THE UNIVERSITY OF TOKYO

東京大学情報基盤センター INFORMATION TECHNOLOGY CENTER, THE UNIVERSITY OF TOKYO

JCAHPC

$\mathcal{H}$ACApK : Parallel $\mathcal{H}$-matrices Library

Akihiro Ida

Contact: IDA, Akihiro <ida@cc.u-tokyo.ac.jp>

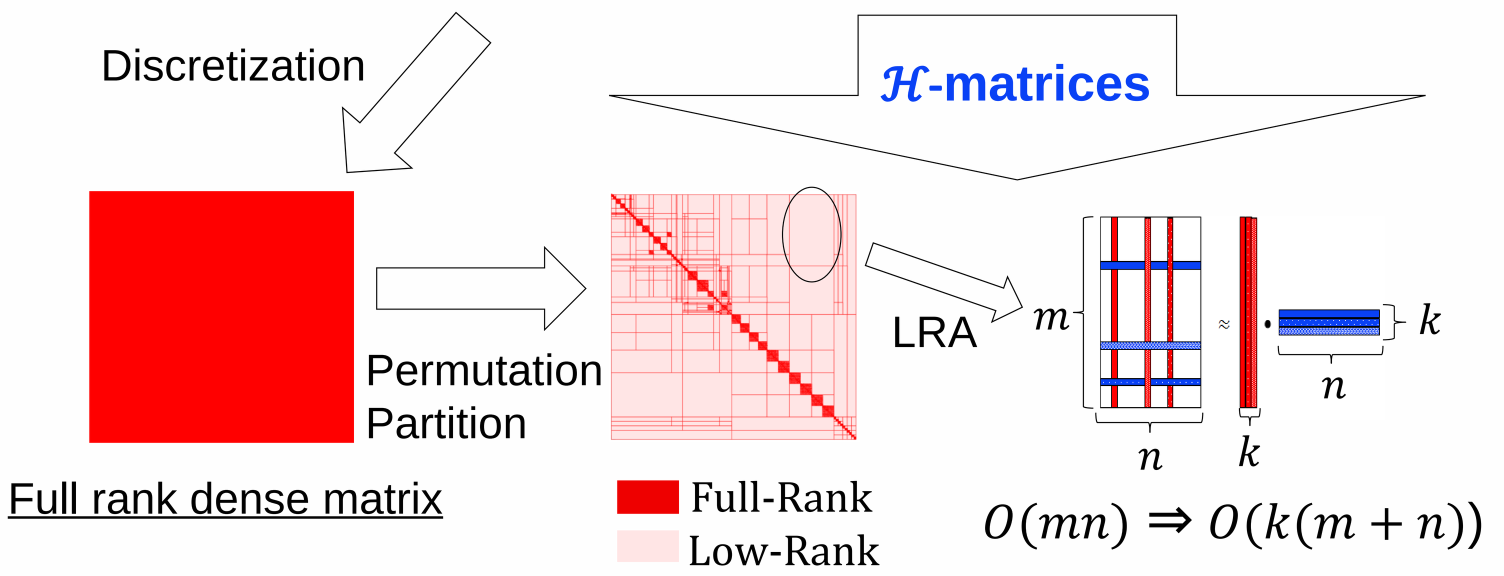
URL: http://www.cspp.cc.u-tokyo.ac.jp/ida/

