

ADVANCED REVIEW

High-Performance Statistical Computing (HPSC): Challenges, Opportunities, and Future Directions

Sameh Abdulah¹ | Mary Lai O. Salvaña²  | Ying Sun³ | David E. Keyes¹ | Marc G. Genton³ 

¹Applied Mathematics and Computational Science Program, King Abdullah University of Science and Technology, Thuwal, Saudi Arabia

²Department of Statistics, University of Connecticut, Storrs, Connecticut, USA | ³Statistics Program, King Abdullah University of Science and Technology, Thuwal, Saudi Arabia

Correspondence: Marc G. Genton (marc.genton@kaust.edu.sa)

Received: 28 July 2025 | **Revised:** 29 September 2025 | **Accepted:** 3 November 2025

Editor-in-Chief: David W. Scott

Keywords: GPUs | high-performance computing | mixed-precision computing | parallel statistical algorithms | statistical computing

ABSTRACT

We recognize the emergence of a statistical computing community focused on working with large computing platforms and producing software and applications that exemplify high-performance statistical computing (HPSC). The statistical computing (SC) community develops software that is widely used across disciplines. However, it remains largely absent from the high-performance computing (HPC) landscape, particularly on platforms such as those featured on the www.top500.org or Green500 lists. Many disciplines already participate in HPC, mostly centered around simulation science, although data-focused efforts under the artificial intelligence (AI) label are gaining popularity. Bridging this gap requires both community adaptation and technical innovation to align statistical methods with modern HPC technologies. We can accelerate progress in fast and scalable statistical applications by building strong connections between the SC and HPC communities. We present a brief history of SC, a vision for how its strengths can contribute to statistical science in the HPC environment (such as HPSC), the challenges that remain, and the opportunities currently available, culminating in a possible roadmap toward a thriving HPSC community.

This article is categorized under:

Software for Computational Statistics > High Performance Software

Software for Computational Statistics > Software/Statistical Software

Algorithms and Computational Methods > Methods for High Performance Computing

1 | Introduction

Statistical computing (SC) is a foundational discipline that aims to merge statistical theory with computational techniques to turn data into actionable insights (Kennedy and Gentle 2021). At its core, it involves designing and implementing novel algorithms, simulations, and models to solve complex problems across various domains, including climate science, economics, and machine learning. Unlike general-purpose computing, statistical

computing is grounded in data analysis, probabilistic reasoning, and statistical inference, drawing heavily on probability theory, statistics, and numerical methods (Givens and Hoeting 2012). As data continues to scale in both size and complexity, statistical computing has become indispensable, not only for managing large datasets but also for enabling predictive modeling and informed decision-making in real time. Programming languages such as R (Ihaka and Gentleman 1996), Julia (Bezanson et al. 2017), and Python (Van Rossum and Drake Jr. 1995) have

Sameh Abdulah and Mary Lai O. Salvaña contributed equally to this work.

© 2025 Wiley Periodicals LLC.

played a critical role in this evolution, empowering researchers to rapidly post-process, analyze, and visualize data. Together, these tools have made SC a central engine of modern data science.

At the same time, the field is facing a shift that is both technical and cultural: the convergence of SC with high-performance computing (HPC). The rapid growth of data and demand for more sophisticated analytics has pushed traditional, single-processor workflows to their limits. In response, SC is moving to distributed and parallel computing frameworks that leverage the massive scale of modern HPC architectures. To recognize this transition, we introduce the term *high-performance statistical computing (HPSC)*, which formalizes the critical intersection where advanced statistical methods meet cutting-edge computational infrastructure. HPSC is not simply a technical move. It indicates a fundamental shift in the conceptualization and development of scalable solutions to statistical problems. Scientific applications are already extending the limits of HPC capabilities, particularly in areas involving large-scale simulations, uncertainty quantification, and real-time inference. Recognizing HPSC as a separate domain enables us to articulate its challenges more effectively, tap into its unexploited potential, and encourage deep collaboration between the HPC and SC communities.

The advancement of HPSC relies on a foundation of interdisciplinary collaboration, which brings together the strengths of modern computational science, statistics, and computer engineering. Progress in this emerging domain demands not only technical expertise in both statistics and computer science but also the ability to bridge these fields meaningfully. This integration requires a deep understanding of statistical literature, algorithmic design, parallel computing architectures, and hardware, as well as an understanding of domain-specific challenges. When these fields intersect, they enable the development of high-performance implementations that preserve statistical accuracy while fully leveraging the computational power of HPC platforms. This collaborative approach will become increasingly critical to advancing the field as data volumes continue to grow and computational demands increase, driving the next wave of scalable, statistically sound solutions to meet the challenges of modern data analysis.

To date, a lot more work within the SC community has been invested in adopting dataflow technologies, that is, frameworks such as Apache Spark (Hariadi et al. 2020), Dask (Rocklin 2015), or TensorFlow (Abadi et al. 2016), that represent computations as directed acyclic graphs of operations on data than in exploring message passing interface (MPI)+X approaches, that is, a hybrid parallel programming paradigm that combines MPI (Gropp et al. 1999) for distributed-memory communication with an additional model “X” (such as OpenMP (Chandra 2001) for multicore CPUs or CUDA for GPUs) for shared-memory or accelerator-level parallelism (Hori et al. 2021), despite the latter being better suited for many of the most computationally demanding tasks in the field. There are multiple factors behind this trend. First, dataflow architectures are often perceived as more accessible, thanks to their relatively simple parallel execution models, compared with MPI-based systems, which require explicit message passing and distributed memory management.

Such frameworks abstract away much of this complexity. This helps statisticians use them even if they do not fully understand parallel programming. Second, the need for interactive computing has strongly influenced choices. Tools like R and Python are deeply tied to exploratory, iterative analysis, where users develop models, visualize outputs, and debug interactively. Moreover, dataflow systems integrate more naturally into cloud-based and batch-processing environments, making them a compelling choice for both industry and large-scale analytics. Third, most statisticians and data scientists primarily work with multicore processors rather than large HPC clusters. This has led to a strong emphasis on shared-memory parallelism (e.g., OpenMP, threading in R and Python) rather than distributed memory paradigms, such as MPI.

While dataflow technologies offer some distributed computing capabilities, they are often employed in a manner that still aligns with a multicore approach rather than fully leveraging the power of supercomputing. Even when statisticians engage with distributed computing, it is often through tools such as Dask (Rocklin 2015), Ray (Moritz et al. 2018), or Spark, which offer a gentler learning curve and allow scaling across nodes without requiring low-level programming expertise. While these tools have clear advantages in usability and accessibility, we confirm that MPI+X remains significantly underutilized in SC. Supercomputing environments offer high efficiency, lower latency, and access to highly optimized numerical libraries. These advantages can substantially accelerate tasks such as large-scale simulations, Monte Carlo methods, and Bayesian inference. Fully embracing HPC technologies can unlock performance levels far beyond what current dataflow frameworks typically provide. In the following sections, we delve deeper into these challenges and present a roadmap for making MPI+X approaches more accessible, scalable, and impactful for the SC community.

This paper attempts a comprehensive review of HPSC, examining the current landscape of its challenges, opportunities, and future directions in a field that is rapidly evolving. Our analysis includes recent advances in distributed computing frameworks, parallel processing algorithms, and modern hardware architectures, which have significantly expanded the capabilities of SC. This review focuses on how current statistical applications do not fully utilize the MPI+X approach, thereby missing out on significant performance gains. Through a systematic survey of the literature and prevailing practices, we identify key technological trends, methodological breakthroughs, and emerging research trajectories that are shaping the future of HPSC. The scope of this paper bridges both theory and practice, with a focus on how statistical methodology can effectively align with HPC infrastructure to meet today’s growing computational demands. Herein, we address critical issues such as scalability, reproducibility, and accessibility, highlighting strategies that have proven successful and pointing to areas where further innovation is needed. Our goal is to provide a valuable resource for researchers and practitioners working at the intersection of statistics and HPC, and to offer a forward-looking perspective on the evolving role of HPSC in data-intensive science.

