jh190015-NAJ

Developing Accuracy Assured High Performance Numerical Libraries for Eigenproblems

Takahiro Katagiri

（Nagoya University）

Final Report for FY2019
Joint Usage/Research Center for Interdisciplinary Large-scale Information Infrastructures
JHPCN: 学際大規模情報基盤共同利用・共同研究拠点　第12回 シンポジウム
2020年7月9日(木) オンライン開催

Significance of Research (1/2)

- Eigenproblem is one of essential numerical problems for several numerical simulations. Its accuracy, however, is not well-assured in many conventional numerical computations.
- Basic Linear Algebra Subprograms (BLAS) is a frequently used to perform linear algebra computations. Ensuring the accuracy of the computational results of BLAS operations is a still crucial problem now. Even in solving linear equations using LAPACK is also a typical example, because LAPACK is rich in BLAS operations, especially matrix-matrix multiplication (MMM) operations for solving linear equations.

Significance of Research (2/2)

- We focus on the following three topics:
  1. Developing an accuracy assured numerical libraries for eigenproblems;
  2. Development of high-performance implementation and AT technology for the developed accuracy assured numerical libraries;
  3. Discussing an extension for non-liner problems based on obtained knowledge of accuracy assured algorithms.

Roles of Project Members

- Prof. Katagiri: High-performance implementation of Osaki method for recent multicore CPUs, and applying auto-tuning technologies.
- Prof. Hwang: Non-linear algorithms for actual engineering problems.
- Dr. Marques: Algorithms and implementations for eigenproblem.
- Prof. Nakajima: Sparse iterative algorithms for liner equation solvers, such as parallel preconditioners.
- Prof. Ogita: Iterative refinement algorithm to assure accuracy of real symmetric eigenproblem.
- Prof. Ohshima: GPGPU implementations.
- Prof. Ozaki: Accurate MMM algorithm (Ozaki method)
- Prof. Wang: Eigenvalue algorithms for actual engineering problems.

Research Plan

- This proposal is planned as three year's. FY2019 is the first year.
- The Year 1st (FY2019):
  - Topic 1: Performance evaluation of high-performance implementations for UNC-HPC libraries between multi-core and many-core CPUs and a GPU.
  - Topic 2: Designing accuracy assured libraries for real symmetric eigenproblem.
  - Topic 3: Discussing extension to non-linear problems.

TOPIC 1: PERFORMANCE EVALUATION OF HIGH-PERFORMANCE IMPLEMENTATIONS FOR UNC-HPC LIBRARIES BETWEEN MULTI-CORE AND MANY-CORE CPUS AND A GPU.

Development of Verified Numerical Computations for Applications

Takeshi Ogita

Tokyo Woman's Christian University

SC18, Dallas, USA

Nov. 14, 2018

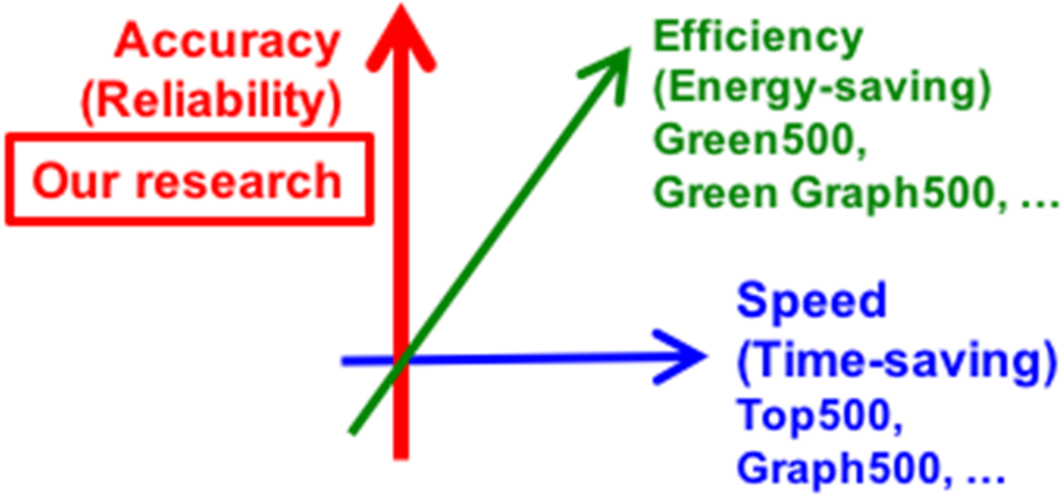
Verified Numerical Computations
VNC-HPC
High-Performance Computing

Outline

Goal of our project

Develop a super high-performance computing environment that can solve various challenging problems caused by numerical errors.

Introduce the axis of accuracy into high-performance computing on the K and Post-K computers.



Performance
= Speed * Accuracy

Research Organization

Project period: August 2016 - March 2020 (3.5 years)

	Organization	Representative	Role
Leader	Tokyo Woman's Christian Univ.	OGITA, Takeshi	Project leader, Development of algorithms for accurate numerical computations
Partners	Waseda Univ.	KASHIWAGI, Masahide	Development of algorithms for verified numerical computations
	Nagoya Univ.	KATAGIRI, Takahiro	Development of benchmark methods and implementation
	Shibaura Inst. Tech.	OZAKI, Katsuhisa	Development of fast and accurate matrix multiplication methods

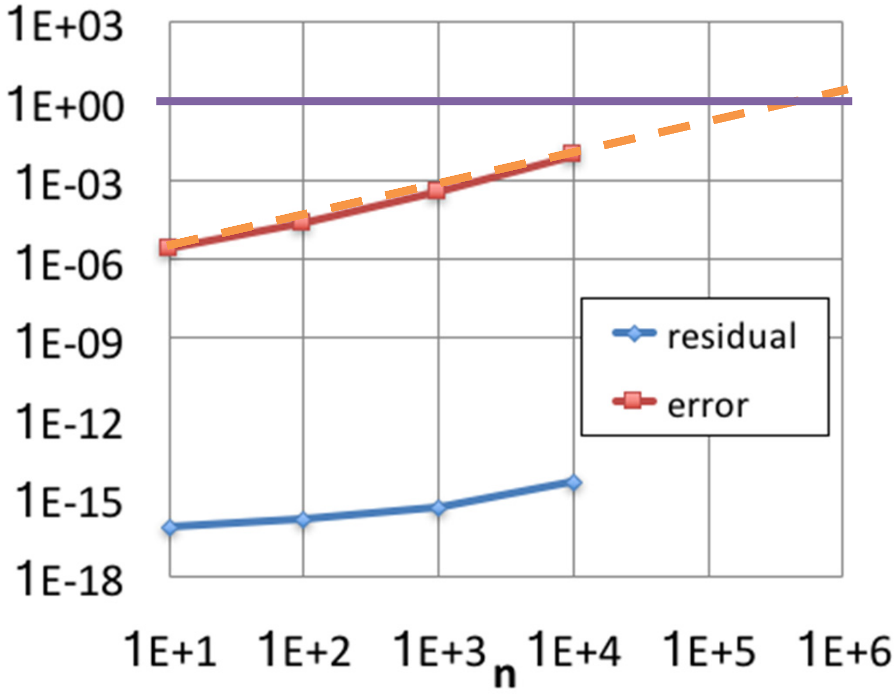
Why this is necessary

- In numerical computations, computed solutions suffer from several numerical errors:
  - discretization errors: FEM, FDM, etc.
  - rounding errors: floating-point arithmetic
  - truncation errors: iterative solvers
- In particular, in HPC environments,
  - rounding errors are highly accumulated in large-scale problems,
  - results are often not reproducible (due to parallelization and others).

