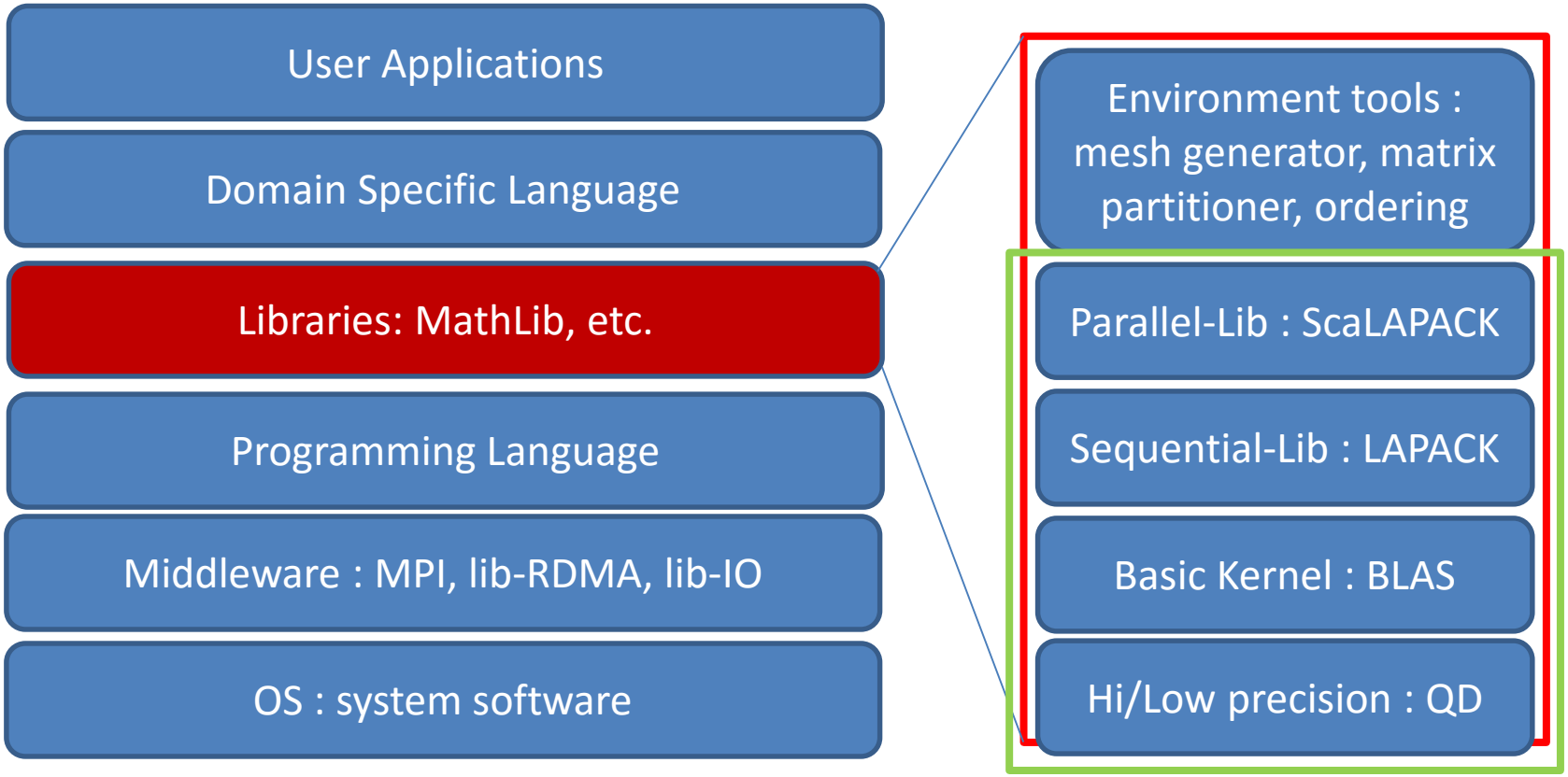
京
K computer

Outline

- 背景
  - Quick review for Mathlib and Co-design in this study
- 技術トレンドレビュー
- 何を進めているか?EigenExa開発を中心に
  - アプリケーション連携
  - CA for the EigenExa project
  - Co-designとしてなり立っているか

Mathematical (numerical) library

- Where is the position of Mathlib? System-software? Middleware? applications?



ユーザーによる特徴分け

- Top Runners
  - 大枠のコードは殆ど変えない
    - コードは定型で何年も使い続けるので、ソルバーに頼らなくてもよいくらい独自にチューニング
- Fluent Users who change their code often
  - プログラムを頻繁に変えるのでライブラリを利用
    - どのライブラリがよいのかを熟知している
- Beginners
  - ライブラリの存在を知らない？
    - 結果、生産性も悪く性能の出ないコードを流している

Way to Co-design

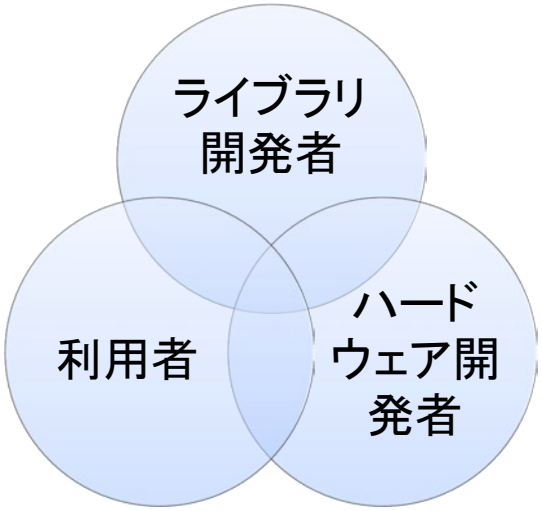
Codesign:

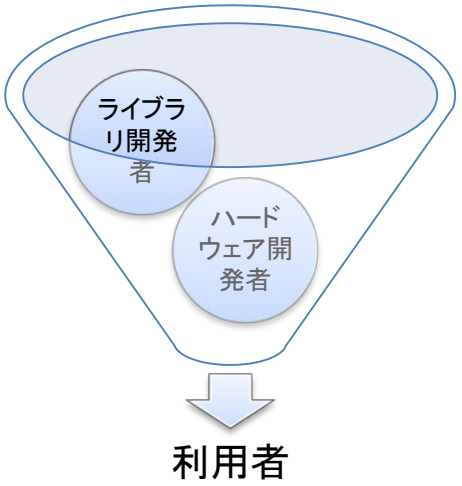
to design jointly (Collins Dictionary)

Lars ALBINSSON, "… The Co-Design Scenario should therefore *include also other interactions, services and staff* that might be part of the overall concept. *The wide scope of the CoDesign Scenario allows* the IT artifact to be designed in its context, and other stakeholders' perspectives to be considered, visualized and explored in the design process. …

Co-Design Scenarios *help people be more innovative* by letting everyone explore ideas. This helps *reduce the feeling of risk by giving a clearer understanding* of what the proposal really mean. It also allows a nuanced critical thinking, in that *objections can also be explored and help improve the design, rather than just rejecting it*. … "

Which is the Present Co-Design topology?





センター等では
利用者と開発者の間にセンター運用管理者がはいるため、なかなか三すくみのコデザインは難しい。
運用も含めた4すくみのコデザインは「ポスト京」開発プロジェクトで実現されている?

Our cases

- 実ユーザーの利用実態
  - どんなライブラリが使われているのか？
  - 問題の種類、規模
- 数値ライブラリの現状調査
  - ライブラリの能力の実態
  - 得意な問題
- 両者の知識共有
  - ハードウェア情報は運用もしくはベンダーから得て共有
  - ユーザと開発者でずれはないか？
  - ライブラリがターゲットとする問題規模の相違は？
- 機能改善と実応用

ヒアリング

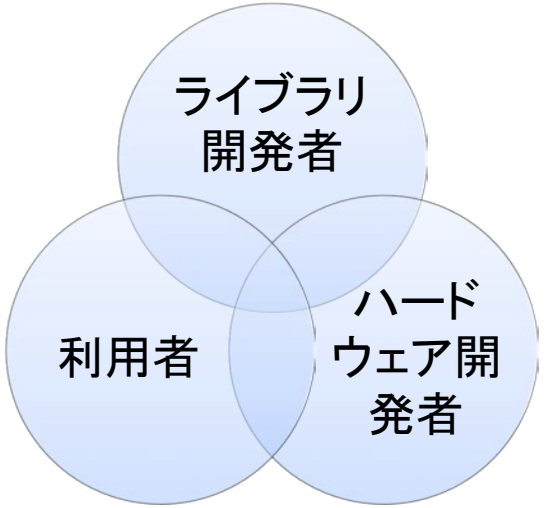
- Hearing from the Strategic programs for innovative research (SPIRE);
  - **Life Science, climate science, material science, industrial field, elementary science, etc.**

Summary of enquires:

- Frequently used calculation (algorithms) and numerical libraries:
  Basic Matrix calculation (GEMM, inversion) / BLAS, PBLAS, ScaLAPACK
  Eigenvalue / ScaLAPACK, EigenExa
  3D-FFT / custom-made, in-house codes, FFTW, FFTE
  Random Number generator / custom-made, built-in function
  System of linear eqs. / no use of general numerical library, Direct linear solver, in-house solver or embedded into the source code
- Not only real number but complex number
- Single-precision is applied to the part where accuracy is not required.
- For capacity computing, problem size is small, smaller than L2 cache.

Fortunately or Unfortunately,
top runners do not care about numerical libraries severely at the moment!

