Boosting Earth System Model Outputs And Saving PetaBytes in Their Storage Using Exascale Climate Emulators

Sameh Abdulah[1,7], Allison H. Baker[2,8], George Bosilca[3,9], Qinglei Cao[4,10], Stefano Castruccio[5,11], Marc G. Genton[1,7], David E. Keyes[1,7], Zubair Khalid[1,6,12], Hatem Ltaief[1,7], Yan Song[1,7], Georgiy L. Stenchikov[1,7], and Ying Sun[1,7]

[1]Extreme Computing & Statistics & Earth Science, King Abdullah University of Science and Technology, KSA
[2]Computational and Information Sciences Lab, NSF National Center for Atmospheric Research, USA
[3]NVIDIA, USA
[4]Department of Computer Science, Saint Louis University, USA
[5]Department of Applied and Computational Mathematics and Statistics, University of Notre Dame, USA
[6] Department of Electrical Engineering, Lahore University of Management Sciences, Pakistan
[7]*{Firstname.Lastname}@kaust.edu.sa* [8]*abaker@ucar.edu* [9]*gbosilca@nvidia.com*
[10]*qinglei.cao@slu.edu* [11]*scastruc@nd.edu* [12]*zubair.khalid@lums.edu.pk*

***Abstract*—We present the design and scalable implementation of an exascale climate emulator for addressing the escalating computational and storage requirements of high-resolution Earth System Model simulations. We utilize the spherical harmonic transform to stochastically model spatio-temporal variations in climate data. This provides tunable spatio-temporal resolution and significantly improves the fidelity and granularity of climate emulation, achieving an ultra-high spatial resolution of 0.034°(∼3.5 km) in space. Our emulator, trained on 318 billion hourly temperature data points from a 35-year and 31 billion daily data points from an 83-year global simulation ensemble, generates statistically consistent climate emulations. We extend linear solver software to mixed-precision arithmetic GPUs, applying different precisions within a single solver to adapt to different correlation strengths. The `PaRSEC` runtime system supports efficient parallel matrix operations by optimizing the dynamic balance between computation, communication, and memory requirements. Our BLAS3-rich code is optimized for systems equipped with four different families and generations of GPUs, scaling well to achieve 0.976 EFlop/s on 9,025 nodes (36,100 AMD MI250X multi-chip module (MCM) GPUs) of Frontier (nearly full system), 0.739 EFlop/s on 1,936 nodes (7,744 Grace-Hopper Superchips (GH200)) of Alps, 0.243 EFlop/s on 1,024 nodes (4,096 A100 GPUs) of Leonardo, and 0.375 EFlop/s on 3,072 nodes (18,432 V100 GPUs) of Summit.**

***Index Terms*—Dynamic runtime systems, High-performance computing, Mixed-precision computation, Spatio-temporal climate emulation, Spherical harmonic transform, Task-based programming models.**

I. Justification for the Gordon Bell Prize

Exascale climate emulator developed using 318 billion hourly and 31 billion daily observations for generating climate emulations at ultra-high spatial resolution (0.034° ∼3.5 km). Modeling climate data using spherical harmonics. Mixed-precision computations. `PaRSEC` dynamic runtime system. Running on 9,025 nodes on Frontier, 1,936 nodes on Alps, 1,024 nodes on Leonardo, and 3,072 nodes on Summit, with the hybrid Flop/s rates 0.976 EFlop/s, 0.739 EFlop/s, 0.243 EFlop/s, and 0.375 EFlop/s, respectively.

II. Performance Attributes

Problem size	54,486,360 spatial locations across the globe at a spatial resolution of 0.034° (∼3.5 km)
Category of achievement	Scalability and peak performance
Type of method used	Spherical Harmonic Transform (SHT) and Cholesky factorization
Results reported on basis of	Cholesky factorization
Precision reported	Double and mixed-precision
System scale	- 0.976 EFlop/s on 9,025 nodes of Frontier (36,100 AMD MI250X multi-chip module (MCM) GPUs) equivalent to 72,200 AMD Graphics Compute Dies (GCDs) - 0.739 EFlop/s on 1,936 nodes of Alps (7,744 NVIDIA Grace-Hopper Superchips (GH200)) - 0.243 EFlop/s on 1,024 nodes of Leonardo (4,096 NVIDIA A100 GPUs) - 0.375 EFlop/s on 3,072 nodes of Summit (18,432 NVIDIA V100 GPUs)
Measurement mechanism	Timers, Flops

III. Overview of the Problem

Climate change, evident in rising temperatures, extreme weather events, sea-level rise, and ecosystem disruption, poses significant risks and urgently requires action due to intensified heatwaves, storms, droughts, floods, and biodiversity loss [1], [2]. We stand at a critical juncture where converging

Authors are listed alphabetically by their last names.

SC24, November 17-22, 2024, Atlanta, GA, USA
979-8-3503-5291-7/24/$31.00 ©2024 IEEE

advancements in technology and methodologies, including the integration of exascale supercomputers, offer unprecedented opportunities to tackle the existential threats of climate change. Our work leverages exascale computing to develop scalable climate emulators that complement high-resolution climate projections, a capability not practically affordable by traditional climate models alone due to their expensive computational demands and petabyte-scale storage requirements.

Conventional climate models use conservation laws expressed as partial differential equations (PDEs) to represent the interactions within Earth's systems, providing insights into climate change causes, mitigation impacts, and the potential trajectories of future climate conditions [3], [4]. Climate modeling began in the 1950s, and over the years, models have become increasingly sophisticated as computing power, observations, and our understanding of the climate system have advanced, with Earth System Models (ESMs) now the state-of-the-art models available to climate scientists [5].

Throughout 80 years of modern computing, capabilities deemed unimaginable by one generation have become routine expectations of the next. Gordon Bell papers have regularly expanded the headroom of applications, in terms of resolution, fidelity, and/or performance [6], which we do here for the next Coupled Model Intercomparison Project (CMIP) campaign. The CMIP campaigns support detailed comparisons of ESMs by enabling systematic analysis, validation, and enhancement. CMIPs are "resource-intensive and time-consuming to run" [7], with each new phase becoming increasingly complex in terms of experimental design, number of model intercomparisons, and the growth in data requested. For example, the most recent CMIP Phase 6 (CMIP6) effort resulted in roughly 28 PB from 45 modeling institutes hosted by the Earth System Grid Federation (ESGF) [8]. For comparison, CMIP3 generated 40 TB of data, and CMIP5 generated 2 PB of data [9]. For individual modeling centers, generating and preparing the data for the CMIP efforts is typically a monumental task. For CMIP6, the National Center for Atmospheric Research (NCAR) ran 37,000 years of climate with the Community Earth System Model (CESM), consuming 190 million CPU hours and resulting in 2 PB post-processed time series data [7]. Similarly, the Goddard Institute for Space Studies (GISS) contribution to CMIP6 required 104.5 million CPU hours on the NASA Center for Climate Simulation (NCCS) Discover supercomputer and generated 147 TB of simulation data [10]. Moreover, financial constraints limit the storage capacity available for climate simulations. For example, managing data at NCAR incurs costs of approximately $45 per TB annually. This results in substantial financial burdens for projects with petabyte-scale storage needs and can limit science objectives. In a recent survey of the NCAR computing community, 70% of respondents indicated that disk volume limitations have a moderate or severe impact on their work.

Computational demands and storage requirements for ESMs continue to escalate as the climate community progresses toward ultra-high-resolution (i.e., kilometer-scale) simulations. Some scientists believe that simulations at these so-called "global storm-resolving" scales are necessary to understand better how weather and extremes will be affected by climate change [11]. The recent DYAMOND (DYnamics of the Atmospheric general circulation Modeled On Non-hydrostatic Domains) project [12] exemplifies the computational cost of kilometer-scale simulations. In the initial DYAMOND experiments, the nine atmosphere-only models in the project ran for only 40 days, due to both computational and petabyte-scale storage costs (note that short timesteps are typically needed for high-res model runs). At the grid spacing of 5 km or less needed to capture small-scale processes, the average DYAMOND model output rate was just 6 SDPD (simulated-days-per-day), and 3D variable output was severely limited. For example, each output sample for the ICON model simulations was $\sim$1 TB [12], and the 3.25 km resolution SCREAM (Simple Cloud-Resolving E3SM Model, whose developers were awarded the first Gordon Bell Prize for Climate Modeling in 2023 [13]) contribution to DYAMOND produced $\sim$4.5 TB of data per simulated day [14]. Note that even a coarser quarter-degree horizontal-grid resolution ($\sim$25 km), which is still considered high-resolution for climate, is quite costly. For example, E3SM consumed almost 100 million core hours on Argonne's Theta supercomputer for a single 50-year-long simulation [15]. Because the model resolution significantly impacts the trade-off between the computational costs and representation of the climate system, addressing both computational and data storage challenges is essential for advancing climate modeling capabilities.