Verified Numerical Computations
VNC-HPC
High-Performance Computing

Example: Accuracy of computed solutions of *Ax* = *b*

A: random, cond(*A*) = 1E+10



n = 100, vary cond(*A*)

cond(*A*)	Residual	Error
1.0E+03	2.07E-16	5.08E-14
1.0E+06	1.00E-16	2.59E-11
1.0E+09	8.23E-17	1.98E-08
1.0E+12	9.40E-17	1.59E-05
1.0E+15	6.49E-17	1.02E-02

LINPACK check (residual)

$$\frac{\|b - A\widehat{x}\|_\infty}{\|A\|_\infty \|\widehat{x}\|_\infty + \|b\|_\infty}$$

correctness of the implementation

Relative error

$$\max \left| \frac{\widehat{x}_i - x_i}{x_i} \right|$$

correctness of the results

What we can do now

- Verified solutions of dense linear systems
  - For general matrices, verification cost is around 1-5 times more than LU factorization.
  - For symmetric positive definite matrices, verification cost is almost nothing using Cholesky factorization.
  - The accuracy of computed solutions can be improved up to the limit of working precision with a little cost.
  - For ill-conditioned cases, verification cost is adaptive.
- Verified solutions of sparse linear systems
  - Direct solvers are required in most cases.
  - Applicable range is limited (diagonally dominant, M-matrix, symmetric and positive definite), but fast.

Verification Algorithm

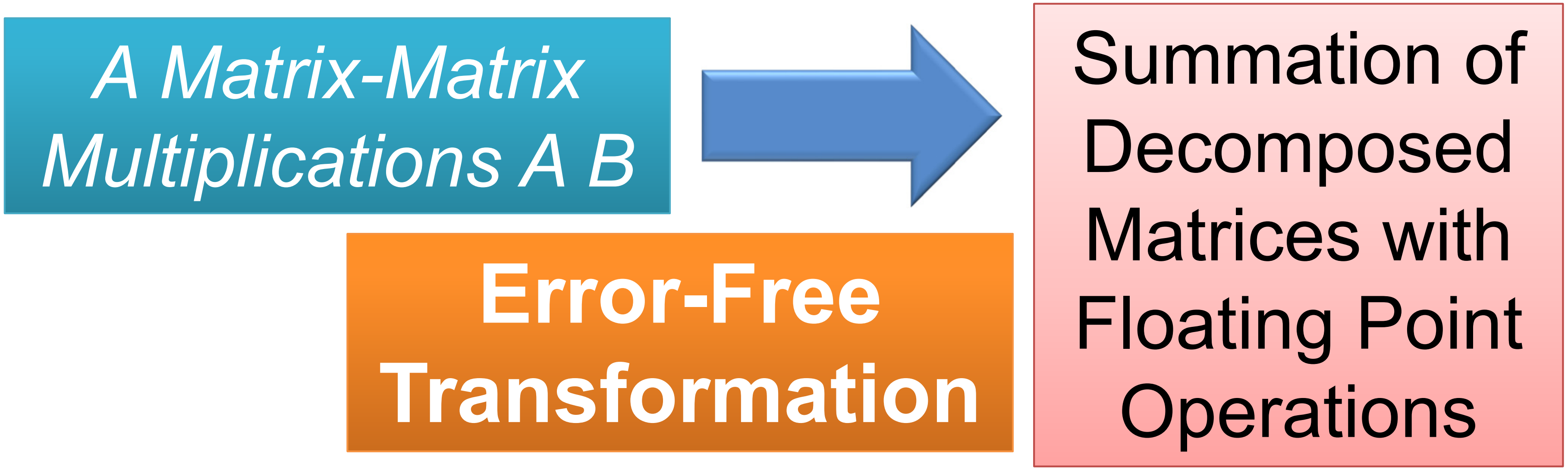
1. Solve a discretized linear system $Ax = b$.
   - ➢ $\hat{x}$: a computed solution
2. Solve a linear system $Ay = e$ where all elements of e are 1's.
   - ➢ $\hat{y}$: a computed solution
3. Verify M-property of A using $\hat{y}$. $(\hat{y} > 0 \;\; \Rightarrow \;\; A\hat{y} > 0)$
4. Compute an error bound using

$$\|x - \hat{x}\|_\infty \le \frac{\|\hat{y}\|_\infty \|b - A\hat{x}\|_\infty}{1 - \|e - A\hat{y}\|_\infty}$$

if $\|e - A\hat{y}\|_\infty < 1$.

HIGHLY ACCURATE MMM: OZAKI METHOD

Overview of High Precision Matrix-Matrix Multiplications (MMM) Algorithm (Ozaki Method †1)（1/3）



A Matrix-Matrix Multiplications A B

Error-Free Transformation

Summation of Decomposed Matrices with Floating Point Operations

$$C = AB = \sum_{q=1}^{r} C_q$$

$$C_q \in F^{m \times p}$$

F : A Set of Floating Point Numbers.
A : A Matrix with *m* * *n.*
B : A Matrix with *n* * *p.*
C : *A* * *B*

†1 K. Ozaki, T. Ogita, S. Oishi, S.M. Rump: Error-Free Transformation of Matrix Multiplication by Using Fast Routines of Matrix Multiplication and its Applications, Numerical Algorithms, Vol. 59, No.1, pp.95-118, 2012.

Other Part of The Error Free Transformation in Ozaki Method

- **Part of MMMs**

n_A : The number of decomposed matrices from matrix A.

n_B : The number of decomposed matrices from matrix B.

```
Function $EF$ = EFT_Mul($A$, $B$)
    [$A$, $n_A$] := Split_A;  [$B$, $n_B$] := Split_B;
      $k$ := 1;
    for $i$ =1: $n_A$
        for $j$ =1: $n_B$
            $EF\{k\}$ := $\underline{A}\{i\} * \underline{B}\{j\}$;    $k$ := $k$ + 1;
    end;  end;
end
```

Multiple BLAS implementation

- A High Precision Summation:

$$AB = \sum_{k=1}^{n_A \cdot n_B} EF^{(k)}$$

⇒ Faithful Algorithm

Faithful Algorithm†

Round-off the true answer to the nearest left or right floating number.

Ozaki Method for MMM

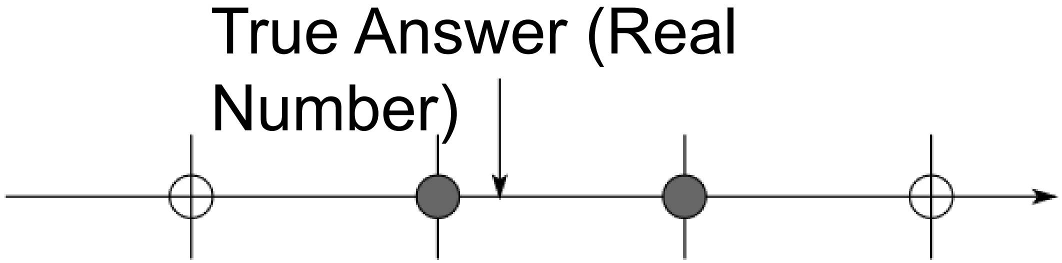


Figure 1: faithful rounding

Accuracy Assured

†Siegfried M. Rump, Takeshi Ogita, Shin'ichi Oishi: *Accurate Floating-Point Summation Part I: faithful Rounding*, SIAM Journal on Scientific Computing, **31**:1 (2008), 189-224.