技術トレンド



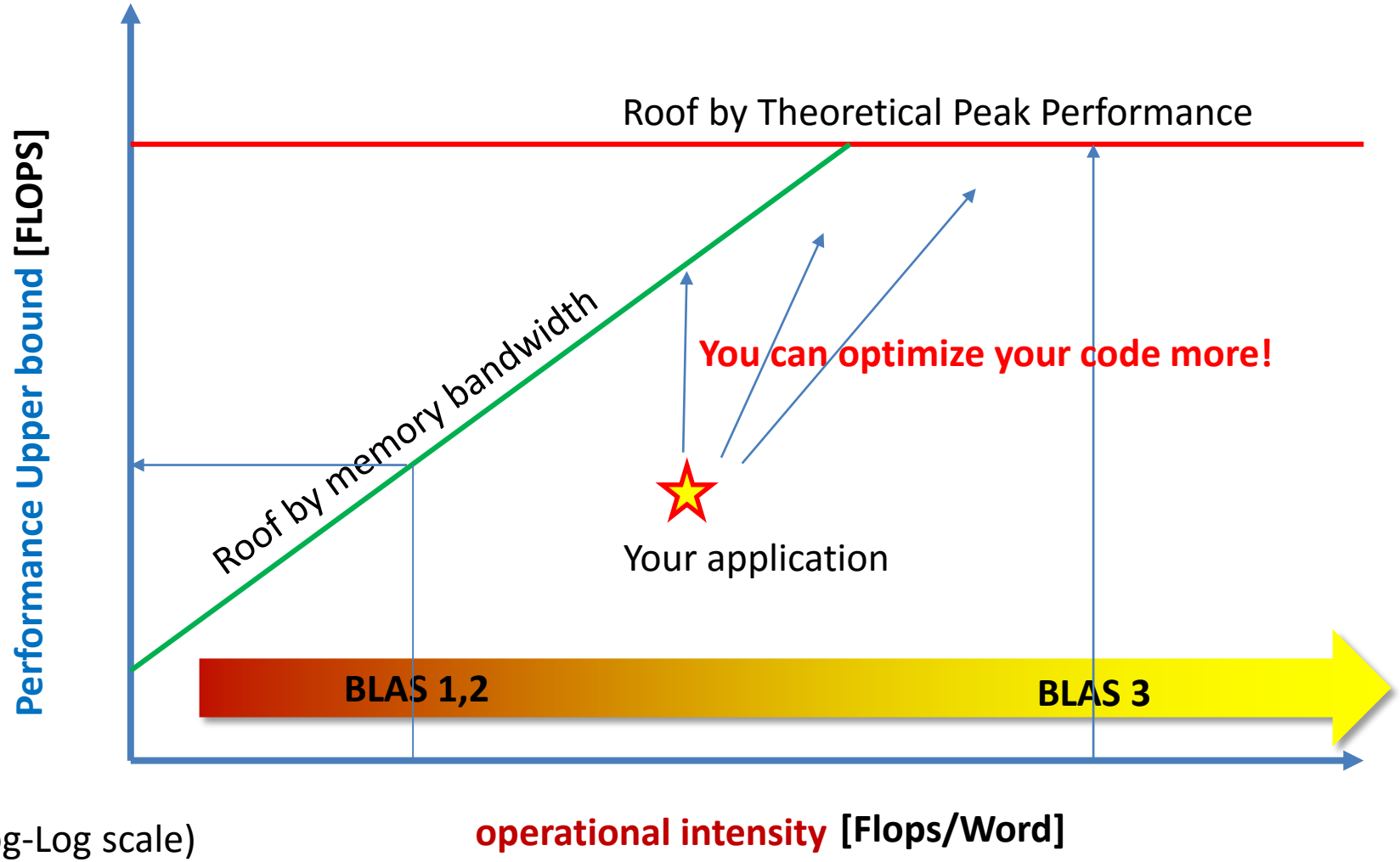
数値計算におけるトレンド

新しくもないが

- **ブロックアルゴリズム(Block algorithm) → cache awareness**
  - operational intensityの増加
  - GEMM (80-90%の理論性能ピーク)
- **タイリング(Tiling) → cache and memory hierarchy**
  - ブロックアルゴリズムの一種だが、**2次元ブロックかつデータレイアウトへの変更を伴う**.
    - A(1:NX,1:NY) → A(1:BX,1:BY, 1:NX/BX, 1:NY/BY)

      Submatrix is contiguously stored in memory
  - L1/L2キャッシュへの連続的なメモリアクセス、格納の保証
  - TLBミスの軽減

Roofline model



(Log-Log scale)

Recent trends of Optimization for LA

- **Communication avoiding (→ latency awareness)**
  - Communication is much **expensive** than any floating point operations appearing a sequential code.
  - Overlapping communication and calculation is well-known and effective, but it is difficult. Its possibility depends on data-dependency.
  - Communication avoidance is another approach to reduce communication overhead even if calculation cost increases.
  - Followings are successful algorithms
    - CAQR (Communication Avoiding QR decomposition)
    - MPK (Matrix Powers Kernel)

K computer

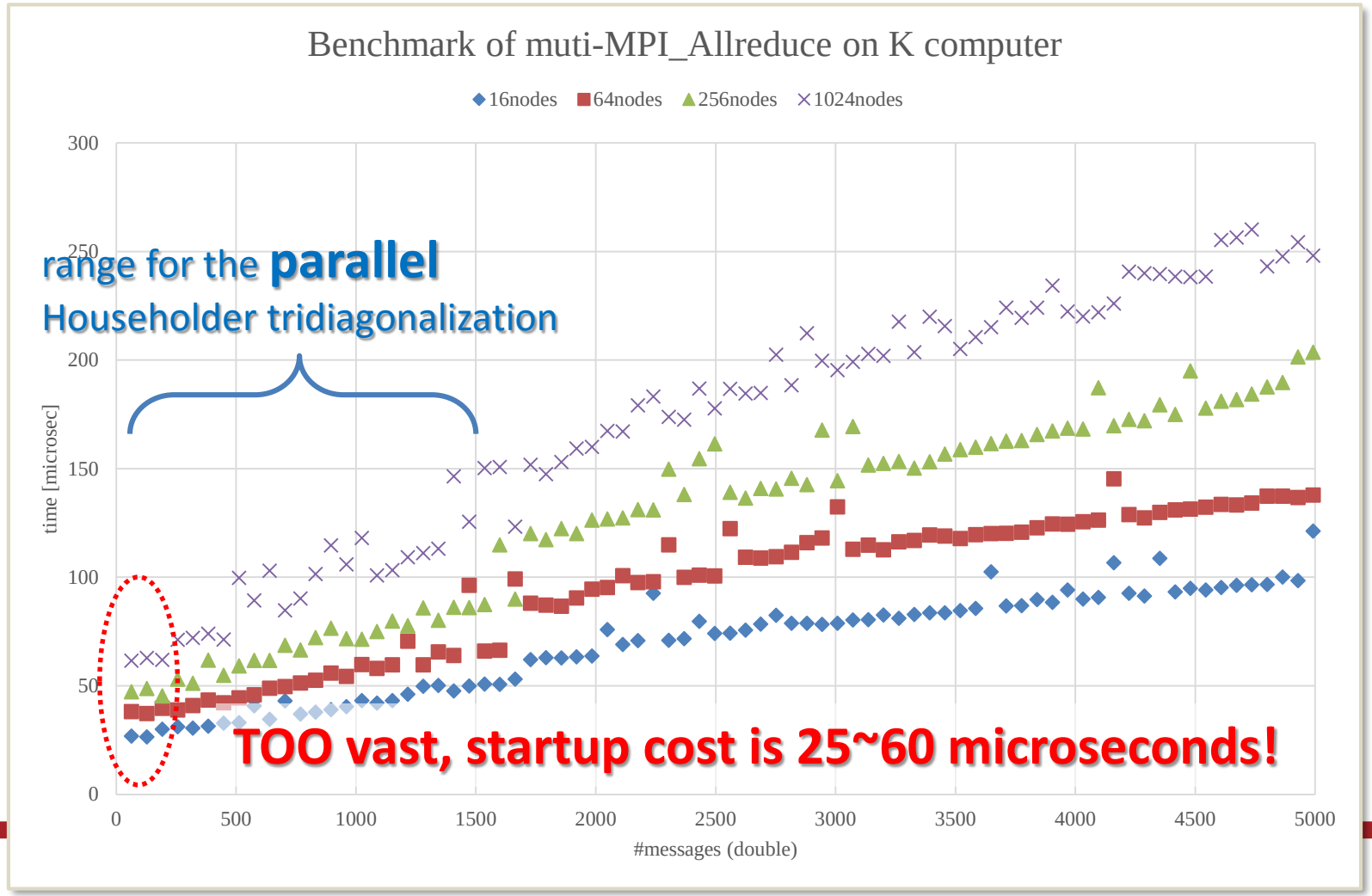
- K computer is the first **10PFLOPS** supercomputer system, and it was launched at September 2012 for the regular operation.
- Maximum **82,944** processors (SPARC64 VIIIfx, 2GHz*8cores/proc., totally 663,552cores on the system) interconnected with Tofu (6D torus/mesh)
- During the **monthly** special operation week (2 or 3days), we are permitted to submit full node jobs.

For further info. http://www.aics.riken.jp/en/k-computer/about/.

- So we must consider O(1)~O(100K) parallelism, very divergence.
- To conduct the numerical library group is very challenging, because Programming cost increases rather than the before K was developed.

Allreduce is an expensive operation

- One of our concerns of Householder tridiagonalization is… ‘It needs tons of MPI_Allreduce operations’.



Recent trends of Optimization for LA

- **Communication avoiding (→ latency awareness)**
  - Communication is much **expensive** than any floating point operations appearing a sequential code.
  - Overlapping communication and calculation is well-known and effective, but it is difficult. Its possibility depends on data-dependency.
  - Communication avoidance is another approach to reduce communication overhead even if additional calculations are induced.
  - Followings are successful algorithms
    - CAQR (Communication Avoiding QR decomposition)
    - MPK (Matrix Powers Kernel)

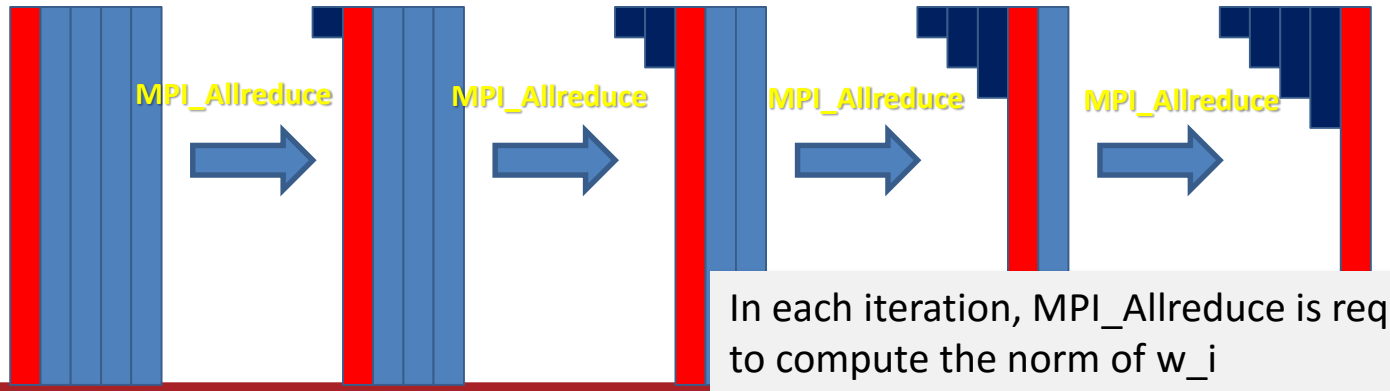
CAQR

- TSQR=QR decomposition algorithm for a Tall-Skinny matrix
  - W: m x b, m>>b (#rows is larger than #columns)