This paper is organized as follows. Section 2 shows the historical development of HPC and SC, highlighting their progressive

convergence. Section 3 surveys the literature, illustrating how HPSC is transforming various application domains. Section 4 identifies key computational and methodological challenges in adapting statistical algorithms to modern HPC platforms. In Section 5, we explore opportunities where HPC can enable scalable, precise, and energy-efficient statistical analysis. Section 6 outlines future directions, focusing on specialized hardware, federated inference, standardization, and new algorithmic paradigms. Finally, Section 7 presents a roadmap for building a sustainable and inclusive HPSC community.

2 | HPC and SC: A History

In recent years, the evolution of computing hardware has entered a new phase, driven by the slowing of transistor scaling and the growing demand for high-performance processing. This marks a turning point from the earlier era when Moore's Law reliably predicted rapid gains in transistor density. In 1965, Gordon Moore, co-founder of Intel, extrapolated 4 years of data to predict an exponential increase in the number of transistors per integrated circuit (Moore 1965). The ensuing decades bore him out with a doubling period of approximately 18 months. However, that pace has since slowed. By 2005, the rise in single-chip central processing unit (CPU) performance, powered by millions of transistors, began to face serious thermal and power limitations (Sutter 2005). In response, hardware designers shifted focus from increasing clock speeds to integrating multiple cores on a single chip, introducing the era of multicore processors. These architectures introduced multiprocessing execution, placing new demands on software to manage parallelism effectively. In parallel with the shift toward multicore CPUs, another revolution was quietly taking shape. At the end of the 20th century, NVIDIA introduced the first stand-alone graphics processing unit (GPU), the "GeForce 256," designed to accelerate gaming graphics (NVIDIA 1999). What began as a specialized tool for rendering images quickly evolved into a powerhouse for general-purpose computing. With far greater parallelism and memory bandwidth than traditional CPUs, GPUs have become indispensable accelerators in modern HPC, at the expense of less memory per processor. Today, GPUs play a central role not only in gaming but also in high-performance applications, including deep learning, scientific simulation, and large-scale statistical inference. Modern HPC systems capitalize on this capability, combining GPUs with multicore CPUs to solve problems that exceed the limits of conventional computing. These platforms are specifically designed to address computational challenges that require vast processing power, high memory throughput, and low-latency communication. Supercomputers, the summit of HPC infrastructure, integrate thousands of CPUs and GPUs connected by ultra-fast interconnects such as InfiniBand (Grun 2010) and Slingshot (De Sensi et al. 2020). This tightly coupled architecture enables rapid coordination across processing units, allowing for the efficient execution of large-scale simulations and data-intensive workloads. Beyond traditional processors, modern HPC systems increasingly incorporate specialized accelerators, such as Tensor Processing Units (TPUs, Jouppi et al. 2017) and Field-Programmable Gate Arrays (FPGAs, Putnam et al. 2014), to optimize specific tasks, including deep learning inference and cryptographic operations.

To maximize performance, many existing HPC environments rely on heterogeneous computing frameworks that integrate

CPUs and GPUs into cooperative workflows. These systems excel at handling highly parallel workloads, often with dramatic gains in throughput and efficiency. Performance is commonly evaluated across several dimensions: FLOPS (Floating-Point Operations Per Second), which measures raw computational speed; memory bandwidth, which affects data transfer rates; a start-up time for an action that must be amortized over the amount of data transmitted or processed; and scalability and parallel efficiency, which reflect how well performance holds as the workload or system size increases; and energy efficiency.

The modern era of HPC began with the 1976 debut of the Cray-1, which delivered roughly 133 MFLOPS. Progress accelerated quickly, producing more HPC machines. The Cray X-MP (1982) reached 800 MFLOPS, the Intel iPSC Hypercube (1985) surpassed 1.6 GFLOPS, the Cray Y-MP (1988) hit 2.67 GFLOPS, and the CM-5 (1991) achieved 131 GFLOPS (Dongarra et al. 1990). These advances fueled breakthroughs across science, defense, and industry, introducing HPC as a foundational technology for large-scale computation. Over the past three decades, technologically leading nations have engaged in an ongoing race to develop and possess the world's fastest supercomputers, as represented in the biannual TOP500 rankings (TOP500 2024). These machines have redefined what is computationally possible, powering breakthroughs in many applications, including physics, climate modeling, AI, and engineering. Their rise has been fueled by rapid innovations in CPU, GPU, and system architecture, often driven by advancements in the industry. Figure 1 shows this trajectory from 1993 to 2024, highlighting the top-performing supercomputers at each point in time. Peak performance, expressed on a logarithmic scale in FLOPS, highlights the exponential growth trend that has characterized HPC. From gigaflops in the early 1990s to exaflops today, the curve reveals a consistent doubling of computational power roughly every 1.5 to 2 years, mirroring Moore's Law and the scaling of parallelism. Some systems, such as ASCI Red, BlueGene/L, Summit, Fugaku, and most recently Frontier and El Capitan, mark significant inflection points in this evolution. If this trend can be sustained, zettaflop-scale systems are expected to emerge as early as the 2030s, introducing a new era of extreme-scale computing for many scientific fields.

SC also has a long history dating back to the mid-20th century. One of the earliest documented statistical packages, the Bio-Medical Data Package (BMDP), was developed in the 1960s at the UCLA Health Computing Facility for IBM mainframes (Dixon 1983). Designed for parametric and nonparametric statistical analyses, BMDP marked the first wave of statistical computing. The second wave emerged in the late 1960s and early 1970s with the release of SPSS (1968), Genstat (1970), and SAS (1971) (Nie et al. 1970). These packages introduced powerful tools such as regression, ANOVA, and cluster analysis to mainframe computing, revolutionizing statistical analysis in research, and were written in FORTRAN. The third statistical computing wave began with the development of the S language in the 1970s, led by John Chambers and his team at Bell Labs (Chambers 2008). Originally developed as a general-purpose statistical tool for academic users, S was later acquired by Insightful Corporation and rebranded as S-PLUS,

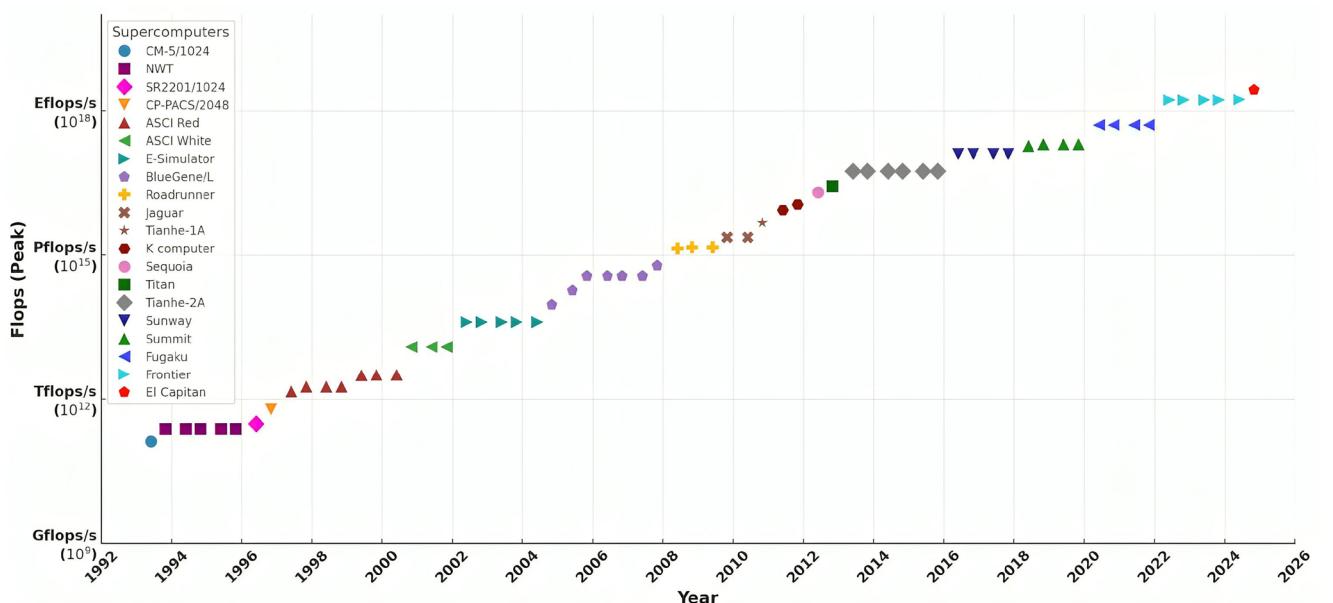


FIGURE 1 | Evolution of supercomputer peak performance over time: From CM-5 to El Capitan.

which emerged as a leading commercial statistical software in the 1990s. Around the same time, Robert Gentleman and Ross Ihaka introduced R (Ihaka and Gentleman 1996), establishing a new standard for open, extensible, and efficient statistical computing. Their work, published in the *Journal of Computational and Graphical Statistics*, laid the foundation for modern statistical software and established key principles for efficient statistical computation that remain relevant today. The late 1990s and early 2000s saw the convergence of this foundation with advances in parallel architectures and algorithms, setting the stage for a new chapter. This paper introduces the fourth wave in SC, that is, HPSC. Unlike previous waves, HPSC is not defined by a specific language or software, but by a broader shift toward scalable and parallel approaches that leverage the power of modern HPC systems.

