ExaNLA Survey Response Report

Generated on: 10/17/2025 at 12:49:09 PM

Submission Details

Library Name: PLASMA
Version: 25.5.27
Contact Name: Piotr
Email: luszczek@icl.utk.edu
Organization: MIT Lincoln Lab

Selected NLA Operations

1. Symmetric/Hermitian Eigenvalue Problems
2. Matrix-Matrix Multiplication (GEMM)
3. Cholesky Factorization

1. Codes Information

Basic information about your application/simulation codes.

Library Name:
    PLASMA

Current Version:
    25.5.27

Contact Information:
    Not specified

    Name:
        Piotr

    Email:
        luszczek@icl.utk.edu

    Organization:
        MIT Lincoln Lab

Application Domain:
    Not specified

    What is the primary application domain of your codes?:
        Other: Numerical Linear Algebra

    Materials Science:
        Not specified

        What are the main functionalities of your code?:
            Not specified

        If you selected "Other", please specify::
            Not specified

    Climate/Weather Modeling:
        Not specified

        What are the main functionalities of your code?:
            Not specified

If you selected "Other", please specify::
Not specified

Fluid Dynamics:
Not specified

What are the main functionalities of your code?:
Not specified

If you selected "Other", please specify::
Not specified

Other Domain Functions:
Not specified

What are the main functionalities of your code?:
Linear systems, least squares, eigenvalue pairs and singular triplets

Use Case Information:
Not specified

Does your codes have multiple distinct use cases?:
Yes, multiple distinct use cases

Which use case are you describing in this submission?:
linear system solve

Library Description:
The PLASMA library solves linear systems, least squares problems, and also computes either eigenvalue pairs or singular triplets. The majority of BLAS functionality is also provided.

2. Matrix-Matrix Multiplication (GEMM)

Matrix-Matrix Multiplication (GEMM):
Yes

Matrix Properties:
Not specified

Matrix Structure:
Dense matrices, Banded matrices, Triangular matrices, Tall-and-skinny matrices

Matrix Distribution:
Not specified

Matrix Storage Format:
Dense (column-major/row-major), Multiple formats (conversion as needed), Other: tile block

Which types of matrix multiplications do you perform?:
Standard multiplication (AB), Scaled multiplication (±AB), A
Full GEMM (±AB + ²C), Transpose multiplication (A@B, AB@),
multiplication (A†B, AB†)

Typical Dimensions:
Not specified

Matrix Size Range:
Very Large (10,000 - 100,000)

Typical Matrix Shapes:
Square matrices (m "H n "H k), Tall-skinny matrices (m >>
Wide-short matrices (m small, n >> k), Block-outer product (k small, m and n large), Block-inner product (m and n small, k large), General rectangular (no dominant pattern), Varies significantly by operation

Batch Size:
Not applicable

Distributed-Memory NLA Library Usage:
Not specified

General Distributed Memory Libraries (CPU/GPU):
Custom distributed implementation

Special/Advanced Implementations:
Fused operations (e.g., GEMM + bias, GEMM + activation, or custom fused kernels)

Are there any NLA libraries you are interested in using (but have not yet adopted)?:
Not specified

Future Requirements:
Not specified

Desired Features:
Better mixed precision support, Auto-tuning capabilities

Benchmarking Requirements:
Not specified

Benchmark Input Types:
Synthetic / random matrices

Can You Provide Data or Mini-apps?:
Yes, both matrices and mini-apps

Scaling Requirements:
Both strong and weak scaling needed

Working Precision:
Single precision (32-bit), Double precision (64-bit), Mixed precision (e.g., FP32 multiplication with FP64 accumulation)

3. Cholesky Factorization

Cholesky Factorization (A = LL^T):
Yes

Diagonal Dominance:
Strictly diagonally dominant

Condition Number:
Varies widely / Not known

Matrix Properties and Structure:
Dense

Matrix Distribution:
Not specified

Matrix Storage Format:
Dense (column-major/row-major), Other: tile block

Matrix Dimensions:
Very large (100,000 – 1,000,000)

Factorization Tolerance:
Machine precision

Working Precision:
Double precision (64-bit), Single precision (32-bit)