    W is distributed in a row-block fashion
- Algorithm 1: Normal Householder QR

$$Q = E_b$$
$$\textbf{for } i = 1 \text{ to } b \textbf{ do}$$
$$(u_i, \beta_i) = h(w_i, i)$$
$$[Q, W_{*,i+1:b}] = (I - \beta_i u_i u_i^T)[Q, W_{*,i+1:b}]$$

$$h(x,i) = \begin{cases} \left((0_{1:i-1}; x_{i:n}) + \text{sign}(\|x\|, x_i)e_i, \frac{2}{\|x\|^2}\right) \\ (0,1) \quad (\text{if } \|x\| = 0) \end{cases}$$



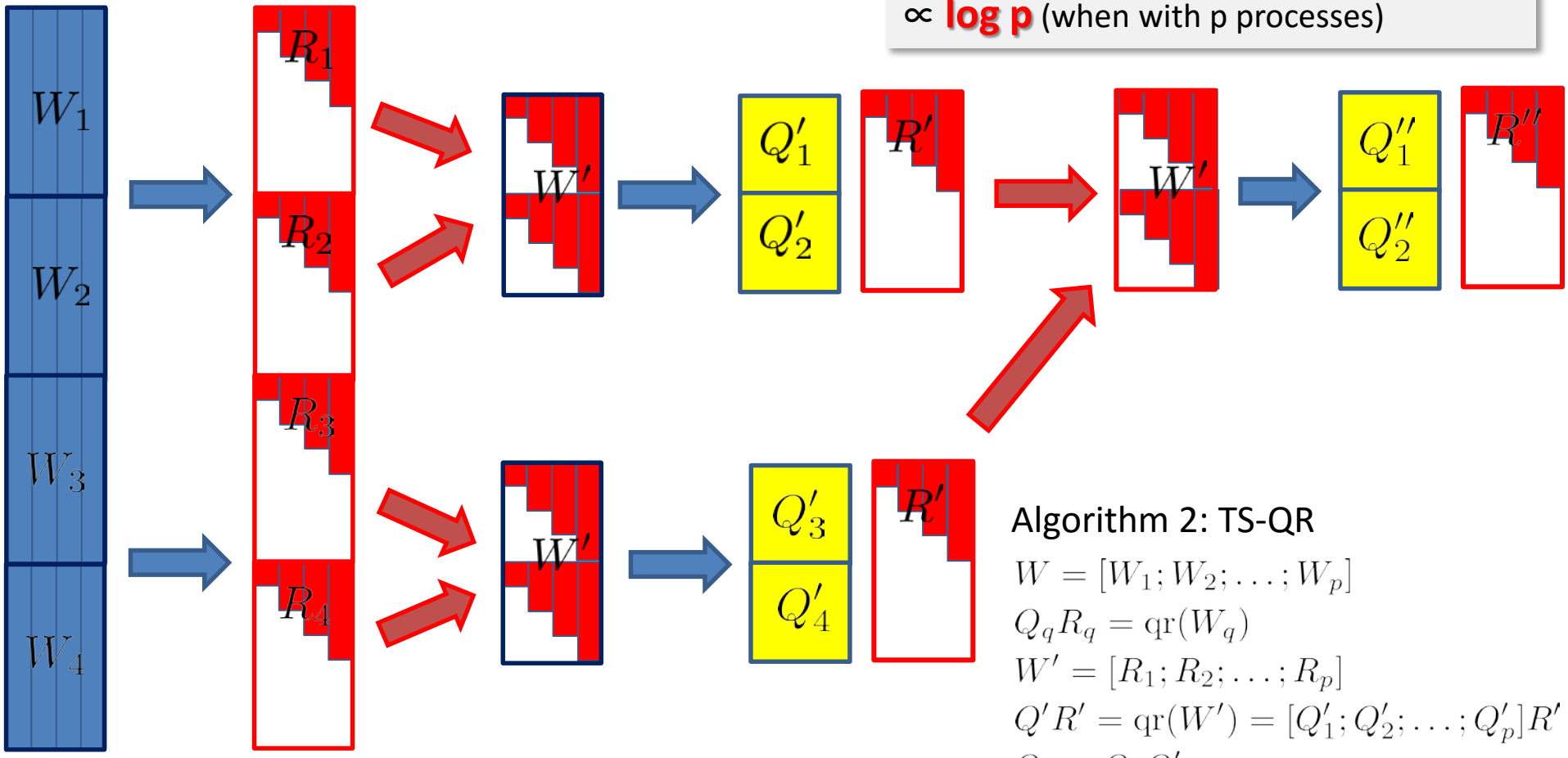
In each iteration, MPI_Allreduce is required to compute the norm of w_i

$\propto$ **b log p** (when with p processes)

CAQR

- In case of p=4

MPI_Allreduce or p-t-p comms are needed.
If binary-tree merge is applied,
$\propto$ **log p** (when with p processes)



Algorithm 2: TS-QR

$W = [W_1; W_2; \ldots; W_p]$
$Q_q R_q = \mathrm{qr}(W_q)$
$W' = [R_1; R_2; \ldots; R_p]$
$Q'R' = \mathrm{qr}(W') = [Q'_1; Q'_2; \ldots; Q'_p]R'$
$Q_q = Q_q Q'_q$
$Q = [Q_1; Q_2; \ldots; Q_p], R = R'$

Recent trends of Optimization for LA

- **Asynchronous algorithms (→ less sync. higher parallelism)**

  by E. Chow and A. Patel, Fine-grained Parallel Incomplete LU Factorization, SIAM J. Sci. Comp., 37, pp. C169-C193 (2015).

  Preconditioning by Incomplete LU Factorization

  → System of non-linear equations by asynchronous Element-by-element updates

$$A = L^{\top} U \rightarrow a_{ij} - \sum l_{ik} u_{kj} = 0$$

  → $F(x^{(k-1)})=x^{(k)}$, converged by the fixed-point scheme

  → If an iterative method has a structure of contraction mapping, the method converges.

- Iterative method for solving a linear equation by hybrid Gauss-Seidel+Jacobi is also available in an asynchronous fashion, and was discussed in early 1990-2000s.

  → Old and forgotten algorithm might be …

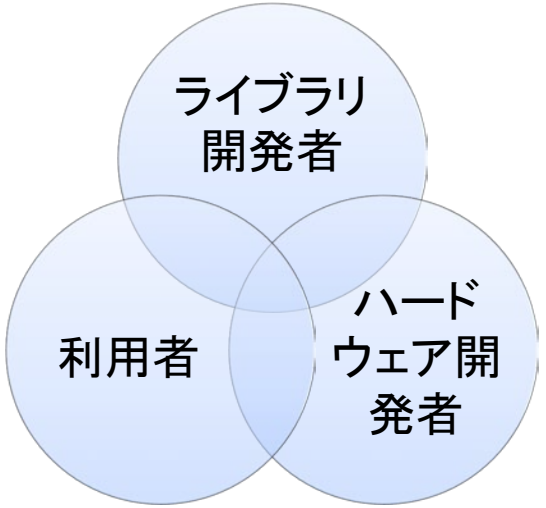
Summary of Optimization for NA

- **Block algorithm**
- **Tiling algorithm**
  - They are intended for ‘cache/memory hierarchy’
- **Communication Avoiding algorithm**
  - Latency awareness
  - A new principle like recurrent QR factorization is quite effective. Even the calculation cost increases, we can pay the additional cost for the CA-based approach.

Similar issues “Synchronous Avoiding”, “Synchronous Reducing”, “Communication Reducing”, also other concept of “Automatic-tuning”, are thought as helpful techniques.

- **Asynchronous algorithm / task schedule**
  - For higher parallelism and scalability
  - It conceals global synchronization costs.
  - (Local or Remote) Atomic operations will be required.
  - DAG-based approach in the PLASMA & DPLASM project
- **Reproducibility**
- **Resiliency**

What are we going to do ?



EigenExa Project

- The standard eigenvalue solver for real-symmetric dense matrices:

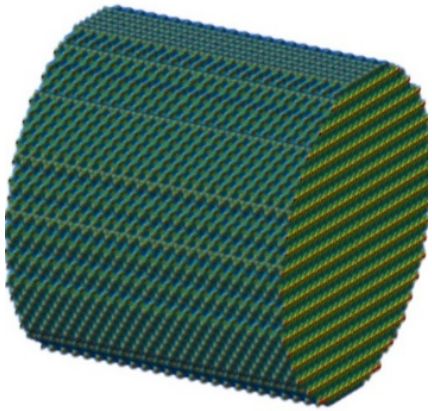
$$Ax = \lambda x$$

- Two big trends in HPC Numerical Linear Algebra are applied
  1. **Block algorithm**
     - Reduce memory transfer to overcome the wall of memory
  2. **Communication Avoiding and asynchronous algorithm**
     - Reduce times of data communication to overcome the wall of network latency
     - Consequently, Block algorithm results in CA

