Sustainably Modeling a Sustainable Future Climate

Rabab Alomairy[1], Sameh Abdulah[2], Qinglei Cao[3], Marc G. Genton[2], David E. Keyes[2], and Hatem Ltaief[2]

[1]Computer Science & Artificial Intelligence Laboratory, Massachusetts Institute of Technology, USA,
rabab.alomairy@mit.edu
[2]Division of Computer, Electrical, and Mathematical Sciences and Engineering,
King Abdullah University of Science and Technology, KSA,
{*Firstname.Lastname*}*@kaust.edu.sa*
[3]Department of Computer Science, Saint Louis University, USA, *qinglei.cao@slu.edu*

2025 IEEE High Performance Extreme Computing Conference (HPEC) | 979-8-3315-7844-2/25/$31.00 ©2025 IEEE | DOI: 10.1109/HPEC67600.2025.11196699

***Abstract*—Voluminous highly resolved climate data requires high-performance statistical modeling frameworks that are both accurate and sustainable on modern supercomputing platforms. Spatio-temporal modeling methods, such as Maximum Likelihood Estimation (MLE), primarily rely on performing a dense Cholesky factorization of large covariance matrices, which becomes a significant bottleneck at scale. We present an enhanced version of `ExaGeoStat`, a high-performance geostatistical modeling framework, designed to address this challenge through tile-based matrix compression, mixed-precision arithmetic, and architecture-aware scheduling. Building upon our previous work on Fugaku, an Arm A64FX system, we extend our results to three supercomputers featuring diverse hardware architectures: Frontier (AMD MI250X GPUs), Alps (NVIDIA GH200 GPUs), and Shaheen III (AMD EPYC Genoa CPUs). We demonstrate that adaptive algorithms targeting low-rank and low-precision opportunities can deliver up to $4\times$ speedup, $2\times$ memory savings, and over 70% energy reduction while maintaining application-acceptable accuracy. Our work demonstrates the feasibility of running accurate climate-emulation workloads on next-generation AI hardware in a power-efficient manner, thereby advancing the sustainability of climate data science itself.**

I. Introduction

Spatio-temporal statistical modeling with Maximum Likelihood Estimation (MLE) has shifted from data limitations to computational challenges involving Cholesky factorizations of large dense covariance matrices. This factorization is performed within an optimization loop for estimating the model parameters, requiring a different matrix to be processed at each iteration. Given its quadratic memory complexity and cubic computational cost for the number of correlated observations, Cholesky factorization emerges as the primary bottleneck when attempting to leverage the newly available massive datasets from observations and simulations. This bottleneck can be addressed effectively through massive parallelism, a tile-based data structure for the matrix, adaptive tile rank, adaptive tile precision, and a dynamic runtime system to schedule tile operations and undertake the required conversions, thereby minimizing memory footprint, data motion, and computation while guaranteeing user-specified accuracy. A practical implementation of this solution is provided by the `ExaGeoStat` software [1].

The `ExaGeoStat` software was executed in 2022 on Fugaku, the world's leading supercomputer at the time, which is a CPU-only system, as detailed in a 2023 publication [2] that was a finalist for the 2022 Gordon Bell Prize. Since then, supercomputer architectures have advanced significantly, necessitating a reassessment of the implementation of MLE for climate modeling and inference. Thus, in this work, we integrate GPU support into our method by adopting a hybrid execution model that assigns tile low-rank (TLR) computations to CPUs and dense tile computations to GPUs. Furthermore, we incorporate mixed-precision (MP) techniques in both dense and low-rank computations to enhance performance and energy efficiency. To evaluate portability and architectural adaptability, we port `ExaGeoStat` to three different systems, utilizing MP arithmetic and TLR matrix approximations to exploit the strengths of each hardware platform. The software was first ported to Frontier, featuring AMD Instinct MI250X GPUs and ranked second on the TOP500 list, then deployed on Alps, ranked eighth and recognized as the top NVIDIA-powered system with Hopper GPUs. To establish a modern CPU baseline, it was also extended to Shaheen III (ranked 47th), enabling both MP and TLR+MP capabilities on dual-socket, 96-core AMD EPYC Genoa processors.

Emerging supercomputer systems offer major improvements in the sustainability of computing for climate applications, but only if applications target the architectural sweet spots. The main motivation for their high-performance low-precision matrix engines is the training of neural network models. These offer unprecedented efficiency, as measured in operations per unit of energy. The system ranked at the top of the "Green500" list has improved in energy efficiency by a factor of about 15 over the past decade. However, many climate models do not utilize this hardware efficiently. Adapting scientific computations to emerging AI-oriented hardware while preserving application-worthy accuracy is a partial answer to the threat to the climate posed by supercomputing itself. It is hardly conscionable to exacerbate climate change while seeking to respond to the need to understand the changing climate. Indeed, our algorithmic advances – constituting a veritable *renaissance* in computational linear algebra – apply in many other domains beyond climate data sciences.

The intended impact of our application is to produce actionable results that support decision-makers in drawing inferences from spatial and spatio-temporal climate models.

979-8-3315-7844-2/25/$31.00 ©2025 IEEE
Authorized licensed use limited to: SAINT LOUIS UNIVERSITY LIBRARIES. Downloaded on March 31,2026 at 21:38:23 UTC from IEEE Xplore. Restrictions apply.

Such insights are critical for a wide range of real-world applications, including the optimal sizing and siting of renewable energy infrastructure and storage based on solar and wind availability, as well as temperature and weather variability on the demand side. Our models also aid in crop planning by capturing solar and precipitation cycles, inform the scheduling and staffing of weather-dependent tourism operations, and support preparedness for natural disasters and climate-related healthcare emergencies. By enhancing the scalability and efficiency of statistical climate emulation, our work enables faster and more accurate responses to these pressing societal and environmental challenges.