Today, HPC includes a broad spectrum of technologies, most of which involve some form of parallel computing. Two dominant tracks have emerged: one rooted in the scientific community, the other in industry. In science, HPC is structured by national supercomputing centers focused on simulation science, typically leveraging the MPI+X programming model (Bernholdt et al. 2020; Hori et al. 2021). Interestingly, Hori et al. (2021) use R for portions of their data analysis and graphics. In contrast, the business world has adopted large-scale data centers, driven by cloud computing, search engines, and e-commerce, which favor dataflow-based parallelism (Kamburugamuve et al. 2018). These divergences reflect fundamental differences in application needs, as well as a degree of cultural and technical isolation between the two communities. At first glance, one might assume the dataflow model is better suited to statistical computing due to its focus on data processing. However, the reality is more complex and in many cases, the opposite holds true. A careful comparison of three widely used algorithms in computational statistics¹ shows that MPI-based implementations consistently outperform their dataflow counterparts, often by significant margins (Kamburugamuve et al. 2018). The MPI model, grounded in mathematical structures such as matrix

algebra, aligns naturally with the computational patterns of statistical analysis. While dataflow systems offer strengths in scalability and data management, especially in long-running services, they are optimized for use cases distinct from the demands of HPSC.

3 | HPSC in the Literature

HPSC applies HPC to complex statistical problems, enabling massive data analysis, large-scale simulations, and models that were once infeasible. This convergence now benefits many fields. This section highlights areas where HPSC already delivers impact and continues to drive innovation.

3.1 | Climate Science

HPSC has significantly advanced climate science by enabling scalable inference, prediction, and uncertainty quantification for massive spatio-temporal environmental datasets. In the literature, a notable contribution is the *ExaGeoStat* framework (Abdulah et al. 2018a, 2018b, 2019), which introduced tile-based, architecture-agnostic solvers for dense and approximate Gaussian process (GP) likelihoods, supporting exact and approximate maximum likelihood estimation (MLE) for spatio-temporal data modeling on CPU-GPU clusters. Tile algorithms partition matrices into cache-sized tiles, enabling efficient multicore/GPU parallelism (Gustavson 2001; Buttari et al. 2009).

A key innovation in *ExaGeoStat* is the integration of tile low-rank (TLR) approximations, where each matrix tile is individually compressed in low-rank (Abdulah et al. 2018b), with mixed-precision techniques, where each tile is computed at a precision level (FP64/FP32/FP16) chosen adaptively based on matrix norms or application-specific features (Abdulah et al. 2019; Cao et al. 2022, 2023). This approach reduces memory footprint and

computational cost while preserving likelihood fidelity (Cao et al. 2022).

Later extensions to *ExaGeoStat* pushed beyond Gaussian assumptions (Mondal et al. 2022, 2023), added hybrid-precision solvers with dynamic scheduling via PaRSEC (Abdulah et al. 2021), and enabled containerized deployments across diverse HPC systems (Abdulah, Ejarque, et al. 2024). These innovations enabled large-scale inference on existing supercomputers with notable speedups in modeling evapotranspiration fields. To improve usability, *ExaGeoStat* was wrapped as the R package *ExaGeoStatR* (Abdulah et al. 2023), bridging high-performance modeling and the R computing environment. Within this ecosystem, Salvaña et al. (2021, 2022) developed massively parallel MLE solvers for multivariate and spatio-temporal GPs, leveraging distributed dense linear algebra and global parallel particle swarm optimization (Schutte et al. 2004) to support exascale kriging for climate and pollution data. Geng et al. (2023, 2025) further enhanced performance by implementing a GPU-accelerated modified Bessel function, which is integrated into *ExaGeoStat* for the efficient evaluation of the Matérn covariance function. Other *ExaGeoStat* features include convolutional neural networks-based partitioning of nonstationary spatial domains (Nag et al. 2025), diagnostics for predictive efficiency in low-rank models (Hong et al. 2021), and large-scale simulation capabilities showcased in three international competitions (Huang et al. 2021; Abdulah et al. 2022; Hong et al. 2023), which benchmarked performance across a range of covariance structures and approximations. *ExaGeoStat*'s contribution to HPSC in climate science was recognized when Abdulah, Baker, et al. (2024) received the ACM Gordon Bell Prize for Climate Modeling.

Beyond *ExaGeoStat*, HPSC-enabled tools have advanced climate modeling through GPU-accelerated profile likelihoods for dense spatial covariance matrices (Xu and Brown 2024), decentralized low-rank inference (Shi et al. 2025), sparsity-discovering kernels (Noack et al. 2023), parallel particle-based Bayesian estimation (Katzfuss and Hammerling 2017), and parallel Markov Chain Monte Carlo (MCMC) techniques (Solonen et al. 2012). Moreover, Abdul Fattah et al. (2025) introduced sTiles, a GPU-accelerated framework for sparse Cholesky factorization of structured matrices, with a particular focus on block arrowhead types commonly arising in Bayesian inference and scientific computing.

A GPU-powered neural network has also demonstrated significant potential for fast estimation of spatial covariance parameters (Gerber and Nychka 2021). These methods address scale and communication bottlenecks while supporting robust, interpretable inference (Mann 2020). Additional contributions include an out-of-core GPU-based geostatistical interpolation (Allombert et al. 2014), a parallel kriging technique based on the k-d tree method (Wei et al. 2015), and GPU-accelerated likelihood estimation software in R. Efficiency improvements in accelerating Cholesky decomposition using a mixed-precision approach (Cao et al. 2023), GPU implementation of Vecchia approximations (Pan et al. 2024; James and Guinness 2024; Pan, Abdulah, Genton, and Sun 2025; Pan, Abdulah, Abduljabbar, et al. 2025), and parallel approaches to inference of spatial

extremes (Castruccio et al. 2016) further illustrate how HPSC reshapes both methodology and practice in climate science.