$\mathcal{H}$-matrices : approximation technique for dense matrices

► Dense matrix : $O(N^2)$ ⇒ $\mathcal{H}$−matrix: $O(N\log N)$

$$\mathcal{g}[u](x) = \int_{\Omega} g(x,y)\,u(y)\mathrm{d}y$$

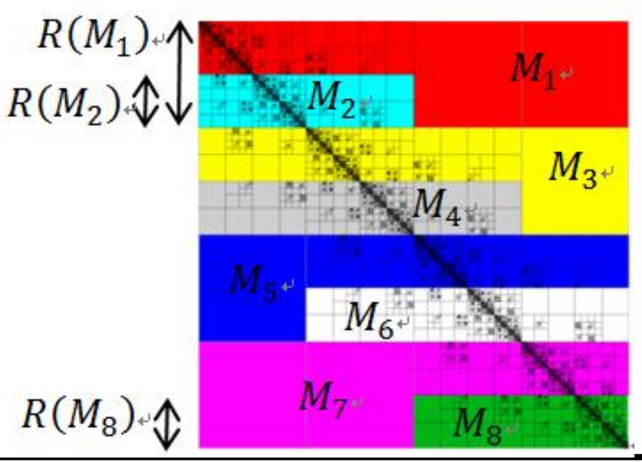
Degenerate kernel: $g(x,y) \cong \sum_{\nu=1}^{r} g_1^{\nu}(x) g_2^{\nu}(y)$



$O(mn) \Rightarrow O(k(m+n))$

Assignment strategy for parallel $\mathcal{H}$ -matrices

► Basically slice the H-matrix in row direction, but not cut submatrices.



- For MPI-parallelization
  - ① Minimize the maximum height $R(M_k)$ as much as possible
    - ⇒ Reducing transferred data size
  - ② Minimize the load imbalance among MPI processes

Overview

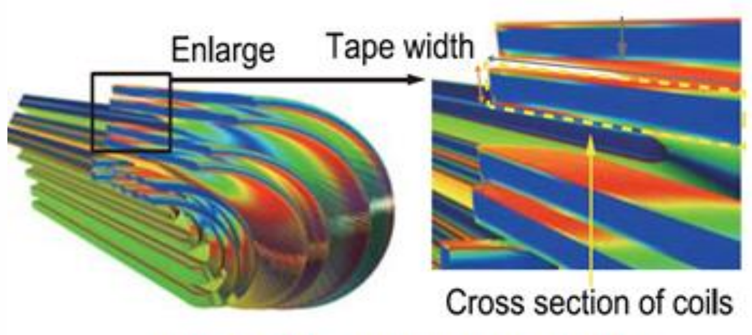
- Open-source software library for $\mathcal{H}$-matrices (MIT license).
- CPU-based clusters, and recently ported to multi-GPU platforms
- For large-scale simulations based on the boudary integral equation method
- Downloaded from web-site : http://ppopenhpc.cc.u-tokyo.ac.jp/

Application examples

► Superconductor

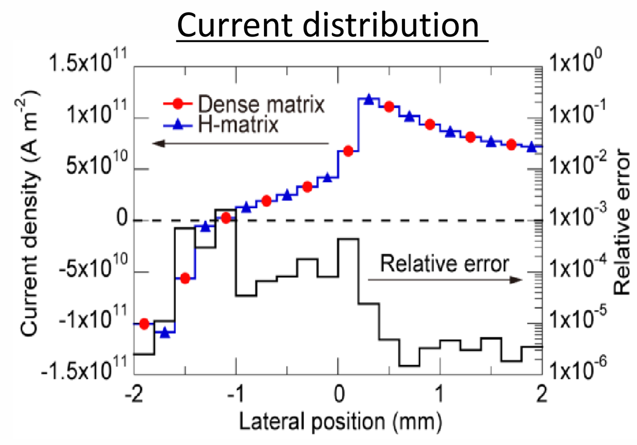
- Current density is calculated in coils wound with coated conductors.

Model (N=1,516,060)

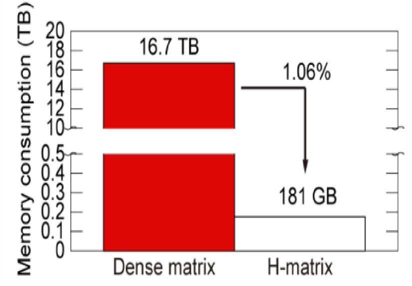


- Transcribed from Tominaga et al (2018)