The `ExaGeoStat` software statistical modeling solution has two steps: (1) a "training phase," which consists of applying MLE to fit the parameters of a statistical model using data at space-time points from observation or first-principles simulation and (2) an "inference phase," called "kriging" in statistics, which consists of estimating data at other points where it is missing and required. We concentrate here on the key computational kernel required for both phases: the Cholesky factorization of the underlying covariance matrix, which possesses as many rows and columns as the number of spatio-temporal measurements that inform it.

In the literature, modeling based on ensemble averages of first-principles simulations is a traditional means of delivering the benefits mentioned above. It relies on fundamental physics but suffers from inefficient utilization of modern hardware, operating at a low fraction of peak performance due to mismatches between memory bandwidth and computational throughput. Moreover, it requires running many simulations to achieve reliable statistical averages. In contrast, modeling based on machine learning has emerged as a new approach to achieving similar goals. While it relies less on first principles, methods such as Physics-Informed Neural Networks (PINNs) [3] can incorporate physical laws as regularizers when desired. It suffers from the need to train a massive number of parameters in a neural network – millions or billions or beyond – with the risk of overfitting, in contrast to the small number in statistical models. Indeed, neural network training and inference are responsible for the lion's share of the compound annual growth rate in US electrical consumption of approximately 18% between 2018 and 2023, with estimates ranging up to 27% between 2023 and 2028.

Our statistical modeling of climate data, called emulation, occupies a niche between simulation and machine learning. Like simulation, it uses physical knowledge, in the form of assumed correlation functions. Like machine learning, it uses contemporary tensor-core intensive computer hardware with high efficiency. All three approaches have value and complement one another. Statistical emulation offers unique advantages in terms of energy efficiency when paired with the correct algorithm for the architecture.

II. Background

Climate and weather data $\mathbf{Z}_n$ are often modeled as realizations of a Gaussian random field. The underlying spatial correlations are captured by a dense $n \times n$ covariance matrix $\mathbf{\Sigma}(\boldsymbol{\theta})$, defined via a parametric kernel $C(\mathbf{h}; \boldsymbol{\theta})$, with separation vector $\mathbf{h}$ between observations. MLE is used to estimate the parameter $\boldsymbol{\theta}$ by maximizing the Gaussian log-likelihood:

$$\ell(\boldsymbol{\theta}) = -\frac{n}{2}\log(2\pi) - \frac{1}{2}\log|\mathbf{\Sigma}| - \frac{1}{2}\mathbf{Z}_n^\top \mathbf{\Sigma}^{-1}\mathbf{Z}_n, \quad (1)$$

where evaluating the log-determinant and inverse requires Cholesky factorization and scales as $\mathcal{O}(n^3)$ in time and $\mathcal{O}(n^2)$ in memory. Prediction at m new locations is obtained from the conditional Gaussian distribution of $\mathbf{Z}_m$ given $\mathbf{Z}_n$:

$$\mathbf{Z}_m = \mathbf{\Sigma}_{mn}\mathbf{\Sigma}_{nn}^{-1}\mathbf{Z}_n. \quad (2)$$

Here, $\mathbf{Z}_m$ denotes the vector of unknown measurements of size m, while $\mathbf{Z}_n$ represents the vector of known measurements of size n, and assuming that $\mathbf{Z}_n$ has a zero-mean function [1]. We accelerate these computations by integrating MP arithmetic, TLR approximations, and GPU offloading into a Cholesky-based solver, enabling efficient prediction.

All the optimizations we applied to the underlying linear algebra operations leverage PaRSEC [4], [5], a task-based runtime system designed for executing fine-grained computations on distributed, heterogeneous architectures. Representing algorithms as Directed Acyclic Graphs (DAGs) enables asynchronous, dependency-driven scheduling that minimizes idle time and improves hardware utilization [6]. PaRSEC adapts task placement based on architecture-specific characteristics such as compute capability and memory hierarchy. To simplify algorithm expressions, it offers Domain-Specific Languages (DSLs); in this work, we use the Parameterized Task Graph (PTG) model to describe dependencies compactly through Job Data Flow (JDF) specifications, allowing for the efficient generation and execution of tasks across computing resources.

III. Related Work

Hierarchical matrix methods, including tile low-rank (TLR) approximations, have become essential in scientific computing for reducing memory usage and algorithmic complexity, especially in large-scale problems [7]–[9]. These methods exploit the structure of the matrix by compressing off-diagonal tiles up to a given accuracy threshold, enabling faster computations with reduced resource demands. However, most existing TLR approaches apply a uniform precision level across all tiles, missing opportunities to further optimize performance through precision-aware techniques.

To address this, [10] proposed a method that leverages the Frobenius norm of individual tiles relative to the global matrix norm to determine whether a tile can be safely computed in lower precision. Our previous work [2] use this idea on CPUs by combining mixed-precision with TLR to accelerate the computations. More recent studies have applied this norm-based precision adaptation to GPU architectures for dense matrices [11], exploiting hardware accelerators like NVIDIA and AMD. In this work, we advance the state of the art by enabling hybrid CPU-GPU execution, where dense tiles are offloaded to GPUs with adaptive precision, while low-rank tiles are processed on CPUs. The PaRSEC runtime system

979-8-3315-7844-2/25/$31.00 ©2025 IEEE
Authorized licensed use limited to: SAINT LOUIS UNIVERSITY LIBRARIES. Downloaded on March 31,2026 at 21:38:23 UTC from IEEE Xplore. Restrictions apply.

orchestrates these computations, efficiently scheduling tasks and managing data movement across heterogeneous resources.