Climate emulators play a pivotal role in alleviating the computational burden and storage requirements associated with climate modeling and simulations, as highlighted by their use in the Intergovernmental Panel on Climate Change (IPCC) assessment report (AR6) to forecast warming across various emission scenarios. These emulators, capable of controlling numerous climate system aspects through parameter variation, can be tuned to mimic ESMs behavior using simulated data such as CMIP6 and generate climate projections within seconds on a standard desktop computer. This enables multiple runs with varied parameter values for a single emissions scenario, which is crucial for encompassing the uncertainty range in future climate projections. Emulators have been part of the climate modeling landscape for almost as long as the development of more advanced climate models (e.g., [16]–[23]). In Figure 1, we review these contributions through the lens of the computational cost associated with climate emulator design (more details provided in the next section). We note that the current climate emulators have not yet attained a spatial resolution finer than 100 km. Moreover, only those emulators designed with an assumption of axial symmetry (stationarity with respect to longitude) have managed to achieve a daily temporal resolution. In this context, our work makes the following notable contributions:

- We have developed a computationally efficient exascale climate emulator that surpasses existing emulators by a factor of 245,280 in terms of spatio-temporal resolution,

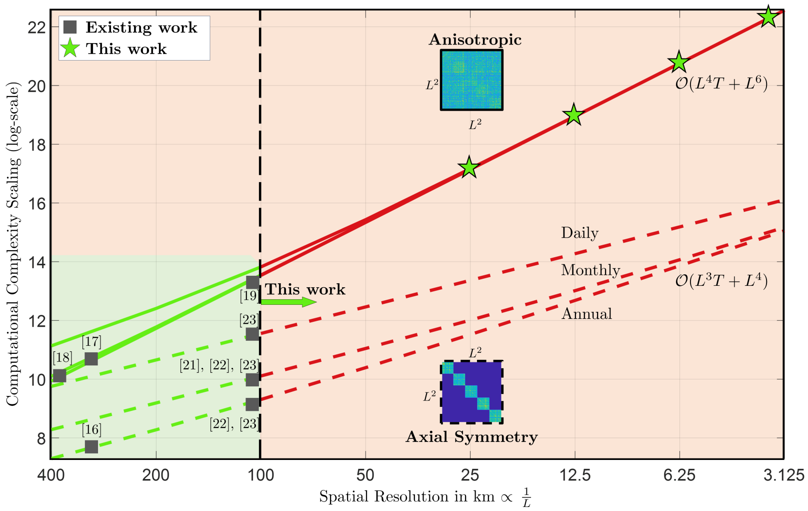


Fig. 1: Overview of existing work vs. this work. Analysis of computational cost associated with the emulator design for different spatial resolutions and the review of the existing emulators. The computational complexity scales as $\mathcal{O}(L^3T + L^4)$ and $\mathcal{O}(L^4T + L^6)$ for axially symmetric and longitudinally anisotropic models, respectively, where T is the count of temporal data points and L parameterizes the spatial resolution, reflecting the number of grid points along either latitude or longitude. Existing emulators have achieved spatial resolutions of up to 100 km (approximately 0.901 degrees at the equator) and temporal resolutions down to the daily scale under the axial symmetry (stationarity with respect to longitude) assumption. In contrast, anisotropic models have been limited to an annual temporal scale at 100 km spatial resolution. The climate emulator design in this work (indicated by a green star) significantly advances the fidelity and granularity of climate emulation by achieving an unprecedented spatial resolution ($\sim$3.5–100 km) and hourly temporal resolution, surpassing the existing emulators by a factor of 245,280 (28 times in spatial resolution and 8,760 times in temporal resolution).

that is, offering 28 times greater spatial and 8,760 times greater temporal resolution (see Figure 1). This advancement marks a significant milestone in climate science, as it enables the generation of hourly emulations at an ultra-high spatial resolution of $0.034°$($\sim$3.5 km) for the first time — a step towards accurate climate projections and a deeper understanding of climate change and its impacts. Our emulator, capable of generating climate emulations statistically consistent with the simulations, is trained on 318 billion hourly and 31 billion daily data points of surface temperature derived from a global ensemble of simulations spanning 35 years (1988-2022) and 83 years (1940-2022), respectively.

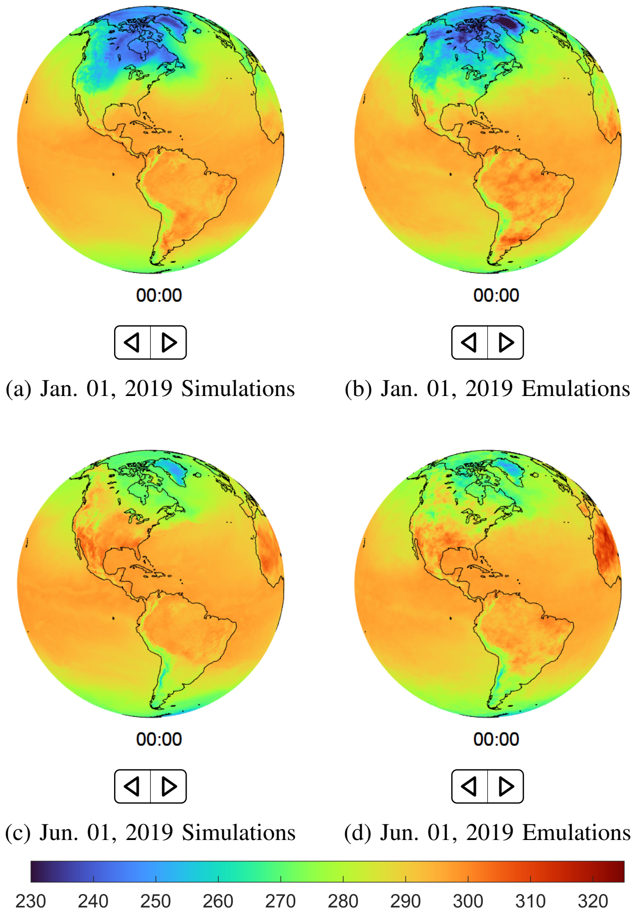
- Our approach addresses the limitations of existing emulators by employing the spherical harmonic transform (SHT) of global data to model anisotropic interactions. We also present HPC innovations for the scalable implementation of the emulator design on the exascale computers. Our implementation applies multiple precisions within a single solver, catering to the covariance strengths of the data. Furthermore, we rely on `PaRSEC` runtime system to support efficient parallel matrix operations and optimize the balance between computation, communication, and memory requirements with tile-centric mixed-precision support. The scalability of the design is demonstrated by generating unprecedented $0.034°$($\sim$3.5 km) hourly climate emulations, equivalent to 477 billion data points for a single year emulation of temperature data, illustrated in Figure 2 at 25 km spatial resolution (the maximum available resolution of ERA5 data).
- We optimize our code for large-scale execution on systems equipped with different families and generations of GPUs, scaling well to achieve 0.976 EFlop/s on 9,025 nodes (36,100 AMD MI250X GPUs) of Frontier (topping the June 2024 Top500 list), 0.739 EFlop/s on 1,936 nodes (7,744 NVIDIA H100 GPUs) of Alps (ranked 6^{th}), 0.243 EFlop/s on 1,024 nodes (4,096 NVIDIA A100 GPUs) of Leonardo (ranked 7^{th}), and 0.375 EFlop/s on 3,072 nodes (18,432 NVIDIA V100 GPUs) of Summit (ranked 9^{th}). Our implementation demonstrates excellent weak scaling efficiency and up to 72% strong scaling efficiency with as many as 12,288 V100 GPUs on Summit. We also rely on KAUST's Shaheen III CPU partition (ranked 23^{rd}) to perform most of the science experiments in this paper.

We thus bridge the gap between ESM simulations and the growing demand for accessible, highly resolved emulations and projections. Since our emulator enhances and complements computationally expensive and storage-intensive simulations, we anticipate substantial savings in storage (in petabytes) needed for storing such simulations, heralding a new era of analysis at ultra-high spatio-temporal resolutions. Furthermore, by shifting the majority of the computational effort from communication-bound sparse double-precision kernels characteristic of PDE-based models to dense low-precision tensor core kernels in the wheelhouse of the GPU commodity market, we open a more sustainable swim lane to climate modeling.

IV. Current State of the Art

A. *Climate Emulators: Review and Computational Challenges*

The development of climate emulators has recently gained prominence due to their role in complementing and enhancing



(a) Jan. 01, 2019 Simulations

(b) Jan. 01, 2019 Emulations

(c) Jun. 01, 2019 Simulations

(d) Jun. 01, 2019 Emulations

Fig. 2: Hourly (24-hour) surface temperatures (in K) generated by climate simulations (ERA5 dataset) and the proposed climate emulator (single realization, statistically consistent with simulations) are plotted in (a) simulations and (b) emulations for January 1, 2019, and in (c) simulations and (d) emulations for June 1, 2019. This figure includes an animation (at 2-hour intervals) for enhanced visualization, optimally viewed in *Adobe Acrobat*.

the ESM simulations (e.g., [17], [24]–[27], see Section VIII for potential uses of climate emulators). At the heart of emulator design is the principle of training a statistical model on a limited set of simulations to generate new data or predict outcomes for new inputs, eliminating the need for a large ensemble of full-scale climate model runs. The development of emulators spans decades, with the conventional approach primarily revolving around the use of Gaussian processes (GPs) and auto-regressive models to capture the intricate spatio-temporal dependencies found in climate simulations driven by underlying physical mechanisms. A primary challenge with this approach is the processing of extensive high-resolution data in global climate simulation, as performing inference with GPs on such large datasets can be computationally prohibitive. Addressing this computational challenge benefits from drawing on existing techniques such as low-rank approximations [28], composite likelihoods [29], stochastic partial differential equations [30], covariance tapering [31], and their combinations [32]. Notably, efficient algorithms to handle large datasets and the distribution of computations across multiple processors have been proposed to reduce the emulation times [16], [17]. [23] presents an innovative emulator tailored for global surface temperature simulations from the recently published CESM2-LENS2 data, featuring a spatial resolution of approximately $1^\circ \times 1^\circ$. By leveraging the SHT, the emulator efficiently transitions simulations from the spatial domain to the spectral domain. Exploiting continuity by changing the discrete basis from grid values to spectral amplitudes leads to significant reductions in both the computational and storage costs, enabling the generation of emulations across annual, monthly, and daily scales. The spectral basis also provides a unified representation of data with different grid resolutions.

The anisotropic (non-stationary) spatial dependence in the output of ESM simulations also presents significant computational challenges associated with estimating and decomposing the covariance matrix that captures this spatial dependence. Common simplifications include assuming isotropy and axial symmetry (stationarity along the longitude), leading to diagonal and sparse covariance matrices, respectively [16], [21]–[23]. This reduces the computational workload for matrix estimation and its Cholesky decomposition.

