SC22: International Conference for High Performance Computing, Networking, Storage and Analysis | 978-1-6654-5444-5/22/$31.00 ©2022 IEEE | DOI: 10.1109/SC41404.2022.00007

Reshaping Geostatistical Modeling and Prediction for Extreme-Scale Environmental Applications

Qinglei Cao[2,6], Sameh Abdulah[1,5], Rabab Alomairy[1,5], Yu Pei[2,6], Pratik Nag[1,5], George Bosilca[2,7], Jack Dongarra[2,3,4,7], Marc G. Genton[1,5], David E. Keyes[1,5], Hatem Ltaief[1,5], and Ying Sun[1,5]

[1]Extreme Computing Research Center,
Division of Computer, Electrical and Mathematical Sciences and Engineering,
King Abdullah University of Science and Technology, Thuwal, KSA
[2]Innovative Computing Laboratory, University of Tennessee, Knoxville, TN USA
[3]The Oak Ridge National Laboratory, Oak Ridge, TN USA
[4]University of Manchester, Manchester, UK
[5]{*Firstname.Lastname*}*@kaust.edu.sa*
[6]{*qcao3, ypei2*}*@vols.utk.edu*
[7]{*bosilca, dongarra*}*@icl.utk.edu*

***Abstract*— We extend the capability of space-time geostatistical modeling using algebraic approximations, illustrating application-expected accuracy worthy of double precision from majority low-precision computations and low-rank matrix approximations. We exploit the mathematical structure of the dense covariance matrix whose inverse action and determinant are repeatedly required in Gaussian log-likelihood optimization. Geostatistics augments first-principles modeling approaches for the prediction of environmental phenomena given the availability of measurements at a large number of locations; however, traditional Cholesky-based approaches grow cubically in complexity, gating practical extension to continental and global datasets now available. We combine the linear algebraic contributions of mixed-precision and low-rank computations within a tile-based Cholesky solver with on-demand casting of precisions and dynamic runtime support from `PaRSEC` to orchestrate tasks and data movement. Our adaptive approach scales on various systems and leverages the Fujitsu A64FX nodes of `Fugaku` to achieve up to 12X performance speedup against the highly optimized dense Cholesky implementation.**

***Index Terms*—Space-Time Geospatial Statistics, Climate/Weather Prediction, Task-Based Programming Models, Dynamic Runtime Systems, Mixed-Precision Computations, Low-Rank Matrix Approximations, High Performance Computing.**

I. JUSTIFICATION FOR THE GORDON BELL PRIZE

Synergistic combination of mixed-precision computations and low-rank matrix approximations. Dynamic task-based runtime system and data movement. Scalability on 48,384 `Fugaku` nodes (2,322,432 cores) for maximum log-likelihood estimation (MLE). Performance speedup up to 12X over FP64 execution while attaining application-worthy accuracy. Incorporation into path-finding software framework for geostatistical applications.

II. PERFORMANCE ATTRIBUTES

Performance Attributes	Our submission
Problem Size	Ten million geospatial locations
Category of achievement	Time-to-solution and scalability
Type of method used	Maximum Likelihood Estimation (MLE)
Results reported on basis of	Whole application
Precision reported	Double, single, and half precision
System scale	48,384 Fujitsu A64FX nodes of `Fugaku`
Measurement mechanism	Timers; Flops

III. OVERVIEW OF THE PROBLEM

Geostatistics is a means of modeling and predicting desired quantities directly from data. It is based on statistical assumptions and optimization of parameters and is often referred to as emulation, in contrast to simulation. It is complementary to first-principles modeling approaches rooted in conservation laws and typically expressed in PDEs. It may draw upon data from simulations and/or from observations. Alternative statistical approaches to predictions from first-principles methods, such as Monte Carlo sampling wrapped around simulations with a distribution of inputs, may be vastly more computationally expensive than sampling from an assumed parameterized distribution based on a much smaller number of simulations. Geostatistics is relied upon for economic and policy decisions for which billions of dollars or even lives are at stake, such as engineering safety margins into developments, mitigating hazardous air quality, siting fixed renewable energy resources, and estimating weather-dependent tourism demands. We consider herein evapotranspiration, important to agricultural irrigation and water resource management, as seen in Figure 1. Climate and weather predictions are among the principal workloads occupying supercomputers around the world and even minor improvements for regular production applications pay large dividends. A wide variety of such predictive codes have

SC22, November 13-18, 2022, Dallas, Texas, USA
978-1-6654-5444-5/22/$31.00 ©2022 IEEE

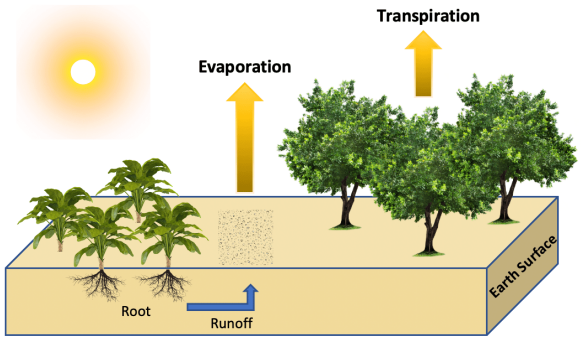


Fig. 1: Evapotranspiration for water evaporation / transpiration.

opportunistically migrated or are migrating to mixed-precision environments and can extract high throughput (Pflops/s) from special hardware components (i.e., matrix engines). However, they still suffer from a large memory footprint that may prevent them from running extreme-scale simulations. In this paper, we design a novel adaptive computing paradigm by combining mixed precisions with low-rank matrix computations and demonstrate it against one important class of such codes.

A main computational kernel of stationary spatial statistics considered herein is the evaluation of the Gaussian log-likelihood function, whose central data structure is a dense covariance matrix of the dimension of the number of (presumed) correlated observations, which is generally the product of the number of observation locations (in space or space-time) and the number of variables observed at each. The covariance matrix is symmetric and positive definite and possesses a mathematical structure arising from its physical origin that motivates approximations of various kinds for high-dimensional problems, especially in view of the demands on storage and factorization of the Cholesky formulation. Classical Cholesky bears quadratic and cubic complexities in the problem dimension for storage and arithmetic, respectively. These costs occur inside the optimization loop that fits statistical model parameters to the input data, which inhibits the application of MLE to high-dimensional problems.

Traditional approximations from the statistics community are rather severe and unrecoverable, such as setting vast proportions of presumed small elements to zero *a priori* or assuming global low-rank structure. The software at the heart of this project, `ExaGeoStat` [1], is designed to provide controllable approximations to extreme-scale MLE problems by introducing novel algorithmic, architectural, and programming model features and packing the power of supercomputers under the high-productivity statistical package R [2].

In her presentation entitled *Implementing Spatial Statistical Methods for Massive Data* in July 2019, the leader of the Statistics and Data Science Section at NCAR stated [3], "Increasing amounts of data are being produced (e.g., by remote sensing instruments and numerical models), while techniques to handle millions of observations have historically lagged behind. While a variety of statistical methods have been developed theoretically to tackle this problem, readily available computational implementations that work with irregularly-spaced observations are still rare." A dense covariance matrix of one million observations contains a trillion entries, which, in double precision, after exploiting symmetry, is a data object of 4 TB whose Cholesky factorization requires $(1/3)\times10^{18}$ flops. The aforementioned brute force reductions of this covariance matrix do not satisfy the accuracy requirements of many geostatisticians. Additional means of reducing this intractable complexity, e.g., dimension-reducing PCA approaches, the independent block approximation method [4], or blockwise or hierarchically low-rank approximations [5], to name a few, are systematically described and evaluated in [6], but lead sometimes to technical limitations that slow down their adoptions in practice.