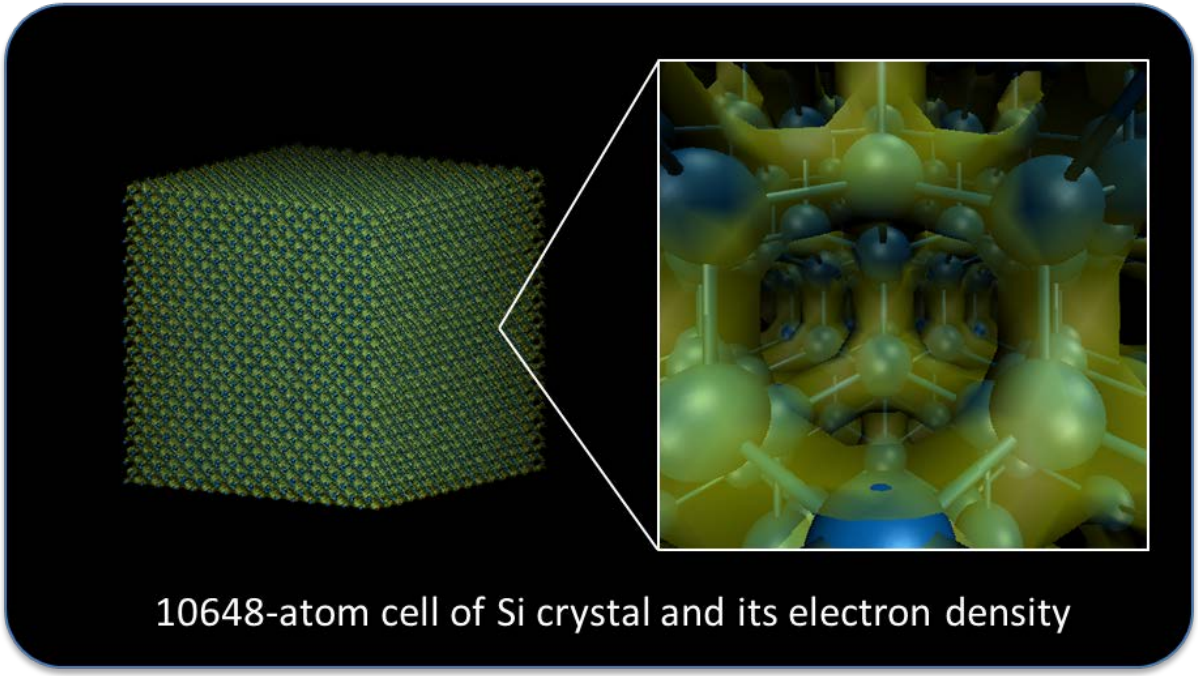
http://www.aics.riken.jp/labs/lpnctrt/EigenExa_e.html

A Large Simulation code developed at AICS

- Grand challenge applications like a nano-material-science simulation demand a high-performance eigenvalue solver.



Silicon nanowire



RSDFT(Real Space Density Function Theory), by Y.Hasegawa, et al @AICS.RIKEN, GBP winner, SC2011

A Large Simulation code developed at AICS

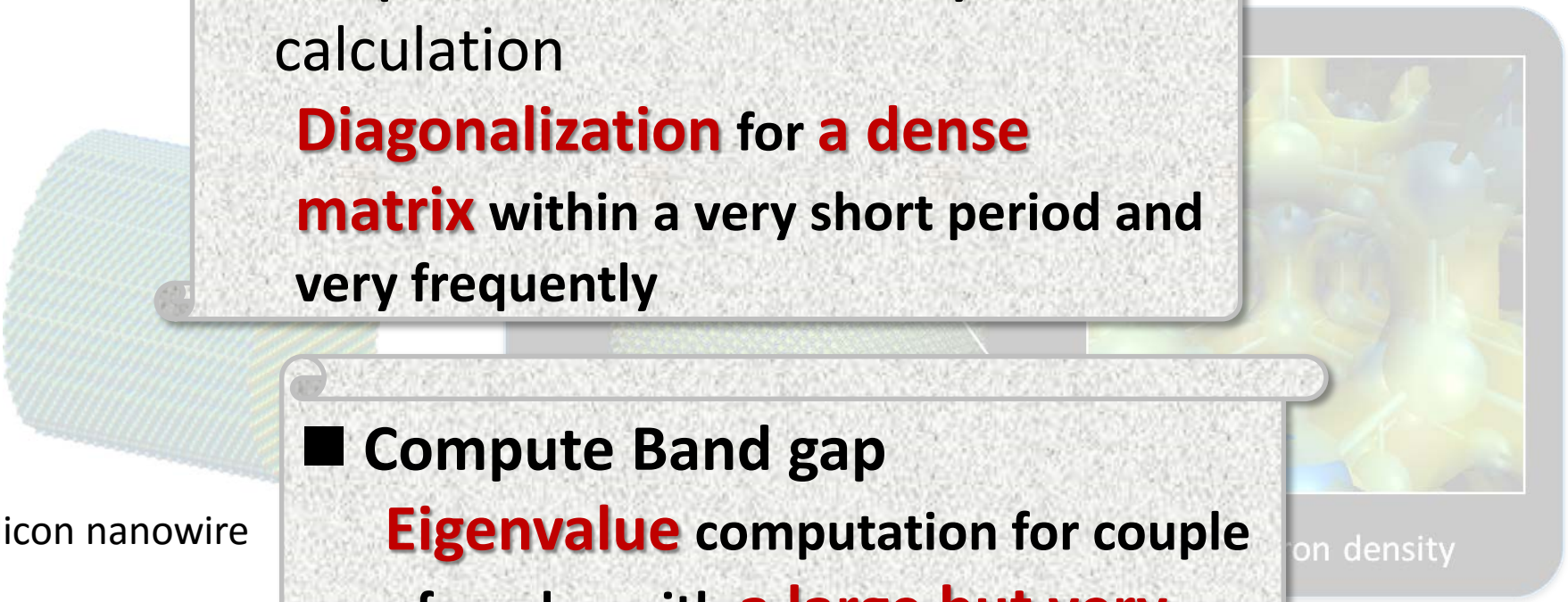
- Grand challenge applications like a nano-material-science simulation demand a high-performance eigenvalue solver.

■ **SCF(Self-Consistent Field)** calculation

Diagonalization for a dense matrix within a very short period and very frequently

■ **Compute Band gap**

Eigenvalue computation for couple of modes with a large but very sparse matrix



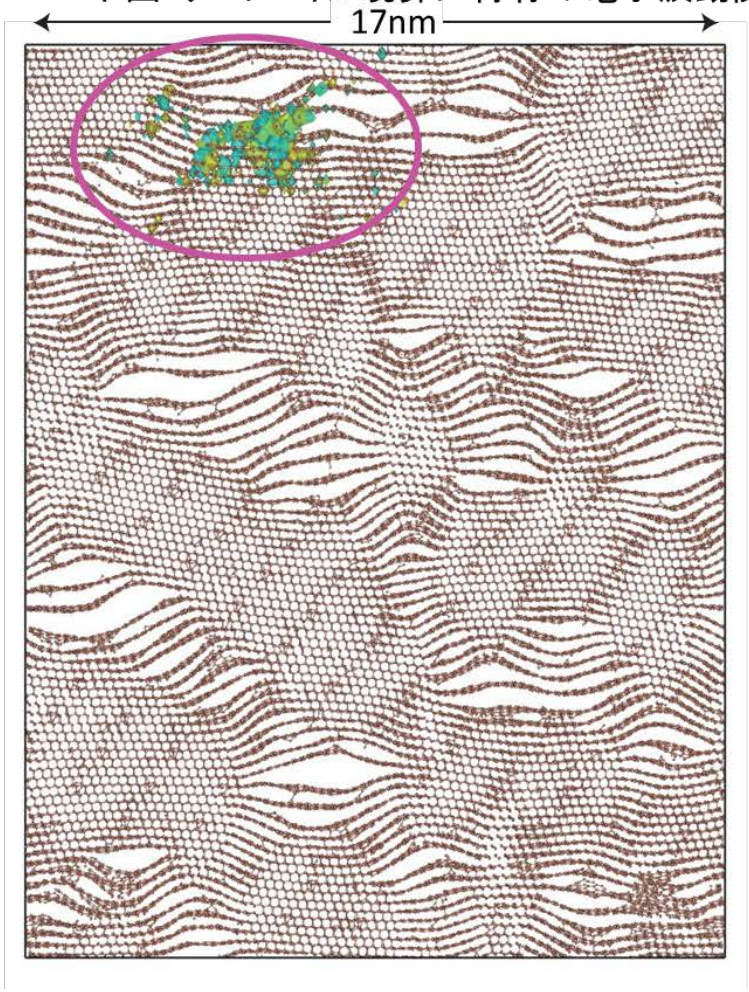
Silicon nanowire

awa, et al

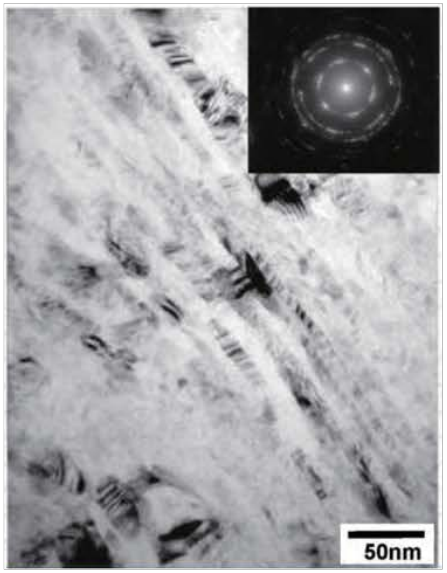
EigenExaの利用例：物質科学分野

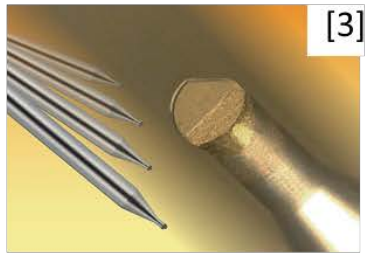
井町宏人，星健夫（鳥取大，JST-CREST）

・大規模電子状態計算コードELSES(http://www.elses.jp/)との接続
・応用：ナノ多結晶ダイヤモンド(＊)研究への応用
　⇒下図：ナノドメイン境界に特有の電子波動関数（M=43万次元行列における固有ベクトル）



(＊)ナノ多結晶ダイヤモンド(超強度材料)
[1] (合成@愛媛大)
T. Irifune, et al. , Nature 421, 599（2003)
[2] (理論研究) T. Hoshi, et al.,
J. Phys. Soc. Jpn. 82, 023710 (2013)
[3] 製品化(住友電工, 2012)