IV. Implementation

A. A Hybrid MP/TLR Framework for Cholesky Factorization

MP algorithms have become central to modern scientific computing, propelled by hardware advances originally designed for AI and big data workloads [12], [13]. This synergy has enabled substantial acceleration of large-scale scientific problems, including research recognized by the Gordon Bell Prize [14]–[16], and has demonstrated effectiveness in applications such as MLE-based climate prediction using Cholesky factorization [17], [18]. TLR approximations offer a complementary algorithmic strategy that reduces both memory footprint and computational complexity by compressing off-diagonal tiles up to a target accuracy threshold, preserving the overall solution quality [19]. Traditionally, MP and TLR have been applied separately, each providing distinct advantages: MP accelerates computations via fast low-precision units, while TLR exploits matrix data sparsity to minimize data movement and resource consumption.

In this work, we combine these two paradigms within a Cholesky solver. Specifically, we assign low-rank tiles, which benefit from compression, to CPUs where memory bandwidth and flexible data structures are more efficiently managed. Dense tiles, on the other hand, are dynamically scheduled on CPUs or GPUs depending on hardware availability, leveraging MP execution to accelerate compute-bound operations and exploit fast low-precision units on modern GPUs.

Our implementation extends the `ExaGeoStat` framework [2], which we enhanced to support heterogeneous hardware targets and multiple precision (FP64/FP32/FP16). To manage the additional complexity introduced by heterogeneous data structures (dense vs. TLR), MP, and hybrid architectures, we rely on PaRSEC runtime system, which coordinates task scheduling, manages dependencies, and orchestrates execution across available resources. It also makes tile-level decisions regarding data representation (dense or TLR) and precision mode, adapting dynamically to runtime conditions and user-defined accuracy requirements. Through this integration, we maximize concurrency and resource utilization while maintaining the accuracy guarantees required by geostatistical applications. This framework opens new opportunities for high-resolution environmental modeling and prediction under tight computational and memory constraints.

B. Precision- and Structure-Aware Runtime Decisions

We extend PaRSEC to incorporate both structure-aware and precision-aware runtime decisions in the context of MP/TLR Cholesky factorization. Earlier work [2] demonstrated the benefits of MP and TLR on homogeneous CPU systems. We unify these approaches into a single adaptive framework that can target heterogeneous architectures, dispatching dense tiles to CPUs or GPUs depending on resource availability and suitability. We adopt a tile-centric heuristic guided by the Frobenius norm of each tile relative to the global matrix norm. In practice, this means PaRSEC monitors each tile's contribution to the matrix norm and selectively lowers precision where error propagation remains within application tolerance. Dense computational kernels handle on-the-fly data conversion to match operand precision with the receiver side.

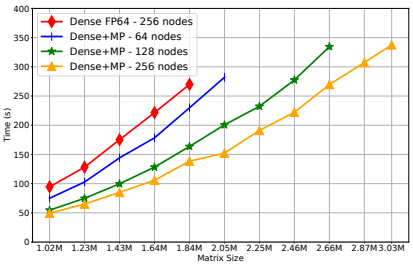
For structure-aware decisions, we implement heuristics to assess tile rank before factorization. If a tile's rank is too high, indicating poor compressibility, it is converted back to dense format. Conversely, low-rank tiles remain compressed, thereby reducing both the memory footprint and computational cost. This approach couples precision- and structure-aware decisions allowing PaRSEC to simultaneously balance computational complexity, projected time-to-solution, and accuracy requirements. The extended runtime also manages task placement and scheduling across heterogeneous resources. It monitors system load and hardware capabilities to dynamically assign dense tiles with high compute intensity to GPUs or CPUs, while coordinating data movement and precision adjustments transparently.

V. Performance Results

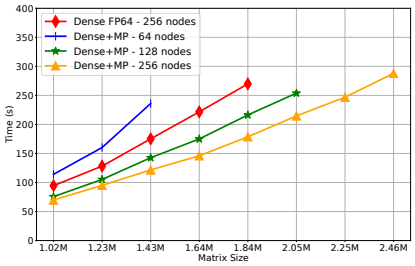
To illustrate the benefits of adapting a traditional algorithm to emerging architectures, we conducted a series of experiments on three systems beyond our original Gordon Bell finalist submission [2] (which reported results on Fugaku only). We generate synthetic matrices with varying correlation strengths, based on the largest size that can be accommodated in full precision in distributed memory. Maintaining that size, we progressively weaken the correlations and observe how much memory and time are saved by adapting the tiled matrix data structure to the problem's physics. We repeat the process with successively smaller sizes to quantify how the savings vary across different sizes. To gain a better understanding of the context of this section, we recommend reading [2].

We performed experiments on ORNL's Frontier supercomputer, ranked second on the TOP500 list[1]. Frontier comprises $9{,}472$ nodes, each equipped with four AMD MI250X GPUs. Each GPU is a multi-chip module containing two AMD Graphics Compute Dies and four terabytes of flash memory. Frontier has a theoretical peak performance of 2.06 EFlop/s in double-precision (DP) operations. Alps at CSCS in Lugano, Switzerland, is ranked eighth on the TOP500. While the system offers diverse resources, we focus on the Grace-Hopper partition, which consists of $2{,}688$ CPU-GPU supernodes. Each supernode includes four NVIDIA GH200 processors, each featuring a 72-core Arm CPU and an NVIDIA H100 Tensor Core GPU with 96 GB of HBM3 memory. This partition has a theoretical peak performance of 353.75 PFlop/s in DP operations. Finally, Shaheen III at KAUST, ranked 47th on the TOP500, comprises $4{,}608$ compute nodes, each equipped with dual-socket, 96-core AMD EPYC processors, and offers a theoretical peak performance of 39.61 PFlop/s.