The approach of this paper is to reshape geostatistical modeling and prediction for extreme-scale environmental applications by combining mixed-precision computations with the mathematically elegant Tile Low-Rank (TLR) matrix approximation technique [7], [8]. Based on tile algorithms [9], the resulting Cholesky factorization takes advantage of the covariance matrix structure, which under a proper ordering [10] clusters the most significant information around the diagonal of the matrix. Instead of completely ignoring the off-diagonal contributions, likely engendering a loss of accuracy, we operate on diagonal, near-diagonal and other identified tiles in double-precision while switching to lower precision and/or performing low-rank matrix approximations for the remaining off-diagonal tiles. This extends the mixed-precision algorithms presented in [11], [12] not only to accelerate the Cholesky factorization during the log-likelihood function evaluation but also to further reduce its associated storage cost. We do so through an adaptive tile-centric approach driven by a performance model and a Frobenius norm-based formula that determine at runtime the data structure (i.e., dense or TLR) and the precision arithmetic (i.e., FP64/FP32/FP16) for each tile, respectively. This synergistic combination inveighs against the predictable load balancing found in traditional dense linear algebra. We must therefore rely on a dynamic runtime system, i.e., `PaRSEC`, to schedule asynchronously tasks operating on dense or TLR data structures, executed in one of the three supported precisions, and to mitigate load imbalance overheads. We further empower `PaRSEC` by embedding necessary decision-making mechanisms to marshal data exchanges with on-demand data conversions. Adapting to application data sparsity is paramount in assessing where and how each algebraic approximation can be applied. While mixed-precision and low-rank algorithmic optimizations may translate into performance gains, we ensure numerical robustness on space and space-time real datasets by validating the accuracy of the statistical parameter estimators, which drive the modeling and the ultimate prediction phases for environmental applications.

IV. CURRENT STATE OF THE ART

A. Climate Modeling and Prediction using MLE

Spatial data associated with climate and weather applications consists of a set of locations regularly or irregularly distributed across a given specific geographical region where each location is linked with one or more climate or environmental variables, such as soil moisture, temperature, humidity, or evapotranspiration. In geostatistics, spatial data are usually modeled as a realization from a Gaussian spatial random field.

1) Space Datasets: Assume a realization $\mathbf{Z}$ of a Gaussian random field $Z(\mathbf{s})$ where $\mathbf{Z} = \{Z(\mathbf{s}_1), \ldots, Z(\mathbf{s}_n)\}^\top$ at a set of n spatial locations $\mathbf{s}_1, \ldots, \mathbf{s}_n$ in $\mathbb{R}^d$, $d \geq 1$. We assume a stationary and isotropic Gaussian random field with mean zero and a parametric covariance function $C(\mathbf{h}; \boldsymbol{\theta}) = \text{cov}\{Z(\mathbf{s}), Z(\mathbf{s}+\mathbf{h})\}$, where $\mathbf{h} \in \mathbb{R}^d$ is a spatial lag vector and $\boldsymbol{\theta} \in \mathbb{R}^q$ is an unknown parameter vector of interest. Here $C(\mathbf{h}; \boldsymbol{\theta})$ depends on the distance between any two locations and we denote by $\boldsymbol{\Sigma}(\boldsymbol{\theta})$ the discrete covariance matrix with entries $\boldsymbol{\Sigma}_{ij} = C(\mathbf{s}_i - \mathbf{s}_j; \boldsymbol{\theta})$, $i, j = 1, \ldots, n$. The matrix $\boldsymbol{\Sigma}(\boldsymbol{\theta})$ is symmetric and positive definite. Statistical inference about $\boldsymbol{\theta}$ is often based on the Gaussian log-likelihood function:

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

The modeling operation depends on computing $\widehat{\boldsymbol{\theta}}$, the parameter vector that maximizes equation (1). However, when the number of locations n is large, the evaluation of the log-likelihood function becomes computationally challenging. Computing the log determinant and carrying out the linear solve require $\mathcal{O}(n^3)$ floating-point operations (flops) on $\mathcal{O}(n^2)$ memory containing a dense n-by-n covariance matrix.

The estimated $\widehat{\boldsymbol{\theta}}$ can be used to predict missing measurements at some other location in the same region. Prediction can be performed from a multivariate normal joint distribution model with m missing measurements $\mathbf{Z}_m$ and n known measurements $\mathbf{Z}_n$ [13], [14]:

$$\begin{bmatrix}\mathbf{Z}_m \\ \mathbf{Z}_n\end{bmatrix} \sim \mathcal{N}_{m+n}\left(\begin{bmatrix}\boldsymbol{\mu}_m \\ \boldsymbol{\mu}_n\end{bmatrix}, \begin{bmatrix}\boldsymbol{\Sigma}_{mm} & \boldsymbol{\Sigma}_{mn} \\ \boldsymbol{\Sigma}_{nm} & \boldsymbol{\Sigma}_{nn}\end{bmatrix}\right), \quad (2)$$

with $\boldsymbol{\Sigma}_{mm} \in \mathbb{R}^{m\times m}$, $\boldsymbol{\Sigma}_{mn} \in \mathbb{R}^{m\times n}$, $\boldsymbol{\Sigma}_{nm} \in \mathbb{R}^{n\times m}$, and $\boldsymbol{\Sigma}_{nn} \in \mathbb{R}^{n\times n}$. The associated conditional distribution is:

$$\begin{aligned}\mathbf{Z}_m|\mathbf{Z}_n \sim \mathcal{N}_m(&\boldsymbol{\mu}_m + \boldsymbol{\Sigma}_{mn}\boldsymbol{\Sigma}_{nn}^{-1}(\mathbf{Z}_n - \boldsymbol{\mu}_n), \\ &\boldsymbol{\Sigma}_{mm} - \boldsymbol{\Sigma}_{mn}\boldsymbol{\Sigma}_{nn}^{-1}\boldsymbol{\Sigma}_{nm}).\end{aligned} \quad (3)$$

Assuming that the observed vector $\mathbf{Z}_n$ has a zero-mean function (i.e., $\boldsymbol{\mu}_n = \mathbf{0}$, hence $\boldsymbol{\mu}_m = \mathbf{0}$), the unknown vector $\mathbf{Z}_m$ can be predicted [13] by solving:

$$\mathbf{Z}_m = \boldsymbol{\Sigma}_{mn}\boldsymbol{\Sigma}_{nn}^{-1}\mathbf{Z}_n. \quad (4)$$

The associated prediction uncertainty is given by:

$$\mathbf{U}_m = \text{diag}[\boldsymbol{\Sigma}_{mm} - \boldsymbol{\Sigma}_{mn}\boldsymbol{\Sigma}_{nn}^{-1}\boldsymbol{\Sigma}_{nm}], \quad (5)$$

where diag denotes the diagonal of a matrix. Computing the last two results requires applying the Cholesky factor of the covariance matrix during the forward and backward substitutions to several right-hand sides.

2) Space-Time Datasets: We consider a Gaussian random field $Z(\mathbf{s}, t)$ where $t \in \mathbb{R}$ represents time. It is still assumed herein to be stationary and isotropic with mean zero and a parametric covariance function $C(\mathbf{h}, u; \boldsymbol{\theta}) = \text{cov}\{Z(\mathbf{s}, t), Z(\mathbf{s}+\mathbf{h}, t+u)\}$, where $u \in \mathbb{R}$ is a temporal lag. Then $\boldsymbol{\Sigma}_{ij} = C(\mathbf{s}_i - \mathbf{s}_j, t_i - t_j; \boldsymbol{\theta})$. The same equations (1)-(5) can be used but now the discrete dimension of $\boldsymbol{\Sigma}(\boldsymbol{\theta})$ is larger due to the temporal dimension; hence the arithmetic complexity is even more challenging in the space-time setting.

3) Covariance Functions: Constructing a corresponding covariance matrix $\boldsymbol{\Sigma}(\boldsymbol{\theta})$ for a set of given locations in MLE modeling or prediction operations requires defining a covariance function to describe the correlation over a given distance matrix. The Matérn family [15] has shown its utility on a wide variety of applications, for example, geostatistics and spatial statistics [16] and machine learning [17].

For the space-time case, a popular covariance function model proposed in [18] has the form:

$$C(\mathbf{h}, u) = \frac{\sigma^2}{a_t|u|^{2\alpha}+1}\mathcal{M}_\nu\left\{\frac{\|\mathbf{h}\|/a_s}{(a_t|u|^{2\alpha}+1)^{\beta/2}}\right\}, \quad (6)$$

where $\mathcal{M}_\nu$ is the univariate Matérn correlation function parametrized with ν, $\sigma^2 > 0$ is the variance parameter, $\nu > 0$ and $\alpha \in (0, 1]$ are the smoothness parameters in space and time, respectively, $a_s, a_t > 0$ are the range parameters in space and time, respectively, and $\beta \in [0, 1]$ is the space-time interaction parameter. When $\beta = 0$, it factors into purely spatial and purely temporal components and the model is classified as separable. Nonzero values of β imply that the correlation structure in space relates to some degree with that in time. The resulting models when $\beta > 0$ are termed nonseparable and deemed more realistic for real-world applications.