Current distribution

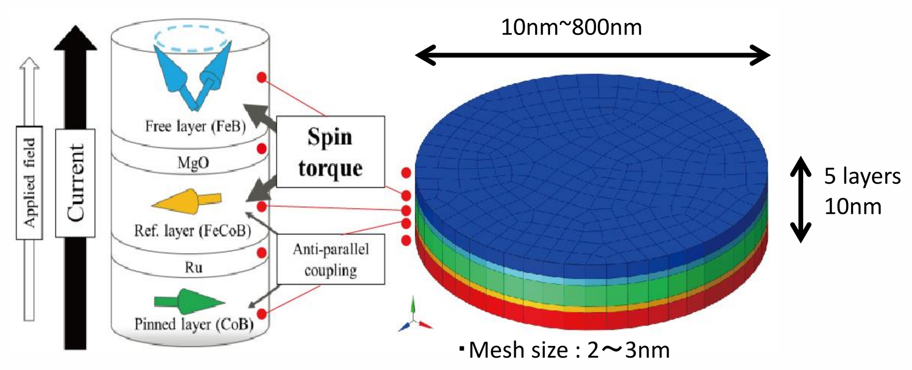


- Potential operator : $\frac{\mu_0 t_s}{4\pi}\int_{S'} \frac{(\nabla\times \boldsymbol{n}'T')\times \boldsymbol{r}\cdot\boldsymbol{n}}{r^3}\mathrm{d}S'$
- Memory usage

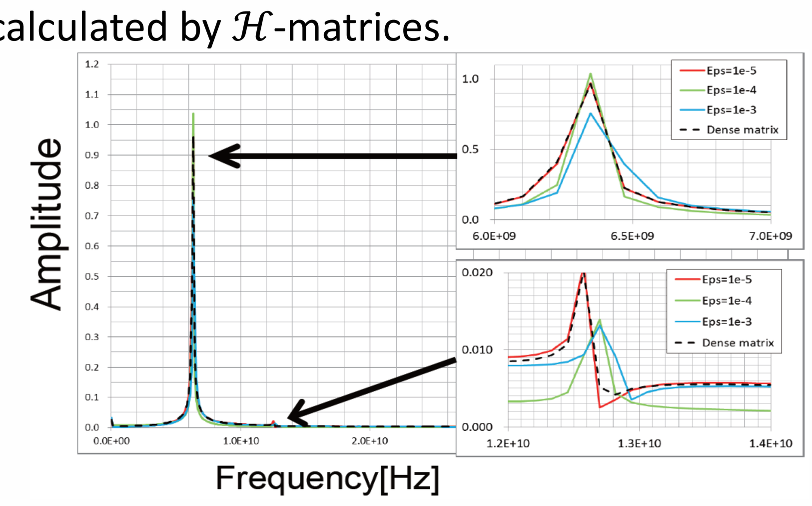


► Micromagnetic simulation of spin torque oscillator

- The factor of a demagnetic field is approximately calculated by $\mathcal{H}$-matrices.



- Transcribed from Ida at al (2017)



- H-matrices accuracy of $\varepsilon = 10^{-5}$ is required to reproduce oscillation processes.