3.2 | Geoscience

HPSC has also become a driving force in geoscience, unlocking scalable modeling, simulation, and inference across massive spatial datasets. Traditional geostatistical methods, such as kriging, which have long been limited by their $\mathcal{O}(N^3)$ computational cost, are now being reimaged through GPU-accelerated and parallel frameworks. Substantial speedups, often exceeding 18X, have been achieved using CUDA for kriging (Hu et al. 2016; Liu et al. 2022; De Luca, Di Luccio, et al. 2022), while OpenCL-based architectures offer similar performance gains (Huang et al. 2016). Hybrid CPU-GPU systems have further extended acceleration to adaptive local kriging for earthquake studies (Chang et al. 2018) and the construction of LiDAR-derived digital elevation models (Danner et al. 2012). Delaunay tetrahedra (Yao et al. 2015), smoothing techniques, such as Savitzky–Golay filtering (De Luca, Galletti, and Marcellino 2022), and massively parallel GP regression (Gramacy et al. 2014; Krasnosky and Roman 2022), performed in GPUs, continue to drive these gains. Beyond interpolation, simulation and surrogate modeling have emerged as key application areas. For instance, Gopinathan et al. (2021) developed a multi-threaded emulation platform for tsunami scenario generation, while Rouholahnejad et al. (2012) proposed a parallel processing software that supports the hydrologic simulator SWAT (Soil Water Assessment Tool). Generalized likelihood estimation methods for hydrological inference (Yin et al. 2020) and fast spatio-temporal weighted regression algorithms (Que et al. 2021) have both been deployed on MPI clusters, fostering an ecosystem of scalable statistical methods. Visualization and simulation tools have also evolved, leveraging GPU rendering (Heitzler et al. 2017), edge/cloud architectures (Mudunuru et al. 2024), and task-based schedulers (Nesi et al. 2022) to handle dynamic, high-resolution geoscientific workflows. Applications span weather prediction (Müller et al. 2019), nuclear radiation modeling (Xiao et al. 2025), atmospheric optics (Russkova 2018), and air pollution modeling (Molnár Jr et al. 2010). Furthermore, seismic hazard modeling (Li et al. 2025), GPU-based surface reconstruction (Yan et al. 2015, 2016), and gap-filling using the modified planar rotator method (Lach and Žukovič 2024) highlight the breadth of geoscience applications now running at scale. Underlying these achievements is a growing recognition that statistical methods, hybrid hardware, and adaptive software must co-evolve. Foundational studies on the usage of heterogeneous computer architectures for geoscientific datasets (Prasad et al. 2015, 2017) reinforce this message. Taken together, these efforts establish HPSC not just as an enabler, but as an essential computational backbone for next-generation geoscientific modeling and discovery.

3.3 | Genomics and Bioinformatics

HPSC has become indispensable in genomics and bioinformatics, enabling scalable, reproducible inference across massive and heterogeneous biological datasets. The integration of distributed architectures and optimized numerical solvers has transformed phylogenetics, genome-wide association studies (GWAS), and

early disease detection. In phylogenetics, parallel implementations such as TREE-PUZZLE (Schmidt et al. 2002), fastDNAml (Stewart et al. 2001), and RAxML (Stamatakis et al. 2008; Stamatakis and Ott 2008) laid the groundwork for likelihood-based inference at scale. These were followed by GPU-accelerated frameworks, such as BEAGLE (Suchard and Rambaut 2009; Ayres et al. 2012, 2019), which are now central to Bayesian phylogenetic engines like MrBayes (Pratas et al. 2009; Ayres and Cummings 2017) and RevBayes (Smith et al. 2024). Recent tools, such as CMAPLE (Ly-Trong et al. 2024), push further, enabling pandemic-scale inference through memory-efficient likelihood evaluation and massively parallel topology search. In GWAS, performance breakthroughs have been achieved through the use of memory-efficient implementations and distributed matrix operations. Tools such as PBOOST (Yang et al. 2015), rMVP (Yin et al. 2021), SAIGE-GPU (Rodriguez et al. 2024), BLUPPmrMLM (Li, Wang, et al. 2024), MPH (Jiang 2024), and Quickdraws (Loya et al. 2025) offer substantial speedups over legacy pipelines. At the extreme scale, Ltaief et al. (2024) demonstrate mixed-precision Kernel Ridge Regression for multivariate GWAS on exascale architectures. Single-cell and spatial omics workflows are also increasingly HPSC-enabled. GPU-accelerated tools, such as SPAC (Liu et al. 2025), enable the scaling of spatial transcriptomics analysis to millions of cells. fastglmpca (Weine et al. 2024) improves GLM-PCA for single-cell RNA sequencing by accelerating Poisson likelihood optimization. ParProx (Ko et al. 2021) enables penalized regression in ultrahigh-dimensional omics datasets. Other advances include parallel co-expression inference (Shealy et al. 2019), GPU-based spike-triggered modeling (Mano and Clark 2017), a parallel minimum spanning tree construction algorithm (Olman et al. 2008), and an HPC-AI hybrid framework (Patil et al. 2024), which achieves over a 200× speedup. Efficient scheduling across multi-GPU nodes (Thavappiragam et al. 2021) and adaptive runtime optimization strategies (Wang et al. 2024) highlight the growing importance of workload balance and architectural efficiency. These innovations collectively demonstrate how HPSC is reshaping the field of genomics and bioinformatics applications. As biological data continues to grow in scale and complexity, the integration of statistical modeling with hardware-optimized computation is enabling unprecedented levels of speed, performance, resolution, and insight.

3.4 | Physics and Astronomy

HPSC has become a foundational enabler in physics, cosmology, and astrophysics, powering both simulation and inference for increasingly complex systems and datasets. GPU-accelerated pipelines lie at the heart of this transformation, driving advancements in a wide array of applications—from Monte Carlo simulations of spin models (Barash et al. 2017) to gravitational wave detection (Talbot et al. 2019; Wysocki et al. 2019; Smith et al. 2020; Bandopadhyay and Moore 2024; Saltas and Oliveri 2025; Jan et al. 2025) and probabilistic programming for particle physics (Baydin et al. 2019). These capabilities are underpinned by GPU-native inference frameworks (Fluke et al. 2011; Szalai-Gindl et al. 2021; Dunn et al. 2022). In cosmology, HPC-compatible libraries such as CosmoPower (Spurio Mancini et al. 2022), CONNECT (Nygaard et al. 2023), JAX-COSMO (Campagne et al. 2023),

OLÉ (Günther et al. 2025), and GLaD (Wang, Suyu, et al. 2025) enable rapid, high-dimensional posterior estimation with orders-of-magnitude improvements in runtime. Many of these tools incorporate likelihood inference (Makinen et al. 2021; Ho et al. 2024; Piras et al. 2024), designed to leverage GPU acceleration for scalability. Large-scale simulations have also benefited dramatically from HPSC advances. Frameworks such as sCOLA (Leclercq et al. 2020) and Farpoint (Frontiere et al. 2022) leverage exascale computing to model cosmic structure formation, while GPU-accelerated linear algebra routines (Rodrigues et al. 2014) and eigensolver libraries like ELPA (Küs et al. 2019) serve core needs in quantum and accelerator physics. Emerging infrastructures are increasingly hybrid and cloud-enabled. HPC platforms that integrate CPU/GPU architectures (Amadio et al. 2016; Boyle et al. 2017) along with modular frameworks, such as Π4U (Hadjidoukas, Angelikopoulos, Papadimitriou, and Koumoutsakos 2015), CUDAHM (Szalai-Gindl et al. 2021), and AI4HPC (Sarma et al. 2024), and HPC-portable languages, such as Julia in high-energy physics (Stewart et al. 2025), exemplify the fusion of high-performance engines with machine learning-driven inference. Despite these gains, challenges persist. Ensuring reproducibility across heterogeneous systems (Jarp et al. 2012; Tiwari et al. 2015; Krupa et al. 2021), standardizing inference error propagation, and generalizing emulators across physical domains remain open problems. Yet as HPSC continues to mature, it is becoming the statistical and computational backbone of physics-informed, simulation-driven discovery, transforming how uncertainty, complexity, and scale are addressed in modern physics.

3.5 | Economics

HPSC is also rapidly reshaping economics, enabling researchers to tackle high-dimensional, nonlinear, and spatially structured models that were once computationally out of reach. In the literature, Aldrich et al. (2011) demonstrated how GPUs can be leveraged to solve dynamic equilibrium models, while Creel and Goffe (2008) provided a broader tutorial on utilizing multicore architectures and parallel computing tools. Overviews by Aldrich (2014) and Fernández-Villaverde and Zarruk Valencia (2018) helped codify GPU and parallel programming strategies tailored for economics. These foundations have since expanded across diverse econometric domains. Parallel Bayesian inference and adaptive sampling for econometric models have pushed the limits of inference in dynamic systems, as demonstrated by Baştürk et al. (2016). Scalable toolkits, such as those introduced by Chib et al. (2023), and the sequential Monte Carlo frameworks advocated by Herbst and Schorfheide (2014), have improved tractability in complex posterior spaces. GPU-accelerated likelihood evaluations (White and Porter 2014), high-performance indirect likelihood inference methods (Creel and Zubair 2012), and stochastic volatility models (Leon-Gonzalez and Majoni 2025) underscore the power of hardware-aware design. Other key developments include embarrassingly parallel bootstraps (Delgado and Parmeter 2013), and parallelizable linear transformation methods for mixed-frequency data (Qian 2016). Comparative studies of MPEC vs. NFXP (Dong et al. 2022) have clarified the computational trade-offs in structural estimation. In high-frequency settings, GPU-enabled Bayesian Hawkes process models (Holbrook et al. 2021, 2022)