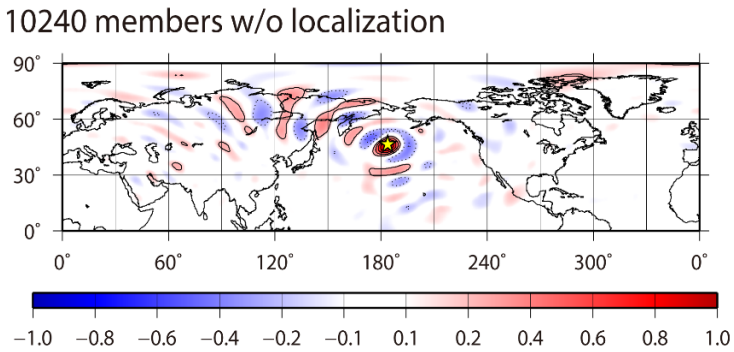


「LETKF」による世界最大規模の全球大気のアンサンブルデータ同化

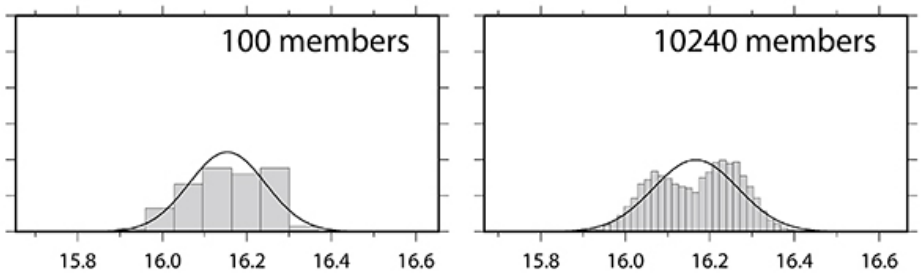
アンサンブルデータ同化システム「LETKF」の固有値計算部分に、EigenExaを組み込むことで、１回のLETKFの計算を125分から15分へ、約8倍高速化。さらに、EigenExaとの相乗効果により、4,608ノード（約590テラFLOPS。京全体の約20分の１）を使って、理論ピーク性能比 で44%超にあたる263テラFLOPSという極めて高い実行効率を達成。

これにより、これまでのアンサンブルデータ同化で用いられてきた100個程度以下のアンサンブルを大幅に上回る10,240個のアンサンブルで3週間分という世界最大規模の全球大気のアンサンブルデータ同化を行うことに成功した。

本研究により、例えば日本から１万km以上離れた場所の観測データをもとに、日本の大気状態の推定精度を向上する可能性が明らかになり、天気予報シミュレーションの精度向上が期待される。
また、これまでの100個のアンサンブルでは確認できなかった大気状態のばらつきが非ガウス性を持つことを確認しており、このような非ガウス性を考慮したさらに高度なデータ同化手法の開発にも期待される。



アンサンブルデータ同化による18日目の水蒸気量の相関マップ (理化学研究所プレスリリースより)
10,240個のアンサンブルを使うと、北部太平洋にある黄色い星の場所（図の中心付近）にある観測データの影響が、遠くロシア西部（図の左上）にまで及ぶことが分かる。

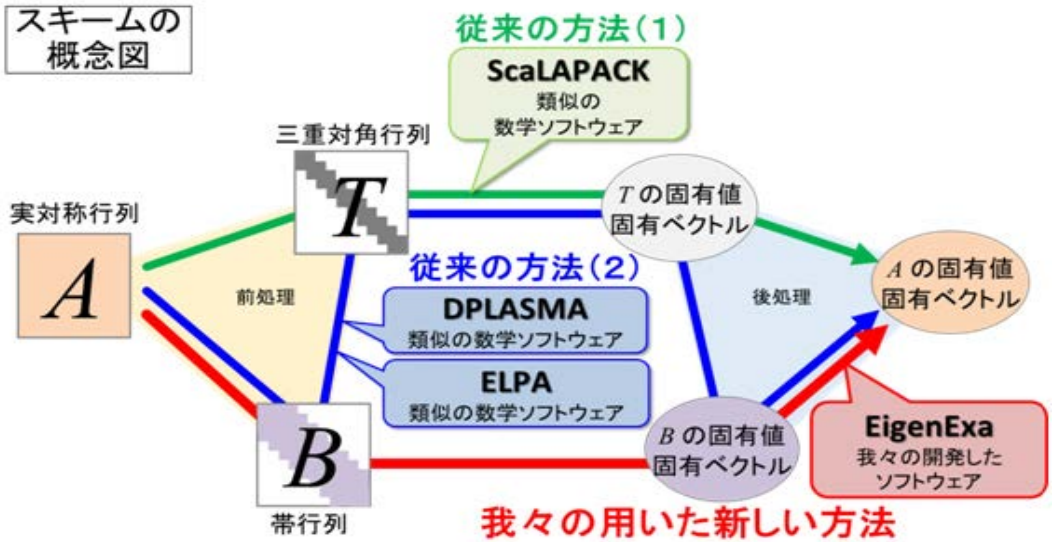


アンサンブルデータ同化による18日目の水蒸気量の誤差を表すヒストグラム
ある地点のある時刻における水蒸気量の誤差を表すヒストグラム。実線はガウス分布（正規分布）関数を示しているが、このような大気状態のばらつきの非ガウス性を直接確認するのは初めてである。

「京」の計算能力を引き出す新開発ソフトウェア「EigenExa（アイゲンエクサ）」

AICS大規模並列数値計算技術研究チームが開発を進める「数値計算パッケージソフト:KMATHLIB」に組み込まれる一要素として「EigenExa（アイゲンエクサ）」を開発し、平成25年8月より利用者向けソフトウェアとして公開している。

京コンピュータの全系(82944ノード)を用いて100万次元規模の乱数行列の対角化を試験したところ、100万次元行列の対角化を約1時間で実行し、性能面でも1.7PFLOPSを記録し理論ピーク性能の16%に達する高速計算を実現。



EigenExaの利用事例のあるソフトウェア等の例
○Platypus QM/MM:酵素反応機構などの生体高分子の精密解析ソフトウェア
○RSDFT:結晶・界面・分子などの広範な物理系に対する密度汎関数法による電子状態計算ソフトウェア
○PHASE:密度汎関数理論に基づく物質中の電子状態の計算ソフトウェア
○ELSES:大規模系の電子構造計算、分子動力学計算による様々なプロセスのシミュレーションソフトウェア
○NTChem:数千原子分子系の第一原理電子状態計算や数百原子分子系の化学反応過程追跡計算ソフトウェア
○Rokko:並列ライブラリ等を利用する大規模密行列、大規模疎行列の対角化を統一的に利用するためのライブラリ
○LETKF:並列計算性能を重視したアンサンブルデータ同化手法
○POD:多次元データから低次元成分を抽出する解析法

EigenExa Project

- The standard eigenvalue solver for real-symmetric dense matrices:

$$Ax = \lambda x$$

- Two big trends in HPC Numerical Linear Algebra are applied
  1. **Block algorithm**
     - Reduce memory transfer to overcome the wall of memory
  2. **Communication Avoiding and asynchronous algorithm**
     - Reduce times of data communication to overcome the wall of network latency
     - Consequently, Block algorithm results in CA

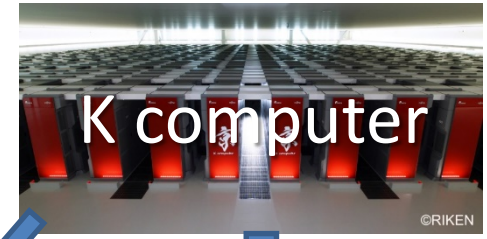
http://www.aics.riken.jp/labs/lpnctrt/EigenExa_e.html

EigenExa Project

- Poritng the EigenExa from K to other systems.

Sorry photo is ES2

T2K
PC cluster



Exa-scale
System??

SX-ACE @ U. Osaka

Photo from:
http://www.hpc.cmc.osaka-u.ac.jp/sx_ace_intro/

BlueGene/Q @ Juelich

Photo from:
http://www.fz-juelich.de/SharedDocs/Bilder/IAS/JSC/EN/galeries/JUQUEEN/juqueen-full.jpg

EigenExa Project

- The standard eigenvalue solver for real-symmetric dense matrices:

$$Ax = \lambda x$$

- Two big trends in HPC Numerical Linear Algebra are applied

1. Block algorithm

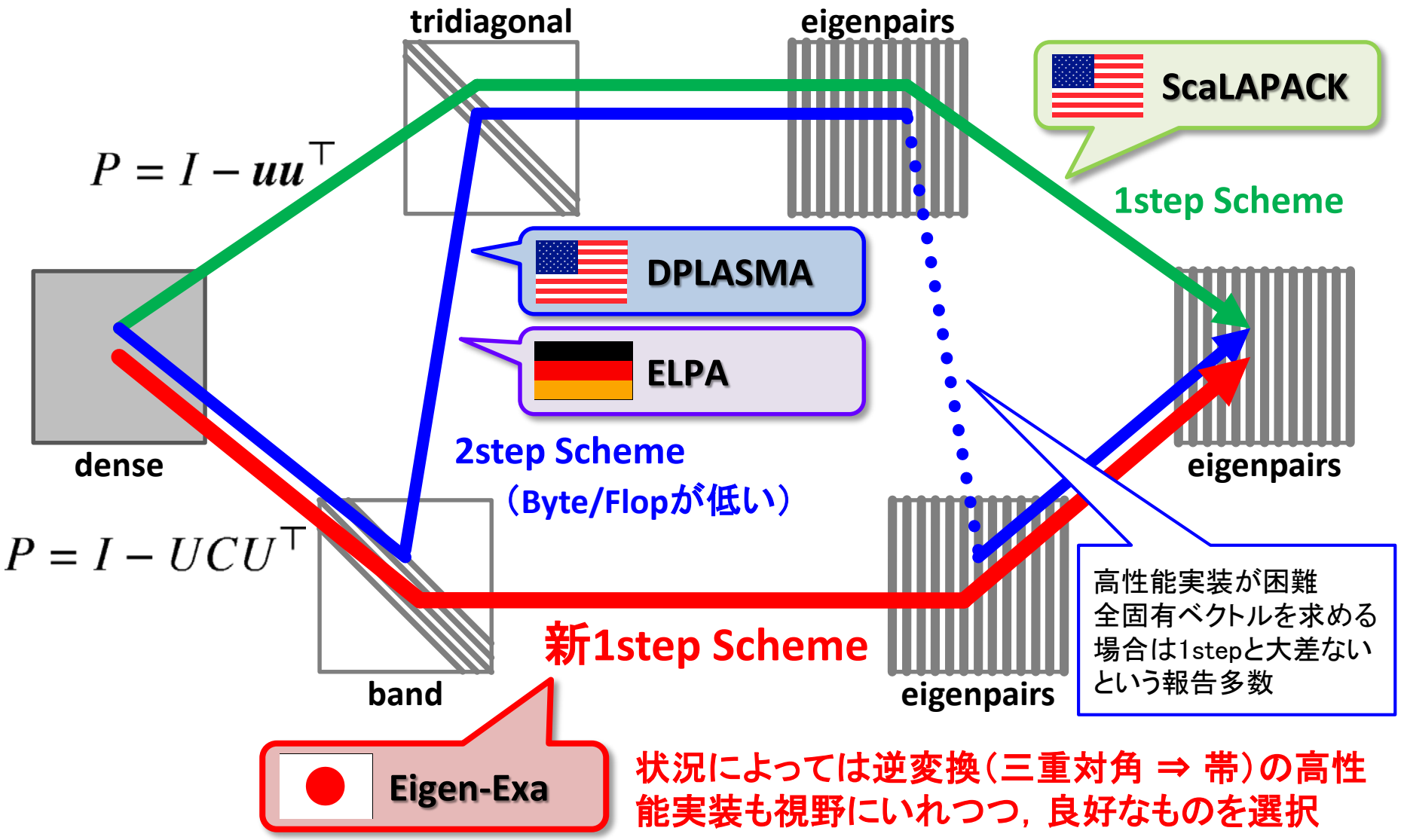
  - Reduce memory transfer to overcome the wall of memory