Lattice $\mathcal{H}$-matrices

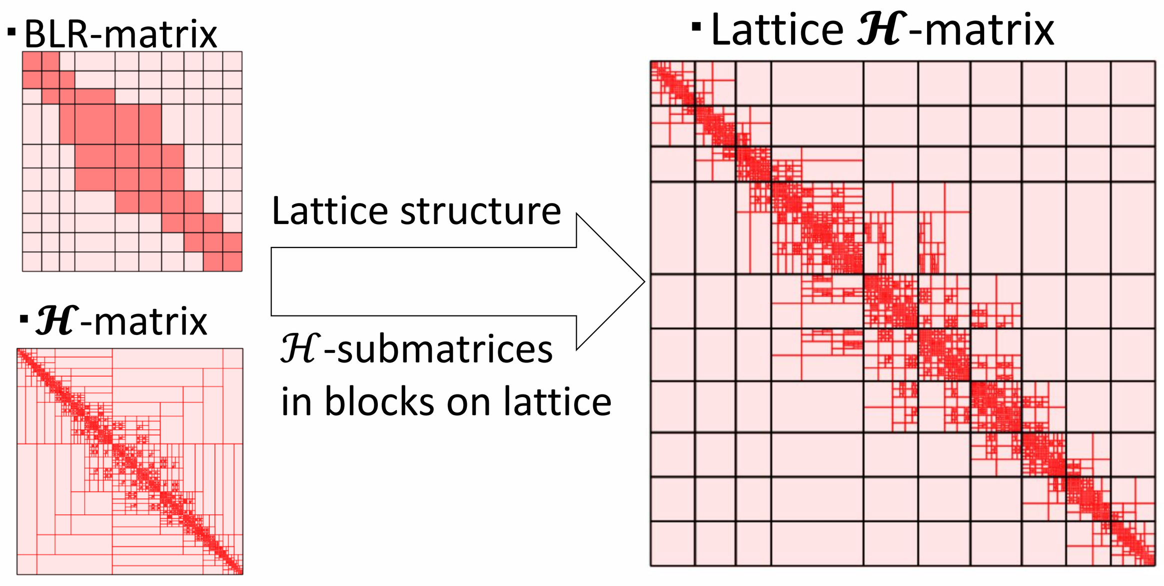
- achieve good load balancing.
- construct efficient communication pattern among MPI processes.

New approach

A new variant of low-rank structured matrices

► High compressibility of $\mathcal{H}$-matrices : $O(N\log N)$

► Convenience of matrix arithmetic with BLR-matrices

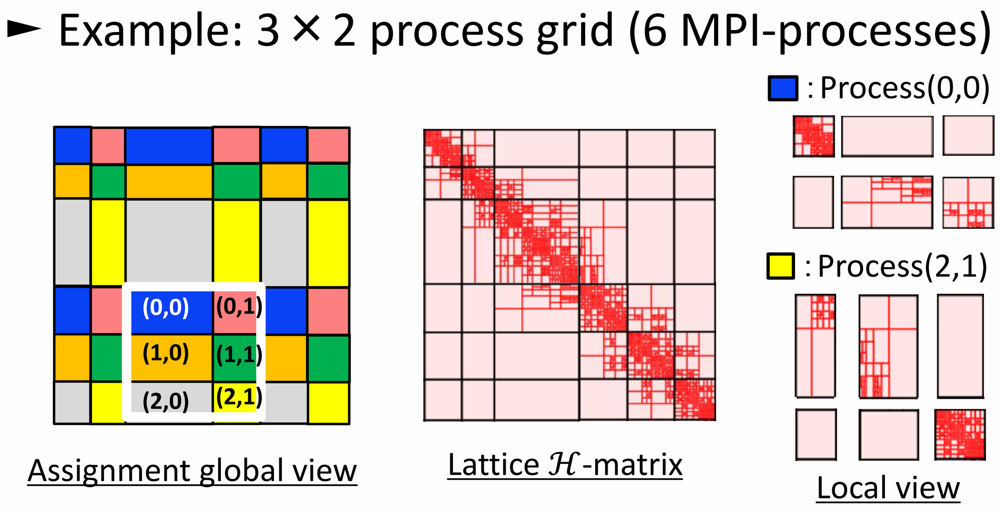


Assignment strategy for lattice $\mathcal{H}$-matrices

► 2D cyclic assignment strategy based on “process grid”

- Assignment for each process : The same colored lattice blocks
- Complex structure : Only needed in serial or thread computing
- Use of existing parallel algorithms for dense matrices

► Example: 3 × 2 process grid (6 MPI-processes)