In Figure 1, we analyze and compare the spatial resolution and computational cost of various emulators proposed in the literature. The computational complexity when assuming axial symmetry scales as $\mathcal{O}(L^3T + L^4)$, where T denotes the number of temporal data points, and L denotes the spatial resolution, reflecting the number of grid points along either latitude or longitude. Emulators designed under this assumption have reached spatial resolutions up to 100 km and temporal resolutions down to the daily scale. Relaxing the axial symmetry assumption increases the computational complexity associated with the emulator design to $\mathcal{O}(L^4T + L^6)$. As a result, emulators accommodating longitudinal variation are typically confined to annual temporal resolutions and moderate spatial resolutions ($\sim$100-500 km) [17]–[19]. We overcome the computational challenges associated with accurately capturing the anisotropic spatial dependencies in ESMs by employing HPC innovations in our scalable implementation. These innovations enable efficient parallel implementation of SHT and the computation and decomposition of the covariance matrix. The proposed HPC innovations, combined with recent scientific advancements, offer pathways to harness the power of supercomputers to develop exascale scalable climate emulators capable of generating climate emulations consistent with the intricate details of climate dynamics and achieving a level of resolution previously unattainable in both space and time.

B. Climate Emulator Design – Overview

We consider spatio-temporal climate data, denoted by $y_t^{(r)}(\theta_i, \phi_j)$ for latitude $\theta_i \in [0, \pi]$, longitude $\phi_j \in [0, 2\pi)$, time point t, and ensemble r, where $i = 1, \ldots, N_\theta$, $j = 1, \ldots, N_\phi$, $t = 1, \ldots, T$, and $r = 1, \ldots, R$. The data in the spatial domain are located on a grid of $N_\theta \times N_\phi$ points, and the total size of the dataset given by $R \times T \times N_\theta \times N_\phi$ depends

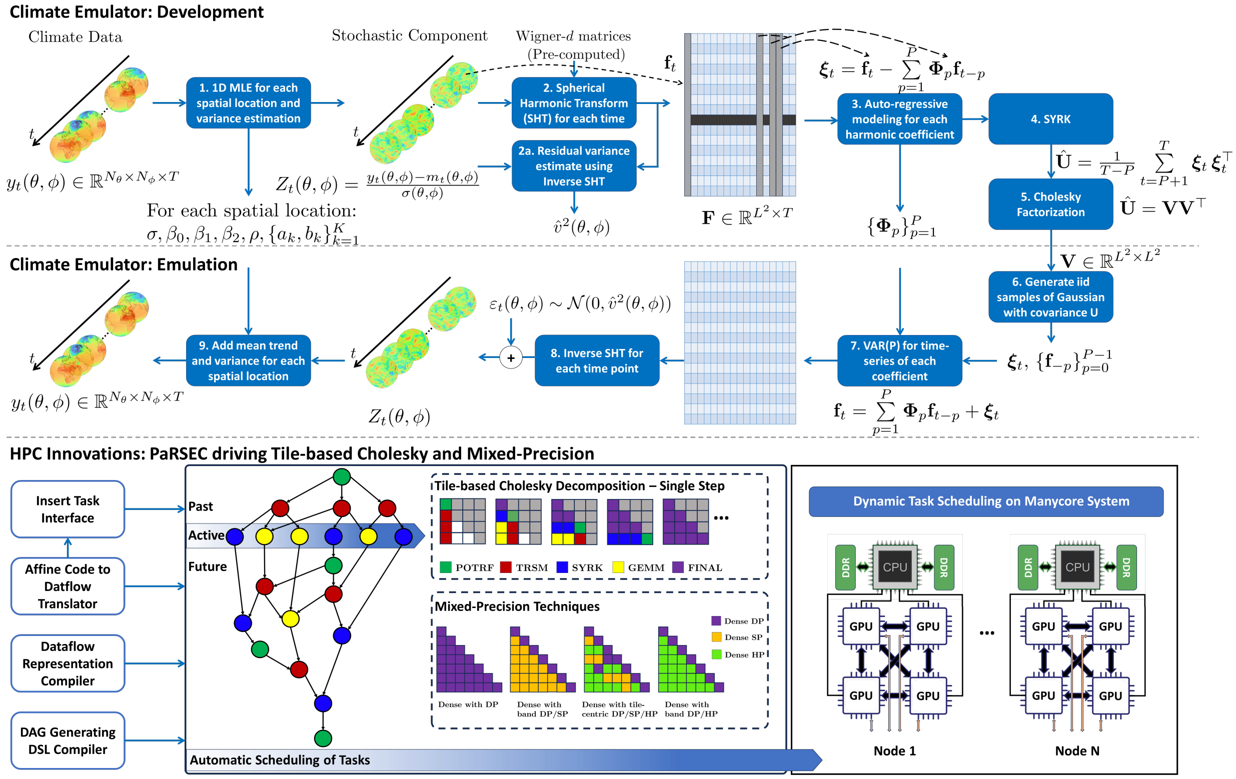

Climate Emulator: Development
Climate Data
t
$y_t(\theta,\phi) \in \mathbb{R}^{N_\theta \times N_\phi \times T}$
1. 1D MLE for each spatial location and variance estimation
For each spatial location:
$\sigma, \beta_0, \beta_1, \beta_2, \rho, \{a_k, b_k\}_{k=1}^K$
Stochastic Component
$Z_t(\theta,\phi) = \frac{y_t(\theta,\phi)-m_t(\theta,\phi)}{\sigma(\theta,\phi)}$
Wigner-d matrices (Pre-computed)
2. Spherical Harmonic Transform (SHT) for each time
2a. Residual variance estimate using Inverse SHT
$\hat{v}^2(\theta,\phi)$
$\mathbf{f}_t$
$\mathbf{F} \in \mathbb{R}^{L^2 \times T}$
$\boldsymbol{\xi}_t = \mathbf{f}_t - \sum_{p=1}^{P} \boldsymbol{\Phi}_p \mathbf{f}_{t-p}$
3. Auto-regressive modeling for each harmonic coefficient
$\{\boldsymbol{\Phi}_p\}_{p=1}^P$
4. SYRK
$\hat{\mathbf{U}} = \frac{1}{T-P} \sum_{t=P+1}^{T} \boldsymbol{\xi}_t \boldsymbol{\xi}_t^\top$
5. Cholesky Factorization
$\hat{\mathbf{U}} = \mathbf{V}\mathbf{V}^\top$
$\mathbf{V} \in \mathbb{R}^{L^2 \times L^2}$
Climate Emulator: Emulation
6. Generate iid samples of Gaussian with covariance U
$\boldsymbol{\xi}_t, \{\mathbf{f}_{-p}\}_{p=0}^{P-1}$
7. VAR(P) for time-series of each coefficient
$\mathbf{f}_t = \sum_{p=1}^{P} \boldsymbol{\Phi}_p \mathbf{f}_{t-p} + \boldsymbol{\xi}_t$
8. Inverse SHT for each time point
$\varepsilon_t(\theta,\phi) \sim \mathcal{N}(0, \hat{v}^2(\theta,\phi))$
+
$Z_t(\theta,\phi)$
9. Add mean trend and variance for each spatial location
$y_t(\theta,\phi) \in \mathbb{R}^{N_\theta \times N_\phi \times T}$
HPC Innovations: PaRSEC driving Tile-based Cholesky and Mixed-Precision
Insert Task Interface
Affine Code to Datflow Translator
Dataflow Representation Compiler
DAG Generating DSL Compiler
Past
Active
Future
Automatic Scheduling of Tasks
Tile-based Cholesky Decomposition – Single Step
...
POTRF
TRSM
SYRK
GEMM
FINAL
Mixed-Precision Techniques
Dense DP
Dense SP
Dense HP
Dense with DP
Dense with band DP/SP
Dense with tile-centric DP/SP/HP
Dense with band DP/HP
Dynamic Task Scheduling on Manycore System
DDR
CPU
DDR
GPU
GPU
GPU
GPU
...
Node 1
Node N

Fig. 3: An overview of the design and development pipeline of the climate emulator, including HPC innovations, detailing each step in the process of developing an emulator and using it to generate emulations (see Section V for details).

on the spatial and temporal resolutions. As an illustration, hourly observations of ERA5 surface temperature are plotted in Figure 2(a) and (c).

Utilizing the SHT, which transforms spatial data on the sphere into the spectral domain based on eigenfunctions of the angular part of the 3D Laplacian, we present the design of an emulator equipped with efficient and adaptable capabilities tailored specifically for managing the complexities inherent in large-scale global climate simulations such as non-stationary spatial dependencies that exist among latitudes and between different geographic regions (e.g., land and ocean), and temporal dynamics. The capability of the proposed emulator for generating climate emulations *statistically consistent* with the simulations has been demonstrated in [23] for different scenarios. Figure 3 provides an overview of the emulator design and the steps involved in generating emulations, including HPC innovations. In the next section, we describe the mathematical formulation and computational complexity associated with designing the emulator and generating emulations.

C. Advancements in Dense Linear Algebra: State-of-the-Art

Dense matrix algorithms are extensively utilized across various applications, from scientific computing to engineering and data analysis. Operations such as Cholesky have seen significant performance improvements due to rapid advancements in hardware technology. With well-designed parallel algorithms, the performance of these dense operations can approach the underlying theoretical limits of the hardware. These algorithms use task-based programming to handle detailed computations efficiently and are known as tile-based linear solvers. They divide the matrix into tiles represented in a Directed Acyclic Graph (DAG) in Figure 3, where nodes and edges represent computational tasks and their dependencies, respectively [33].