2. Communication Avoiding and asynchronous algorithm

  - Reduce times of data communication to overcome the wall of network latency
  - Consequently, Block algorithm results in CA

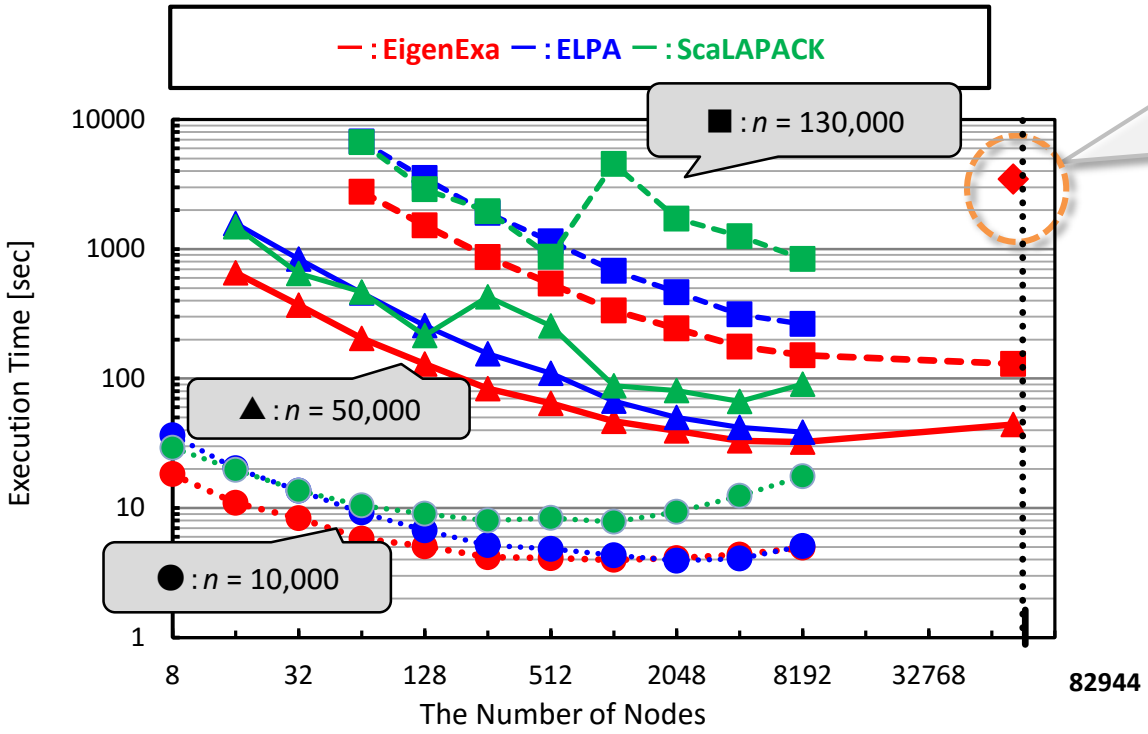
http://www.aics.riken.jp/labs/lpnctrt/EigenExa_e.html

密行列固有値エンジンEigenExa



World Largest Dense Eigenvalue Computation

- We have successfully solved a world largest-scale dense eigenvalue problem (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). It is the world highest performance for solving an eigenvalue problem of a dense matrix.



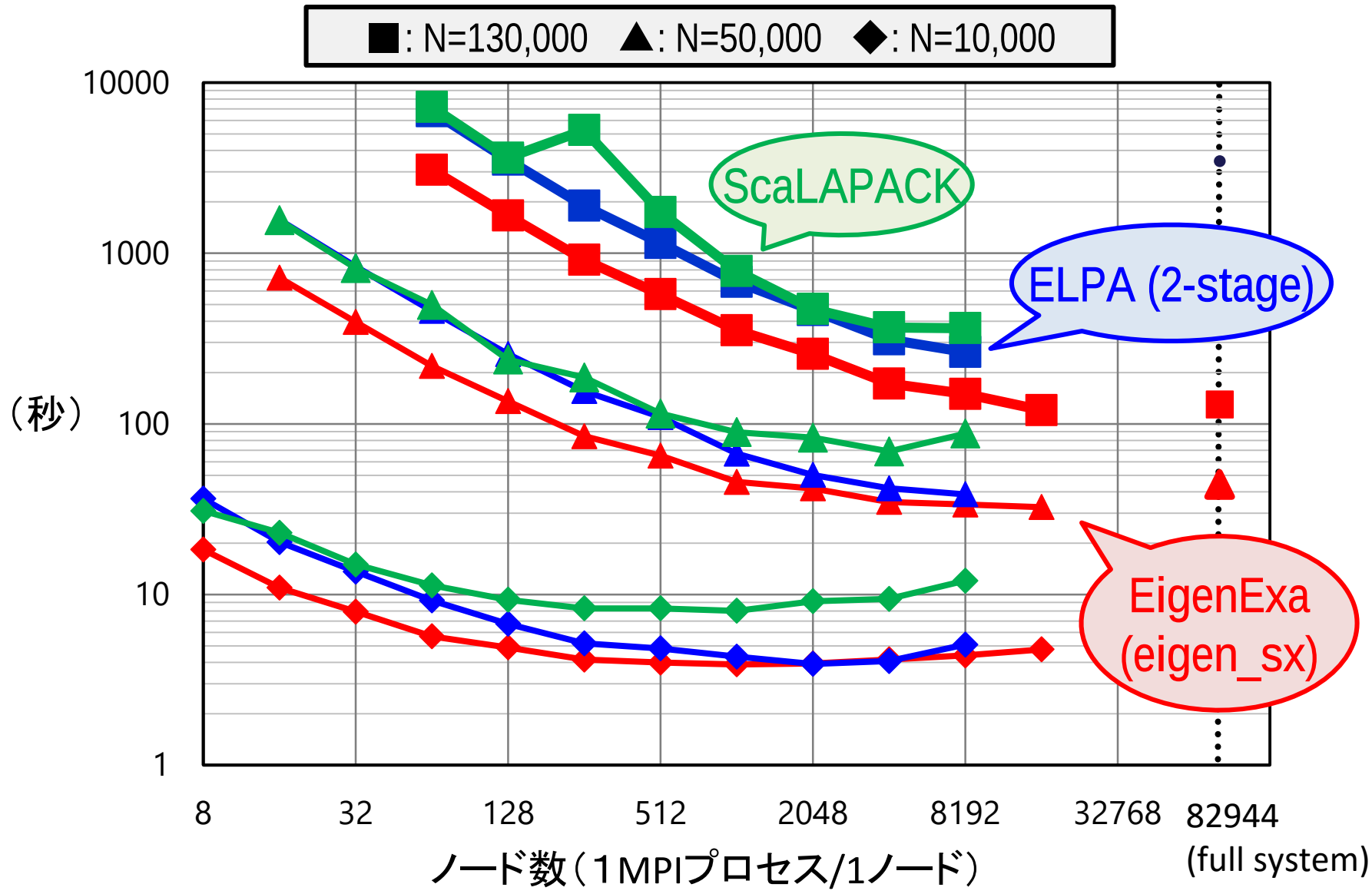
◆ : n = 1,000,000

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

- ✓ 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)

京コンピュータにおけるEigenExaの性能 （2015年最新データ）



物質科学分野での高性能利用技術

– 数理的ミドルウェアとして複合ソルバを開発・共有（星）

一般化固有値問題ライブラリの複合化:

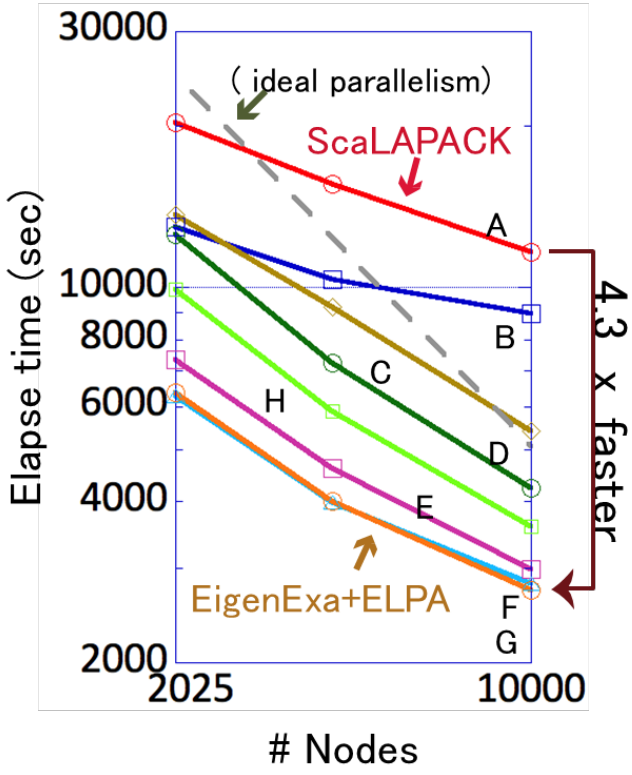
: ScaLAPACK, EigenExa, ELPA(http://elpa.rzg.mpg.de/)