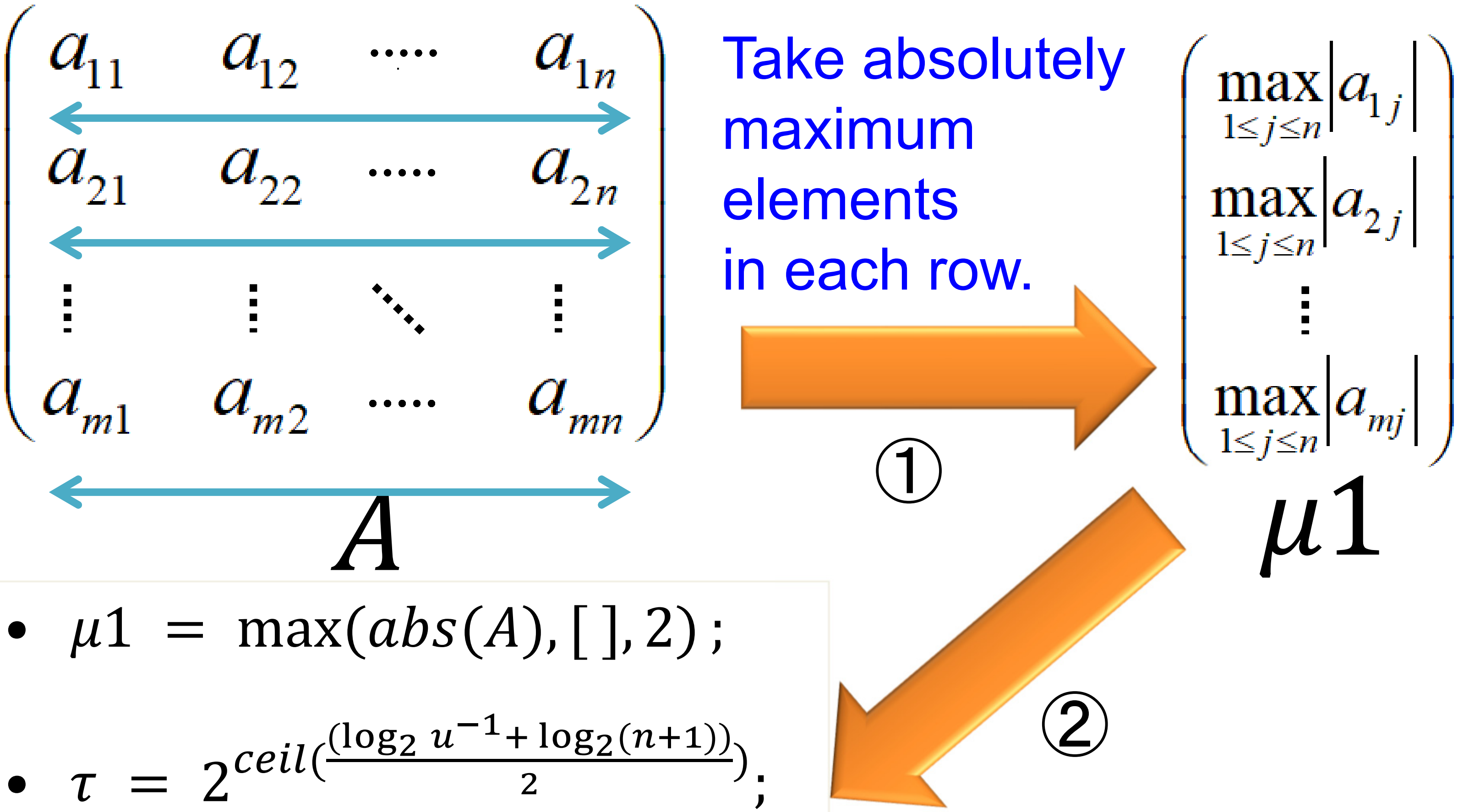
Characteristics of Ozaki Method

- Ozaki method can establish high precision for MMM with extremely dispersed elements.
- Computational complexity of Ozaki method depends on range of input elements.

(1) If dispersion of elements of matrix is large:
Sparse matrix can be utilized after error free translation to reduce computational complexity.

(2) If dispersion of elements of matrix is small:
Cannot reduce computational complexity.
But, Conventional high performance implementations (BLAS dgemm) of dense MMM can be utilized.

Error-Free Transformation (1/3)



$$A = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{pmatrix}$$

Take absolutely maximum elements in each row.

①

$$\mu 1 = \begin{pmatrix} \max_{1\le j\le n}|a_{1j}| \\ \max_{1\le j\le n}|a_{2j}| \\ \vdots \\ \max_{1\le j\le n}|a_{mj}| \end{pmatrix}$$

②

- $\mu 1 \; = \; \max(abs(A), [\,], 2)\,;$
- $\tau \; = \; 2^{ceil(\frac{(\log_2 u^{-1} + \log_2(n+1))}{2})};$

$$\boxed{t_A = 2^{ceil(\log_2(\mu 1))}\ \tau}$$

Take maximum elements of products in each column.

* $ceil()$: Compute minimum integer number

Error-Free Transformation (2/3)

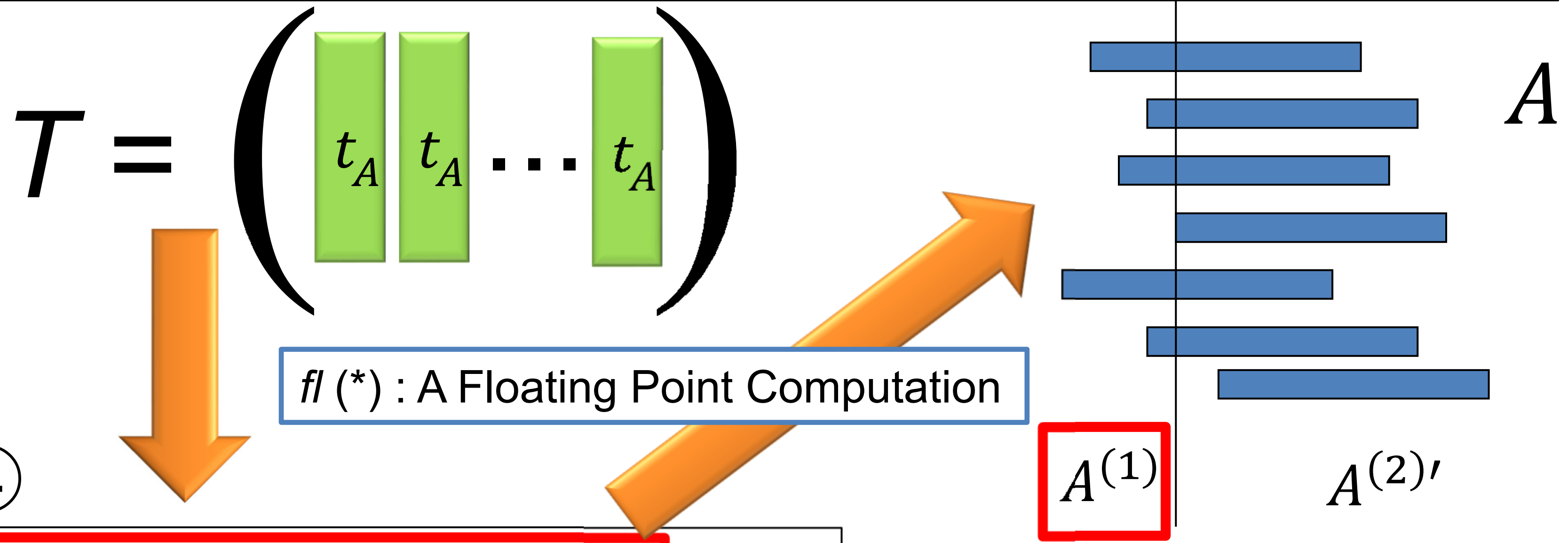
③

- Make *T* as:

$$T=[t_{A,}\ t_{A,}\ \ldots,\ t_A],$$

where, $T_{ij} > A_{ij}$.

Maximum number of products in each column.



$T = \begin{pmatrix} t_A & t_A & \cdots & t_A \end{pmatrix}$
A
$A^{(1)}$
$A^{(2)\prime}$