B. A Three-Decade (R)Evolution in Dense Linear Algebra

Dense matrix algorithms (e.g., the Cholesky factorization) have witnessed an exponential performance enhancement over the last 30 years, closely following the hardware trend in HPC with HPL as the proxy benchmark. Taking advantage of the surface-to-volume effect characterized by compute-bound workloads, such algorithms achieve sustained performance close to theoretical performance peak. With the advent of manycore architectures, these bulk-synchronous algorithms were redesigned to expose more concurrency using fine-grained computations, expressed via task-based programming models. The resulting tile algorithms [9] can be represented as Directed Acyclic Graphs (DAGs), where nodes and their edges correspond to computational tasks and their data dependencies, respectively. The recent hardware development in mixed-precision floating point arithmetic supported by dedicated matrix engines to accelerate AI workloads has created opportunities for traditional dense linear algebra algorithms to exploit fast matrix operations performed in lower precisions. While this improves absolute performance in dense linear algebra, it does not address the curse of dimensionality encountered when solving large dense problems. This is where TLR matrix approximations [7], [8] come to the rescue by leveraging the

data sparsity structure of the matrix operator to reduce not only the algorithmic complexity but more importantly the memory footprint, allowing more of the workingset to dwell in fast memory. It becomes critical to employ a dynamic runtime system framework to address the challenges in orchestrating these tasks, intrinsically heterogeneous by their data structures (dense or TLR) and their actual precisions (i.e., IEEE 754 FP64/FP32/FP16).

C. Dynamic Runtime Systems

Task-based dynamic runtimes have been introduced to schedule fine-grained computational tasks onto the underlying hardware resources. They execute tasks in an asynchronous fashion and break out from the overly constraining bulk-synchronous programming model. They target shared- and distributed-memory systems, possibly equipped with GPU accelerators. They reduce process idle time during the execution of imbalanced workloads. They implement various scheduling heuristics to reduce remote and expensive data movement, while favoring data locality. In particular, `StarPU` [19] and `OmpSs` [20] provide a convenient task-insertion API that abstracts the hardware complexity. The user is still in charge of ensuring sequential numerical correctness of the task-based code before these runtimes proceed with the scheduling onto parallel resources. This separation of concerns has enabled wide adoption of these runtimes in the community, which eventually led to the support of tasks into the `OpenMP` standard. These runtimes build the DAG dynamically and unroll it as computational progress occurs. However, both runtimes suffer from sequential task insertion, which along with the DAG pruning phase, may have potential limitations on scalability [21]. There are numerous runtimes (HPX [22], Legion [23], Charm++ [24], etc.) that employ asynchronous many-tasking executions. In this paper, we focus on `PaRSEC` [25], which relies on a domain-specific language that represents the entire DAG in a compressed and parametrized manner. This permits to identify and leverage collective communications that are crucial in dense matrix algorithms. `PaRSEC` decouples the data distribution from the actual task operations. This level of abstraction provides the technical basis for our novel research contributions.

V. INNOVATIONS REALIZED

A. Combining MP with Dense/TLR for Cholesky-based Solver

The rise of mixed-precision (MP) algorithms in scientific computing rides the technology wave of machine learning and analytics on big data problems. Because of the convergence of HPC and big data [26] at the forefront of digital innovation in, e.g., healthcare, security, and climate/weather modeling, hardware vendors have invested heavily during the last decade in designing chips that provide fast and energy-efficient low precision floating-point units [27], [28]. This hardware/algorithm synergism for MP has proved to be a game-changer for solving challenging scientific problems, such as those necessary for Gordon Bell submissions [29]–[31]. Moreover, MLE-based climate/weather prediction applications can benefit from MP [11], [12] during the Cholesky factorization, as shown in Fig. 2(a-c).

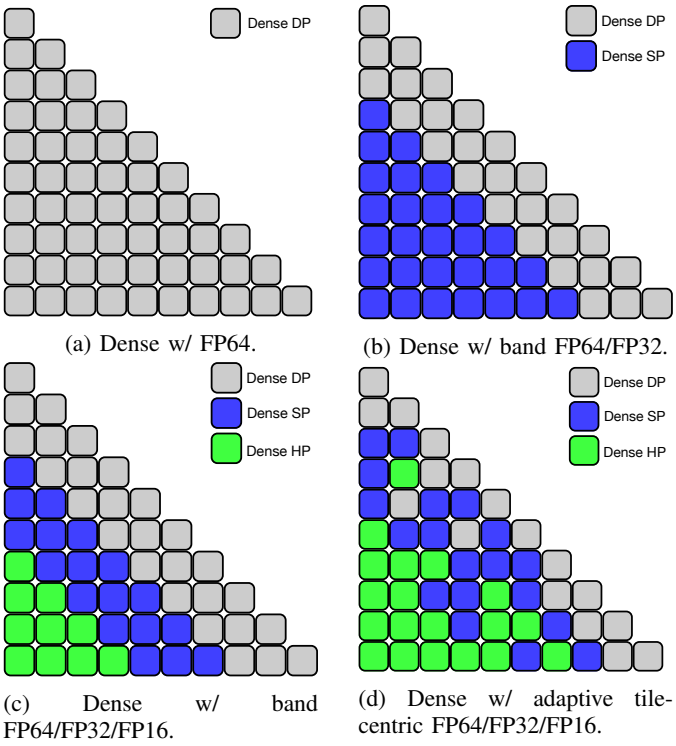
TLR matrix approximations make up yet another algorithmic class that further reduces memory footprint and algorithmic complexity [7], [8]. TLR approximations exploit data sparsity within the matrix operator by compressing off-diagonal tiles up to a target accuracy threshold, without degrading the overall accuracy sought from the solution beyond a user-specified tolerance. TLR has the potential to dramatically improve the resolution and accuracy of climate/weather predictions for a given memory footprint and computer time allocation, as illustrated in Fig. 3(a-b) of [32].

These two computing paradigms (i.e., MP and Dense/TLR) have thus far been considered as *separate swim lanes*. They have independently mobilized researchers to identify opportunities within legacy numerical algorithms and to leverage hardware features accordingly. The overarching opportunity is to determine which computational phases within an algorithm are amenable to algebraic approximations or lower precision execution while maintaining and delivering the required level of accuracy for the final result. Given this fertile landscape, we are migrating geostatistics modeling and prediction to MP *and* TLR matrix algorithms, synergistically extracting benefits of each.

Geostatistics augments first-principles modeling approaches for the prediction of environmental phenomena, given the availability of measurements at a large number of locations. One of the main computational kernels is the evaluation of the Gaussian log-likelihood function performed via MLE. The required evaluation of the determinant of the covariance matrix and the application of its inverse translate into iteratively solving a number of large systems of linear equations. The large dense symmetric and positive definite covariance matrix can be processed with the dense MP+TLR Cholesky factorization to reduce memory footpring and algorithm complexity and enable solving space-time geospatial datasets otherwise intractable. We extend the `ExaGeoStat` framework [1] to simultaneously execute MP and TLR operations within the Cholesky factorization on the covariance matrix. We rely on the `PaRSEC` dynamic runtime system not only to orchestrate the task scheduling but also to supervise the tile-centric decision for the choice of the data structures (dense or TLR) and precision (FP64/FP32/FP16).

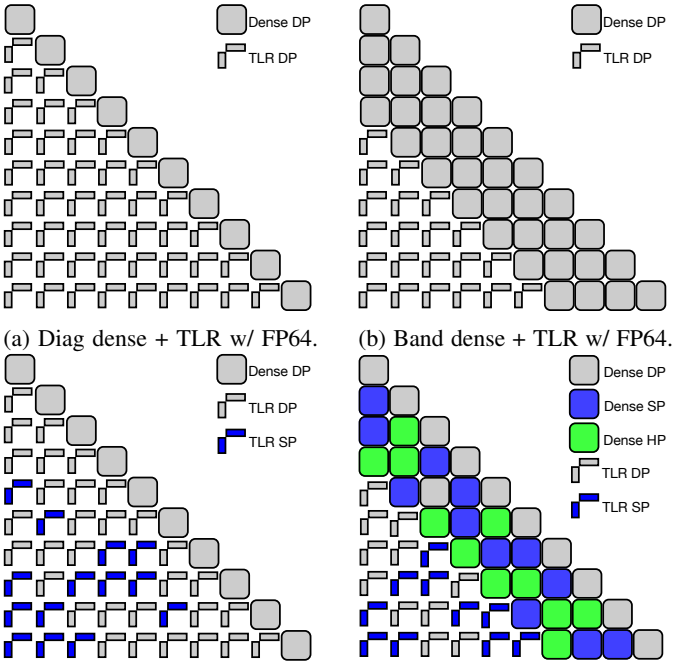
B. Empowering `PaRSEC` with Structure-Aware and Precision-Aware Runtime Decisions

We extend the features of `PaRSEC` [25], a dynamic, task-based runtime system that takes advantage of the concise Parameterized Task Graph DSL [33]. `PaRSEC` allows us to focus on algorithmic features and program in a style independent of the data distribution to reach unprecedented levels of efficiency for solving extreme-scale linear algebra matrix operations [25]. This effort is part of a larger ongoing research collaboration, as earlier described in [12], [32], [34], [35]. We further extend this work to seamlessly combine MP+Dense/TLR Cholesky



(a) Dense w/ FP64. (b) Dense w/ band FP64/FP32. (c) Dense w/ band FP64/FP32/FP16. (d) Dense w/ adaptive tile-centric FP64/FP32/FP16.