have supported inference on spatio-temporal contagion data, while parallel Gibbs samplers have been applied to generalized autoregressive conditional heteroscedastic-intertemporal capital asset pricing models with extreme-value distributions (Khanthaporn and Wichitakorn 2023). Nonparametric and hierarchical methods continue to stretch computational resources. The *np* package (Hayfield and Racine 2008) enables flexible estimation via kernel methods but remains computationally intensive. Recent advances, such as hierarchical geographically weighted regression with parallel backfitting (Hu et al. 2024) and hierarchical panel data models for stochastic metafrontiers (Amsler et al. 2023) demonstrate how HPSC can address spatial and group heterogeneity at scale. Big-data-focused toolkits built on distributed computing frameworks, such as Apache Spark (Hariadi et al. 2020), have also emerged as robust solutions for econometric modeling on massive datasets (Bluhm and Cutura 2020). Discrete games now leverage GPU-powered simulation and equilibrium search methods (Chung et al. 2023). Parallel algorithms for solving finite mixture models (Ferrall 2005) and parameterized expectations in macroeconomics (Creel 2008) further highlight the convergence of algorithmic innovation and hardware acceleration. Altogether, these developments showcase how algorithmic design, hardware optimization, and software engineering converge to redefine econometric methods in the era of complex data and real-time policy modeling.

3.6 | Finance

HPSC has revolutionized modern finance, enabling real-time analytics, high-dimensional inference, and complex simulation-based estimation across a wide range of applications, including portfolio optimization, volatility modeling, credit risk scoring, and derivative pricing. Early studies by Zenios (1999) highlighted the role of HPC in accelerating Monte Carlo-based security pricing and Value-at-Risk models, while Liyanage et al. (2017) demonstrated how such simulations could be radically optimized through vectorization and parallelization. Building on this, Guo et al. (2023) introduced *HXPY*, a high-performance data pipeline that leverages SIMD (Single Instruction Multiple Data) and CUDA to outperform legacy tools in financial time-series processing. Parallelization has also enhanced nonparametric risk-neutral density estimation, as seen in bandwidth-optimized approaches for options pricing (Monteiro and Santos 2023). Stochastic simulation frameworks for pricing and sensitivity analysis (Dixon et al. 2012, 2014; Kucherenko et al. 2023) continue to benefit from high-throughput GPU implementations. Moreover, machine learning (ML) integration is a growing trend. CUDA-enabled ML pipelines have been applied to real-time stock forecasting (Kumari et al. 2020), while parallelized ML approaches have accelerated credit risk scoring (Hentosh et al. 2022). Scalable Bayesian workflows for financial time series (Casarin et al. 2016) push the limits of high-dimensional modeling. Algorithmic innovations, such as parallel minorization–maximization algorithms for heteroscedastic regression (Nguyen et al. 2016), have helped bridge the gap between statistical accuracy and computational speed. Recent works highlight the importance of the design trade-offs (Inggs et al. 2013; Jain et al. 2015; Mudnic et al. 2017), emphasizing the need to balance model complexity, runtime constraints, and inference accuracy. Advances in massively

parallel market environments for financial reinforcement learning (Wang, Holzer, et al. 2025) and high-fidelity calibration objectives for agent-based financial market simulations (Wang, Ren, and Yang 2025) underscore the convergence of algorithmic modeling and high-performance engineering, while parallel Bayesian inference for stock price prediction on Apache Spark (Hariadi et al. 2020) further exemplifies this trend. As financial systems demand greater speed and precision, HPSC has become a critical pillar in the computational infrastructure of quantitative finance.

3.7 | Recurring Principles Across HPSC Applications

Across the different domains shown above, common principles highlight how HPSC is evolving. Approximation strategies, such as low-rank representations and randomized methods, can reduce computational and memory costs while maintaining statistical accuracy. Mixed-precision arithmetic and the use of GPUs and other accelerators are emerging as effective ways to balance performance with numerical stability across various fields. Parallel design patterns, including divide-and-conquer, task and data parallelism, and MapReduce, provide reusable frameworks for scaling statistical algorithms. Numerous studies have demonstrated the benefits of software and hardware co-design, where statistical methodologies are closely integrated with HPC libraries, containerization, and distributed frameworks. In conclusion, these recurring ideas highlight that while applications differ, HPSC is driven by a shared set of strategies that enable scalability, efficiency, and reproducibility across disciplines. Table 1 summarizes the different principles discussed in the above subsections.

4 | HPSC Challenges

Leveraging HPC resources to accelerate statistical analysis presents challenges different from those of traditional computational science. Many statistical algorithms were initially designed for sequential execution and modest data sizes, making their transition to parallel, distributed, and heterogeneous architectures far from a straightforward process. These algorithms often exhibit complex data dependencies, irregular memory access, and heightened sensitivity to numerical precision, factors that complicate scaling on modern HPC platforms. Meeting these demands requires more than raw compute power. It requires algorithm redesign, minimizing data movement between computing units, ensuring numerical stability at scale, and building robust, portable, and reproducible software. In the following section, we outline the core challenges facing HPSC, including parallelization strategies, memory and data management, numerical precision, and implementation bottlenecks, with a focus on what is needed to achieve scalability and accuracy.

4.1 | Adapting Statistical Algorithms for Parallel Environments

One of the main challenges in using HPC for statistical applications is that many fundamental algorithms were designed to be sequential by default. As a result, they often fail to scale

TABLE 1 | Recurring principles across HPSC applications.

Principle	Domains	Representative techniques/tools
Approximations	Climate, Geoscience, Genomics, Finance	Tile/Block approximation, Vecchia, sketching, variational inference, approximate risk-neutral densities
Mixed-precision computing	Climate, Genomics, Finance	FP64/FP32/FP16 solvers, stochastic rounding, kernel ridge regression, energy-efficient likelihood evaluation
GPU/accelerator utilization	Physics, Climate, Finance, Genomics	CUDA/OpenCL kernels, cuBLAS/cuSPARSE, BEAGLE phylogenetics, GPU-accelerated MCMC, portfolio optimization, volatility modeling
Federated/distributed frameworks	Healthcare, Finance, Climate	Spark, Dask, Ray, Federated Gaussian Processes, Hadoop-based financial risk analysis
Software/hardware co-design	Climate, Genomics, Finance, Statistics	ExaGeoStat, RScalAPACK, bigGP, pbDR, RCOMPSS; domain-specific HPC toolkits for finance (HXPY, stochastic simulation frameworks)

effectively. Unlike simulation-based scientific codes, statistical routines frequently involve iterative, data-dependent logic that resists naive parallelization. The limited research on this issue focuses on the parallelization of statistical algorithms. Unlocking HPC performance thus requires more than just access to advanced hardware; it demands a redesign of the whole algorithm. This may involve restructuring existing methods to expose parallelism or developing new algorithms specifically designed for distributed environments and accelerators. This can be achieved by identifying independent computational tasks and blocks, optimizing data locality (i.e., keeping data physically close to where they are processed) and communication patterns (i.e., how parallel tasks, such as processes or threads, exchange data during computation), and balancing workloads across different processing units. Without such adaptations, statistical applications cannot fully exploit the power of HPC platforms.

The design of parallel algorithms always relies on the architecture of the target hardware. It also adds a layer of complexity for developers to consider. Thus, optimizing performance across various systems, such as multicore CPUs and GPU-accelerated clusters, requires careful alignment between the algorithmic structure and hardware characteristics. One of the central challenges is picking an appropriate parallel programming model and design pattern that fully exploits the available parallelism in the algorithm while minimizing overhead. Parallel programming models provide abstractions that define how computational tasks are structured, executed, and coordinated concurrently across multiple processing units. Examples include OpenMP (Chandra 2001), which enables shared-memory parallelism using threads; MPI (Hori et al. 2021), which supports distributed-memory parallelism through message passing; and CUDA (NVIDIA 2025), which allows fine-grained parallel programming on NVIDIA GPUs.