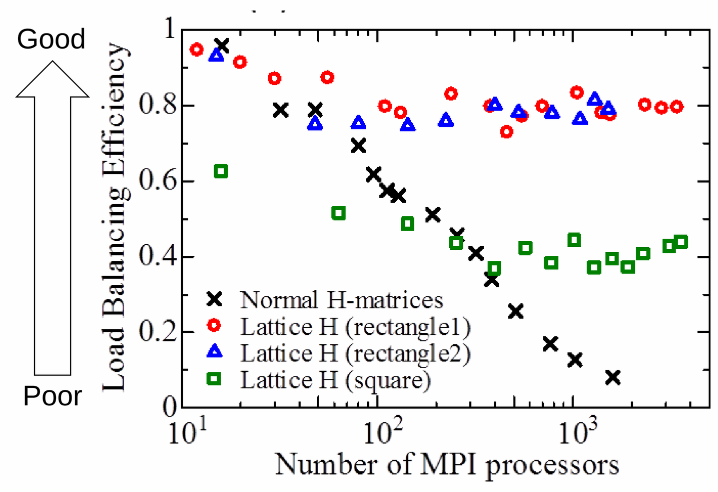


Load balance among MPI processes

► Load-balancing efficiency

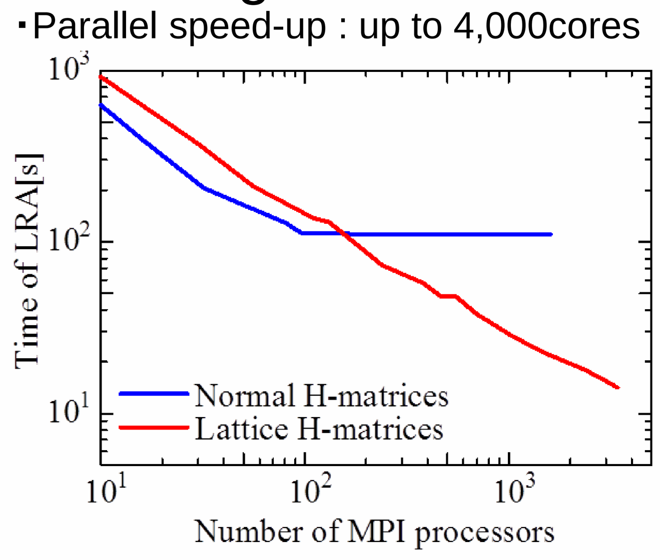
$E := \frac{\mu}{\mu_{\mathrm{pmax}} N_p}$ μ : Memory usage (entire matrix) μ_{pmax} : The largest memory (an MPI process)

- Normal H-matrices : Efficiency E rapidly decays
- Lattice H-matrices : Decay rate is moderate