Fig. 2: Mixed-precision dense computations.



(a) Diag dense + TLR w/ FP64. (b) Band dense + TLR w/ FP64. (c) Diag dense + TLR w/ FP64/FP32. (d) Band dense + TLR w/ FP64/FP32/FP16.

Fig. 3: Combining mixed precision and TLR approximations.

factorization with numerically-driven runtime decisions to reduce time-to-solution.

1) When to use MP: MP selection may be based in a brute force way on a band structure, as shown in Fig.2(c). This approach has been previously analyzed [11], [12] on large-scale homogeneous x86 systems, i.e., full-scale `Shaheen II` (KAUST, Saudi Arabia), half-scale `HAWK` (HLRS, Germany), and `Summit` (ORNL, USA). This fast-path approach permits to empirically determine the band size. It may engender more operations than required in case actual low precision tiles reside in a band region with high precision. Figure 2(d) shows how a tile-centric precision-aware decision can be incorporated into the dataflow of any matrix computations. It is precisely in such scenarios where user-productivity is challenged that one may need to sacrifice performance for code simplicity. However, with a Frobenius norm-based adaptive formula (see Section IV.*C*), we can lower the precision of selected tiles depending on the ratio of the tile norm to the global matrix norm. Algorithm 1 presents the MP dense Cholesky factorization composed of nested loops calling the four dense kernels, with GEMM being the most time-consuming kernel. The $^+$ sign on the parameter of the functions indicates the precision-lead operand for which the corresponding precision kernel will be called. This means that `PaRSEC` will move and convert on-the-fly the operands with the * sign to match the precision at the receiver side.

2) When to Use Dense Over TLR: We integrate into `PaRSEC` a tile-centric structure-aware heuristic to decide at runtime whether a specific tile should remain dense or approximated with TLR instead, depending on its rank. This choice of data structures is guided by a formula that estimates at runtime, right after the generation/compression of the matrix and just before the Cholesky factorization starts, the required number of flops and estimated time-to-solution. If the rank of a given tile is high enough that the overhead of compression during the Cholesky factorization is not justified, `PaRSEC` takes the decision to translate the tile back to its original dense structure. This decision leads to establishment of a band structure since tiles with high ranks are typically located around the diagonal. This process is illustrated in the matrix transformation going from Fig. 3(a) to Fig. 3(b) with a band of three tiles. This runtime decision has been previously studied on two large-scale HPC systems, i.e., full-scale `Shaheen II` [32], [34], [35] and `Fugaku` [36]. However, the decision therein did not take into account the precision of each tile as we address it here.

The structure-aware runtime decision relies only on the number of flops/time-to-solution of the actual matrix opera-

Algorithm 1: Pseudo-code of the MP Dense Cholesky Factorization.

```
for k = 0 to NT − 1 /* Panel Factorization (PF) */
    DPOTRF (C_kk^+)
    for m = k + 1 to NT − 1
        TRSM (C_kk^*, C_mk^+)
    for m = k + 1 to NT − 1
        DSYRK (C_mk^*, C_mm^+)
    for m = k + 2 to NT − 1 /* Trailing Submatrix Update */
        for n = k + 1 to m − 1
            GEMM (C_mk^*, C_nk^*, C_mn^+)
```

tions, while the precision-aware runtime decision depends only on the required accuracy of the application.

C. Supporting MP+Dense/TLR for Space-Time Datasets

In previous work [11], [12], we introduced a MP matrix algorithm (see Fig. 2(c)) to accelerate the Cholesky factorization during the log-likelihood function evaluation for 2D space models in the context of environmental applications. Since we are modeling behaviors that are continuous in time, by considering the time dimension herein, the resulting space-time model can deliver more accurate predictions than uncorrelated spatial snapshots, especially with the precision-aware tile-centric runtime decision (see Fig. 2(d)).

Supported by the flexibility, efficiency and scalability of the `PaRSEC` runtime, `ExaGeoStat` is now able to efficiently solve geospatial applications on large-scale distributed-memory systems.

VI. HOW PERFORMANCE WAS MEASURED

We validate the numerical robustness of our mixed precision dense+TLR approach by training on real datasets until convergence, followed by prediction. We report time-to-solution and scalability up to 48,384 `Fugaku` nodes of a single iteration of MLE that is a proxy of the overall simulation. We link against the vendor optimized numerical library SSL for BLAS/LAPACK routines. However, we have to disable sector cache optimizations due to an incompatibility discovered on SSL when using task-based programming model powered by MPI+Pthreads. This drops sustained node performance to 65% of peak. This has several implications, as described in Section VIII.

A. Real Dataset Descriptions

We describe two real datasets (i.e., space and space-time) that we use in this study to assess the accuracy and the performance of our adaptive approaches, i.e., MP+dense/TLR. Figure 4 shows the geographical regions for both datasets.

We consider 2D high-resolution soil moisture data on January 1st, 2004 coming from the top layer of the Mississippi River basin, U.S [37]. This dataset has around 2M spatial locations. We randomly pick 1M locations as the training dataset and 100K locations for testing.

The Evapotranspiration (ET) 2D space-time dataset, obtained from the NASA Goddard Earth Science Data Information and Services Center (GES DISC) [38], includes water evaporation and transpiration processes to move the water from the earth's surface into the atmosphere. We consider monthly aggregated data over the Central Asia Region consisting of approximately 83K spatial locations. We collected such datasets from 2001 to 2021, giving 12 spatial fields each year for 21 years. To remove the temporal trend in the data and make its mean stationary, we take the average for each location for a given month for years from 2001 to 2020 and subtract it from 2021 monthly aggregates for that exact location. This gives us 12 correlated spatial fields, which are residuals of the observations for the year 2021, yielding around 1M space-time locations. Along with the temporal stationarity, spatial stationary is also required for running a spatio-temporal Gaussian model on the data. We have fitted a simple linear regression model of the observations with the locations for each month separately and then subtracted the predictions based on this linear model from the observations. The resulting values or residuals of our variable of interest are now approximately stationary and display Gaussianity.

B. Performance Model for Structure-Aware Runtime Decision

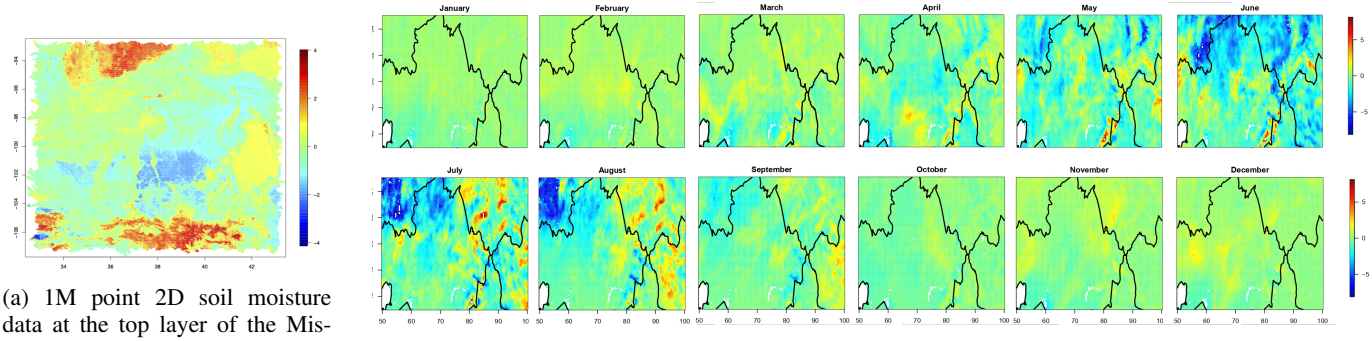
We develop and embed a performance model into `PaRSEC` to facilitate the structure-aware runtime decision (dense or TLR). This performance model is driven by the algorithmic complexity of dense GEMM (compute-bound) and TLR GEMM (memory-bound) kernels. The flops of the latter depend on the accuracy tolerance, set to $1e-8$ for this application. Figure 5 reports the time-to-solution (left y-axis) as well as its ratio (right y-axis) of a dense FP64 `GEMM` to a `TLR` FP64 `GEMM` on a single core of the A64FX node. FP64 `TLR GEMM` can be more expensive than FP64 dense `GEMM` when the rank exceeds a threshold (i.e., 200), determined by the arithmetic complexities of `TLR`. We need then to factor the actual precision of the tile into the final runtime decision, which leads to Algorithm 2 for deploying the structure-aware decision mechanism. `PaRSEC` will then automatically determine how many tiles need to be stored back in the original dense format after compression.