On modern hardware, such as GPUs, there is a trend towards supporting operations at lower precision, a development that AI workloads capitalize on for accelerated training and inference. For instance, NVIDIA V100, A100, and H100 can run single-precision (SP) / half-precision (HP) 2X/16X, 16X/32X, 14.7X/29.5X faster than double-precision (DP) arithmetic, respectively (assuming, if supported, tensor core operations and the SXM interface). This shift also presents an opportunity in dense linear algebra to exploit fast matrix operations performed at reduced precision, particularly in scientific applications that can tolerate reduced computational accuracy.

Mixed-precision matrix approximations capitalize on hardware trends and substantially reduce memory usage. This reduction allows more of the working set to reside in fast memory, enhancing performance [34], [35]. To effectively manage these operations, which vary inherently in their data structures and precision levels (i.e., DP/SP/HP), it is essential to implement a dynamic runtime system framework that adeptly orchestrates these heterogeneous tasks.

D. Increased Productivity with Dynamic Runtime Systems

Task-based dynamic runtimes have been proposed as an alternative for programming challenges raised by complex hierarchical systems, with the goal of facilitating the scheduling of fine-grained "work units", or tasks, onto the underlying hardware resources and managing the communications. Augmented with various scheduling strategies, they are able to reduce idle time for imbalanced applications and reduce expensive data movements while favoring data locality. These runtime systems support shared- and distributed- memory systems, including support for accelerators. The result is an asynchronous execution model offering adaptive capabilities beyond the bulk synchronous SPMD mainstream.

The programming API varies between the different frameworks, going from a relatively easy to depict sequential declaration of tasks and their dependencies for OpenMP [36], `StarPU` [37], `OmpSs` [38], `HPX` [39], `Charm++` [40], to a more complex, static or dynamic, symbolic representation of the graph of tasks for `PaRSEC` [41] and `Legion` [42]. In all cases, the users remain responsible for the correct description of the graph of tasks, and the runtime is responsible for the effective unrolling of the tasks and resulting communications onto the available resources, providing a separation of concerns between algorithmic description and execution.

`PaRSEC` [41], the runtime used in this work, relies on a domain-specific language (DSL) that represents the entire DAG in a compressed and parametrized manner. This permits leveraging collective communications that are crucial in dense matrix algorithms, provides opportunities for communication/computations overlap, and better uses data locality and caching. `PaRSEC` decouples the data distribution from the actual task operations, providing a level of abstraction allowing for type changes, or adaptive precision conversion, at the communication level, based on the needs for successors.

V. Innovations Realized

A. Climate Emulator Design – Formulation

We represent the climate data in terms of deterministic and stochastic components as [22], [23]

$$y_t^{(r)}(\theta_i, \phi_j) = m_t(\theta_i, \phi_j) + \sigma(\theta_i, \phi_j) Z_t^{(r)}(\theta_i, \phi_j), \quad (1)$$

where $Z_t^{(r)}(\theta_i, \phi_j)$ is the stochastic component at spatial location (θ_i, ϕ_j), time point t, and ensemble r. The terms m_t and σ represent the deterministic functions responsible for the mean trend and standard error, respectively, and are shared across all ensembles. The mean trend m_t for each spatial location can be modeled using an infinite distributed lag model to relate it with the radiative forcing (RF) trajectory x_t (annual scale) as [17], [22], [23]

$$\begin{aligned} m_t =& \beta_0 + \beta_1 x_{\lceil t/\tau \rceil} + \beta_2(1-\rho) \sum_{s=1}^{\infty} \rho^{s-1} x_{\lceil t/\tau \rceil - s}, \\ &+ \sum_{k=1}^{K} \left\{ a_k \cos\left(\frac{2\pi t k}{\tau}\right) + b_k \sin\left(\frac{2\pi t k}{\tau}\right) \right\}, \end{aligned} \quad (2)$$

where $\beta_0(\theta_i, \phi_j)$ is the intercept, and $\beta_1(\theta_i, \phi_j)$ and $\beta_2(\theta_i, \phi_j)$ are slopes for the current and past RF, respectively. Here $\beta_2(\theta_i, \phi_j)\{1 - \rho(\theta_i, \phi_j)\}\rho(\theta_i, \phi_j)^{s-1}$ are the lag weights that decrease the impact of past RF exponentially by $\rho(\theta_i, \phi_j) \in [0, 1]$. The choice of K depends on the number of terms used to model the periodic (e.g., intraday or interannual) variations. τ is a parameter capturing the frequency of periodic variations with a value $\tau = 12$, $\tau = 365$ and $\tau = 8760$ for monthly, daily, and hourly temporal resolutions, respectively. Leveraging the independence, we can efficiently estimate the parameters in (2) and σ by using 1D maximum likelihood estimation (MLE) method [22], [23] for each spatial location with the computational complexity $\mathcal{O}(T)$.

1) *Modeling in the Spherical Harmonic Domain:*

We model the stochastic component in terms of spherical harmonics as

$$Z_t^{(r)}(\theta_i, \phi_j) = \sum_{\ell=0}^{L-1} \sum_{m=-\ell}^{\ell} (z_t^{(r)})_{\ell,m} Y_{\ell,m}(\theta_i, \phi_j) + \varepsilon_t^{(r)}(\theta_i, \phi_j),$$

where $Y_{\ell,m}(\theta_i, \phi_j) = \sqrt{\frac{2\ell+1}{4\pi} \frac{(\ell-m)!}{(\ell+m)!}} P_\ell^m(\cos\theta_i) \exp(\mathrm{i} m \phi_j)$ and $(z_t^{(r)})_{\ell,m}$ represent an orthonormal spherical harmonic basis function and coefficient, respectively, of degree $\ell \geq 0$ and order $|m| \leq \ell$. Since we have truncated the representation in terms of basis functions at degree $L - 1$, the last term $\varepsilon_t^{(r)}(\theta_i, \phi_j)$ accounts for the remaining information and is assumed to be independent and follow $\mathcal{N}(0, v^2(\theta_i, \phi_j))$. The harmonic coefficients $(z_t^{(r)})_{\ell,m}$ are computed by the SHT given by [43]:

$$(z_t^{(r)})_{\ell,m} = \int_{\theta=0}^{\pi} \int_{\phi=0}^{2\pi} Z_t^{(r)}(\theta, \phi) \overline{Y_{\ell,m}(\theta, \phi)} \sin\theta \mathrm{d}\theta \mathrm{d}\phi, \quad (3)$$

which can be re-formulated by dropping the dependence on t and r for succinct representation as

$$(z)_{\ell,m} = \int_{\theta=0}^{\pi} G_m(\theta) Y_{\ell,m}(\theta, 0) \sin\theta \mathrm{d}\theta, \quad (4)$$

with $Y_{\ell,m}(\theta, \phi) = Y_{\ell,m}(\theta, 0) e^{\mathrm{i}m\phi}$ and

$$G_m(\theta) = \int_{\phi=0}^{2\pi} Z(\theta, \phi) e^{-\mathrm{i}m\phi} \mathrm{d}\phi = 2\pi \sum_{\ell=|m|}^{L-1} (z)_{\ell,m} Y_{\ell,m}(\theta, 0),$$

which can be used to represent the data in the form

$$Z(\theta, \phi) = \sum_{m=-(L-1)}^{L-1} G_m(\theta) e^{-\mathrm{i}m\phi} \quad (5)$$

with an assumption that the data are band-limited at degree L. Noting the relationship between Wigner-d function, denoted by $d_{m,n}^{\ell}(\theta)$ for degree ℓ and orders $|m|, |n| \leq \ell$, and the spherical harmonic [44]: $Y_{\ell,m}(\theta, 0) = \sqrt{\frac{2\ell+1}{4\pi}} d_{m,0}^{\ell}(\theta)$, and the expansion of Wigner-d function in terms of complex exponentials [44]: $d_{m,0}^{\ell}(\theta) = \mathrm{i}^{-m} \sum_{m'=-\ell}^{\ell} d_{m',0}^{\ell}(\frac{\pi}{2})\, d_{m',m}^{\ell}(\frac{\pi}{2})\, e^{\mathrm{i}m'\theta}$, we express $G_m(\theta)$ after changing the order of summation as

$$G_m(\theta) = \sum_{m'=-(L-1)}^{L-1} K_{m,m'}\, e^{\mathrm{i}m'\theta}, \quad (6)$$

where $K_{m,m'} = \mathrm{i}^{-m} \sum_{\ell=\max(|m'|,|m|)}^{L-1} \sqrt{(\pi)(2\ell+1)}\, f_{\ell,m} \times d_{m',0}^{\ell}(\frac{\pi}{2})\, d_{m',m}^{\ell}(\frac{\pi}{2})$. Substituting (6) in (4) yields

$$(z)_{\ell,m} = \sum_{m''=-(L-1)}^{L-1} S_{\ell,m,m''} \sum_{m'=-(L-1)}^{L-1} K_{m,m'}\, I(m' + m'') \quad (7)$$

where $S_{\ell,m,m''} = \mathrm{i}^{-m}\sqrt{\frac{2\ell+1}{4\pi}} d^{\ell}_{m'',0}\left(\frac{\pi}{2}\right)\ d^{\ell}_{m'',m}\left(\frac{\pi}{2}\right)$, and $I(q)$ is an integral of the form

$$I(q) = \int_{\theta\in[0,\pi]} e^{\mathrm{i}q\theta}\sin\theta \mathrm{d}\theta = \begin{cases} \delta_{|q|,1}\ \frac{\mathrm{i}q\pi}{2}, & q \text{ odd}, \\ \frac{2}{1-q^2}, & q \text{ even}, \end{cases} \tag{8}$$

where $\delta_{|q|,1}$ is a Kronecker delta function.