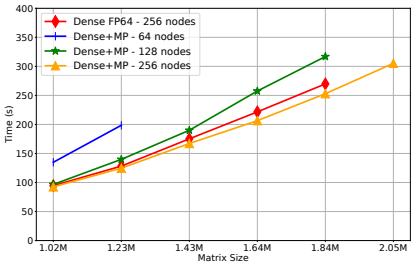
979-8-3315-7844-2/25/$31.00 ©2025 IEEE
Authorized licensed use limited to: SAINT LOUIS UNIVERSITY LIBRARIES. Downloaded on March 31,2026 at 21:38:23 UTC from IEEE Xplore. Restrictions apply.



(a) Weak Correlation (WC).



(b) Medium Correlation (MC).



(c) Strong Correlation (SC).

Fig. 1: Performance of the Matérn 2D space model with varying correlation levels (weak, medium, and strong) using 64/128/256 nodes of Alps with a total of 256/512/1024 GPUs, respectively.

A. *Performance on GPU-based Systems: Alps and Frontier*

Due to the lack of TLR implementations on GPU-based systems, we apply only MP solution for our problem on Alps and Frontier. To illustrate the memory savings achievable by leveraging low-precision cores on modern GPUs, we generate a series of model problems with random locations in a two-dimensional domain. The realizations of random fields over these observation points exhibit varying levels of correlation that adaptive precision techniques can exploit. In spatial statistics, the Matérn kernel is one of the most widely used models for spatial data. It is defined by three primary statistical parameters: variance (σ^2), range (β), and smoothness (ν). For further details, see [2]. In our study, we evaluate performance across three different levels of correlation, as characterized by β, to reflect the behavior of real-world datasets.

Performance results on Alps with GH200 Grace Hopper GPU are shown in Figure 1 for different spatial correlations and Figure 2 for a strongly correlated space-time kernel. We focus on the strong kernel here, but weak and medium space-time correlations also exhibit behavior similar to the spatial-only kernel. A comparison of timings is provided in Table I. For the AMD MI250X ORNL Frontier machine, Figure 3 illustrates the performance of the weak correlation space-time kernel across various node configurations and problem sizes, scaling up to a matrix size of 3.26M.

1) Memory Saving: Weaker correlations can exploit smaller precisions, where full-precision mantissas would "fall off the end" of arithmetic operations with disparate magnitude tile arguments. This adaptation reduces memory requirements, allowing larger datasets (in terms of the number of floating-point words of all sizes) to reside higher in the memory hierarchy. By gradually weakening the correlation from Strong Correlation (SC) to Medium Correlation (MC) and finally to Weak Correlation (WC), the memory efficiency improves.

With traditional full-precision as the baseline, on a system with 64 nodes (256 GPUs), SC matrices save 20% of the memory compared to full precision. Similarly, at 128 nodes (512 GPUs), SC matrices save 12.5%, and at 256 nodes (1024 GPUs), they save 11%. These savings are meaningful and allow larger matrices to fit into memory while preserving,

[1]https://TOP500.org/, June 2025

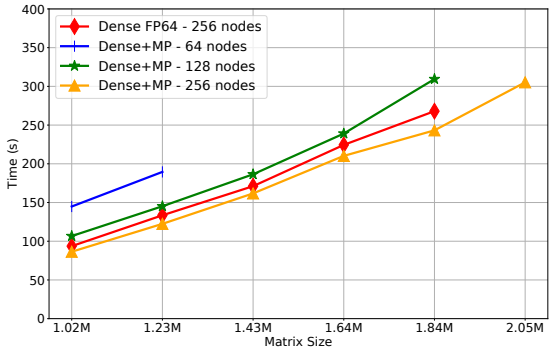


Fig. 2: Performance of a strongly correlated Matérn 2D space-time system using 64/128/256 nodes of Alps with a total of 256/512/1024 GPUs, respectively.

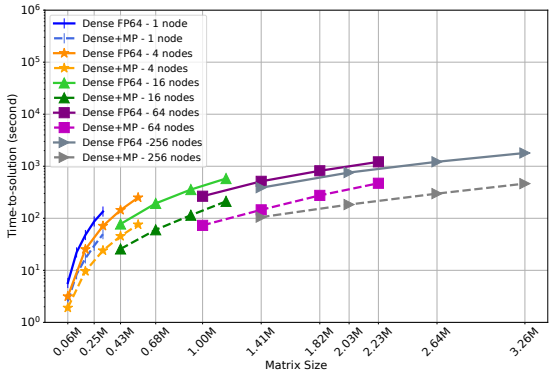


Fig. 3: Performance of Matérn 2D space-time model with weak correlation for varying matrix sizes on different numbers of nodes with a total number of 8, 32, 128, 512, and 2,048 GPUs, respectively, comparing Dense, and Dense+MP approaches on Frontier.

taking into account the relatively strong spatial coupling. Adaptive precision for MC matrices saves 40% memory at 64 nodes, 25% at 128 nodes, and 22% at 256 nodes. At 64 nodes, WC matrices achieve 80% memory savings, effectively doubling the matrix size that can fit into memory compared to full-precision. At larger configurations, WC matrices save 62.5% memory at 128 nodes and 67% at 256 nodes. Across

979-8-3315-7844-2/25/$31.00 ©2025 IEEE
Authorized licensed use limited to: SAINT LOUIS UNIVERSITY LIBRARIES. Downloaded on March 31,2026 at 21:38:23 UTC from IEEE Xplore. Restrictions apply.

TABLE I: Execution times for different correlation strengths for the largest full-precision matrix across GPU configurations on Alps.