C. Adaptive Precision-Aware Runtime Decision

We implement in `PaRSEC` a decision that adapts the precision of a tile at runtime. Each tile A_{ij} of matrix A is stored in a precision derived by thresholding the ratio of the Frobenius norms of the tile and the global matrix [39]. Let u_{high} and u_{low} be the machine epsilons of the high precision and a given lower precision, respectively, and NT the number of tiles in one dimension. Then a tile A_{ij} with Frobenius norm less than $u_{high} * \|A\|_F/(NT * u_{low})$ may be stored in the lower precision. The matrix $\hat{A}$ thus formed satisfies $\|\hat{A} - A\|_F \leq u_{high}\|A\|_F$. (The Frobenius norm of the global matrix is accumulated tile-by-tile during generation, and a copy of the global matrix need not be stored.)

Algorithm 2: Auto-tuning `band_size_dense`.

Input : Matrix data descriptor
Generate the matrix with `band_size_dense` $= 1$
Globalize the rank distribution to all the processes
Set $ID = 1$ and initialize $fluctuation$
do
  $ID := ID + 1$
  $time\_to\_solution\_dense$ = total time-to-solution of `TRSM` and `GEMM` (FP64/FP32/FP16) of all tiles in sub-diagonal with `band_ID` $= ID$ if executing in dense format
  $time\_to\_solution\_tlr$ = total time-to-solution `TRSM` and `GEMM` (FP64/FP32) of all tiles in sub-diagonal with `band_ID` $= ID$ if executing in low-rank format
while $time\_to\\_solution\_dense < fluctuation \times time\_to\\_solution\_tlr$;
Output: `band_size_dense` $= ID - 1$



(a) 1M point 2D soil moisture data at the top layer of the Mississippi River basin.

(b) 2021 monthly total evapotranspiration (ET) residuals over the Central Asia Region.

Fig. 4: Visualizations of two real datasets in space (soil moisture) and space-time (ET) domains.

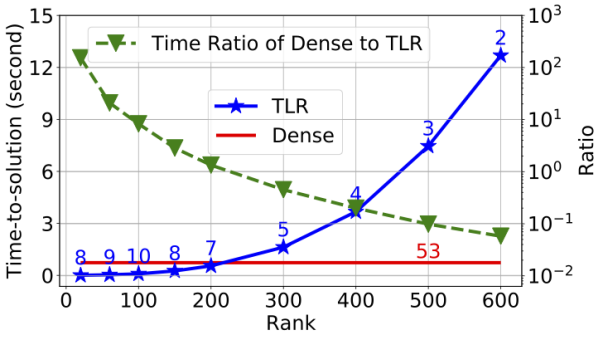


Fig. 5: FP64 `TLR GEMM` vs dense `GEMM` on `Fugaku`.

D. Strong and Weak Scaling Experiments

We provide strong scaling experiments up to 16K `Fugaku` nodes of three Cholesky variants: the reference FP64 dense, the MP dense, and the MP+dense/TLR. The first two are compute-bound and can extract high sustained peak performance. The latter variant is memory-bound and achieves only a small percentage of the theoretical peak performance. However, it may attain up to an order of magnitude speedup compared to the two dense variants. Weak scaling is of high interest when performing parallel optimization to accelerate and improve the training, as discussed in [40]. The particle swarm optimization [41] approach requires launching a set of independent executions for the log-likelihood function that allows parallel execution of the MLE operation. This translates into executing several Cholesky factorizations in an embarrassingly parallel fashion. The single tightly-connected MLEs are then synchronized in a loose manner at each iteration and the parallel optimizer procedure carries on until convergence.

E. Description of HPC Systems

`ExaGeoStat` has been ported to most of the major commercial hardware architectures of the Top10 supercomputers [12], [32], [35], including IBM+NVIDIA (Summit) and AMD (Hawk). In this paper, we run and validate our accuracy experiments on `Shaheen II`, a Cray XC40 supercomputer with 6,174 nodes composed of two-socket 16-core Intel Haswell (AVX2) processor and 128GB of main memory, using the Cray Aries network interconnect. For half precision support, we trim the operands of the GEMM kernel to FP16 and call an SGEMM BLAS routine to accumulate in FP32, similar in functionality to what cuBLAS provides for mixed precision SGEMM on NVIDIA GPUs. We run our performance benchmarking campaign on `Fugaku`, a Fujitsu ARM (SVE) system with 152,064 A64FX nodes composed of four 12-core core memory groups (CMGs) and 32GB of main memory, connected through the TofuD interconnect. `Fugaku` does not support mixed-precision HGEMM with FP32 accumulation, but only pure FP16 HGEMM.

VII. Performance Results

A. Qualitative Analysis using Synthetic Data

In this section, we assess the performance of our proposed adaptive computation approaches using synthetic datasets. The assessment consists of evaluating the ability of the proposed approaches to recover the space and the space-time model parameters. We simulate 100 space realizations at 50K spatial locations from the Matérn space 2D kernel and 100 space-time realizations at 5K spatial locations in 10 time-slots from the Matérn space-time 2D kernel. For each kernel, we have three different combinations of parameters that represent different space/space-time interactions between the spatial locations. These sets of parameters combinations have been used to generate synthetic datasets using the *ExaGeoStat* software [1].

We assess the effectiveness of an approach by its ability to recover the parameter set used to generate the synthetic dataset. Figure 6 shows boxplots of 100 synthetic datasets in the space domain. Three sets of correlation parameters are used, i.e., weak (0.03), medium (0.1), and strong (0.3). As shown from the boxplots, both the MP+dense and MP+dense/TLR approaches compete with dense FP64 in recovering the spatial parameters in most of correlation settings. For weak and medium correlations, the proposed adaptive approaches more efficiently estimate the underlying model parameters. Strongly spatially correlated data is more sensitive to precision loss engendered by the MP and/or TLR numerical approximations. This may require further tightening the tolerance for TLR or investigating new floating-point

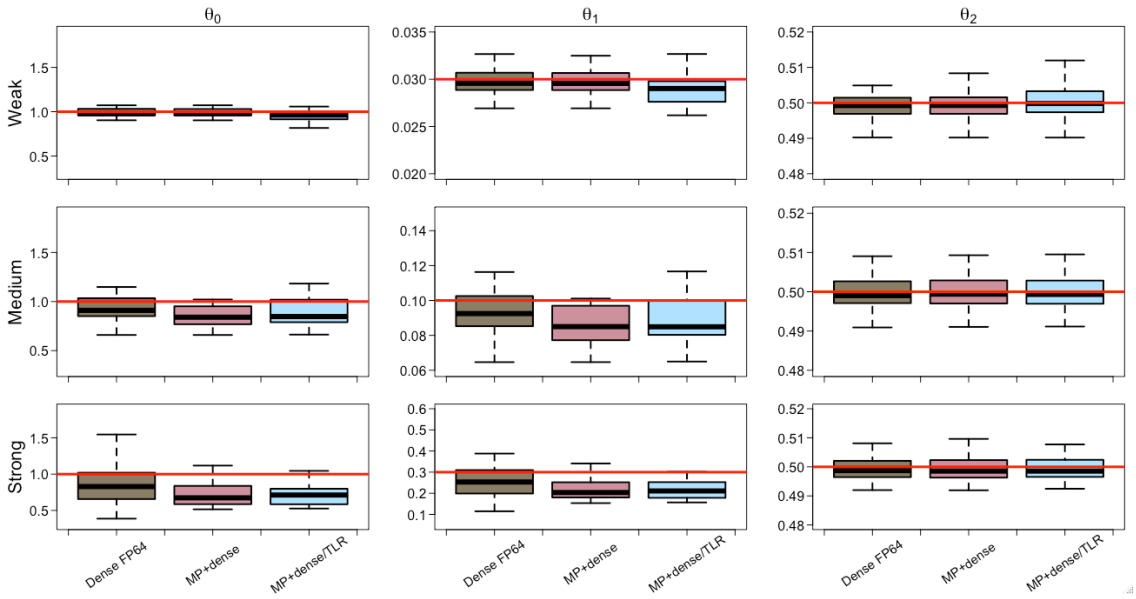


Fig. 6: Boxplots of parameter estimates using 100 samples of $50K$ synthetic space datasets with different degrees of space dependence. Weak/medium/strong correlations in space θ_1 = 0.03, 0.1, 0.3, respectively. The red line represents the true values.

representations (e.g., BF16 and TF32, not currently supported on `Fugaku`) to obtain the proper estimates.