fl (*) : A Floating Point Computation

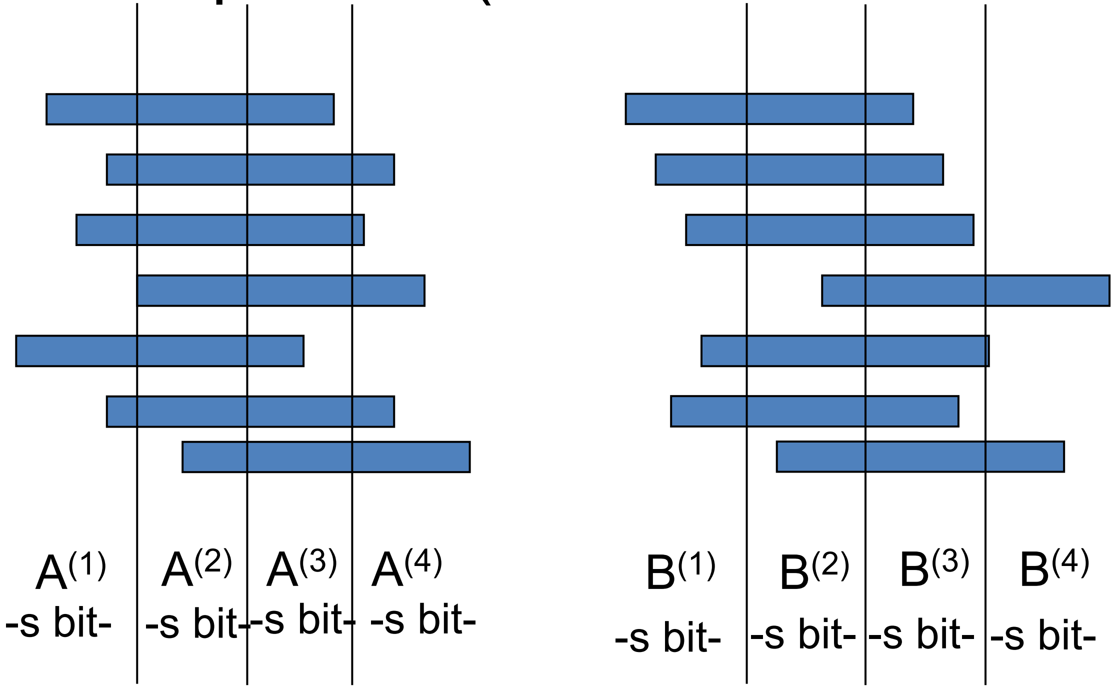
④

- $A^{(1)} = fl((A+T)-T);$
- $A^{(2)\prime} = fl(A-A^{(1)});$

Extract values which exceed range of expression of products with respect to round-off error.

Error-Free Transformation (3/3)

- An image of decomposition (Error free transformation)



Matrix size = n x n

$$s = floor((\log_2(u^{-1}) - \log_2(n))/2)$$ [bit]

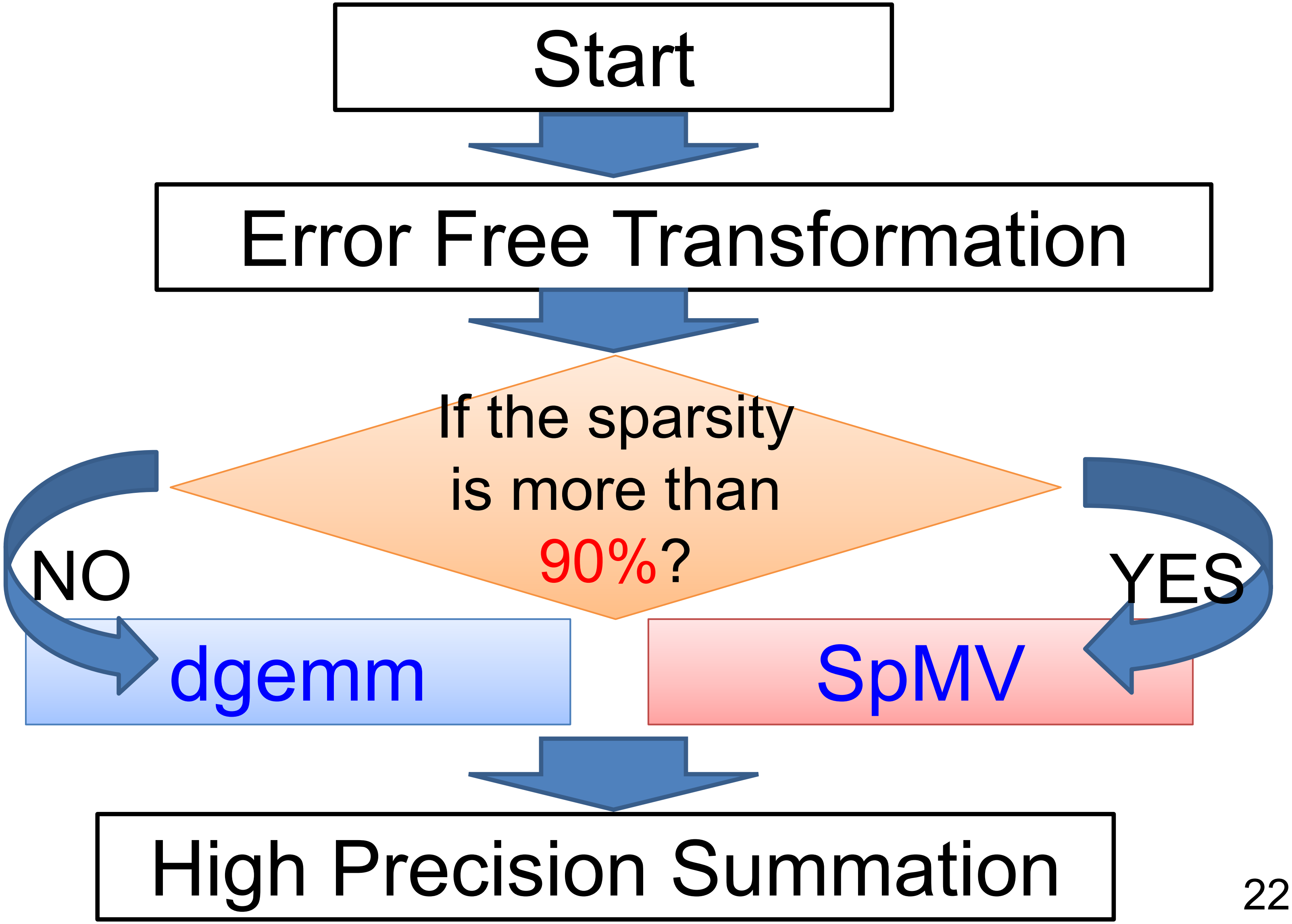
Ex.) If *double precision*, then it should take:

$floor(\,(53 - \log_2(n)\,/2)$ [bit],

And if matrix size is $n = 1024$, then it should take $floor\left(\frac{53-10}{2}\right) = 21$ [bit].

$floor$: under rounding for the first digit of floating point number.

Strategy of Using Sparse Matrix for Our Implementation



Start

Error Free Transformation

If the sparsity is more than 90%?