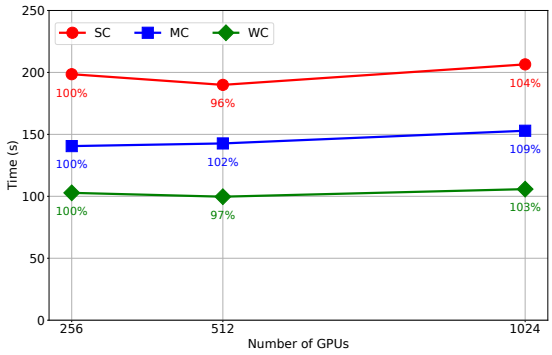
Nodes	GPUs	Matrix size	Time (s) Dense (FP64)	Time (s) SC (Dense+MP)	Time (s) MC (Dense+MP)	Time (s) WC (Dense+MP)
64	256	1.024M	159.27	134.84	114.05	74.95
128	512	1.64M	266.49	239.04	174.99	128.28
256	1024	1.84M	262.37	247.44	178.69	138.41

all configurations, WC matrices enable handling matrices up to twice as large as those in full-precision without appreciable sacrifice in accuracy [2], showcasing their potential for addressing large-scale problems.

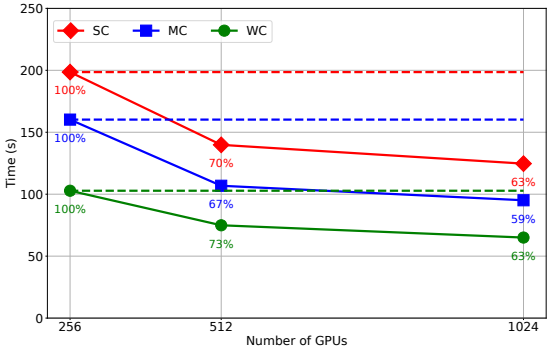
2) Time Saving: An analysis of execution time for a matrix of fixed dimensions on Alps, with precision adapted to varying correlation levels, reveals a clear trend of performance improvement, as reported in Table I. SC matrices provide modest speedups, with execution times reduced by up to 13% compared to full-precision representation. MC matrices show more substantial improvements, offering speedups in the range of 26-34%. WC matrices achieve the most significant gains, with execution times reduced by up to 52%, depending on the system configuration. These improvements stem from the reduced computational overhead associated with weaker correlations, allowing more efficient data handling and computation.

At smaller scales, such as 64 nodes with 256 GPUs, the performance benefits are most pronounced, with WC matrices achieving execution times over 51% faster. Even at larger scales, such as 256 nodes with 1024 GPUs, WC matrices maintain substantial speedups of around 47%, highlighting their scalability and efficiency. These results demonstrate that WC matrices maximize memory savings and significantly accelerate computation, making them an ideal choice for large-scale systems where both performance and resource efficiency are critical. By leveraging MP and adapting to the correlation level, this approach enables faster computation for large datasets while maintaining scalability across varying hardware configurations. Figure 3 also reports a significant performance gain when engaging matrix engines of AMD GPUs on Frontier with low-precision computations using the Dense+MP configuration on up to 256 nodes (a total of 2,048 GPUs), as opposed to Dense FP64 only, reaching up to a 4.1X performance speedup.

3) Weak and Strong Scalability: Figure 4 illustrates the weak and strong scalability on 256, 512, and 1,024 GPUs of Alps. For weak scalability, the efficiency shows only a slight drop, reaching 59% for strong correlation on 1,024 GPUs. The results demonstrate good strong scalability, achieving an efficiency of 63%, which is close to that of full-precision computations (red line). We believe this performance can be further improved by enabling CUDA-aware MPI within the PaRSEC dynamic runtime system and leveraging Remote Direct Memory Access (RDMA).



(a) Weak scalability.



(b) Strong scalability.

Fig. 4: Weak and strong scaling of the Matérn 2D space-time model with varying correlation levels (weak, medium, and strong) on Alps using Dense+MP.

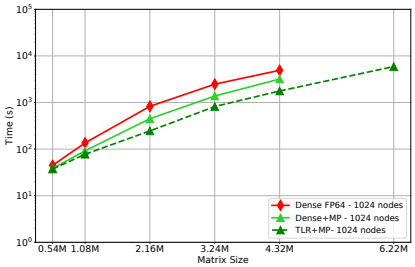
B. Performance on an x86-based System: Shaheen III

Introducing TLR into the MLE workflow accelerates computation by applying low-rank approximations per tile. To demonstrate the advantages of incorporating TLR, we adopt the coupled (TLR+MP) algorithm on CPU-based systems, specifically on Shaheen III, and augment our previous results on the ARM-based Fugaku system. Like MP techniques, we observe even greater benefits from TLR when the spatial correlation is weaker than in scenarios with stronger correlations.

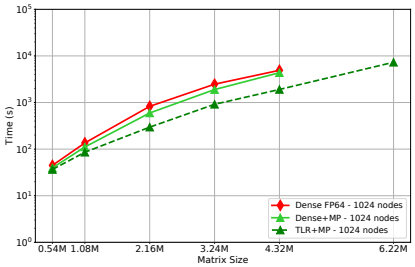
Figures 5 and 6, along with Table II, demonstrate the performance of Shaheen III under varying correlation levels in spatial and spatio-temporal scenarios. The results highlight how TLR further enhances performance, making Shaheen III competitive with the new GPU-based systems. Additionally, we analyze the system's energy consumption across different scenarios to provide a comprehensive performance evaluation.