B. Real Dataset Accuracy Analysis

In this section, we show the effectiveness of the proposed approximation methods in estimating the statistical model parameters and predicting missing values at new locations using our two main datasets, i.e., the soil moisture and the evapotranspiration (ET) datasets.

The original soil moisture dataset at the top layer of the Mississippi River basin has around 2M locations. We selected 1M spatial locations for training and 100K locations for testing. Table I reports the results of the estimation. After obtaining the estimated parameters, we predict using Equation (4) the measurements at the space locations included in the 100K testing dataset and compute the prediction errors measured by the mean square prediction error (MSPE). The results show very close estimations between the three variants, i.e., dense FP64, MP+dense, and MP+dense/TLR. Moreover, the prediction errors closely match, confirming the effectiveness of the proposed adaptive approximation methods relative to the dense FP64 default. The results identify medium space correlation between the given spatial locations ($\theta_1 = 0.15$) with rough random field ($\theta_2 = 0.44$) for this real dataset. This medium correlation gives more opportunities to represent the covariance matrix tiles in lower accuracy without impacting the overall accuracy of both the parameter estimation and the prediction.

For the ET dataset, we select 1M locations for the twelve months of the year, i.e., around 83K locations each month, for training the space-time model. The testing data (for prediction) was also selected throughout the year with the size of 100K locations. Table II shows the results of the estimated parameters and predictions error using the three computation approaches.

The parameter estimates in Table II imply several properties regarding the given space-time dataset. First, the correlation in space between different locations is very high and could be described as a strong correlation. This strong space correlation makes most of the matrix values important and increases the number of dense FP64 tiles when using the adaptive approach. Second, as shown by the table, the estimation coming from the two approximation approaches is very close to the result of the dense FP64 variant, and this is also shown by the MSPE values on the given testing dataset, which demonstrates better tolerance to precision loss for the prediction phase. Third, the non-separable parameter represents the interaction between the space and the time dimensions. In the ET dataset, the estimated value shows medium interaction, i.e., 0.18. Some studies drop this value to reduce the complexity of the optimization process from six parameters to five. However, it may dramatically impact the prediction accuracy as illustrated in [40].

C. GEMM Performance Evaluation on A64FX

It is crucial to assess the GEMM performance kernel on A64FX. The Fujitsu SSL numerical library provides GEMM performance for the three precisions close to the theoretical peak, thanks to Sector Cache Optimizations (SCO) that allocate a fast throughput region on HBM2 to stream data to registers. SCO makes a strong assumption that GEMM calls can only be made in parallel multithreaded OpenMP. Task-based programming models usually rely on sequential task performance. Tasks are then executed in an asynchronous parallel fashion following the data dependencies captured in the DAG. We therefore had to disable SCO due to memory conflicts that engender a segfault. Our parallel implementation

TABLE I: Qualitative assessment of the MLE based on the adaptive approach using the soil moisture 2D space dataset.

Approach	Variance (θ_0)	Range (θ_1)	Smoothness (θ_2)	Log-Likelihood (llh)	MSPE
Dense FP64	0.6720	0.1730	0.4358	-52185.7336	0.0330
MP+dense	0.6751	0.1740	0.4357	-52185.7643	0.0330
MP+dense/TLR	0.6621	0.1882	0.3921	-52188.2341	0.0332

TABLE II: Qualitative assessment of the MLE based on the adaptive approach using the ET space-time dataset.

Approach	Variance (θ_0)	Range (θ_1)	Smoothness (θ_2)	Range-time (θ_3)	Smoothness-time (θ_4)	Nonsep-param (θ_5)	Log-Likelihood (llh)	MSPE
Dense FP64	1.0087	3.7904	0.3164	0.0101	3.4890	0.1844	-136675.1	0.9345
MP+dense	0.9428	3.8795	0.3072	0.0102	3.4941	0.1860	-136529.0	0.9348
MP+dense/TLR	0.9247	3.7756	0.3068	0.0102	3.5858	0.1857	-136541.8	0.9428

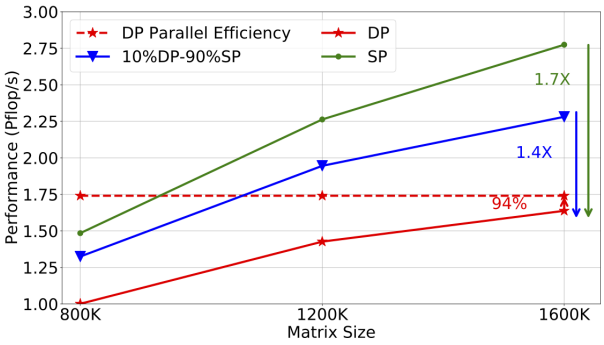


Fig. 7: Mixed-precision Cholesky on 1024 nodes; tile size 800.

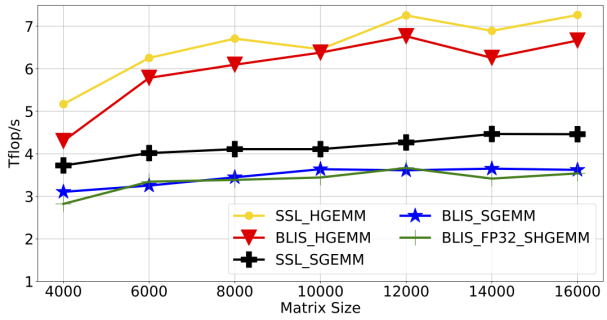


Fig. 8: BLIS and SSL single node performance with Pthreads.

of GEMM calling sequential GEMM kernels via Pthreads scores only 65% of the peak performance on single node, as opposed to 90% when running SSL multithreaded GEMM with SCO enabled.

Even with SCO disabled, we reach a performance efficiency using Dense FP64 on 1024 Fugaku nodes up to 94% compared to single node performance, as shown in Figure 7. Enabling mixed precision increases the performance throughput as expected. We omit the FP16 HGEMM since our MLE application requires HGEMM with FP32 accumulation. This kernel is not available in SSL at this time.

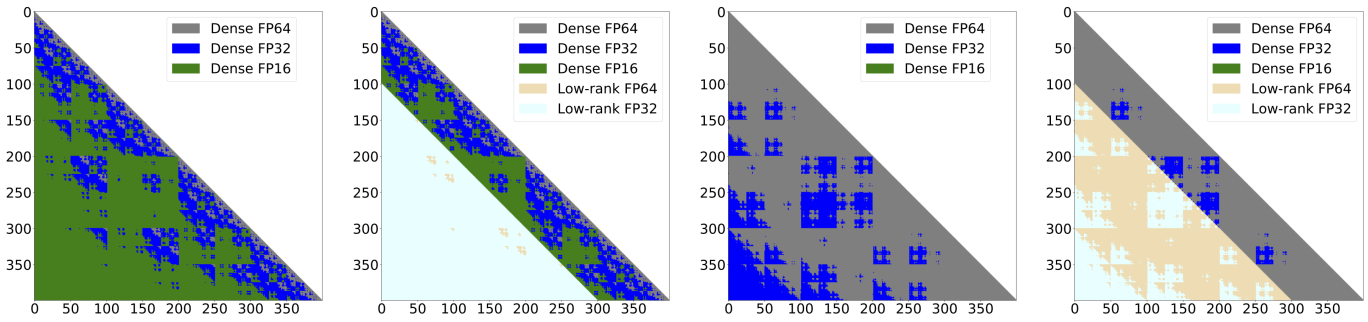
BLIS developer Ruqing Xu recently provided a missing FP32 SHGEMM kernel. Figure 8 shows performance comparisons against SSL without SCO. This enables us to reach the accuracy needed for MLE. In terms of performance, BLIS FP32 SHGEMM achieves lower performance than SSL regular SGEMM. This may be improved with proper hardware support, as seen on NVIDIA GPUs. Therefore, we fall back to SGEMM from SSL for performance, without trading off accuracy.