Equally important is the choice of a suitable communication model, which governs how data and tasks are shared across

processing units. Historically, the fork-join model has been the dominant paradigm, mainly in shared-memory settings. In this approach, execution begins sequentially until a parallelizable block is encountered, at which point multiple threads are spawned, typically mapped to individual cores, to perform concurrent work before merging back into the main thread. While simple and widely supported, this model can harm performance in certain scenarios. Specifically, it limits optimization opportunities related to data synchronization, thread scheduling, and cache efficiency, primarily when threads compete for shared memory resources. To overcome these bottlenecks, other algorithmic patterns are increasingly important, especially in the context of statistical computing, where data and computation structures are often irregular. Strategies such as Divide and Conquer, Task Parallelism, Data Parallelism, Map-Reduce, and Asynchronous Parallelism offer more flexible and scalable avenues for exploiting concurrency. Selecting the best pattern depends on the structure of the given task, the nature of data dependencies, and the characteristics of the underlying hardware. Incorporating these approaches early in algorithm design is essential for achieving high performance in HPC environments. Figure 2 illustrates the six primary parallel computing patterns: Data Parallelism, Fork/Join, Map/Reduce, Divide and Conquer, Task Parallelism, and Pipeline Parallelism. Each subfigure illustrates a distinct execution structure, showcasing how computations or data flow are organized to exploit concurrency in modern high-performance and distributed systems.

Different computing patterns suit different scenarios. Data parallelism is ideal when the same computation is run on many independent pieces, such as bootstrap resamples, cross-validation folds, or Monte Carlo draws. Map/Reduce is appropriate when “group-by and aggregate” at scale is needed, for example, summaries. A pipeline suits end-to-end workflows where data flows through stages. Fork/join fits phased methods that require a global combine step, for example, computing partial gradients or sufficient statistics in parallel, then synchronizing to update

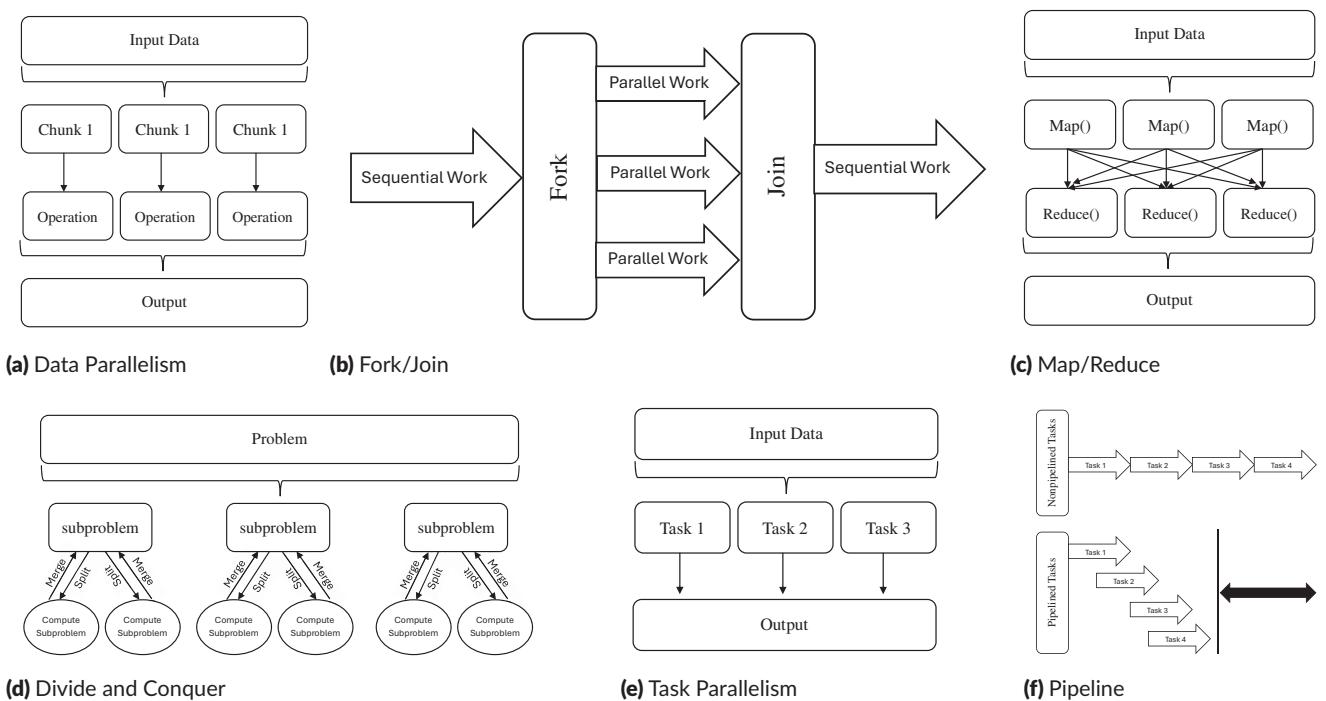


FIGURE 2 | Common parallelization patterns in HPC.

a model. Divide-and-conquer is helpful when the problem naturally splits recursively, such as when partitioning the data or domain, solving subproblems, and merging the results. Task parallelism is more suitable for heterogeneous, dependency-driven work, such as hyperparameter searches, multi-model comparison pipelines, or sparse/graph computations.

A growing number of studies in the literature illustrates how diverse parallel design patterns and programming models can be effectively applied to SC. For instance, Hong et al. (2022) introduced the SOLID algorithm, a divide-and-conquer strategy for sparse predictive modeling and penalized regression in biostatistics. By coupling screening and one-step linearization, SOLID can achieve significant speedups without negatively impacting statistical accuracy. Similarly, Wang et al. (2021) applied divide-and-conquer principles to optimize sparse Cox regression for high-dimensional survival data, showing substantial gains in computational efficiency. Task-based parallelism has also shown strong potential. Hadjidoukas, Angelikopoulos, Kulakova, et al. (2015) used this approach in Bayesian uncertainty quantification, employing adaptive load balancing to schedule numerical differentiation and sampling tasks across multicore CPUs and GPUs. The result was a highly efficient, scalable framework for large-scale inference. In the context of metaheuristic optimization, Santander-Jiménez and Vega-Rodríguez (2016) implemented an asynchronous parallel model, allowing worker threads to proceed independently without synchronization barriers. This design reduced idle time and improved scalability, particularly for stochastic algorithms. For problems with strong data partitioning potential, Yen (2020) introduced a block-splitting method that reformulates fused penalty estimation into a separable structure, enabling efficient data parallelism. Numerous additional studies highlight the breadth of parallel design strategies applied in

SC. These include a wide array of works based on divide-and-conquer strategies (Bai et al. 2011; Loh et al. 2011; Kim 2016; Sabnis et al. 2016; Guhaniyogi et al. 2017; Deng et al. 2017; Xu et al. 2019; Minsker 2019; Roy et al. 2019; Su 2020; Sukkuea and Heednacram 2022; Hong et al. 2022; Vyner et al. 2023; Chen et al. 2024), task-based parallelism (Canales et al. 2016; Ko et al. 2020; Abdulah et al. 2021; Mondal et al. 2022), data parallelism (Michalak et al. 2012; Homrighausen and McDonald 2016), and MapReduce-based approaches (Liu et al. 2010; Mohammed et al. 2014; Ferraro Petrillo et al. 2019; Mwamnyange et al. 2021). In summary, these works demonstrate the flexibility and effectiveness of parallel computing paradigms in addressing the computational demands of modern statistical problems, where they often achieve huge improvements in scalability and runtime without sacrificing accuracy.

4.2 | Data Management and Movement Optimization

Efficient data management and movement are fundamental challenges in HPSC, where datasets grow to terabyte and petabyte scales. Traditional statistical workflows typically assume that data can be held in memory or processed sequentially, assumptions that break down in distributed, high-performance environments. At HPC scales, concerns such as data locality, partitioning strategy, and communication overhead become performance-critical in many cases, where the cost of moving data—whether across nodes or through memory hierarchies—can exceed the benefits gained from parallel computation. Statistical tasks, such as subsampling, shuffling, or conditioning, often induce irregular memory access patterns that are difficult to optimize using conventional memory management techniques.