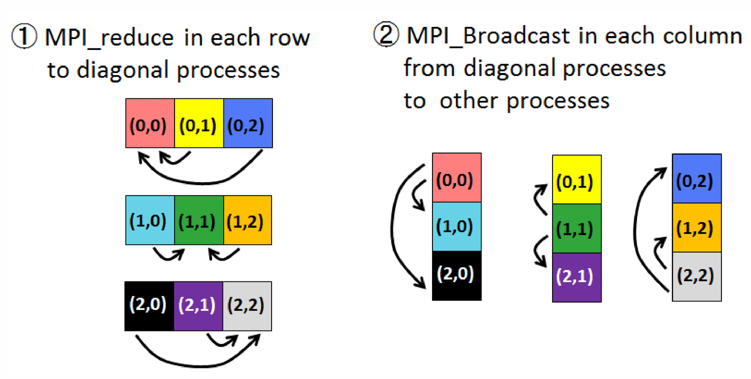
Matrix generation

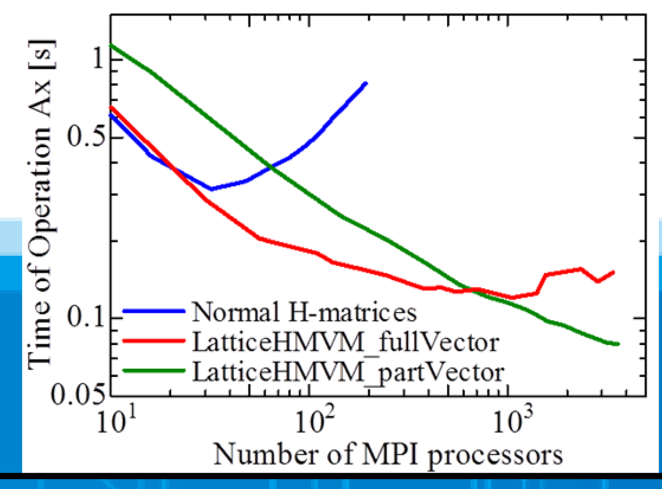
- Parallel speed-up : up to 4,000cores



Matrix-vector product

- Less communication cost than normal $\mathcal{H}$





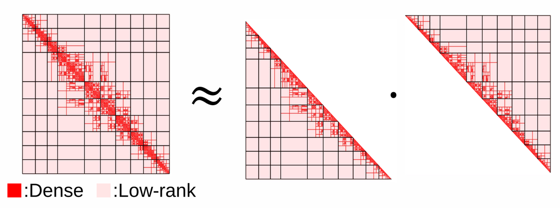
Matrix factorization

► **Scope of application**

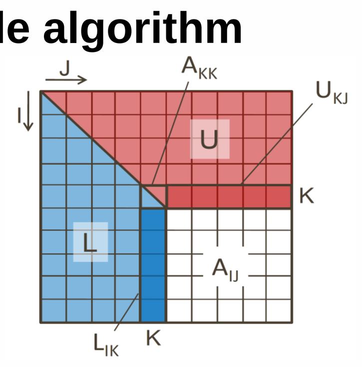
- High accuracy : Instead of non-approximated factorization
- Low accuracy : Preconditioner for Krylov subspace iterative methods

► **LU factorization**

- **Definition**
  - Approximate decomposition with keeping the structure of lattice $\mathcal{H}$-matrices



- **Tile algorithm**

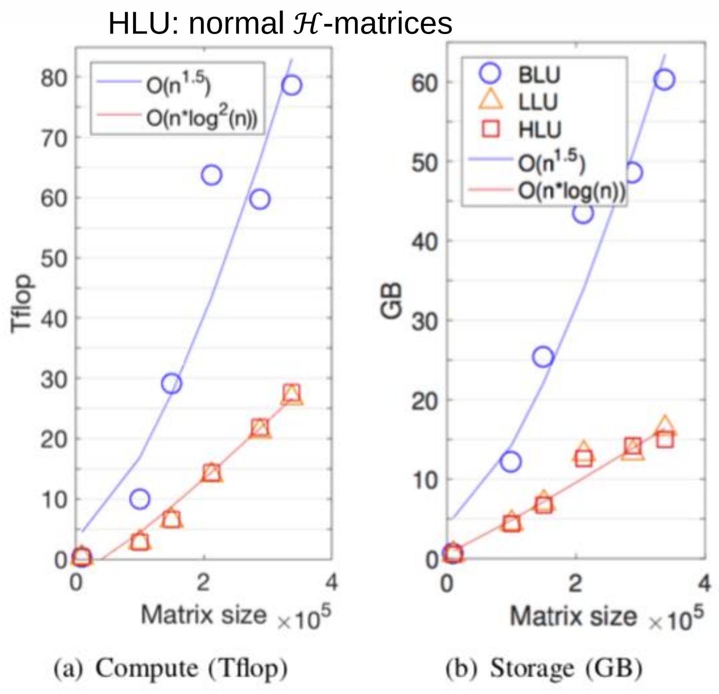


- **Numerical results**

BLU: Block-low-rank

LLU: lattice $\mathcal{H}$-matrices

HLU: normal $\mathcal{H}$-matrices



(a) Compute (Tflop)   (b) Storage (GB)