NO

YES

dgemm

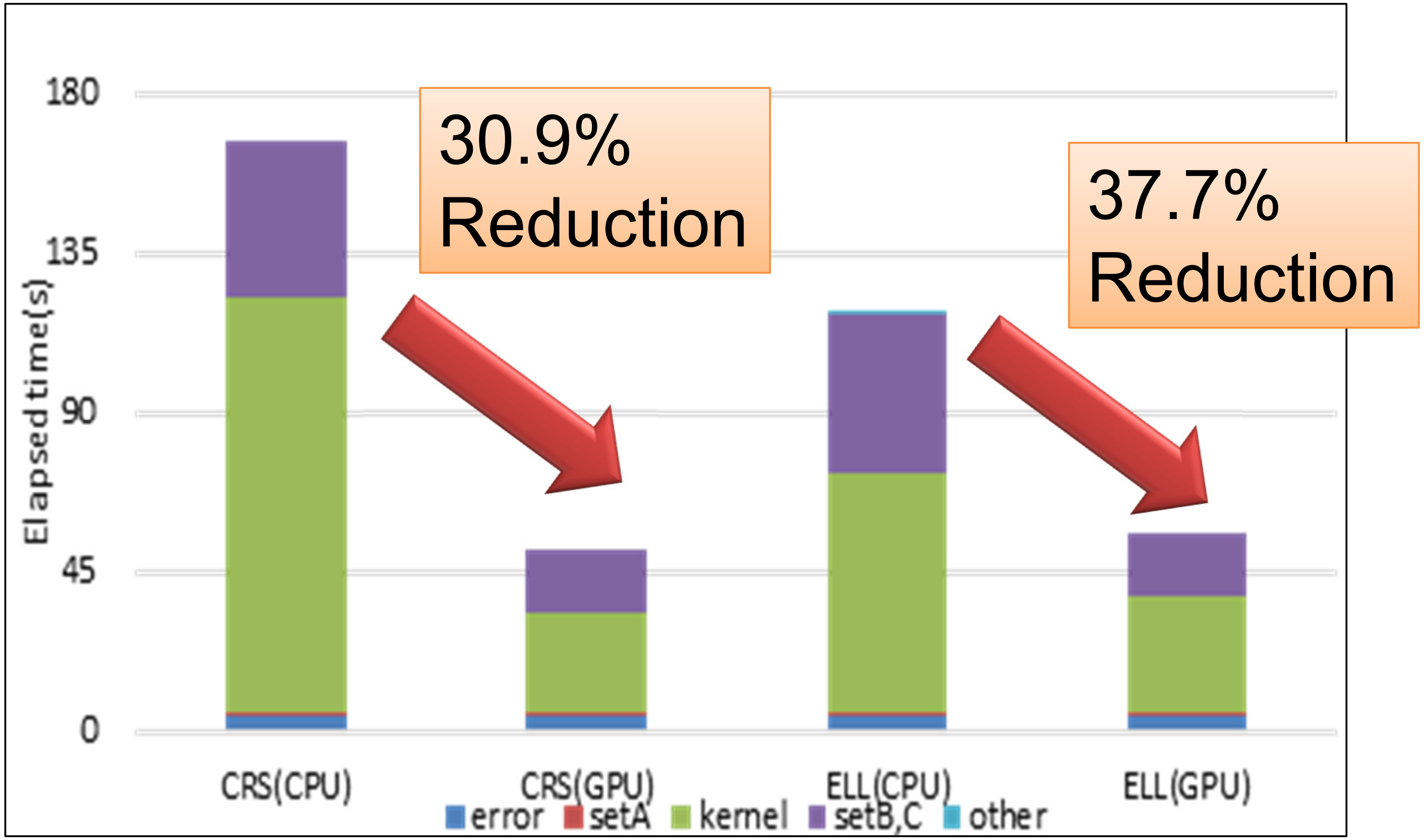
SpMV

High Precision Summation

Sparse Matrix-vector Multiplication (SpMV) Implementation for Ozaki Method

- We describe the calculation time of the SpMV routine in the Compressed Row Storage (CRS) and Ell-pack (ELL) formats in the CPU and GPU environments for a test matrix.
- The whole duration of the routine includes the error-free conversion time, duration of the change to the sparse matrix format, and actual calculation time.
- The error-free conversion time is “error_free”; the conversion time of matrix A to the sparse matrix format and the memory transfer time from the CPU to the GPU is “setA”; the SpMV routine time is “kernel”; the memory transfer time from the CPU to the GPU of the matrix B and from the GPU to the CPU of the matrix C is “SetB,C”; the duration of the remaining operations is given under “other”.

Result of SpMV Implementations for Ozaki Method



30.9% Reduction

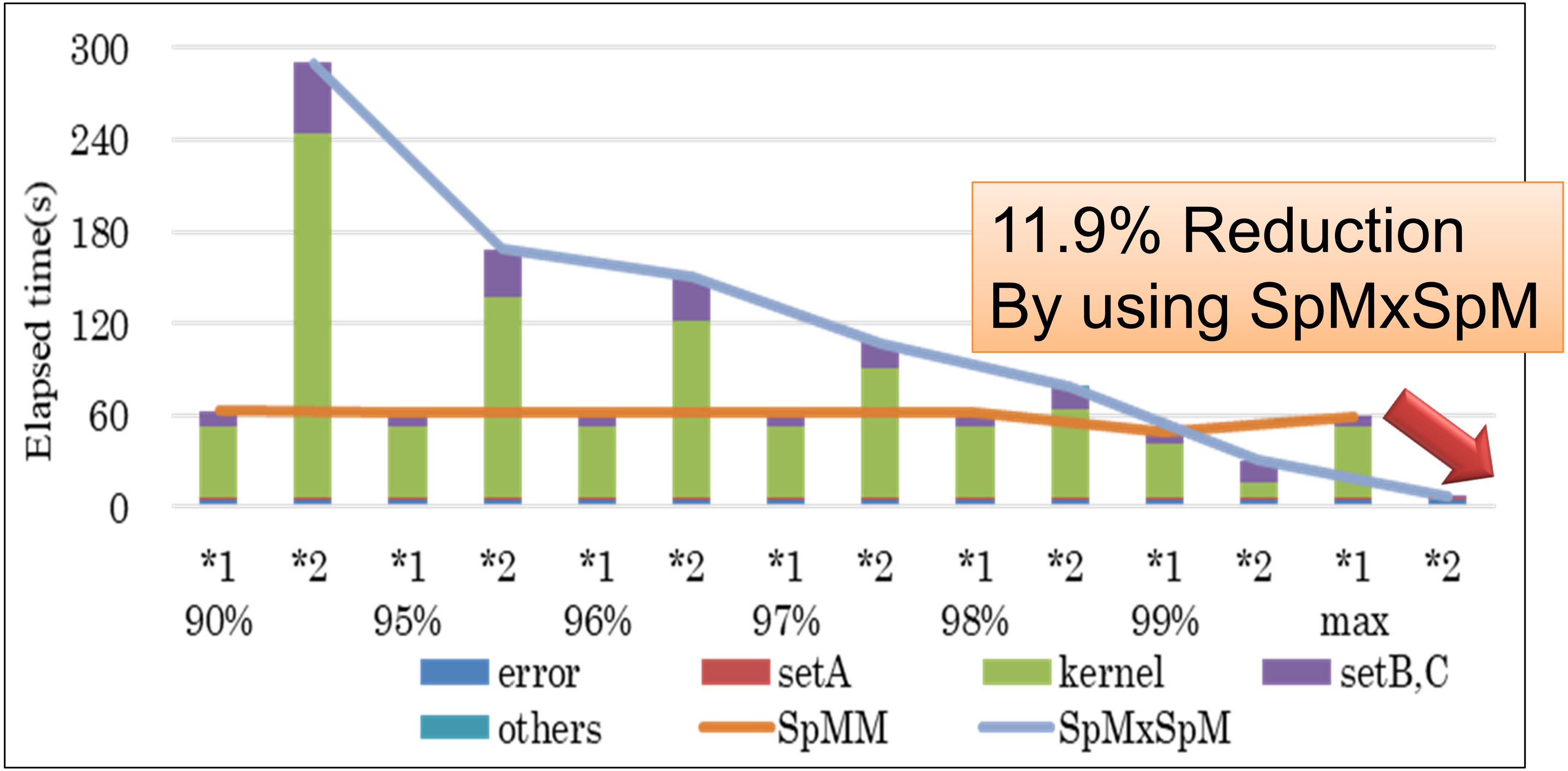
37.7% Reduction

Fig. 2. Execution speed of the SpMV routine with the format and ELL formats in the CPU and GPU environments for a test matrix with N = 10,000 in accurate MMM library. Reedbush-H (U. Tokyo) is used.

Sparse Matrix-Matrix Multiplication (SpMxSpM) Implementation for Ozaki Method

- We have developed an implementation of SpMxSpM with CRS format for Ozaki method in GPU environment.
- We evaluate performance of the SpMxSpM implementation for Ozaki method with cuBLAS. In addition, sparse matrix-matrix (SpMM) implementation for Ozaki method with cuBLAS is also evaluated.

Result of SpMxSpM Implementations for Ozaki Method



11.9% Reduction
By using SpMxSpM

Fig. 3 Execution time between SpMM and SpMxSpM implementations for Ozaki method. X-axis is sparsity of input matrix. "*1" stands for SpMM implementation. "*2" stands for SpMxSpM implementation.

Reedbush-H (U. Tokyo) is used.

TOPIC 2: DESIGNING ACCURACY ASSURED LIBRARIES FOR REAL SYMMETRIC EIGENPROBLEM.

ACCURACY ASSURED LINEAR EQUATION SOLVER

Iterative Refinement

- We check real answer of large-scale linear equations for liner solver with residual iteration refinement by accurate dot product (pseud quadratic accuracy).
- This experiment is using 1750,000 dimensions for linear equations.
- 2500 nodes (80,000 cores) of the Fujitsu PRIMEHPC FX100 in Nagoya University is used.
- The iterative refinement procedure is: (1) an approximate answer is obtained by using LU factorization; (2) A residual iterative refinement is performed.
- The result is as follows:

(First Step) Residual Norm: 4.019007e-14

(Second Step) Residual Norm: 0.000000e+00

- The above result indicates that the real answer is obtained with 2 step iterations. This also shows that the assured procedure we propose is a useful way for large-scale computations.