Figures 5 and Table II compare the times for various problem sizes across node configurations (256, 512, 1,024, and 2,048 nodes). For each configuration, we select the largest matrix that can be accommodated within the memory of the nodes using FP64 representation. The results show the computation time for various correlation constraints (Strong Correlation (SC), Medium Correlation (MC), Weak Correlation (WC)) and computational approaches (Dense FP64, Dense+MP, and TLR+MP). The results emphasize the per-

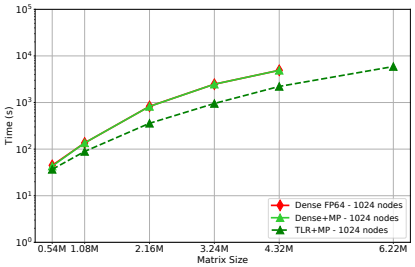
979-8-3315-7844-2/25/$31.00 ©2025 IEEE
Authorized licensed use limited to: SAINT LOUIS UNIVERSITY LIBRARIES. Downloaded on March 31,2026 at 21:38:23 UTC from IEEE Xplore. Restrictions apply.



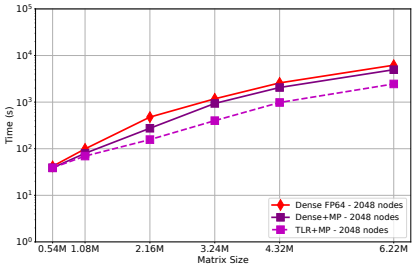
(a) Weak Correlation - 1024 nodes.



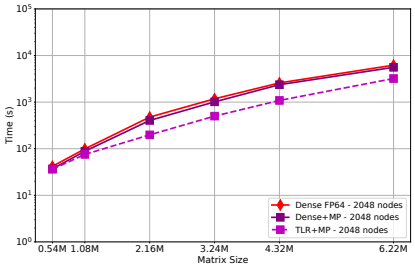
(b) Medium Correlation - 1024 nodes.



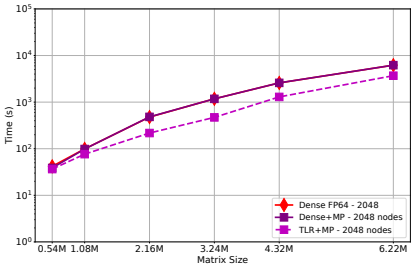
(c) Strong Correlation - 1024 nodes.



(d) Weak Correlation - 2,048 nodes.



(e) Medium Correlation - 2048 nodes.



(f) Strong Correlation - 2,048 nodes.

Fig. 5: Performance of the Matérn 2D space model with varying correlation levels (weak, medium, and strong) across different numbers of nodes of Shaheen III.

TABLE II: Execution times for varying correlation levels (SC, MC, WC) and algorithmic approaches (Dense FP64, Dense+MP, and TLR+MP) for the largest full-precision matrix across different node configurations on Shaheen III, highlighting the impact of MP on dense matrices (Dense+MP) and MP with TLR matrix approximation (TLR+MP) on performance.

	Problem	Time (s)	Time (s): SC		Time (s): MC		Time (s): WC	
# Nodes	Matrix Size	**Dense FP64**	**Dense+MP**	**TLR+MP**	**Dense+MP**	**TLR+MP**	**Dense+MP**	**TLR+MP**
256	2.16M	2859	2847	1036	2180	921	1600	891
512	3.24M	4745	4742	1791	3009	1579	2649	1470
1024	4.32M	4910	4907	2213	4361	1906	3191	1767
2048	6.22M	6202	6207	3890	5575	3206	4962	2456

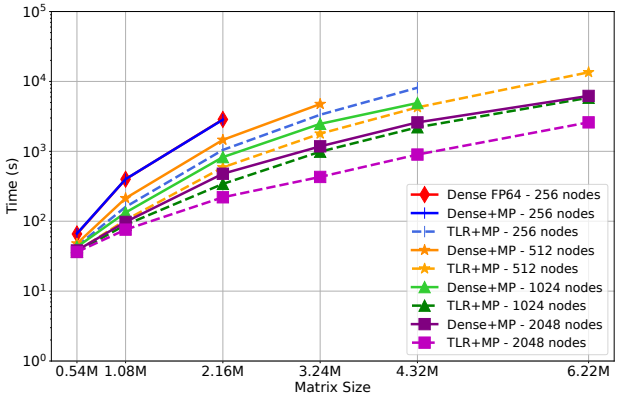


Fig. 6: Performance of a strongly correlated Matérn 2D space-time system on different numbers of nodes of Shaheen III. (The top two curves overlap.)

formance improvements possible with successively weaker correlations and applying advanced numerical techniques, such MP and TLR compression, in reducing computational time.

The impact of correlation level is evident, with Weak Correlation (WC) consistently achieving the shortest execution time. As an example, at 512 nodes using MP, the SC matrix requires 4,742 seconds, the MC matrix takes 3,009 seconds, and the WC matrix reduces this further to 2,649 seconds. Combining MP with TLR compression (TLR+MP) significantly improves performance. At 512 nodes, the WC matrix requires only 1,470 seconds with TLR+MP, compared to 1,791 seconds for the SC matrix and 1,579 seconds for the MC matrix, highlighting the combined advantages of adapting in precision (exploiting smallness) and rank (exploiting smoothness) to weaker correlations.

At 2,048 nodes, the combination TLR+MP delivers the most significant improvements in time across all configurations. Dense FP64 requires 6,207 seconds to solve a matrix of dimension 6.22M, serving as the baseline for comparison. Introducing MP, Dense+MP reduces the time for WC to 4,962 seconds, achieving a 20% reduction compared to Dense FP64. Further, applying TLR compression (TLR+MP) reduces the time even further to $2,456$ seconds, representing a total improvement of 60.4% compared to Dense FP64 and an additional 50.5% reduction compared to Dense+MP. Figure 6 confirms similar performance trends for the various configurations with spatio-temporal scenarios in the presence of strong correlations.