The complexity increases further when dealing with high-dimensional, heterogeneous data. Designing scalable data pipelines requires a smooth understanding of how to efficiently move and store data without compromising analytical accuracy. To meet these demands, researchers increasingly turn to out-of-core algorithms (Yildirim et al. 2023), data compression techniques (Kriemann et al. 2022), and communication-avoiding methods (Shohdy et al. 2016). Some strategies, such as locality-aware partitioning (i.e., carefully distributing data across processing units to maximize data locality) and asynchronous data prefetching (i.e., loading data into memory before they are needed for computation), aim to reduce latency and improve throughput, though often at the cost of added complexity or overhead. Ultimately, constructing data pipelines that align with the architecture's computational and memory hierarchy is essential for unlocking the full potential of HPSC. As such, data movement is not merely a systems concern; it is a central design constraint in SC at scale that can heavily impact performance.

4.3 | Numerical Stability and Precision Issues

Numerical stability is a critical challenge in SC, and its importance grows with the scale and complexity of modern HPC workloads (Altman et al. 2004). Large-scale computations amplify the impact of rounding errors, cancellation, and other numerical artifacts that may be negligible at smaller scales. Statistical methods such as Bayesian inference, maximum likelihood, and covariance matrix inversion are sensitive to small errors in input data or intermediate results. When computations are distributed across thousands of processing units, tiny floating-point rounding errors from individual operations can accumulate, sometimes leading to noticeable differences in large-scale results. While this is a general concern, today's advancements in low-precision computing, driven by GPUs and specialized accelerators, introduce an additional layer of complexity. Lower-precision formats, such as FP16 and BFLOAT16, offer significant improvements in performance and energy efficiency. However, these gains often come at the expense of reduced numerical accuracy. This accuracy degradation is especially problematic for iterative solvers, gradient-based optimization, and matrix decompositions, which form the backbone of many statistical algorithms (Abdullah et al. 2021; Salvaña et al. 2024). Balancing speed and robustness in this setting requires a careful redesign of the algorithm. One promising mitigation strategy is the use of stochastic rounding, which reduces systematic bias introduced by low-precision arithmetic and improves the stability of iterative computations at scale (Croci et al. 2022). However, the challenges go beyond precision. Reproducibility becomes increasingly a problem in parallel environments, where thread scheduling, non-deterministic reductions, and hardware-level optimizations can lead to variations in results. In statistical applications, especially those with downstream scientific or policy implications, ensuring numerical and statistical reproducibility is not optional; it is a foundational requirement. Addressing these concerns requires both algorithmic safeguards and systems-level support for deterministic, high-precision computing when needed.

4.4 | Software Design for Portability and Reproducibility

Designing statistical software for HPC requires a careful balance between performance, portability, and reproducibility. Portability means that code can run efficiently across diverse architectures, including CPUs, GPUs, and emerging accelerators, without requiring extensive rewrites. Achieving this typically involves adopting abstraction layers and performance-portable programming models, such as Kokkos (Sunderland et al. 2016) and RAJA (Hornung and Keasler 2014), which decouple hardware-specific details from algorithmic logic. In addition to performance-portable programming models, software portability can be further enhanced through the use of containerization. Containerization is a lightweight virtualization technique that packages an application together with its dependencies, libraries, and runtime into a single portable unit. Containerization tools, such as Docker and Singularity (Schulz et al. 2016; Abdullah, Ejarque, et al. 2024), also play a key role, enabling consistent execution environments across platforms while simplifying deployment and dependency management.

Moreover, reproducibility is essential, particularly in statistical applications where numerical results directly inform scientific conclusions or policy decisions. Ensuring reproducibility requires careful control over software environments, versioning, data provenance, and the generation of random numbers. This often involves workflow orchestration systems, standardized data formats, and comprehensive logging of configuration states. When these practices are integrated early in the development lifecycle, teams can build HPC-ready statistical software that is not only high-performing but also robust, maintainable, and verifiable. Embedding portability and reproducibility as core design goals is fundamental to advancing reliable HPSC.

4.5 | Implementation Challenges on Heterogeneous Architectures

Deploying statistical algorithms on heterogeneous HPC architectures, including CPUs, GPUs, and specialized accelerators, also presents numerous practical implementation challenges. These systems demand explicit control over data movement across complex memory hierarchies, careful load balancing, and hardware-specific performance tuning. Statistical workloads are often characterized by irregular computation, conditional branching, and fine-grained data dependencies, which do not map cleanly onto massively parallel devices, such as GPUs. As a result, porting statistical algorithms typically requires re-thinking the core data structures, optimizing memory access patterns, and integrating parallel libraries such as cuBLAS (GPU-accelerated linear algebra), Thrust (parallel algorithms and data structures), or oneAPI (cross-architecture portability) to extract performance. Portability adds further complexity. Achieving efficient execution across vendor-specific platforms, such as NVIDIA's CUDA and AMD's ROCm, requires either developing separate backends or using abstraction layers that can introduce overhead or limit optimization. Addressing these issues demands a modular software architecture that cleanly separates hardware-agnostic logic from performance-critical

kernels, enabling both scalability and maintainability across diverse systems. Language-level limitations can further impede adoption. For example, R, the widely used programming language in the SC community, does not directly support GPU computing or parallel execution on heterogeneous systems. This gap underscores the need for broader community investment in extending R's capabilities for high-performance environments and heterogeneous distributed systems. Efforts, such as the *MPCR* package for multi- and mixed-precision computing (Salvaña et al. 2024), the *future* package in R, which offers a unified and extensible framework for concurrent and parallel programming with a conceptually simple and expressive syntax (Bengtsson 2020), and the integration of R with the COMPSS runtime system for distributed execution via the *RCOMPSS* package (Zhang et al. 2025), represent important steps forward. However, these tools are still in their early stages and require further development, optimization, and broader community adoption to fully enable scalable statistical computing on modern, heterogeneous HPC platforms.

5 | HPC-Driven Opportunities in Statistical Computing

In this section, we examine how HPC extends the boundaries of what is possible in SC. It highlights key opportunities for accelerating inference, enhancing model fidelity, and reducing reliance on heuristics or coarse approximations. From enabling finer-grained analysis to supporting complex, multiscale models, HPC is not only an enabler of speed; it is a motivation for reimaging the scale and scope of statistics itself.

5.1 | Parallel Statistical Algorithms

Parallel statistical algorithms are foundational to advancing computational workflows in modern, data-intensive applications. Many core methods, such as Monte Carlo simulations, bootstrap resampling, expectation–maximization (EM), and MCMC, naturally lend themselves to parallel execution due to their iterative structure and inherent concurrency (Suchard et al. 2010; Bottou et al. 2018). Distributing these computations across multicore CPUs, GPUs, or hybrid architectures dramatically reduces runtime and unlocks scalability for larger datasets and increasingly complex models. This capability enables faster convergence, richer model exploration, and real-time inference across diverse domains—including Bayesian inference, spatial statistics, biostatistics, and machine learning. Realizing these gains, however, demands not just access to powerful hardware but also algorithmic innovation and careful alignment between statistical fidelity and implementation strategy.

Designing efficient parallel algorithms requires thoughtful orchestration of memory, communication, and computation. Modern approaches often blend shared- and distributed-memory paradigms, optimizing for cache locality, interconnect bandwidth, and memory hierarchy (Rauber and Rünger 2013). Load balancing, whether through static partitioning or dynamic strategies such as work stealing (i.e., idle processors “steal” tasks from the queues of busy processors, improving overall load balance), is essential to ensure high utilization, particularly on

heterogeneous systems (Blumofe and Leiserson 1999). As systems grow more complex, adaptive scheduling and intelligent resource allocation become increasingly important. Hardware architecture significantly shapes performance. Optimizing for multicore CPUs, GPUs, and emerging accelerators entails understanding their respective execution models, memory layouts, and compute hierarchies (Abdelfattah et al. 2016; Ashari et al. 2014). Heterogeneous systems exacerbate these challenges, necessitating hybrid execution strategies that dynamically coordinate across different device types and workloads.