Solving Linear Equations

- We evaluate assured accuracy computation for solving linear equation. Given accuracy is improved by the iterative refinement procedure shown in previous slide.
- We set a real answer with (1,1,1,…,1)^T.
- 2500 nodes (80,000 cores) of the Fujitsu PRIMEHPC FX100 in Nagoya University is used.
- The result is:

(1 million dimension) Upper bound of error: 1.111484e-16

(1.5 million dimension) Upper bond of error: 1.113360e-16

- The above result indicates that the obtained accuracy is almost full for double precision computation. Hence the accuracy assurance can be adaptable for very large-scale computations on distributed memory supercomputers.

ACCURACY ASSURED STANDARD SYMMETRIC EIGENPROBLEM SOLVER

Iterative Refinement (Eigenproblem)

- We made a proto type implementation of assured accuracy library for standard symmetric eigenproblem.
- PDSYEVD (a ScaLAPACK routine） is used for this implementation. For test matrix, a symmetric matrix with elements generated by uniform distribution [0, 1].
- The Fujitsu PRIMEHPC FX100 in Nagoya University is also used.

Performance Evaluation (Varying Nodes)

- FX100
- $n = 50{,}000$
- PDSYEVD（ScaLAPACK）
- According to Fig. 4, there is a scalability for the ratio. This means that the ratios of verification time to computation time of eigenvalue are getting smaller according to number of nodes.
- This is a nice result to adapt the library of accuracy assurance to several applications.

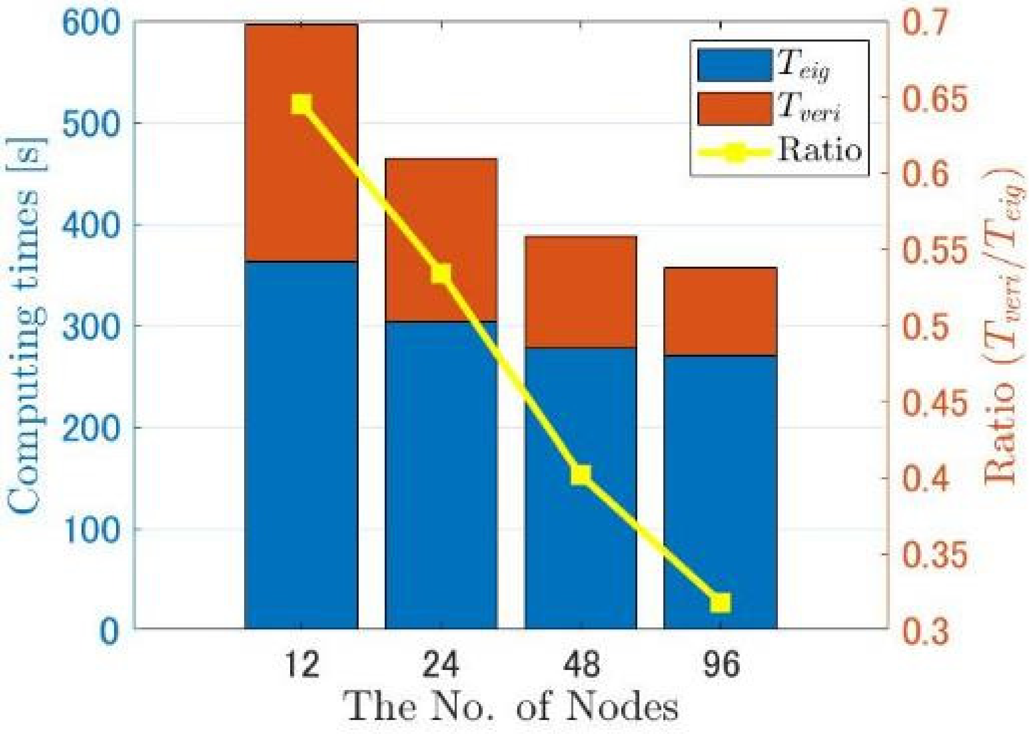


Fig. 4 Ratios of execution time (Tveri / Teig). Tveri stands for verification time. Teig stands for computation time of eigenvalues.

iPerformance Evaluation (Weak Scaling)

- FX100
- $n = 125{,}000 \sim 500{,}000$
- PDSYEVD（ScaLAPACK）
- We fix number of dimensions per node, while number of nodes increases. This is weak scaling evaluation.
- Fig. 5 shows that execution time for assured accuracy computation can be occupied up to 40%~50% to computation time of eigenvalues.
  This is acceptable ratio for large-scale computation.

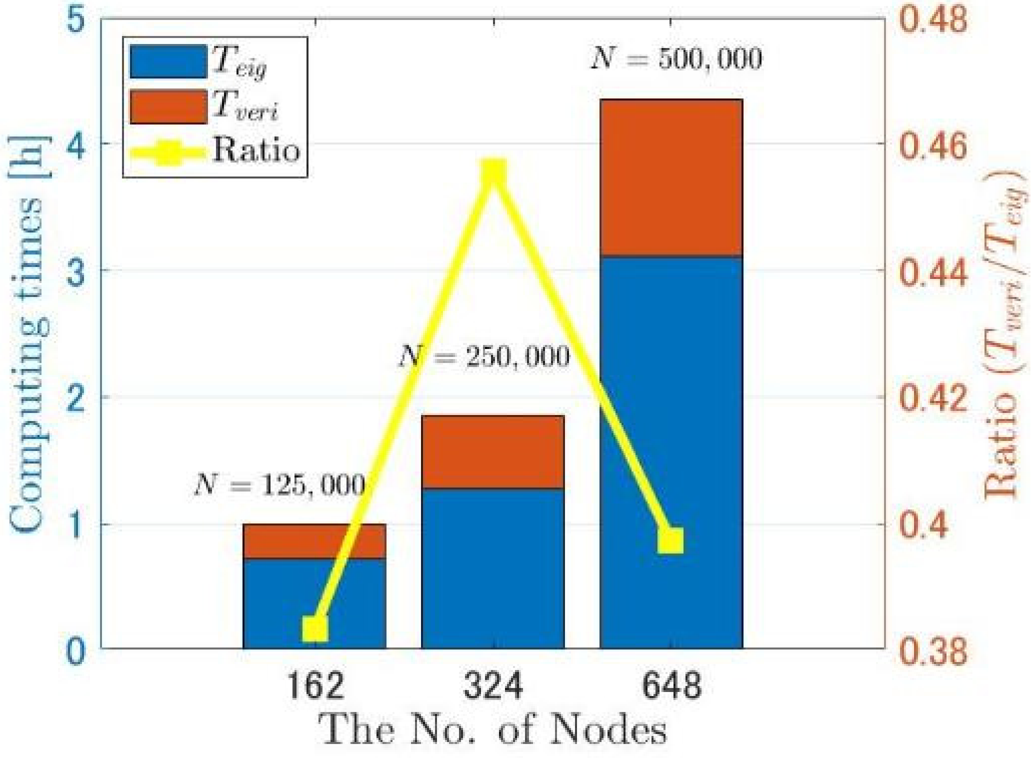


Fig. 5 Weak Scaling Result.

Performance Evaluation (Accuracy)

- FX100
- $n = 500{,}000$
- PDSYEVD（ScaLAPACK）
- λ_i :i-th smaller approximate eigenvalue.
- r_i :upper error bound of λ_i by accuracy assurance.
- We fix number of dimensions per node, while number of nodes increases. This is weak scaling evaluation.
- Fig. 6 shows that upper bound of calculated error is 60% at the worst. This indicates that the calculated result is never included “duplicate eigenvalues” for the eigenproblem with dimension of 500,000.
- We cannot proof this without the techniques for accuracy assurance for the eigenproblem.