H. Imachi and T. Hoshi, in press
(preprint: http://arxiv.org/abs/1504.06443)

計算例:京での強スケーリングテスト(他に、Oakleafやintel-CPUマシンでのテストもあり)

- 最大100万次元(フルノード計算)まで。
- 43万次元行列での強スケーリング性
  種々の複合ソルバーの比較

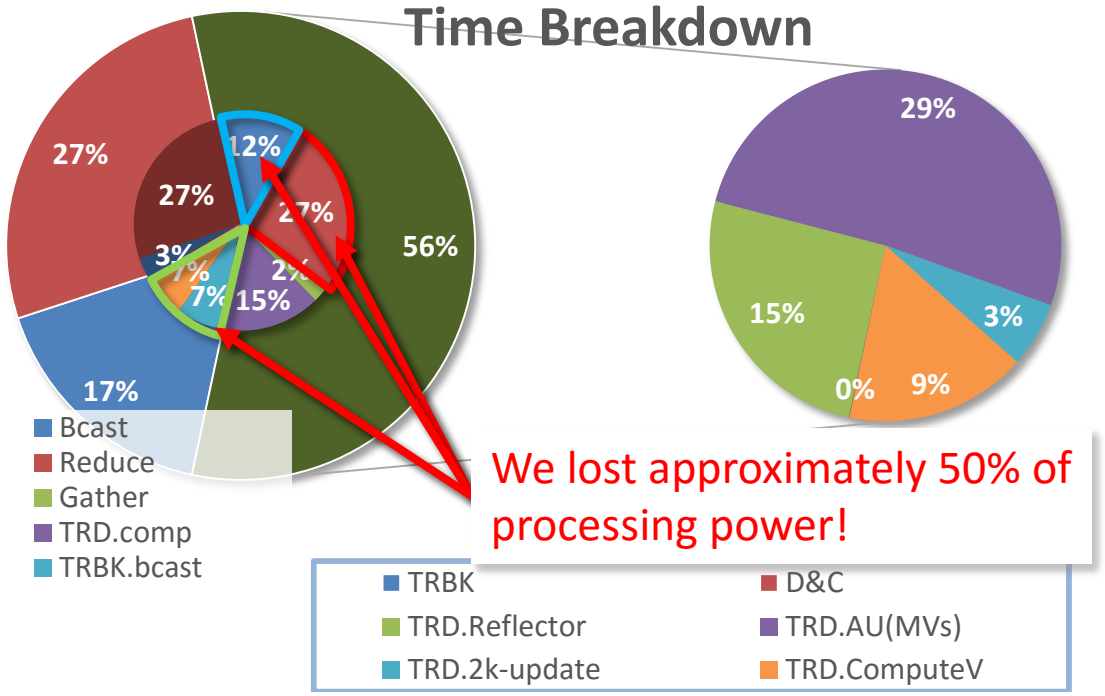
	SEP solver	Reducer
A	ScaLAPACK	ScaLAPACK
B	EigenExa(sx)	ScaLAPACK
C	ScaLAPACK	ELPA
D	ELPA(2)	ELPA
E	ELPA(1)	ELPA
F	EigenExa(s)	ELPA
G	EigenExa(sx)	ELPA
H	EigenExa(sx)	EigenExa(sx)



Why are we concerned about communication cost?

```
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 5417097.565200453
GFLOPS
D&C 921.8044028282166 3130319.580211733 GFLOPS
TRBAK= 573.9026420116425
COMM= 533.7601048946381
   573.9026420116425 3484911.644577213 GFLOPS
   182.3303561210632 5484550.248648792 GFLOPS
   152.0370917320251 6577342.335399065 GFLOPS
   0.1022961139678955 7.379654884338379
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]
```

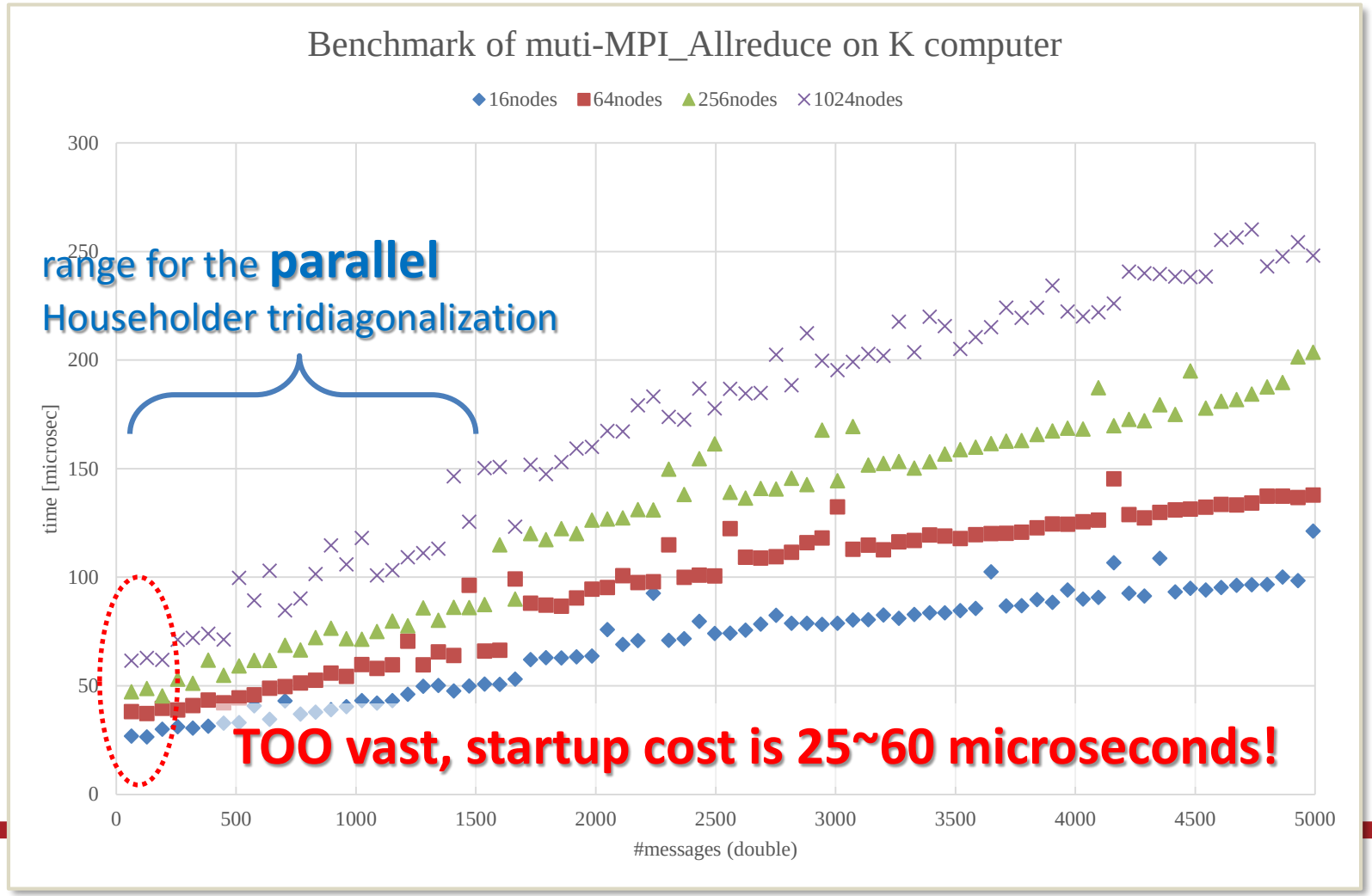
The World Largest Dense Eigenvalue Computation



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

Allreduce is an expensive operation

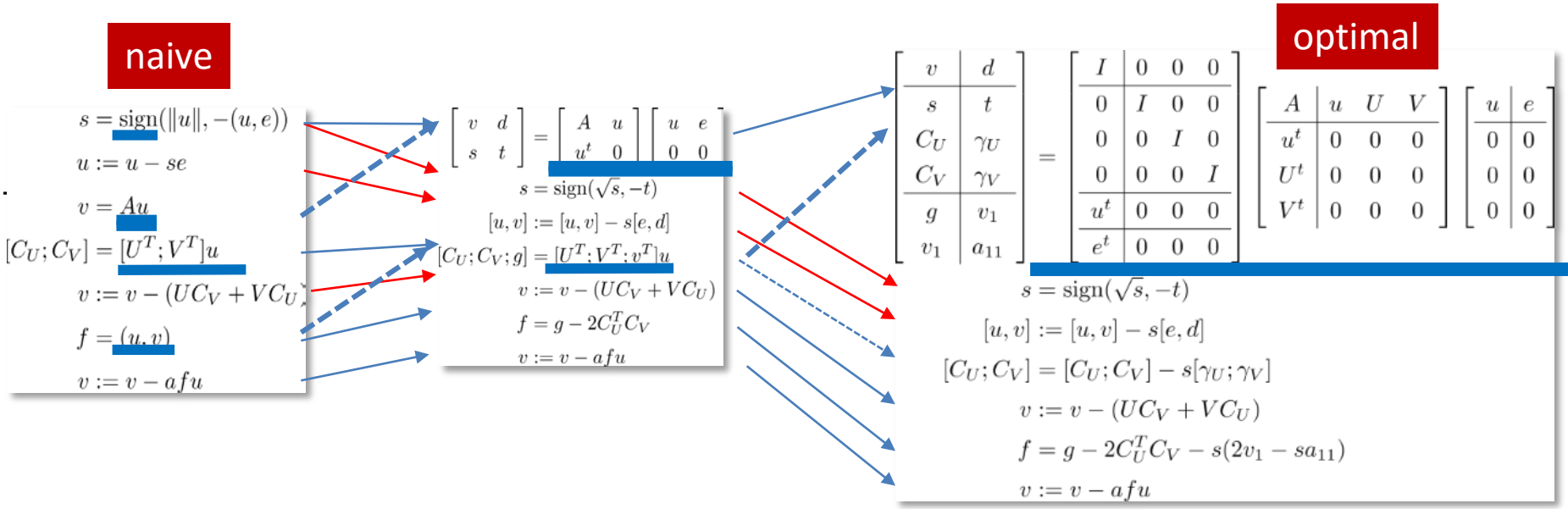
- One of our concerns of Householder tridiagonalization is… ‘It needs tons of MPI_Allreduce operations’.