These results highlight the remarkable effectiveness of

979-8-3315-7844-2/25/$31.00 ©2025 IEEE
Authorized licensed use limited to: SAINT LOUIS UNIVERSITY LIBRARIES. Downloaded on March 31,2026 at 21:38:23 UTC from IEEE Xplore. Restrictions apply.

TABLE III: Time to solution for weakly correlated systems of similar problem size with a number of nodes appropriate to the minimum memory required: Dense+MP on Alps and Frontier, and TLR+MP on Shaheen III.

System	# Nodes	# PUs (GPUs/ CPUs)	Algorithms	Matrix Size	Time (s)
Alps	256	1024	Dense+MP	3.07M	337.54
Frontier	256	2048	Dense+MP	3.26M	463.90
Shaheen III	2048	4096	TLR+MP	3.24M	497.94

TLR+MP in reducing computational overhead and improving performance at large node counts. Combining MP and TLR compression enables efficient scaling to handle larger matrices while delivering significant reductions in time, making it a compelling approach for high-performance computing at scale.

C. Performance Comparisons Across Systems

Table III shows performance comparisons across the three systems: Alps, Frontier, and Shaheen III. With the algorithmic solution, TLR+MP enabled with FP64/FP32 support only, Shaheen III and its x86 technology can compete against accelerator-based systems like Frontier with matrix engines for low-precision computations, including FP16. For the minimum node resources with the memory required to accommodate the problem, Alps ultimately outperforms Frontier and Shaheen III, thanks to the highest throughput for FP64/FP32/FP16 achieved via tensor cores.

D. Energy Saving

To demonstrate the impact of Dense+MP and TLR+MP on energy efficiency compared to performing the entire computation in Dense FP64, we utilize a Matérn 2D space kernel for a matrix size of 3.24M on 512 nodes of Shaheen III. The results demonstrate substantial improvements in computation time and energy consumption by adopting Dense+MP. Dense FP64 requires the highest energy consumption at $2{,}020$ megajoules (MJ) and a running time of $4{,}746$ seconds. Switching to MP (Dense+MP) reduces energy consumption to $1{,}065$ MJ, achieving a 47.3% improvement, and decreases computation time to $2{,}650$ seconds, representing a 44.2% reduction. This demonstrates the efficiency of MP in lowering computational and energy overheads while preserving accuracy.

Including TLR compression with mixed precision (TLR+MP) further enhances these gains, reducing energy consumption to 575 MJ, a total improvement of 71.6% compared to Dense FP64 and an additional 46% reduction compared to Dense+MP. The execution time is also reduced to $1{,}471$ seconds, representing a 69% reduction compared to Dense FP64 and a 44.5% improvement over Dense+MP. These findings highlight the compounding benefits of combining MP and TLR compression.

VI. Summary and Future Work

We extend the spatio-temporal geostatistical modeling framework ExaGeoStat in two directions that track evolving opportunities in supercomputer architecture: the nearly exclusive employment of GPUs on the world's most capable systems, supplying typically well over 90% of their peak flop/s, and the shrinking precision of the highest-performing matrix engines of these GPUs. These trends are welcoming to MLE computations, particularly if memory working sets can be compressed so that the data resides within the memory available to the GPU. We demonstrate a tile-based algorithm that can exploit the mathematical structure of the vast majority of climate data sets, where, under proper ordering, the smooth decay of correlations in space and time translates into low rank in the vast majority of off-diagonal tiles and where the small magnitude of such correlations further permits the use of reduced precisions for a majority of such tiles.

We extend our demonstrations of these optimizations from Fugaku in [2] to two of the world's currently most interesting architectures: NVIDIA Grace Hopper in Alps and AMD MI250X in Frontier. We also compare them to the new AMD Epyc Genoa CPUs in Shaheen III, equipped with large last-level caches that can be leveraged with algebraic compression via tile low-rank. The latter comparison reveals the additional memory, time, and energy benefits from exploiting the TLR structure of the covariance matrix. Porting this capability to GPUs is a natural next step.

Adapting from default dense double precision to Dense+MP and TLR+MP for Cholesky factorizations over the range of covariance matrices considered yields dividends of up to 3–4X in the three critical metrics of memory footprint, time-to-solution, and energy. The performance opportunities available from reduced rank and reduced precision do not, however, extend indefinitely, because, after a point that depends on the architectural balance of the nodes and the network, the costs of data motion outweigh any further computational savings. Tile size and desired accuracy are user-set parameters that affect the potential gains from these algorithmic enhancements of a default dense high-precision Cholesky. Tuning these parameters to specific application-architecture combinations can pay off handsomely, but such tuning is beyond the scope of this short preface.

VII. Acknowledgments

These architectural and performance updates to Reference [2], *Reshaping Geostatistical Modeling and Prediction for Extreme-Scale Environmental Applications* were made possible by the gracious accommodations of CSCS in Lugano, Switzerland (special thanks, Maria Grazia Giufredda) for the use of Alps, OLCF at Oak Ridge National Laboratory (special thanks, Ashley Barker, Ryan Landfield, Veronica Vergara, and Matthew Ezell) for the use of Frontier, and the KAUST Supercomputing Laboratory (special thanks, Bilel Hadri, Habeeb Khan, and Wijdan Shahin) for the use of Shaheen III.

References

[1] S. Abdulah, H. Ltaief, Y. Sun, M. G. Genton, and D. E. Keyes, "ExaGeoStat: A high performance unified software for geostatistics on manycore systems," *IEEE Transactions on Parallel and Distributed Systems*, vol. 29, no. 12, pp. 2771–2784, 2018.