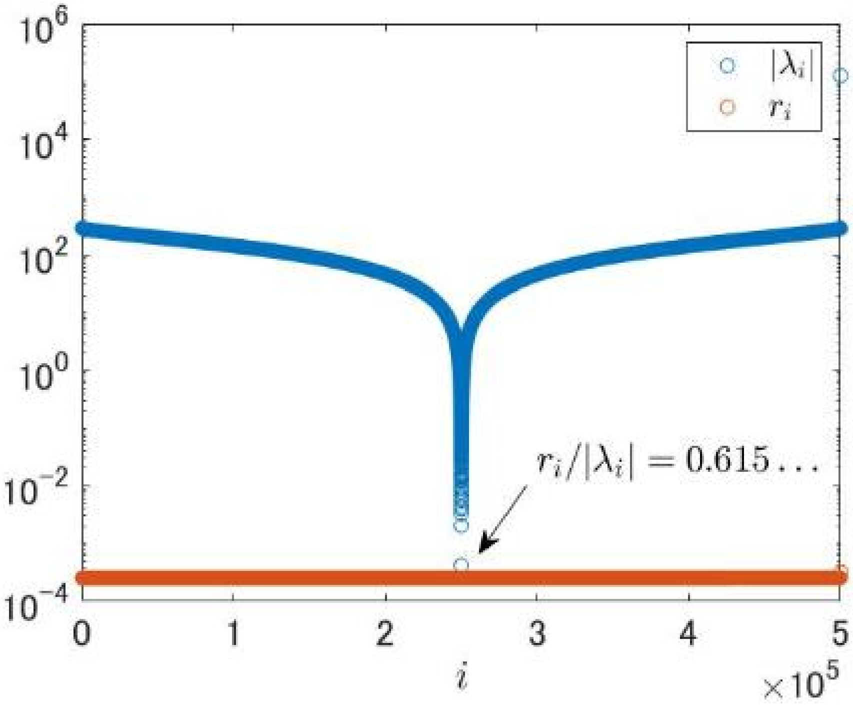


Fig. 6 Errors of computed eigenvalues to real answer.

TOPIC 3: DISCUSSING EXTENSION TO NON-LINEAR PROBLEMS.

Parallel smoothed aggregation multilevel Schwarz preconditioned Newton-Krylov algorithms for Poisson-Boltzmann problems

Introduction

- This work aims to improve our one-level parallel tool for simulating the colloidal phenomena modeled by the Poisson-Boltzmann equation (PBE) to a multilevel tool.
- The multilevel algorithm used the geometric mesh information to construct the multilevel preconditioner.
- The multilevel PBE solver was built on the top of **PETSc**, **Trilinos/ML**, and **METIS**.

Poisson-Boltzmann equation

- PBE is given on a 3D bounded domain Ω by

$$\begin{cases} \nabla^2\psi &= \frac{\exp(z_z\psi)-\exp(-z_c\psi)}{z_c+z_a} & \text{on } \Omega, \\ \psi &= g_D & \text{on } \Gamma_D \text{ of Dirichlet boundary}, \\ \frac{\partial\psi}{\partial n} &= 0 & \text{on } \Gamma_N \text{ of Neumann boundary}, \end{cases}$$

  where ψ is the electric potential, $z_c : z_a$ denotes the valence of cation and anion.
- Finite element method (FEM) discretizes the PDE.

Newton-Krylov-Schwarz framework

Input: nonlinear function $F(x)$, form Jacobians $J_F(x)$, mesh information for R_i^δ.

Output: approximate solution $x^{(k)}$

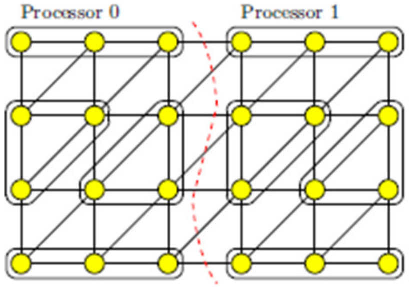
```
define Schwarz type preconditioner $M_k^{-1} = \sum_i^{np} (R_i^\delta)^T J_i^{-1} R_i^\delta$.
while $\|F(x^{(k)})\| > \epsilon_r \|F(x^{(0)})\|$ and $\|F(x^{(k)})\| > \epsilon_a$ and $\|\lambda_x s^{(k)}\| > \epsilon_s$ and $k < m$ do
    solve $M_k^{-1} J_F(x^{(k)}) s^{(k)} = -M_k^{-1} F(x^{(k)})$ for $s^{(k)}$.
    find $\lambda \in (0,1]$ such that $f(x^{(k)} + \lambda s^{(k)}) < f(x^{(k)}) + \alpha\lambda\nabla f(x^{(k)})^T s^{(k)}$.
    $x^{(k+1)} \leftarrow x^{(k)} + \lambda_x s^{(k)}$.
    $k \leftarrow k+1$.
```

- Smoothed aggregation multilevel Schwarz preconditioning

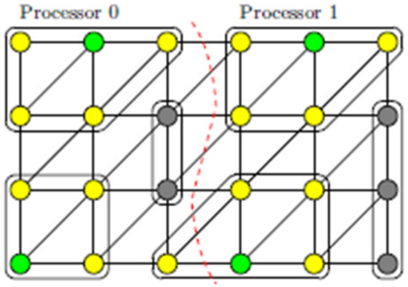
Input: The Jacobian matrix J, residual r, and the total number of levels L.
Output: The preconditioned residual $\hat{r}$.

```
$r^l \leftarrow r$
for $l \leftarrow L$ to 2 step -1 do
    $u^l \leftarrow \sum_{i=1}^{np} (R_i^{k,0})^T (J_i^k)^{-1} R_i^{k,0} r^l$
    $r^{l-1} \leftarrow (I_{l-1}^l)^T (r^l - J^k u^l)$
solve $J^1 u^1 = r^1$ for coarse mesh correction $u^1$.
for $l \leftarrow 2$ to $L$ step 1 do
    $u^l \leftarrow u^l + I_{l-1}^l u^{l-1}$
    $u^l \leftarrow u^l + \sum_{i=1}^{np} (R_i^{k,0})^T (J_i^k)^{-1} R_i^{k,0} (r^k - J^k u^l)$
$\hat{r} \leftarrow u^l$
```

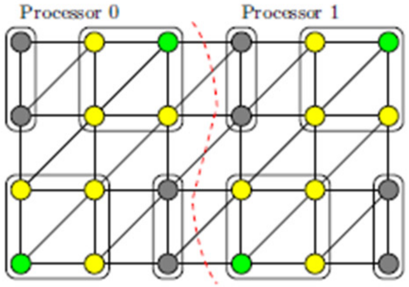
- Parallel multilevel aggregation schemes
  - Parallel maximal independent set (MIS): aggregates contain the nodes distributed on the different cores.
  - Uncoupled MIS (uncoupled): no distributed nodes in aggregates.
  - Graph partition-based scheme (METIS): without distributed nodes in aggregates and nearly the same size of coarse grid independed the employed cores.



(a) MIS



(b) Uncoupled

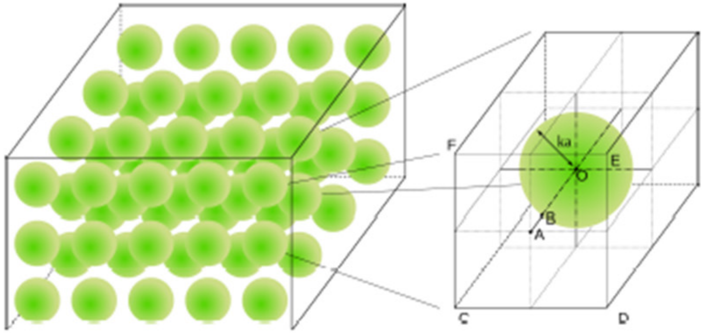


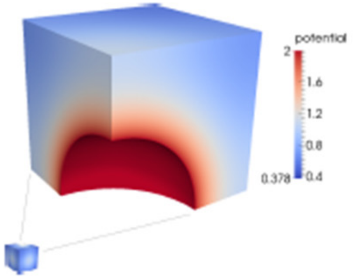
(c) METIS

Numerical results: Array of charged spherical particles

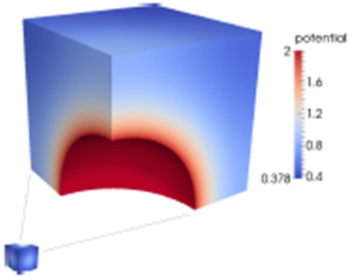
- An array of charged spherical particles immersed in symmetric (1:1) or asymmetrical (1:2) electrolytes.
- About 20 million unknowns tested on 64 to 4,096 cores on the cluster Reedbush-U in University of Tokyo, Japan.
- Two-level and three-level preconditioner and different aggregation schemes