Since the representation of the data in (5) indicates that $Z(\cdot,\phi)$, with complex exponentials $\{e^{\mathrm{i}m\phi}\}$ as basis functions, is band-limited at L, $G_m(\theta)$ can be recovered accurately using the Fast Fourier Transform (FFT) if $N_\phi \geq 2L-1$. We extend the domain of $G_m(\theta)$ to include the points along co-latitude in $(\pi, 2\pi)$ for $\theta \in (0,\pi)$ as $G_m(2\pi-\theta) = 2\pi\sum_{\ell=|m|}^{L-1} f_{\ell,m}Y_{\ell,m}(\theta,\pi) = (-1)^m G_m(\theta)$, where we have used $(-1)^m Y_{\ell,m}(\theta,\pi) = Y_{\ell,m}(\theta,0)$. We use (6) to recover $K_{m,m'}$ exactly by employing inverse FFT provided $G_m(\theta)$ over $2N_\theta - 2$ points along extended co-latitude if $N_\theta > L$. Once we have $K_{m,m'}$ for all $|m|,|m'| < L$, we use (7) for the exact computation of SHT under the assumption that the data are band-limited. We note that the SHT can be computed with sufficient accuracy using the proposed approach even when the data are not band-limited, as illustrated in [23], and the error captured by $\varepsilon_t^{(r)}$ is incorporated in the emulation process.

2) Efficient Implementation: For efficient implementation of SHT, we pre-compute the Wigner-d matrix, thus eliminating the need to recalculate them with each temporal observation. The computation of $G_m(\theta)$ and $K_{m,m'}$ is optimized to $\mathcal{O}(L^2\log L)$ through the FFT, whether along ϕ or an extended θ. Once $K_{m,m'}$ is determined for all $|m|,|m'| < L$ for each observation, the computational complexity of the inner summation in (7) is $\mathcal{O}(L^3)$ for all m and m''. The subsequent outer summation also requires $\mathcal{O}(L^3)$ per observation, resulting in a total complexity of $\mathcal{O}(TL^3)$ for all temporal observations. The Wigner-d functions, specifically for the fixed argument $\pi/2$, can be efficiently calculated using recursion relations, with an $\mathcal{O}(\ell^2)$ complexity for computing $d^{\ell}_{m'',m}(\frac{\pi}{2})$ from $d^{\ell-1}_{m'',m}(\frac{\pi}{2})$, resulting in a total of $\mathcal{O}(L^3)$ across all degrees and orders. The calculation of $S_{\ell,m,m'}$ also takes $\mathcal{O}(L^3)$. As $S_{\ell,m,m'}$ does not depend on the data, it can be pre-computed for $\ell < L$, $|m| \leq \ell$, and $|m'| < L$, necessitating $\mathcal{O}(L^3)$ in storage space. While our pre-computation strategy does not change the asymptotic computational complexity of SHT, it significantly reduces actual computation time. Additionally, we underscore that our proposed SHT computation technique is fully parallelizable, offering a linear computational complexity of $\mathcal{O}(L)$ for computing SHT for different time points simultaneously.

3) Modeling the Temporal Dependence and Constructing a Covariance Matrix: After the computation of SHT of the stochastic component for all time points, we can form a matrix $\mathbf{F}^{(r)} = (\mathbf{f}_1^{(r)},\ldots,\mathbf{f}_T^{(r)}) \in \mathbb{R}^{L^2\times T}$ where $\mathbf{f}_t^{(r)} = ((z_t^{(r)})_{00},\ldots,(z_t^{(r)})_{LL})^\top \in \mathbb{R}^{L^2}$ is a vector of spherical harmonic coefficients for each time slot $t = 1,\ldots,T$. We model the temporal dependence using a vector auto-regressive model of order P, that is, $\mathbf{f}_t^{(r)} = \sum_{p=1}^{P}\Phi_p\mathbf{f}_{t-p}^{(r)} + \boldsymbol{\xi}_t^{(r)}$, where each $\Phi_p \in \mathbb{R}^{L^2\times L^2}$, $p = 1,2,\ldots,P$, is assumed to be a diagonal correlation matrix [23], and $\boldsymbol{\xi}_t^{(r)} \overset{i.i.d.}{\sim} \mathcal{N}_{L^2}(\mathbf{0},\mathbf{U})$. The covariance matrix $\mathbf{U}$ can be empirically evaluated with computational complexity $\mathcal{O}(L^4T)$ as

$$\hat{\mathbf{U}} = \frac{1}{R(T-P)}\sum_{r=1}^{R}\sum_{t=P+1}^{T}\boldsymbol{\xi}_t^{(r)}\left(\boldsymbol{\xi}_t^{(r)}\right)^\top, \tag{9}$$

and then factorized via Cholesky decomposition with computational complexity $\mathcal{O}(L^6)$ as $\hat{\mathbf{U}} = \mathbf{V}\mathbf{V}^\top$. In cases where $R(T-P) < L^2$, we introduce a minor perturbation along the diagonal of $\hat{\mathbf{U}}$ to ensure it remains positive definite. This completes the design of our emulator capable of generating emulations using the matrices $\mathbf{V}$, $\{\Phi_p\}_{p=1}^{P}$ and parameters (or matrices) v, $\sigma, \beta_0, \beta_1, \beta_2, \rho, \{a_k, b_k\}_{k=1}^{K}$ defining the characteristics of spatio-temporal climate data.

B. Generating Emulations

Using $\mathbf{V}$, we first generate $\boldsymbol{\xi}_t^{(r)} \overset{i.i.d.}{\sim} \mathcal{N}_{L^2}(\mathbf{0},\mathbf{U})$ and determine $\mathbf{f}_t^{(r)}$ by employing Φ_p for $p = 1,\ldots,P$ in $\mathcal{O}(L^2T)$. We then compute inverse SHT of the spherical harmonic coefficients $\mathbf{f}_t^{(r)}$ to obtain $(\tilde{Z}_t^{(r)})$ over a grid of N_θ latitudes and N_ϕ longitudes with the computational complexity of $\mathcal{O}(L^3T)$. Add $\varepsilon_t^{(r)} \sim \mathcal{N}(0, v^2)$ to $(\tilde{Z}_t^{(r)})$ to get $(Z_t^{(r)})$. Finally, we use (1) to generate emulations with the overall computational complexity of $\mathcal{O}(L^3T)$.

C. Using `PaRSEC` to Meet Exascale Challenges

Scale changes everything, escalating minor issues to major performance bottlenecks. Scaling up our experiments, especially on platforms where each node has a tremendous computing capability, highlighted two potential issues: the importance of ordering communications, especially collective communications, and the significance of minimizing memory waste. `PaRSEC` has support for both of these capabilities, including user-driven support via annotations in the source code, but this support was not previously tested at this scale.

Tiles from a regularly distributed matrix with varied precision require different amounts of storage. `PaRSEC` has support for dynamic memory allocations based on the incoming data [45], but as the communication layer can reshape the data to adapt it to the precision required by the successors' tasks, this allocation needs to be set dynamically, driven by the sender. Moreover, this dynamic allocation should be able to handle data on both CPU and accelerator memory.

Collective communications in `PaRSEC` are conceptually similar to those in MPI, with the major difference that no group or communicator is known in advance; the group will be dynamically built based on the location of successor tasks and propagated (either physically or symbolically) with the collective message. This means that each node of the propagation tree will locally construct the group of collectives and identify the processes with which they interact before participating in the collective communication. Moreover, as many collective communications are ongoing simultaneously, the original version of `PaRSEC` tried to maximize the overall bandwidth and allow for multiple concurrent collective communications. In other words, `PaRSEC` favored bandwidth over latency, resulting in individually longer collective latency but an overall better bandwidth usage. At scale, especially for strong scaling experiments, such a strategy is sub-optimal, as it creates starvation points. Thus, we realigned the priorities of the collective communications to focus on individual latency first, leading to faster and more regular time-to-solution.

D. Mixed-Precision Computation Innovations

Recent GPUs from various vendors are designed to deliver high performance at lower precision levels, catering to modern AI workloads. In earlier research, we introduced mixed-precision (MP) computation for large-scale single matrices, utilizing tile-based algorithms to distribute precision across different tiles according to application features. For example, precision distribution can be based on bandwidth, as seen in band-based MP computations [34], [46], or it can be tailored around numeric features, as demonstrated in tile-centric MP approaches [47]. All these studies were around CPU-based systems, including Intel, ARM, and Fujitsu A64FX Systems.

Recently, we expanded this approach to single and multiple GPUs [35], achieving improved performance and reduced power consumption for environmental applications. This advancement aims to enhance the accuracy of predictions using existing geospatial data. Here, we broaden the capabilities of the `PaRSEC` runtime system and mixed-precision computation using tile-based linear algebra algorithms to large-scale GPU-based systems, specifically, Frontier (AMD MI250X), Alps (NVIDIA H100), Leonardo (NVIDIA A100), and Summit (NVIDIA V100), and we supplement our study with efficiency and scalability assessment results for climate emulators.

VI. HOW PERFORMANCE WAS MEASURED

A. Spatio-temporal Dataset, Upsampling and Accuracy

We consider the surface temperature measured at 2 meters above the surface, encompassing land, sea, and inland water bodies, as spatio-temporal climate data. We use the ERA5 dataset, which provides the surface temperature spanning the last 83 years (1940-2022) at a spatial resolution of 0.25° corresponding to $N_\theta = 721$ latitude points and $N_\phi = 1440$ longitude points and the spherical harmonic band-limit $L = 720$. In anticipation of future climate simulations and experiments expected to feature even higher spatial resolutions, we perform spline interpolation to upscale the data to higher spatial resolutions consistent with the band-limits of 720, 1,440, 2,880, and 5,219 (see green stars in Figure 1). This upscaling of the number of grid points serves dual purposes: i) to demonstrate the scalability of the design and computational performance of our emulator and ii) to ensure our approach remains relevant and aligned with advancements in climate modeling. We use $K = 5$ to capture periodic variations along the time and $P = 3$ for auto-regressive modeling of spherical harmonic coefficients, a choice deemed sufficient based on existing related research [22], [23]. For the temporal dimension, we consider hourly and daily resolutions, resulting in $T = 306{,}600$ (35 years, hourly, 1988-2022) and $T = 30{,}295$ (83 years, daily 1940-2022) time slots, after adjusting for the omission of an extra day in leap years. As highlighted earlier, even daily resolution has not been fully adopted in the existing emulator designs primarily due to the significant computational demand to process large amounts of data. We plot the hourly ERA5 data alongside the emulations in Figure 2 for two days. We also plot the daily simulations and emulations for the same two days in Figure 4 for different mixed-precision variants. These figures illustrate the statistical consistency (studied in detail in [23]) between the emulations and simulations at different precision levels employed in the tile-based decomposition of the covariance matrix.