D. Matrix Heatmaps of Precision/Structure-Aware Decisions

Figure 9 shows matrix heatmaps of our structure-aware and adaptive precision-aware runtime decisions for weak/strong correlations (WC/SC) for Matérn 2D space on a matrix of dimension 1M. We see that WC creates more opportunities to compute in lower precision computations than SC for two Cholesky variants in MP+dense and MP+dense/TLR. Thanks to its precision-aware adaptivity, PaRSEC does much more than task scheduling. PaRSEC engages in the execution of the application in a holistic manner and converts the operands to the corresponding precision on-demand. The memory footprint (MF) to store the dense matrix achieves up to 79% (MP+dense/TLR) reduction compared to dense FP64 (MF=4356GB).

E. Time-to-Solution of Various Cholesky Variants

Figure 10 reports time-to-solution of the Matérn 2D space on 2048, 4096, 8192, and 16384 Fugaku nodes using weak/medium/strong correlations (left to right). While the dense FP64 Cholesky variants are compute-bound and can achieve high absolute performance, they are not competitive with memory-bound MP+dense/TLR Cholesky variants in terms of elapsed time. Moreover, the memory footprints of the dense variants are prohibitive in solving large-scale geostatistics applications and can only handle the smaller matrix sizes considered. If we assume we have enough memory capacity to host a matrix of size 9M in pure FP64, the MP+dense/TLR variant of Cholesky can achieve up to 12X speedup on 16K Fugaku nodes using weak correlations, which is representative of our soil moisture real datasets. Figure 11 shows the performance of the Matérn 2D space-time kernel using strong correlation on 4096 and 48384 Fugaku nodes. This is representative of our ET real datasets. The MP+dense/TLR approach achieves up to slightly less than an order of magnitude speedup compared to pure dense FP64 on 4096 nodes, since ranks are higher and opportunities for low precision computations are rare. The performance superiority of MP+dense/TLR approach compared to pure dense FP64 on 48384 nodes is further reduced due to strong scalability limitations, i.e., there may not be enough tasks to keep the computational resources busy. Exposing more tasks and optimizing PaRSEC runtime system are both areas



(a) WC, `MP+dense` (MF 1607GB).
(b) WC, `MP+Dense/TLR` (MF 915GB).
(c) SC, `MP+dense` (MF 3877GB).
(d) SC, `MP+Dense/TLR` (MF 1830GB).

Fig. 9: Adaptive decision map for Matérn 2D space on 1M matrix with tile size 2700 using Weak/Strong Correlations (WC/SC).

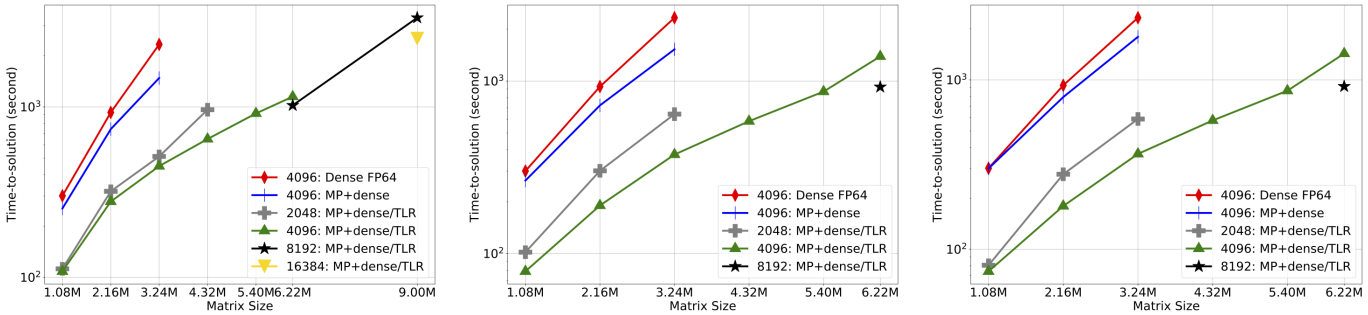


Fig. 10: Performance of Matérn 2D space on 2048, 4096, 8192, and 16384 `Fugaku` nodes. WC; MC; SC.

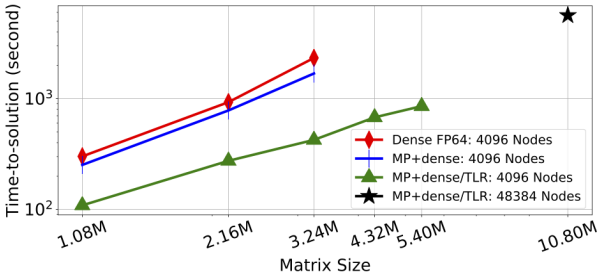


Fig. 11: Performance of Matérn 2D space-time of strong correlation on 4096 and 48384 `Fugaku` nodes.

for improvements to pursue in future work. However, the gain in memory footprint remains significant, allowing to handle larger problem sizes for the same allocated resources. Moreover, once decent strong scaling is attained for a given problem size, hybrid parallelization with weak scaling can be triggered using particle swarm optimization, as studied in [40], reaching effectively full `Fugaku` scale.

VIII. IMPLICATIONS

Tile low-rank and mixed-precision algorithms for problems dominated by covariance matrix operations can be applied across a wide variety of scientific domains beyond environmental applications. This campaign is one of many ongoing that demonstrate how traditional HPC applications may turn out to be resilient to precision loss in specific computational phases. Many calls to "blackbox" linear algebra routines inside of applications that dominate supercomputer centers likely oversolve, relative to the precision needed to fully exploit the resolving power of the discretization or the output requirements. The APIs of much scientific software should be extended to support MP and TLR, in particular in the ARM software ecosystem. Identification of the operations within an application that are resilient with respect to precision loss requires interdisciplinary interaction. Moreover, adaptive tuning of file rank and precision puts new demands on runtime load balancing for scaling to distributed and shared memory environments. We demonstrate how one fine-grained task-based programming model can be extended to accommodate them.

In uncertainty quantified optimization, not yet considered in this paper, but a relevant extension in policy contexts, the inverse of the covariance again plays a central role. The Bayesian UQ application and its solution can follow naturally upon our work.

ACKNOWLEDGMENTS

For computer time, this research used the resources of the Supercomputing Laboratory (KSL) `Shaheen II` at King Abdullah University of Science & Technology (KAUST) in Thuwal Saudi Arabia and the supercomputer `Fugaku` provided by RIKEN through the HPCI System Research Project (Project ID: hp200310 and ra010009). We effusively thank Mitsuhisa Sato / Miwako Tsuji (Riken) and Bilel Hadri (KSL)

for their support on Fugaku and Shaheen II, respectively, as well as Ruqing Xu from The University of Tokyo for providing the BLIS FP32 SHGEMM kernel on A64FX.

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.

[2] R Core Team, *R: A Language and Environment for Statistical Computing*, R Foundation for Statistical Computing, Vienna, Austria, 2022. [Online]. Available: https://www.R-project.org/

[3] D. Hammerling, H. Huang, and L. Blake, "Implementing Spatial Statistical Methods for Massive Data," in *Joint Statistics Meetings*, 2019.

[4] M. Stein, "Limitations on Low-Rank Approximations for Covariance Matrices of Spatial Data," *Spatial Statistics*, vol. 8, pp. 1–19, 2014.

[5] W. Hackbusch, "A Sparse Matrix Arithmetic Based on H-matrices. Part I: Introduction to H-Matrices," *Computing*, vol. 62, no. 2, pp. 89–108, 1999.

[6] Y. Sun, B. Li, and M. G. Genton, "Geostatistics for Large Datasets," in *Advances and challenges in space-time modelling of natural events*. Springer, 2012, pp. 55–77.