979-8-3315-7844-2/25/$31.00 ©2025 IEEE
Authorized licensed use limited to: SAINT LOUIS UNIVERSITY LIBRARIES. Downloaded on March 31,2026 at 21:38:23 UTC from IEEE Xplore. Restrictions apply.

[2] Q. Cao, S. Abdulah, R. Alomairy, Y. Pei, P. Nag, G. Bosilca, J. Dongarra, M. G. Genton, D. E. Keyes, H. Ltaief *et al.*, "Reshaping geostatistical modeling and prediction for extreme-scale environmental applications," in *SC22*. IEEE, 2022, pp. 1–12.
[3] M. Raissi, Z. Wang, M. S. Triantafyllou, and G. E. Karniadakis, "Deep learning of vortex-induced vibrations," *Journal of Fluid Mechanics*, vol. 861, pp. 119–137, 2019.
[4] G. Bosilca, A. Bouteiller, A. Danalis, M. Faverge, T. Hérault, and J. J. Dongarra, "PaRSEC: Exploiting Heterogeneity to Enhance Scalability," *Computing in Science & Engineering*, vol. 15, no. 6, pp. 36–45, 2013.
[5] A. Bouteiller, T. Herault, Q. Cao, J. Schuchart, and G. Bosilca, "Parsec: Scalability, flexibility, and hybrid architecture support for task-based applications in ecp," *The International Journal of High Performance Computing Applications*, vol. 39, no. 1, pp. 147–166, 2025.
[6] R. Alomairy, F. Tome, J. Samaroo, and A. Edelman, "Dynamic Task Scheduling with Data Dependency Awareness Using Julia," in *2024 IEEE High Performance Extreme Computing Conference (HPEC)*. IEEE, 2024, pp. 1–7.
[7] P. Amestoy, C. Ashcraft, O. Boiteau, A. Buttari, J.-Y. L'Excellent, and C. Weisbecker, "Improving Multifrontal Methods By Means Of Block Low-Rank Representations," *SIAM Journal on Scientific Computing*, vol. 37, no. 3, pp. A1451–A1474, 2015.
[8] K. Akbudak, H. Ltaief, A. Mikhalev, and D. Keyes, "Tile Low Rank Cholesky Factorization for Climate/Weather Modeling Applications on Manycore Architectures," in *32nd International Conference on High Performance, Frankfurt, Germany*. Springer, 2017, pp. 22–40.
[9] R. Alomairy, W. Bader, H. Ltaief, Y. Mesri, and D. Keyes, "High-Performance 3D Unstructured Mesh Deformation Using Rank Structured Matrix Computations," *ACM Transactions on Parallel Computing*, vol. 9, no. 1, pp. 1–23, 2022.
[10] N. J. Higham and T. Mary, "Mixed precision algorithms in numerical linear algebra," *Acta Numerica*, vol. 31, pp. 347–414, 2022.
[11] H. Ltaief, R. Alomairy, Q. Cao, J. Ren, L. Slim, T. Kurth, B. Dorschner, S. Bougouffa, R. Abdelkhalak, and D. E. Keyes, "Toward Capturing Genetic Epistasis from Multivariate Genome-Wide Association Studies Using Mixed-Precision Kernel Ridge Regression," in *SC24*. IEEE, 2024, pp. 1–12.
[12] "NVIDIA Tensor Cores," https://www.nvidia.com/en-us/data-center/tensorcore/, 2019, [Online; accessed June 2019].
[13] "Google Tensor Processing Unit (TPU)," https://cloud.google.com/tpu/, 2019, [Online; accessed June 2019].
[14] W. Joubert, D. Weighill, D. Kainer, S. Climer, A. Justice, K. Fagnan, and D. Jacobson, "Attacking the Opioid Epidemic: Determining the Epistatic and Pleiotropic Genetic Architectures for Chronic Pain and Opioid Addiction," in *SC18*. IEEE, 2018, pp. 717–730.
[15] Y. Liu, X. Liu, F. Li, H. Fu, Y. Yang, J. Song, P. Zhao, Z. Wang, D. Peng, H. Chen *et al.*, "Closing the Quantum Supremacy gap: Achieving Real-time Simulation of a Random Quantum Circuit Using a new Sunway Supercomputer," in *SC21*, 2021, pp. 1–12.
[16] S. Abdulah, A. H. Baker, G. Bosilca, Q. Cao, S. Castruccio, M. G. Genton, D. E. Keyes, Z. Khalid, H. Ltaief, Y. Song *et al.*, "Boosting earth system model outputs and saving petabytes in their storage using exascale climate emulators," in *SC24: International Conference for High Performance Computing, Networking, Storage and Analysis*. IEEE, 2024, pp. 1–12.
[17] S. Abdulah, H. Ltaief, Y. Sun, M. G. Genton, and D. E. Keyes, "Geostatistical Modeling and Prediction Using Mixed Precision Tile Cholesky Factorization," in *HiPC26)*. IEEE, 2019, pp. 152–162.
[18] M. L. O. Salvaña, S. Abdulah, H. Ltaief, Y. Sun, M. G. Genton, and D. E. Keyes, "Parallel space-time likelihood optimization for air pollution prediction on large-scale systems," in *Proceedings of the Platform for Advanced Scientific Computing Conference*, 2022, pp. 1–11.
[19] N. Al-Harthi, R. Alomairy, K. Akbudak, R. Chen, H. Ltaief, H. Bagci, and D. Keyes, "Solving acoustic boundary integral equations using high performance tile low-rank LU factorization," in *ISC20*. Springer, 2020, pp. 209–229.

979-8-3315-7844-2/25/$31.00 ©2025 IEEE
Authorized licensed use limited to: SAINT LOUIS UNIVERSITY LIBRARIES. Downloaded on March 31,2026 at 21:38:23 UTC from IEEE Xplore. Restrictions apply.