B. Performance Measured in Flop/s

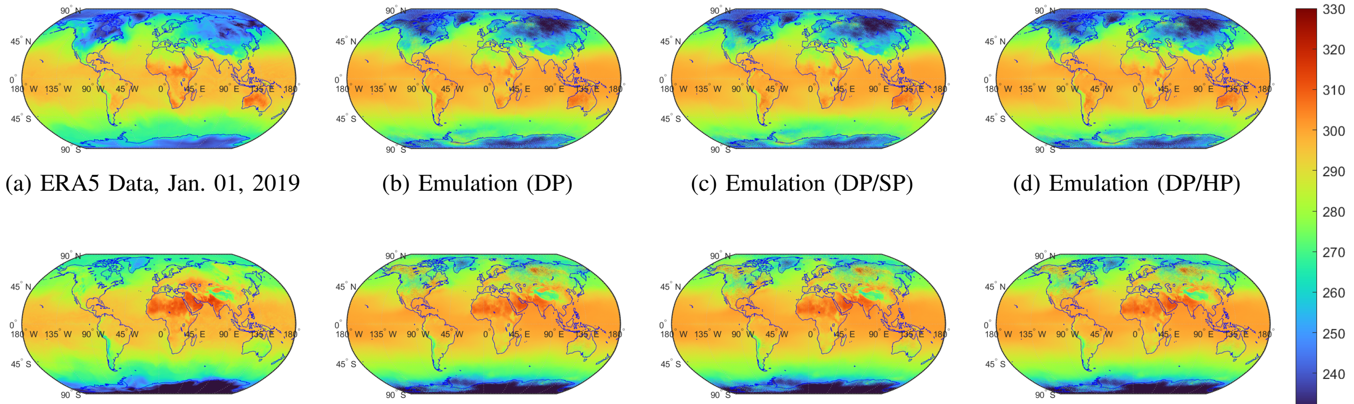
We assess the numerical robustness of our mixed-precision approach by applying the climate emulator with various L values, specifically, 720, 1,440, 2,880, and 5,219. We report the Flop/s and scalability achieved on up to 9,025 nodes on Frontier (36,100 AMD MI250X GPUs), 1,936 nodes on Alps (7,744 GH200 GPUs), 1,024 nodes on Leonardo (4,096 A100 GPUs), and 3,072 nodes of Summit (18,432 V100 GPUs). To optimize performance, we fine-tuned our code and integrated it with the highly optimized BLAS/LAPACK libraries available on each of these systems.

We conducted experiments on the four supercomputers, utilizing four precision variants: full double-precision (DP); a single band as DP with the remaining tiles as SP (DP/SP); DP with 5% as SP and the remaining tiles as HP (DP/SP/HP); and a single band as DP with the remaining tiles as HP (DP/HP). The performance results were reported in terms of achieved Flop/s for a single Cholesky factorization, which is required after the SHT step to emulate new climate datasets. We varied the problem sizes to encompass all spatial resolutions down to 0.034°. The performance data spanned different numbers of nodes across the various systems, limited by availability but nevertheless providing a comparison of our computational capability across different configurations and resolutions. This approach allowed us to assess the efficiency of each variant under different computational loads and system scales.

C. Strong and Weak Scaling Efficiency

Having greatest availability on about-to-be-retired Summit, we gave it a triumphant swan song of strong and weak scaling experiments using up to 2,048 nodes (12,288 NVIDIA V100 GPUs) with four Cholesky decomposition variants: the reference DP, and variants DP/SP, DP/SP/HP, and DP/HP. For the weak scaling experiments, we assigned the same amount of data to each GPU while varying the number of GPUs. This approach helped us evaluate how effectively our MP code can maintain performance per GPU as the scale of the system increases. It allowed us to assess the consistency of performance across different GPU counts, to understand how the application scales horizontally.

For strong scalability, we used a constant workload that 512 nodes of Summit would typically handle and tested this workload on larger configurations of 1,024 and 2,048 nodes. This allowed us to measure the scaling efficiency across the different precision variants. Specifically, we examined how



(a) ERA5 Data, Jan. 01, 2019 (b) Emulation (DP) (c) Emulation (DP/SP) (d) Emulation (DP/HP)

(e) ERA5 Data, Jun. 01, 2019 (f) Emulation (DP) (g) Emulation (DP/SP) (h) Emulation (DP/HP)

Fig. 4: ERA5 surface temperatures (in K) on (a) Jan. 1, 2019, and (e) Jun. 1, 2019. The corresponding emulated temperatures using DP, DP/SP, and DP/HP, are plotted in the same row.

the performance per GPU with the fixed workload decreased with adding more GPUs and computed the efficiency as the performance ratio per GPU on the 512-node configuration. This metric helps in understanding the effectiveness of resource utilization and in pinpointing potential overheads or inefficiencies in the parallelization of the code, especially as we scale up to a larger number of GPUs.

D. Attributes of the HPC Systems

This study focuses on four GPU-based systems. Frontier at ORNL consists of 9,472 nodes, each equipped with four AMD MI250X GPUs; each GPU is a multi-chip module (MCM) with two AMD Graphics Compute Dies (GCDs) and four Terabytes of flash memory. Frontier has a theoretical peak performance of 1.71 EFlop/s in DP operations. Recently listed Alps at CSCS consists of many resources; we focus on the Grace-Hopper partition of 2,688 CPU-GPU supernodes, each equipped with four NVIDIA GH200 with 72 ARM cores CPU and one NVIDIA H100 Tensor Core GPU with 96GB of HBM3. Alps has a theoretical peak performance of 353.75 PFlop/s in DP operations. Leonardo at CINECA comprises 3,456 nodes, each with four NVIDIA A100 64GB GPUs and boasting a theoretical peak performance of 306.31 PFlop/s in DP. Summit at ORNL supercomputer consists of 4,608 compute nodes, each equipped with two 22-core IBM Power9 CPUs and six NVIDIA Tesla V100 GPUs, with over 600 GB of coherent memory accessible by both CPUs and GPUs and a theoretical peak performance of 200.79 PFlop/s in DP.

VII. PERFORMANCE RESULTS

We analyze performance on the four above-mentioned systems, covering various problem sizes and weak and strong scalability efficiency on Summit.

A. Performance of Mixed-Precision Cholesky on Summit

In [34], we propose a banded-mixed-precision approach through PaRSEC at large-scale. At this scale, tile-based mixed-precision Cholesky factorization has a special communication pattern. First, POTRF (c, c) on the diagonal tile broadcasts to TRSMs in column c for a given tile-based matrix. Secondly, TRSM (r, c) broadcasts to GEMMs in row r, GEMMs in column c, and SYRK (r, c). Due to different precisions, a task might receive a tile at a different precision than it operates on, necessitating a precision conversion. This conversion can occur at the sender or the receiver. If we apply down-precision conversion at the sender, we boost performance by reducing communication. Send-based conversion enhances performance, as exemplified by TRSM to GEMMs conversions, which reduces repeated conversions across successive GEMMs. An evaluation with single and multiple GPUs of sender/receiver-based conversions was introduced in [35]. In this work, we have optimized the performance by dynamically applying sender-based conversion and compared the performance to our previous work [34] on 128 nodes of Summit. As shown in Figure 5, we can obtain a speedup up to 1.53X with DP/HP. The other two curves have slightly improved, i.e., 1.15X and 1.06X for DP and DP/SP, respectively, because the advantage in communication is hindered by the computation overhead, resulting in a negligible performance gain compared to the DP/HP variant.

Figure 6 shows the performance comparisons of several precision configurations: DP, DP/SP, DP/SP/HP, and DP/HP on 2,048 nodes of Summit. The red dashed line represents the peak performance of DP on 2,048 Summit nodes. The figure illustrates that DP Cholesky achieves 61.7% of the peak performance. The figure also reports the speedups achieved by the three MP configurations compared to DP computation, showing speedups of 2X, 3.2X, and 5.2X for DP, DP/SP/HP, and DP/HP, respectively. The figure also demonstrates that with a problem size of approximately 8M, DP/HP Cholesky on 2,048 nodes can achieve up to 304.84 PFlop/s.

B. Scalability of Mixed-Precision Cholesky on Summit

Figure 7 (left) illustrates the weak scalability of Summit performing MP Cholesky decomposition. The reported performance is normalized to reflect single-node performance and facilitate a more granular analysis. As the number of nodes increases from 64 to 2,048, each maintains performance levels per node for different configurations. Figure 7 demonstrates excellent scaling efficiency when maintaining the same work-

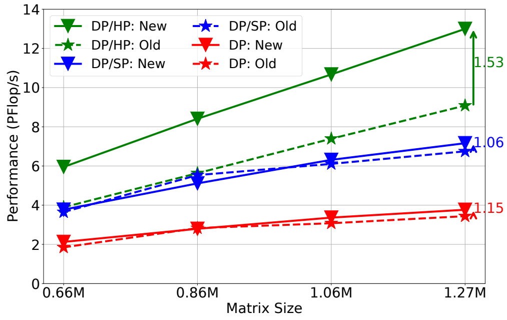


Fig. 5: Performance of Cholesky factorization for DP, DP/SP, and DP/HP on 128 nodes of Summit (768 NVIDIA V100 GPUs) compared to the work in [34], with varying covariance matrix sizes.