Aggregation scheme	np	2-level			3-level		
		Newton iter. (A. linear)	Time (sec.)	Eff. (%)	Newton iter. (A. linear)	Time (sec.)	Eff. (%)
MIS	64	4 (8.0)	131.1	100.0	4 (10.3)	121.5	100.0
	128	4 (7.5)	59.4	110.4	4 (9.8)	54.2	112.1
	256	4 (7.0)	28.8	113.8	4 (9.8)	26.2	115.9
	512	4 (7.5)	14.0	117.1	4 (10.5)	13.8	110.0
	1,024	4 (8.5)	7.8	105.0	4 (11.3)	8.3	91.5
	2,048	4 (7.8)	4.9	83.6	4 (11.5)	6.8	55.8
	4,096	4 (8.3)	4.0	51.2	4 (12.3)	9.4	20.2
Uncoupled	64	4 (8.0)	103.5	100.0	4 (10.3)	93.6	100.0
	128	4 (7.8)	51.2	101.1	4 (9.8)	45.8	102.2
	256	4 (7.8)	25.6	101.1	4 (9.8)	22.9	102.2
	512	4 (7.8)	12.5	102.7	4 (10.3)	12.0	97.5
	1,024	4 (9.0)	7.0	92.4	4 (10.8)	7.3	80.0
	2,048	4 (7.8)	4.3	75.2	4 (9.8)	5.3	55.2
	4,096	4 (8.3)	3.6	44.9	4 (9.8)	6.8	21.5
METIS	64	4 (14.3)	94.4	100.0	3 (23.7)	75.4	100.0
	128	4 (14.5)	46.7	101.1	4 (20.8)	49.4	76.3
	256	4 (14.3)	23.3	101.3	4 (18.5)	24.3	77.6
	512	4 (15.0)	12.2	96.7	4 (18.0)	12.8	73.6
	1,024	4 (14.8)	6.8	86.7	4 (17.0)	8.1	58.2
	2,048	4 (15.0)	4.7	62.8	4 (16.3)	7.5	31.4
	4,096	4 (16.0)	3.8	38.8	4 (16.3)	13.4	8.8





(a) 1:1 electrolytes



(b) 1:2 electrolytes

- Parallel performance evaluation

Compared the 1-level additive nonoverlapping Schwarz preconditioner, parallel smoothed aggregation multigrid (SAMG) preconditioner (5-level V cycle), and tuned 2-level METIS aggregation preconditioner.

Array of charged spherical particles

np	One-level block Jacobi			One-level nonoverlapping Schwarz			SAMG			Two-level SA Schwarz (METIS)		
	N. iter.	Time (sec.)	Eff. (%)	N. iter.	Time (sec.)	Eff. (%)	N. iter.	Time (sec.)	Eff. (%)	N. iter.	Time (sec.)	Eff. (%)
Symmetrical 1:1 electrolytes												
64	3 (1040.3)	933.6	100.0	3 (463.0)	269.2	100.0	4 (12.3)	103.1	100.0	4 (14.3)	94.4	100.0
128	3 (1058.3)	510.2	91.5	3 (469.0)	130.7	103.0	4 (11.3)	50.3	102.1	4 (14.5)	46.7	101.1
256	3 (1082.7)	294.5	79.3	3 (476.0)	58.9	114.3	4 (10.5)	25.1	103.1	4 (14.3)	23.4	101.3
512	3 (1060.3)	197.8	59.0	3 (490.3)	29.2	115.2	4 (10.3)	13.6	95.1	4 (15.0)	12.2	96.7
1,024	3 (1105.7)	226.5	25.8	3 (517.7)	15.5	108.5	4 (10.0)	12.1	53.0	4 (14.8)	6.8	86.7
2,048	3 (1180.0)	471.7	6.2	3 (554.7)	9.0	93.5	4 (9.8)	38.2	9.2	4 (15.0)	4.7	62.8
4,096	3 (1223.7)	2313.3	0.6	3 (572.3)	4.9	85.8	4 (9.3)	212.2	0.8	4 (16.0)	3.8	38.8
Asymmetrical 1:2 electrolytes												
64	5 (919.8)	1395.9	100.0	5 (395.4)	402.1	100.0	5 (11.2)	129.4	100.0	5 (13.2)	118.5	100.0
128	5 (967.0)	785.2	88.9	5 (403.6)	196.5	102.3	5 (10.4)	63.2	102.4	5 (13.4)	58.4	101.5
256	5 (978.6)	449.8	77.6	5 (409.4)	89.2	112.7	5 (9.6)	31.0	104.4	5 (13.4)	28.8	102.9
512	5 (987.0)	306.9	56.9	5 (420.2)	43.6	115.3	5 (9.6)	16.5	98.0	5 (14.2)	14.8	100.1
1,024	5 (984.4)	356.0	24.5	5 (437.0)	22.9	109.7	5 (9.4)	14.9	54.3	5 (14.0)	8.4	88.2
2,048	5 (978.2)	630.3	6.9	5 (462.6)	13.0	96.7	5 (8.8)	44.2	9.1	5 (14.2)	5.4	68.6
4,096	5 (972.4)	3047.0	0.7	5 (481.0)	6.6	95.2	5 (8.6)	251.0	0.8	5 (15.0)	4.4	42.1

- Conclusions

We develop a multilevel version of the PBE solver by adding the smoothed aggregation type coarse mesh space to further improve the algorithmic algorithmic scalability of the one-level NKS algorithm.

Future Prospects

- Topic 1: UNC-HPC libraries between multi-core and many-core CPUs and a GPU.
  - According to our results, we found several performance changes based on computer environments, such as CPU or GPU. In addition, sparsity of input matrix is also crucial factor.
  - We need to add adaptive selection for several implementations of Ozaki method. To establish this, auto-tuning (AT) technology is one of promising ways.
- Topic 2: Designing accuracy assured libraries for real symmetric eigenproblem.
  - We need to develop high performance implementation of the accuracy assured libraries for real symmetric eigenproblem toward to distributed memory supercomputers. In particular, adaptation of GPU computing is highly required.

Published Papers

- Journal Papers (Refereed)

[1] S.-R. Cai, J.-Y. Xiao, Y.-C. Tseng, F.-N. Hwang, ‘Parallel multilevel smoothed aggregation Schwarz preconditioned Newton-Krylov algorithms for Poisson-Boltzmann problems, Numerical Mathematics: Theory, Methods and Applications’, Vol. 13, pp. 745-769, August 2020

- Proceedings of International Conferences (Refereed)

[2] Fumiya Ishiguro, *Takahiro Katagiri*, *Satoshi Ohshima*, *Toru Nagai*, ‘Performance Evaluation of Accurate Matrix-matrix Multiplications on GPU Using Sparse Matrix Multiplications’, International Conference on High Performance Computing in Asia-Pacific Region (HPCAsia2020), January 2020 (A Poster Presentation)

International conference Papers (Non-refereed)

[3] T. Ogita, ‘Verification methods for numerical linear algebra and applications’, The 9th International Congress on Industrial and Applied Mathematics (ICIAM 2019), July 2019

[4] T. Terao, K. Ozaki, T. Ogita, ‘Verified Numerical Computations for Eigenvalue Problems on Large-Scale Parallel Systems’, 2020 SIAM Conference on Parallel Processing for Scientific Computing, January 2020

[5] T. Ogita, ‘Verified Numerical Computations with HPC’, 3rd International Conference on Modern Mathematical Methods and High Performance Computing in Science & Technology, January, 2020 (An Invited Talk)

[6] T. Ogita, ‘Verified Numerical Computations on Supercomputers’, Workshop on Large-scale Parallel Numerical Computing Technology (LSPANC 2020), January 2020

- Presentations at domestic conference (Non-refereed)

[7] *片桐孝洋*，石黒史也，荻田武史，尾崎克久，*大島聡史*，*永井亨*，「精度保証付き数値計算ライブラリの運用に向けて」，大学ICT推進協議会 2019年度年次大会，AXIES2019予稿集，2019年12月

- Other (patents, press releases, books and so on)

[8] Fumiya Ishiguro, ‘A High-performance Implementation on GPU for Accurate Matrix-Matrix Multiplication Using Sparse Matrix Computations’, A Master Thesis, Graduate School of Informatics, Nagoya University, February 2020 (In Japanese)