Workload Characteristics:
Not specified

Computation Pattern: capability or capacity:
Large-scale single factorizations (e.g., one large matrix at a time, using significant computational resources)

Distributed-Memory Dense NLA Library Usage:
Not specified

- Currently Used Libraries:
  Not specified

- Interested in Using, but not currently using:
  Not specified

Specialized Libraries (Sparse/Structured/Hierarchical):
Not specified

- Currently Used Libraries:
  Not specified

- Interested in Using, but not currently using:
  Not specified

Benchmarking Requirements:
Not specified

- Benchmark Input Types:
  Synthetic / random matrices

- Can You Provide Data or Mini-apps?:
  Yes, both matrices and mini-apps

- Scaling Requirements:
  Both strong and weak scaling needed

4. Standard Eigenvalue Problems (Ax = »x)

Symmetric/Hermitian

Primary Use Cases:
Kohn–Sham equations (standard DFT), GW quasiparticle calculations, Bethe–Salpeter equation (Tamm-Dancoff approximation), Tight-binding models

Matrix Properties and Structure:
Dense

Matrix Properties:
Not specified

- Matrix Distribution:
  Not specified

- Matrix Storage Format:
  Dense (column-major/row-major), Other: tile block

- Positive definiteness:
  Varies depending on the problem

- Eigenvalue distribution:
  Varies

- Problem Scale:
  Very Large (100,000 - 1,000,000)

Computation Requirements:
Not specified

- Percentage of eigenvalues:
  Varies

- What to compute:
  Varies

- Eigenvalue location:
  Varies

Required tolerance/precision:
Not specified

Residual tolerance type:
Absolute residual (||Ax - »x||)

Absolute residual tolerance:
Machine precision

Relative residual tolerance:
Not specified

Hybrid residual tolerance:
Not specified

Orthogonality tolerance:
Machine precision

Working Precision:
Single precision (32-bit), Double precision (64-bit)

Workload Characteristics:
Not specified

Computation Pattern: capability or capacity:
Large-scale single problems (e.g., one large matrix at a time, using significant computational resources)

Distributed-Memory NLA Library Usage:
Not specified

Distributed-Memory Dense Linear Algebra:
Not specified

Iterative Eigensolvers:
Not specified

High-Level & Interface Libraries:
Not specified

Are there any NLA libraries you are interested in using (but have not yet adopted)?:
Not specified

Benchmarking Requirements:
Not specified

Benchmark Input Types:
Synthetic / random matrices

Can You Provide Data or Mini-apps?:
Yes, both matrices and mini-apps

Scaling Requirements:
Both strong and weak scaling needed

5. Generalized Eigenvalue Problems (Ax = »Bx)

Symmetric/Hermitian A, SPD B

Matrix Structure:
A is dense, B is dense

Reduction to Standard Eigenproblem (using B):
Not specified

Reduction to Standard Eigenproblem:
Yes, always

Reduction Method:
Cholesky factorization of B (B = LLW or B = L*L)

Matrix Properties:
Not specified

Eigenvalue distribution:
Varies

Problem Scale:
Very Large (100,000 - 1,000,000)

Computation Requirements:
Not specified

Percentage of eigenvalues:
Varies

What to compute:
Varies

Eigenvalue location:
Varies

Required tolerance/precision:
Not specified

Residual tolerance type:
Absolute residual (||Ax - »x||)

Absolute residual tolerance:
Machine precision

Relative residual tolerance:
Not specified

Hybrid residual tolerance:
Not specified

Orthogonality tolerance:
Machine precision

Working Precision:
Single precision (32-bit), Double precision (64-bit)

Workload Characteristics:
Not specified

Computation Pattern: capability or capacity:
Asynchronous/background processing (can wait for solutions)

Distributed-Memory NLA Library Usage:
Not specified

Distributed-Memory Dense Linear Algebra:
Not specified

Iterative Eigensolvers:
Not specified

High-Level & Interface Libraries:
Not specified

Are there any NLA libraries you are interested in using (but have not yet adopted)?:
Not specified

Benchmarking Requirements:
Not specified

Benchmark Input Types:
Synthetic / random matrices

Can You Provide Data or Mini-apps?:
Yes, both matrices and mini-apps

Scaling Requirements:
Both strong and weak scaling needed