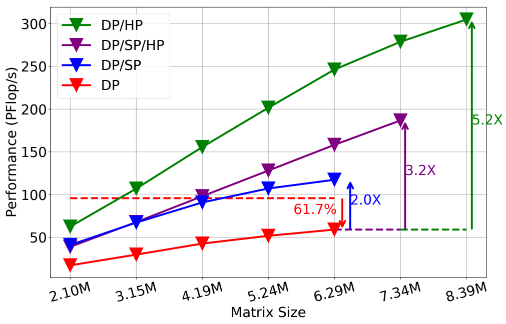


Fig. 6: Performance of Cholesky factorization using four precision levels: DP, DP/SP/HP, DP/SP, and DP/HP on 2,048 nodes of Summit (12,288 NVIDIA V100 GPUs), with varying covariance matrix sizes.

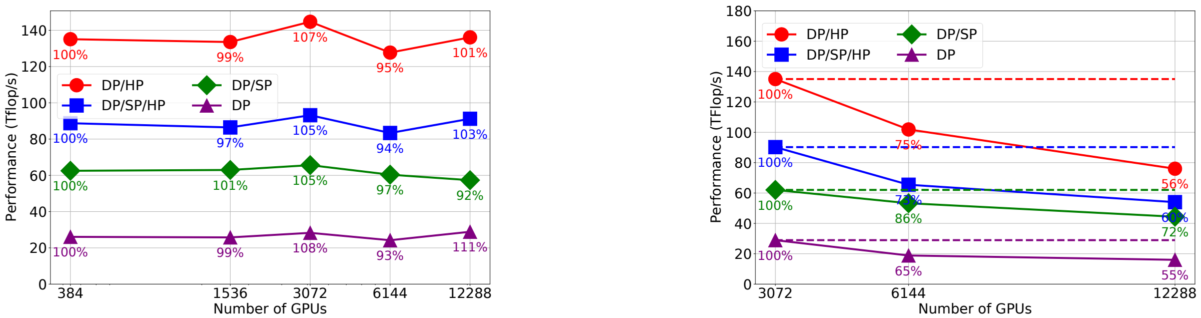


Fig. 7: Weak (left) and strong (right) scaling on 2,048 nodes of Summit (12,288 NVIDIA V100 GPUs), using various precision levels, i.e., DP, DP/SP, DP/SP/HP, and DP/HP.

load per GPU as the number of GPUs increases. Compared to the baseline performance on 384 GPUs, the scaling efficiency ranges from 92% to 111% across various GPU counts, up to 12,288 V100 GPUs, using different MP variants.

Figure 7 (right) illustrates the strong scalability on the Summit system when performing MP Cholesky decomposition. This figure shows how the system handles a fixed total workload as the number of nodes increases from 3,072 to 12,288 GPUs, effectively showcasing the utilization of additional computational resources while maintaining the same problem size. The largest problem size that fits the memory of 3,072 GPUs is used as the fixed workload. The performance metrics are also normalized to show performance per node. As typically expected in strong scalability scenarios, the performance per node tends to decrease when more resources are applied to the same problem size. For instance, with the DP variant, the efficiency drops to about 55% per node when comparing the performance of 2,048 nodes with 512 nodes.

However, efficiency improves noticeably when using mixed-precision variants. Efficiency increases to 72% for DP/SP and 60% for DP/SP/HP, indicating that these configurations mitigate performance degradation more effectively than the pure DP setup. This suggests that employing mixed-precision can be a strategic choice for enhancing computational efficiency in large-scale systems, especially when significantly scaling up the number of nodes. The efficiency of DP/HP in a large number of nodes has dropped to 56% due to less work being required per node compared to the DP/SP variant.

C. *Performance Comparisons of Mixed-Precision Cholesky*

Table I highlights DP/HP performance in PFlop/s of the mixed-precision Cholesky on 1,024 nodes of the four systems, i.e., Frontier, Alps, Leonardo, and Summit, maxing out the device memory with each matrix size in addition to PaRSEC internal memory buffers. Moreover, since Summit has a higher GPU count per node than other systems, we normalize the absolute performance per GPU. We can observe performance improvements in NVIDIA GPU hardware generations. While NVIDIA A100 achieves similar performance to AMD MI250X, NVIDIA GH200 outperforms it by 1.6X. There is still room for further improvements on Frontier and Alps systems by leveraging their network interconnect using CUDA-aware MPI to mitigate data movement overheads. This requires additional support within PaRSEC and will be addressed in future work.

D. *Experiments of Largest Scale Climate Emulators*

Figure 8 reports the largest scale experiments conducted on the Leonardo and Summit systems, using DP/HP configurations with 1,024 and 3,072, respectively. Additionally, we report Alps results using three node counts, i.e., 1,024, 1,600, and 1,932, and Frontier results using four node counts, i.e., 2,048, 4,096, 6,400, 9,025. We achieved 0.243 EFlop/s on the Leonardo system using 1,024 nodes with a problem size of approximately 8.39M. On the Summit system, activating 3,072 nodes yielded 0.375 EFlop/s for a problem size of around 12.58M. On Alps, we measured 0.364, 0.623 and 0.739 EFlop/s on 1,024, 1, 600 and 1,936 nodes with problem sizes

TABLE I: DP/HP Performance on 1,024 nodes of the four systems, i.e., Frontier, Alps, Leonardo, and Summit.

System	Frontier	Alps	Leonardo	Summit
Vendor	AMD	NVIDIA		
Chip	MI250X	GH200	A100	V100
# GPUs	4,096	4,096	4,096	6,144
Matrix Size	8.39M	10.49M	8.39M	6.29M
Performance (PFflop/s)	223.7	384.2	243.1	153.6
TFlop/s/GPU	54.6	93.8	57.2	25

of approximately 10.49M, 14.42M, and 15.73M, respectively. Finally, on the Frontier system, we executed on 2,048, 4,096, 6,400, and 9,025 nodes and achieved 0.316, 0.523, 0.715, and 0.976 EFlop/s with problem sizes of approximately 12.58M, 16.78M, 20.97M, and 27.24M, respectively.

VIII. IMPLICATIONS

Dynamic runtime systems have become critical to scalability and performance. Not only do they insulate computational scientists from the programming aspects, allowing them to focus on algorithmics and science, but they also enable computer scientists to quickly explore and adapt solutions to the evolving needs of applications. The level of performance obtained on the highest-end platforms shows that task-based runtime systems have matured, outpacing their initial node-level scope to become scalable partners for HPC applications that can handle resource management and communication overlap, assuming the algorithm has enough computations.

In this work, we propose a mixed-precision approach that leverages the capability of modern GPUs to perform fast, lower-precision computations with the power of runtime systems. This enables us to develop efficient solvers, such as Cholesky factorization, that can accelerate the performance of applications at their algebraic core. With dynamic runtime systems, we balance load across inhomogeneous tiles of large-scale matrices and hide communication latencies with dense compute-bound algorithms. While this solution is assessed in the context of a climate emulator, it has the potential to benefit a wide range of applications that require rapid computations using dense linear solvers. We have optimized and tested our code across four different GPU architectures – AMD MI250X, NVIDIA GH200, NVIDIA A100, and NVIDIA V100 – on four distinct systems: Frontier, Alps, Leonardo, and Summit.

Our work integrates state-of-the-art HPC to develop extreme-scale climate emulators, showcasing the potential of supercomputers to advance climate modeling at ultra-high resolution and enable significant savings in the petabytes of storage required for storing climate simulations. Our approach uses the SHT for modeling spatio-temporal climate data, accommodating climate data sourced from various spatial resolutions. We establish a sustainable approach to climate modeling on GPUs by shifting computation from communication-bound sparse double-precision kernels inherent in PDE-based models to low-precision tensor core kernels. As evidenced by IPCC assessment reports and our endorsement letters, the exascale climate emulator holds significant potential for the climate community, advancing climate research and policy making.

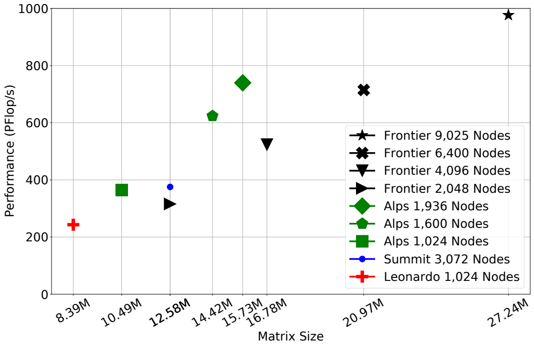


Fig. 8: Performance of largest runs on Summit, Leonardo, Alps, and Frontier; with additional run-up points on Alps and Frontier, all using the DP/HP precision variant.

Climate emulators have already found widespread application in assessing the sensitivity of various physical parameters [24], performing model calibration [25], analyzing emission scenarios [17], studying the internal variability on climate model outputs [26], investigating the persistence of climate anomalies and analyzing the temporal dynamics of climate systems [27]. Extending these applications, this work will enable climate scientists to apply ultra-high spatial resolution models to enhance understanding and predict climate change impacts.

This work also holds significant potential in advancing the development of machine learning (ML) and AI-driven methods for forecasting or prediction applications in climate science. For example, our framework can enhance hybrid approaches that combine physics with ML, thereby generating higher fidelity climate data [48] and refine 3D neural networks recently proposed for weather forecasting [49]. Finally, we highlight the utility of our work for applications beyond climate science, given the flexible, scalable design of the emulator. For instance, our HPC innovations can be used to build an emulator for N-body simulations in cosmology for generating cosmological mass maps required for estimating the cosmological parameters [50]. In summary, we aim to drive the development of robust and multi-variate emulators for generating high-resolution spatio-temporal data that ride the waves of high performance architecture.