- **Computational Complexity analysis of lattice $\mathcal{H}$-LU**

$O(n\log^2 n)$: if $l \propto n$

n:size of matrix, l :size of lattice

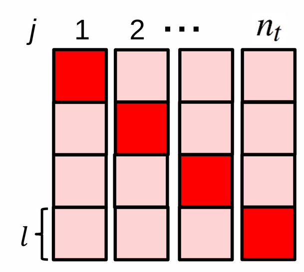
Function	# of calls	Complexity	Total complexity
getrf($\mathcal{H}$)	$\mathcal{O}(n/l)$	$\mathcal{O}(l\log^2 l)$	$\mathcal{O}((n/l)*(l\log^2 l))$
trsm(low-rank)	$\mathcal{O}((n/l)^2)$	$\mathcal{O}(l)$	$\mathcal{O}((n/l)^2 * l)$
trsm($\mathcal{H}$)	$\mathcal{O}(n/l)$	$\mathcal{O}(l\log l)$	$\mathcal{O}((n/l)*(l\log l))$
gemm(low-rank)	$\mathcal{O}((n/l)^3)$	$\mathcal{O}(l)$	$\mathcal{O}((n/l)^3 * l)$
gemm($\mathcal{H}$)	$\mathcal{O}((n/l)^2)$	$\mathcal{O}(l\log^2 l)$	$\mathcal{O}((n/l)^2 * (l\log^2 l))$

Performance of lattice $\mathcal{H}$-matrices

► **QR factorization**

- **Algorithm : Block modified Gram-Schmidt**

for $j = 1,2,\ldots,n_l$ **do**
  $[Q_j, R_{j,j}] := \mathrm{TSQR}(X_j)$
  $R_{j,j+1:n_t} := Q_j^T X_{j+1:n_t}$
  $X_{j+1:n_t} := X_{j+1:n_t} - Q_j R_{j,j+1:n_t}$
end for



- **Orthogonality**

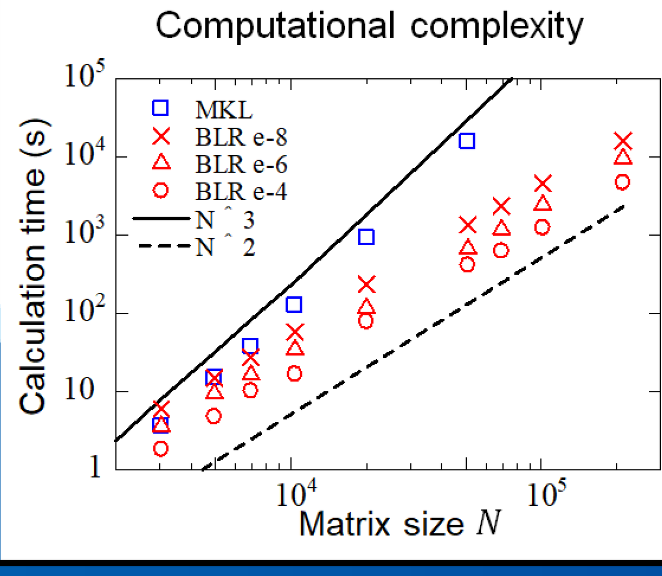
$$Q_j^T Q_k = \begin{cases} I & (j = k) \\ 0 & (j \neq k) \end{cases}$$

- **Computational Complexity analysis**

	# of calls	BLR : $O(N^2)$ if $l \propto \sqrt{N}$ — 1time	BLR — Total	Lattice $\mathcal{H}$: $O(N\log^2 N)$ if $l \propto N$ — 1time	Lattice $\mathcal{H}$ — Total
TSQR	N/l	$O(l^3)$	$O(Nl^2)$	$O(l\log^2 l)$	$O(N\log^2 l)$
$Q^T X$	$(N/l)^2$	$O(N)$	$O(N^2/l^2)$	$O(l)$	$O(N^2/l)$
$X - QR$	$(N/l)^2$	$O(N)$	$O(N^2/l^2)$	$O(l\log l)$	$O((N^2\log^2 l)/l)$

- **Numerical results in the case of BLR**

Computational complexity



Orthogonality