[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] E. Agullo, J. Demmel, J. Dongarra, B. Hadri, J. Kurzak, J. Langou, H. Ltaief, P. Luszczek, and S. Tomov, "Numerical Linear Algebra on Emerging Architectures: The PLASMA and MAGMA Projects," *J. Phys.: Conf. Ser.*, vol. 180, no. 1, 2009.

[10] G. Morton, *A Computer Oriented Geodetic Data Base and a New Technique in File Sequencing*. IBM, 1966.

[11] S. Abdulah, H. Ltaief, Y. Sun, M. G. Genton, and D. E. Keyes, "Geostatistical Modeling and Prediction Using Mixed Precision Tile Cholesky Factorization," in *26th International Conference on High Performance Computing, Data, and Analytics (HiPC)*. IEEE, 2019, pp. 152–162.

[12] S. Abdulah, Q. Cao, Y. Pei, G. Bosilca, J. Dongarra, M. G. Genton, D. E. Keyes, H. Ltaief, and Y. Sun, "Accelerating Geostatistical Modeling and Prediction With Mixed-Precision Computations: A High-Productivity Approach with PaRSEC," *IEEE Transactions on Parallel and Distributed Systems*, vol. 33, no. 4, pp. 964–976, 2021.

[13] M. G. Genton, "Separable Approximations of Space-time Covariance Matrices," *Environmetrics: The Official Journal of the International Environmetrics Society*, vol. 18, no. 7, pp. 681–695, 2007.

[14] N. Cressie and C. K. Wikle, *Statistics for Spatio-Temporal Data*. John Wiley & Sons, 2015.

[15] B. Matérn, *Spatial Variation*. Springer-Verlag, Berlin,, 1986, vol. 36.

[16] J.-P. Chiles and P. Delfiner, *Geostatistics: Modeling Spatial Uncertainty*. John Wiley & Sons, 2009, vol. 497.

[17] S. Börm and J. Garcke, "Approximating Gaussian Processes with H^2-Matrices," in *European Conference on Machine Learning*. Springer, 2007, pp. 42–53.

[18] T. Gneiting, "Nonseparable, Stationary Covariance Functions for Space–time Data," *Journal of the American Statistical Association*, vol. 97, no. 458, pp. 590–600, 2002.

[19] C. Augonnet, S. Thibault, R. Namyst, and P.-A. Wacrenier, "StarPU: a unified platform for task scheduling on heterogeneous multicore architectures," *Concurrency and Computation: Practice and Experience*, vol. 23, no. 2, pp. 187–198, 2011.

[20] A. Duran, R. Ferrer, E. Ayguade, R. M. Badia, and J. Labarta, "A proposal to extend the OpenMP tasking model with dependent tasks," *Intl. Journal of Parallel Programming*, vol. 37, no. 3, pp. 292–305, 2009.

[21] R. Hoque, T. Herault, G. Bosilca, and J. Dongarra, "Dynamic Task Discovery in PaRSEC: A Data-flow Task-based Runtime," in *Proceedings of the 8th Workshop on Latest Advances in Scalable Algorithms for Large-Scale Systems*, ser. ScalA '17, 2017.

[22] T. Heller, H. Kaiser, and K. Iglberger, "Application of the ParalleX execution model to stencil-based problems," *Computer Science - Research and Development*, vol. 28, no. 2-3, pp. 253–261, 2013.

[23] M. Bauer, S. Treichler, E. Slaughter, and A. Aiken, "Legion: Expressing locality and independence with logical regions," in *International Conference for High Performance Computing, Networking, Storage and Analysis, SC*. IEEE, 2012, pp. 1–11.

[24] L. V. Kale and S. Krishnan, "CHARM++: a portable concurrent object oriented system based on C++," in *ACM Sigplan Notices*, vol. 28, no. 10. ACM, 1993, pp. 91–108.

[25] 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.

[26] M. Asch, T. Moore, R. Badia, M. Beck, P. Beckman, T. Bidot, F. Bodin, F. Cappello, A. Choudhary, B. de Supinski, E. Deelman, J. Dongarra, A. Dubey, G. Fox, H. Fu, S. Girona, W. Gropp, M. Heroux, Y. Ishikawa, K. Keahey, D. Keyes, W. Kramer, J.-F. Lavignon, Y. Lu, S. Matsuoka, B. Mohr, D. Reed, S. Requena, J. Saltz, T. Schulthess, R. Stevens, M. Swany, A. Szalay, W. Tang, G. Varoquaux, J.-P. Vilotte, R. Wisniewski, Z. Xu, and I. Zacharov, "Big Data and Extreme-scale Computing: Pathways to Convergence-toward a Shaping Strategy for a Future Software and Data Ecosystem for Scientific Inquiry," *The International Journal of High Performance Computing Applications*, vol. 32, no. 4, pp. 435–479, 2018.

[27] "NVIDIA Tensor Cores," https://www.nvidia.com/en-us/data-center/tensorcore/, 2019, [Online; accessed June 2019].

[28] "Google Tensor Processing Unit (TPU)," https://cloud.google.com/tpu/, 2019, [Online; accessed June 2019].

[29] 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: International Conference for High Performance Computing, Networking, Storage and Analysis*. IEEE, 2018, pp. 717–730.

[30] K. Ando, R. Bale, C. Li, S. Matsuoka, K. Onishi, and M. Tsubokura, "Digital Transformation of Droplet/Aerosol Infection Risk Assessment Realized on" Fugaku" for the Fight Against COVID-19," *arXiv preprint arXiv:2110.09769*, 2021.

[31] 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 *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, 2021, pp. 1–12.

[32] Q. Cao, Y. Pei, K. Akbudak, A. Mikhalev, G. Bosilca, H. Ltaief, D. Keyes, and J. Dongarra, "Extreme-scale Task-based Cholesky Factorization Toward Climate and Weather Prediction Applications," in *Proceedings of the Platform for Advanced Scientific Computing Conference (PASC)*, 2020, pp. 1–11.

[33] A. Danalis, G. Bosilca, A. Bouteiller, T. Herault, and J. Dongarra, "PTG: an Abstraction for Unhindered Parallelism," in *2014 Fourth International Workshop on Domain-Specific Languages and High-Level Frameworks for High Performance Computing*. IEEE, 2014, pp. 21–30.

[34] Q. Cao, Y. Pei, T. Herault, K. Akbudak, A. Mikhalev, G. Bosilca, H. Ltaief, D. Keyes, and J. Dongarra, "Performance Analysis of Tile Low-Rank Cholesky Factorization Using PaRSEC Instrumentation Tools," in *IEEE/ACM International Workshop on Programming and Performance Visualization Tools (ProTools) at SC19*, 2019, pp. 25–32.

[35] Q. Cao, Y. Pei, K. Akbudak, G. Bosilca, H. Ltaief, D. Keyes, and J. Dongarra, "Leveraging PaRSEC Runtime Support to Tackle Challenging 3D Data-Sparse Matrix Problems," in *2021 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*. IEEE, 2021, pp. 79–89.

[36] Q. Cao, R. Alomairy, Y. Pei, G. Bosilca, H. Ltaief, D. Keyes, and J. Dongarra, "A Framework to Exploit Data Sparsity in Tile Low-Rank Cholesky Factorization," in *36th IEEE International Parallel & Distributed Processing Symposium (IPDPS)*, 2022.

[37] N. W. Chaney, P. Metcalfe, and E. F. Wood, "Hydroblocks: a field-scale resolving land surface model for application over continental extents," *Hydrological Processes*, vol. 30, no. 20, pp. 3543–3559, 2016.

[38] A. McNally, J. Jacob, K. Arsenault, K. Slinski, D. Sarmiento, A. Hoell, S. Pervez, J. Rowland, M. Budde, S. Kumar *et al.*, "A hydrologic monitoring dataset for food and water security applications in central asia," *Earth System Science Data Discussions*, pp. 1–32, 2021.

[39] N. J. Higham and T. Mary, "Mixed precision algorithms in numerical linear algebra," *Acta Numerica*, vol. 31, pp. 347–414, 2022.
[40] M. L. O. Salvana, S. Abdulah, H. Ltaief, Y. Sun, M. G. Genton, and D. Keyes, "Parallel Space-Time Likelihood Optimization for Air Pollution Prediction on Large-Scale Systems," in *Proceedings of the Platform for Advanced Scientific Computing Conference (PASC)*, 2022.
[41] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of ICNN'95 - International Conference on Neural Networks*, vol. 4. IEEE, 1995, pp. 1942–1948.