ACKNOWLEDGMENTS

For computer time, this research used Shaheen III at the Supercomputing Laboratory of the King Abdullah University of Science and Technology (KAUST) in Thuwal, Saudi Arabia; Frontier and Summit at the Oak Ridge Leadership Computing Facility at the US DOE's Oak Ridge National Laboratory; Alps at CSCS in Lugano, Switzerland; and Leonardo at CINECA in Bologna, Italy. The authors are deeply grateful to the OLCF, CSCS, and CINECA for the discretionary allocations that allowed partial scaling runs after completing the science runs at KAUST and to NVIDIA for brokering the CSCS and CINECA connections. We also would like to thank Bilel Hadri from the KSL team at KAUST, as well as Thomas Herault, Aurelien Bouteiller, and Joseph Schuchart from the ICL team at the University of Tennessee, Knoxville (UTK), for their essential support in facilitating this work.

REFERENCES

[1] J. Rising et al. The missing risks of climate change. *Nature*, 610(7933):643–651, 2022.

[2] I. P. on Climate Change (IPCC). *Climate Change 2022 – Impacts, Adaptation and Vulnerability: Working Group II Contribution to the Sixth Assessment Report of the Intergovernmental Panel on Climate Change*. Cambridge University Press, 2023.

[3] I. Sognnaes et al. A multi-model analysis of long-term emissions and warming implications of current mitigation efforts. *Nature Climate Change*, 11(12):1055–1062, 2021.

[4] W. Mort. Avoiding dangerous climate change. *Ecoscience*, 14(1):134–135, 2007.

[5] G. Flato et al. Evaluation of climate models. In *Climate change 2013: the physical science basis. Contribution of Working Group I to the Fifth Assessment Report of the Intergovernmental Panel on Climate Change*, pp. 741–866. Cambridge University Press, 2014.

[6] J. Dongarra and D. Keyes. The co-evolution of computational physics and high-performance computing. *Nature Physics Reviews*, 2024. To appear.

[7] S. Mickelson et al. A new end-to-end workflow for the Community Earth System Model (version 2.0) for the Coupled Model Intercomparison Project Phase 6 (CMIP6). *Geoscientific Model Development*, 13(11):5567–5581, 2020.

[8] Earth System Grid Federation. ESGF Data Statistics, 2024. http://esgf-ui.cmcc.it/, Last accessed on 2024-02-27.

[9] V. Balaji et al. Requirements for a global data infrastructure in support of CMIP6. *Geoscientific Model Development*, 11(9):3659–3680, 2018.

[10] NASA Center for Climate Simulation. GISS and NCCS Contribute to CMIP6 International Climate Model Intercomparison Project, 2021. https://www.nccs.nasa.gov/news-events/nccs-highlights/GISS-CMIP6, Last accessed on 2024-02-27.

[11] C. Schär et al. Kilometer-scale climate models: prospects and challenges. *Bulletin of the American Meteorological Society*, 101(5):E567 – E587, 2020.

[12] B. Stevens et al. DYAMOND: the dynamics of the atmospheric general circulation modeled on non-hydrostatic domains. *Progress in Earth and Planetary Science*, 6(61), 2019.

[13] M. Taylor et al. The simple cloud-resolving E3SM atmosphere model running on the Frontier Exascale System. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage, and Analysis (SC'23)*, 2023.

[14] P. M. Caldwell et al. Convection-permitting simulations with the E3SM global atmosphere model. *Journal of Advances in Modeling Earth Systems*, 13(11), 2021.

[15] P. M. Caldwell et al. The DOE E3SM coupled model version 1: description and results at high resolution. *Journal of Advances in Modeling Earth Systems*, 11(12):4095–4146, 2019.

[16] S. Castruccio and M. L. Stein. Global space-time models for climate ensembles. *The Annals of Applied Statistics*, pp. 1593–1611, 2013.

[17] S. Castruccio et al. Statistical emulation of climate model projections based on precomputed GCM runs. *Journal of Climate*, 27(5):1829–1844, 2014.

[18] P. H. G. Philip B. Holden, Neil R. Edwards and R. D. Wilkinson. Emulation and interpretation of high-dimensional climate model outputs. *Journal of Applied Statistics*, 42(9):2038–2055, 2015.

[19] R. Link et al. Fldgen v1.0: an emulator with internal variability and space–time correlation for earth system models. *Geoscientific Model Development*, 12(4):1477–1489, 2019.

[20] S. Castruccio et al. Reproducing internal variability with few ensemble runs. *Journal of Climate*, 32(24):8511–8522, 2019.

[21] J. Jeong et al. A stochastic generator of global monthly wind energy with Tukey g-and-h autoregressive processes. *Statistica Sinica*, 29(3):1105–1126, 2019.

[22] H. Huang et al. Saving storage in climate ensembles: a model-based stochastic approach (with discussion). *Journal of Agricultural, Biological and Environmental Statistics*, 28(2):324–344, 2023.

[23] Y. Song et al. Efficient stochastic generators with spherical harmonic transformation for high-resolution global climate simulations from CESM2-LENS2. *Journal of the American Statistical Association*, 0(0):1–15, 2024.

[24] J. E. Oakley and A. O'Hagan. Probabilistic sensitivity analysis of complex models: a Bayesian approach. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 66(3):751–769, 2004.

[25] W. Chang et al. Fast dimension-reduced climate model calibration and the effect of data aggregation. *The Annals of Applied Statistics*, 8(2):649–673, 2014.

[26] W. Hu and S. Castruccio. Approximating the internal variability of bias-corrected global temperature projections with spatial stochastic generators. *Journal of Climate*, 34(20):8409–8418, 2021.

[27] G. Branstator and H. Teng. Two limits of initial-value decadal predictability in a CGCM. *Journal of Climate*, 23(23):6292–6311, 2010.

[28] N. Cressie and G. Johannesson. Fixed rank kriging for very large spatial data sets. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 70(1):209–226, 2008.

[29] J. Guinness. Gaussian process learning via Fisher scoring of Vecchia's approximation. *Statistics and Computing*, 31(3):25, 2021.

[30] D. Bolin et al. Covariance–based rational approximations of fractional SPDEs for computationally efficient Bayesian inference. *Journal of Computational and Graphical Statistics*, 33(1):64–74, 2024.

[31] C. G. Kaufman et al. Covariance tapering for likelihood-based estimation in large spatial data sets. *Journal of the American Statistical Association*, 103(484):1545–1555, 2008.

[32] A. Datta et al. Hierarchical nearest-neighbor Gaussian process models for large geostatistical datasets. *Journal of the American Statistical Association*, 111(514):800–812, 2016.

[33] E. Agullo et al. Numerical linear algebra on emerging architectures: The PLASMA and MAGMA projects. In *Journal of Physics: Conference Series*, volume 180. IOP Pub., 2009.

[34] S. Abdulah et al. Accelerating geostatistical modeling and prediction with mixed-precision computations: a high-productivity approach with PaRSEC. *IEEE Transactions on Parallel and Distributed Systems*, 33(4):964–976, 2021.

[35] Q. Cao et al. Reducing data motion and energy consumption of geospatial modeling applications using automated precision conversion. In *2023 IEEE International Conference on Cluster Computing (CLUSTER)*, pp. 330–342. IEEE, 2023.

[36] OpenMP. OpenMP 4.5 Complete Specifications, 2015.

[37] C. Augonnet et al. StarPU: a unified platform for task scheduling on heterogeneous multicore architectures. *Concurrency and Computation: Practice and Experience*, 23(2):187–198, 2011.

[38] A. Duran et al. A proposal to extend the OpenMP tasking model with dependent tasks. *Intl. Journal of Parallel Programming*, 37(3):292–305, 2009.

[39] T. Heller et al. Application of the ParalleX execution model to stencil-based problems. *Computer Science - Research and Development*, 28(2-3):253–261, 2013.

[40] L. V. Kale and S. Krishnan. CHARM++: a portable concurrent object oriented system based on C++. In *ACM Sigplan Notices*, volume 28, pp. 91–108. ACM, 1993.

[41] G. Bosilca et al. PaRSEC: exploiting heterogeneity to enhance scalability. *Computing in Science & Engineering*, 15(6):36–45, 2013.

[42] M. Bauer et al. Legion: expressing locality and independence with logical regions. In *IEEE International Conference for High Performance Computing, Networking, Storage and Analysis, SC*, pp. 1–11, 2012.

[43] J. Chowdhury et al. Fast and accurate spherical harmonic transform for spatio-temporal regular grid data. *IEEE Signal Processing Letters*, 31:1825–1829, 2024.

[44] R. A. Kennedy and P. Sadeghi. *Hilbert Space Methods in Signal Processing*. Cambridge University Press, Cambridge, UK, March 2013.

[45] Q. Cao et al. Leveraging PaRSEC runtime support to tackle challenging 3D data-sparse matrix problems. In *2021 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, pp. 79–89. IEEE, 2021.

[46] S. Abdulah et al. Geostatistical modeling and prediction using mixed precision tile Cholesky factorization. In *26th International Conference on High Performance Computing, Data, and Analytics (HiPC)*, pp. 152–162. IEEE, 2019.

[47] Q. Cao et al. Reshaping geostatistical modeling and prediction for extreme-scale environmental applications. In *SC22: International Conference for High Performance Computing, Networking, Storage and Analysis*, pp. 1–12. IEEE, 2022.

[48] S. Yu et al. ClimSim: A large multi-scale dataset for hybrid physics-ML climate emulation. *Advances in Neural Information Processing Systems*, 36, 2024.

[49] K. Bi et al. Accurate medium-range global weather forecasting with 3d neural networks. *Nature*, 619(7970):533–538, 2023.

[50] N. Perraudin et al. Emulation of cosmological mass maps with conditional generative adversarial networks. *Frontiers in Artificial Intelligence*, 4:673062, 2021.