CA for Householder Tridiagonalization?

- Communication Avoiding for Householder transformation not by the idea like CAQR
  - The key idea here is that to take advantage of the distributive law, data transform, esp. collective communications are relieved from data dependency and make it possible to combine them with other communications simultaneously.

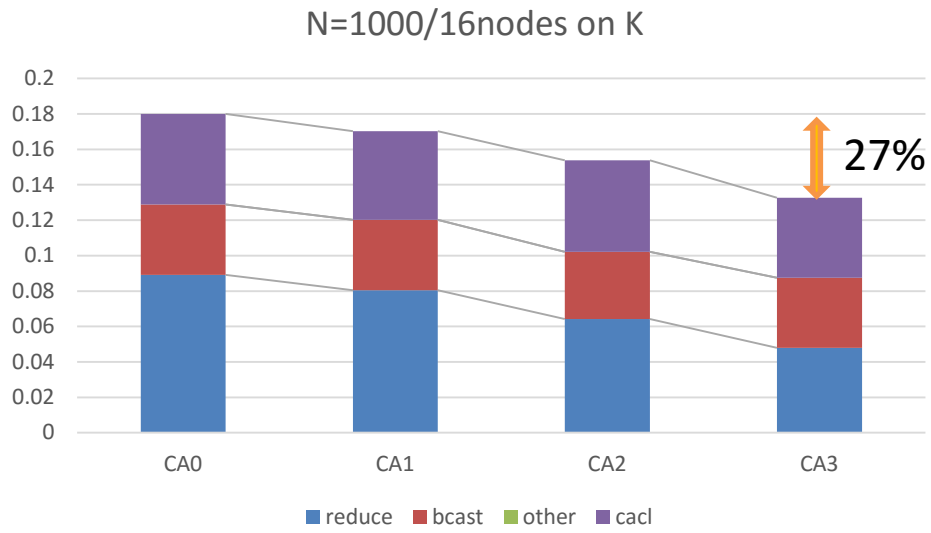
Principles : Distributive Law && Exchange order && Introducing the correction terms && Merge of couple of collective ops.

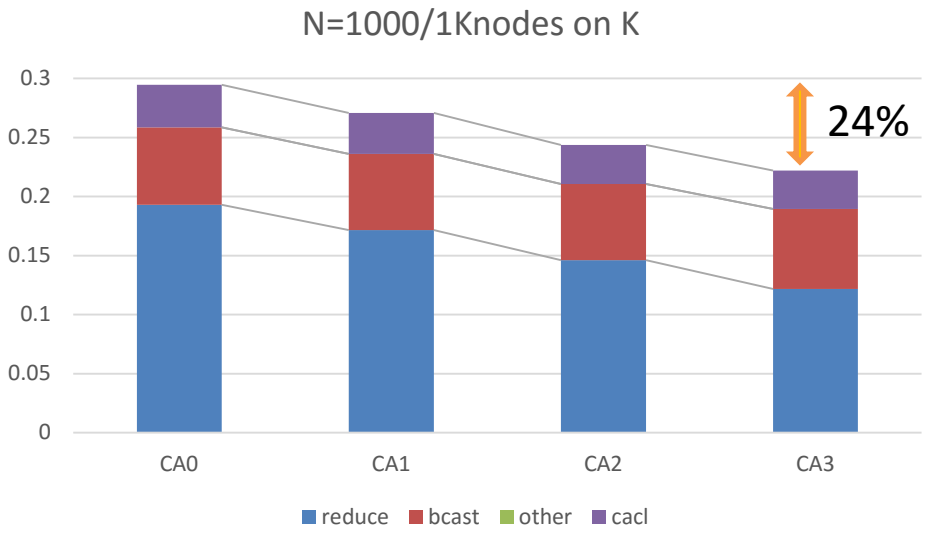


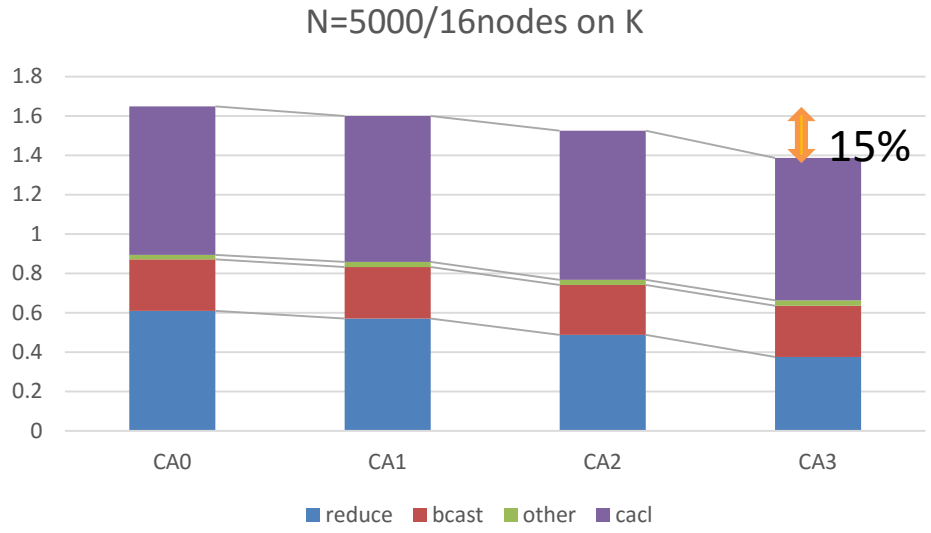
Preliminary results on K computer

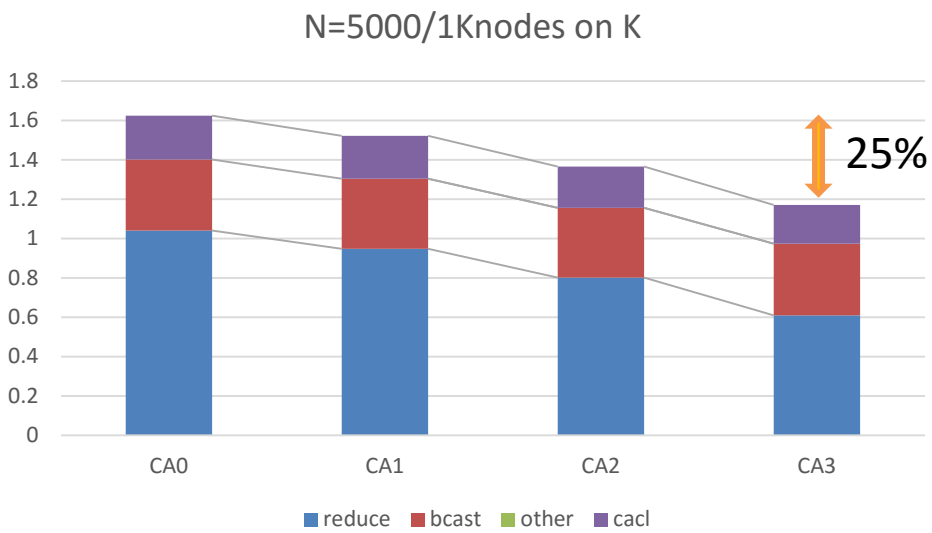
We achieved a 15~25% time-reduction by only communication avoidance!

Reduction of broadcast by CA is future work!!!

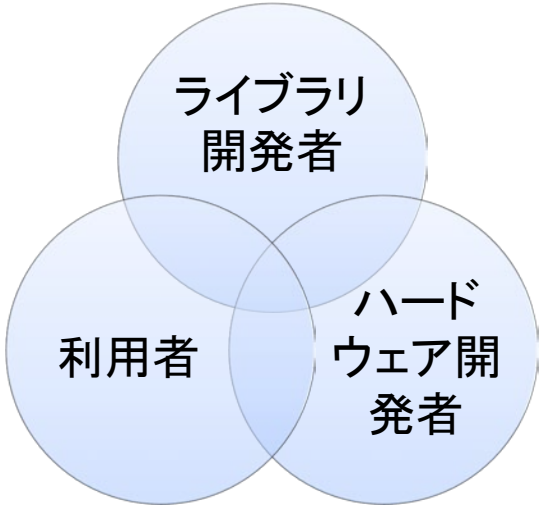








まとめ：
コデザインとして見返し



まとめ：固有値ソルバ

- Dense Eigenvalue Solver
  - EigenExa: Kコンピュータでは数万次元の対角化が10～数秒オーダーで計算可能
  - 一般化固有値問題

  Codesign → 特異値計算への応用
  小規模問題の需要
  テンソルの特異値
  非対称問題への対応
- Sparse Eigen Solver
  - 櫻井先生（筑波大）のSS法をベースとしたz-Paresが有効
  - 内部固有値問題も求解可能

まとめ：数値計算ライブラリ

- その他の解法についても情報提供
  - FFT
    - FFTE(高橋先生筑波大)
    - FFTW
    - ベンダー提供

    上記の多くは空間分割データは2軸でなくてはならない、より自然な並列化である3軸分割向けのFFTも存在して高並列
  - 乱数生成器
    - メルセンヌツイスター

    理研が「京」向けに移植済み、SIMD機能を使わなければ「京」以外での分散並列使用可能

本公演のまとめ

- **Codesign for Numerical libraries**
- **Easier Optimization Recipe**
  - Level3 BLAS(ブロックアルゴリズム)が効果的
  - Roofline model
- **The latest topics**
  - Memory wall と Netowork wall
- EigenExa:
  - World largest scale computing
  - アプリとの連携 → Co-design
- 数値計算ライブラリ全般
  - 本公演がCo-Designの一助になったでしょうか？

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

謝辞/Acknowledgment

- My team members:
  - Yusuke Hirota, Daichi Mukunoki, Masaharu Ohi
  - Prof. Takeshi Fukaya(U. Hokkaido)
  - Prof. Daisuke Takahashi(U. Tsukuba)
  - Prof. Franz Franchetti (CMU)
- JST CREST collaborators
  - Prof. Tetsuya Sakurai (U. Tsukuba)
  - Prof. Yusaku Yamamoto (UEC)
  - Prof. Tateo Hoshi (Tottori U.)
  - Susumu Yamada, Masahiko Machida, Narimasa Sasa (JAEA)
- RIKEN AICS members:
  - Miyoshi, Kondo, Minami, Kuroda, Hasegawa(Alumni), etc.
- All attendees jointing here