A robust and evolving software ecosystem supports parallel statistical computing. Low-level models such as MPI, OpenMP, and CUDA/OpenCL provide fine-grained control over distributed, shared, and GPU-based execution, while high-level frameworks such as Dask, future, and parallel extensions of R and Python help abstract complexity (Rocklin 2015). However, reproducibility and numerical stability remain nontrivial. Parallel execution introduces sensitivity to floating-point accumulation, rounding errors, and nondeterministic execution orders (Higham 2002).

Achieving peak performance requires minimizing data movement and maximizing computational throughput. The Roofline model (Williams et al. 2009) offers a guiding framework for this trade-off, while strong and weak scaling analyses help evaluate algorithmic efficiency across architectures and problem sizes. Recent advances explore communication-avoiding algorithms (Xu et al. 2024; Katagiri et al. 2024), adaptive parallelism responsive to resource availability, and domain-specific abstractions that simplify implementation without impacting performance. As HPC systems continue to scale, resilience becomes a key concern. Techniques such as checkpoint-restart and algorithm-based fault tolerance are increasingly becoming standard (Shohdy et al. 2016). Ultimately, the evolution of parallel statistical algorithms lies at the intersection of hardware innovation, software engineering, and statistical theory. As problem sizes grow and models become increasingly complex, scalable parallelism will remain a defining pillar of HPSC.

5.2 | Parallel Dense and Sparse Linear Algebra Libraries

Parallel dense and sparse linear algebra libraries are essential enablers of scalable statistical computing in the HPC landscape. Many core statistical algorithms, ranging from regression and principal component analysis to high-dimensional model fitting, rely heavily on matrix operations. These libraries provide highly optimized, parallel implementations of operations such as matrix factorizations (LU, QR, Cholesky) and eigenvalue decompositions, tailored for multicore and manycore architectures. For shared-memory systems, libraries such as *OpenBLAS* (OpenBLAS Project Contributors 2025), *Intel MKL* (Intel Corporation 2025), *AMD BLIS* (AMD 2025), and *NVIDIA cuBLAS* (NVIDIA Corporation 2025) harness thread-level parallelism and vectorized instructions. For distributed and GPU-accelerated environments, tools like *ScalAPACK* (ScalAPACK Project 2025), *Elemental* (Elemental Project Contributors 2025), *DPLASMA* (Innovative Computing Laboratory, University of Tennessee 2025a), and *MAGMA* (Innovative Computing Laboratory, University of Tennessee 2025b) enable large-scale

matrix computations by distributing workloads across nodes or GPU devices. These libraries form the computational backbone of many statistical workflows, delivering substantial gains in speed, memory efficiency, and scalability.

Sparse linear algebra libraries are equally vital, particularly for applications involving high-dimensional, yet sparse, data structures. These tools are designed to minimize memory use and accelerate computation by exploiting sparsity patterns in matrices. Libraries such as *SuiteSparse* (Davis 2019) offer efficient routines for sparse matrix factorization and graph-based computations. *Intel MKL* includes optimized sparse BLAS and solvers for Intel hardware, while *cuSPARSE* delivers high-performance sparse operations on NVIDIA GPUs. *PETSc* (Portable, Extensible Toolkit for Scientific Computation) provides scalable sparse solvers, preconditioners, and optimization routines, and is widely adopted in both scientific computing and machine learning applications (Balay et al. 2019). Likewise, *Trilinos* (Heroux et al. 2005) offers a flexible framework for building scalable, domain-specific solvers in HPC environments. Together, these libraries support statistical methods such as sparse regression, large-scale generalized linear models, and spatial modeling—delivering high performance while maintaining numerical stability and accuracy.

A growing body of work demonstrates how these libraries are being integrated into statistical software ecosystems. For example, Yeginath et al. (2005) introduced *RScalAPACK*, which connects the R environment with ScaLAPACK to enable distributed linear algebra without requiring users to manage parallelism explicitly. Implemented in C and Fortran with MPI, it enables scalable matrix operations such as eigenvalue decompositions and SVD while preserving R's familiar syntax. *RScalAPACK* has been adopted in studies such as Elgamal and Hefeeda (2015) for large-scale statistical analysis and SVD computation. Another important contribution is the *bigGP* package (Paciorek et al. 2015), which introduces scalable GP modeling to R by leveraging ScaLAPACK, BLAS, and MPI. By distributing Cholesky decompositions and covariance matrix operations across nodes, *bigGP* enables efficient kriging, likelihood optimization, and high-dimensional GP regression that would otherwise be computationally prohibitive.

Sparse matrix libraries have also been successfully integrated into statistical applications. For instance, Castrillon-Candás et al. (2016) present *SuiteSparse* to accelerate large-scale sparse regression and covariance tapering methods, reducing runtime while preserving analytical fidelity. In biostatistics, Raim et al. (2013) employed the Portable, Extensible Toolkit for Scientific Computation (*PETSc*) (Balay et al. 2019) and its toolkit for advanced optimization (*TAO*) (Benson et al. 2003) to parallelize maximum likelihood estimation for the random-clumped multinomial model. By distributing the optimization workload across an HPC cluster, their method achieved scalable, efficient inference on large datasets.

Altogether, parallel dense and sparse linear algebra libraries represent an indispensable infrastructure for HPSC. They provide the foundation for building scalable algorithms that can fully exploit modern hardware, bridging the gap between statistical sophistication and computational efficiency.

5.3 | HPC Tools for Big Data and Streaming Statistical Analysis

Statistical applications increasingly operate at data scales that strain or exceed the limits of traditional computing. Domains such as spatial statistics have seen explosive growth in data volume due to advances in satellite sensing, remote imaging, and IoT technologies (Brunsdon and Comber 2020). Similar trends are evident in biostatistics, where genomics, medical imaging, and electronic health records (EHR) systems now generate petabyte-scale datasets (Li 2024). The high volume and velocity of modern datasets necessitate high-performance solutions. Conventional systems, constrained by memory and compute limits, often force practitioners to rely on approximations, sampling, or data reduction techniques. While these strategies are helpful, they can also limit the accuracy. By contrast, HPC offers a path toward exact, scalable analysis, delivering both speed and accuracy for high-dimensional problems. Traditional models, designed for small, structured data, struggle with the complexity and scale of these sources. For instance, genomic data often features millions of variables (e.g., gene expressions, mutations) across relatively few samples, creating challenging high-dimensional, low-sample-size (HDLSS) regimes (He et al. 2017). HPC enables practitioners to bypass many of these limitations, allowing scalable inference, full-resolution modeling, and faster iteration cycles across big data landscapes.

Distributed computing frameworks are central to HPC-enabled big data analytics. By distributing data and computation across nodes, these systems provide the necessary memory and throughput to manage massive datasets. However, many statisticians lack training in parallel programming or system-level tuning, creating a barrier to entry. Bridging this gap requires high-level abstractions that expose HPC performance without demanding low-level expertise. Several frameworks have emerged to meet this need. Apache Spark offers a widely adopted, in-memory data processing engine for large-scale analytics, with native support for machine learning and graph computation (Salloum et al. 2016). Dask brings parallelism to Python workflows, scaling familiar libraries such as NumPy, pandas, and scikit-learn to work in distributed settings (Rocklin 2015). Ray provides a flexible, high-performance execution framework built for parallel, AI-centric workloads, and has found growing use in statistical modeling (Moritz et al. 2018).

Despite their potential, the adoption of these tools within the statistical community remains limited, largely due to a lack of integration with traditional statistical environments and workflows. To close this gap, domain-specific frameworks have emerged that combine statistical rigor with HPC efficiency. For example, *XGBoost* (Chen and Guestrin 2016) and *LightGBM* (Ke et al. 2017) offer distributed, high-speed implementations of gradient-boosted trees, optimized for performance on high-dimensional datasets. These tools are increasingly used in applied statistics, epidemiology, and econometrics, providing scalable alternatives to conventional regression and classification models. As statistical workflows evolve, the continued development and adoption of HPC-compatible tools will be essential to fully leverage distributed computing resources in big data settings.

HPC also opens the door to real-time statistical analysis—an emerging priority in domains where decisions must be made with minimal latency. Traditional batch-processing methods are too slow for applications such as patient monitoring, financial risk management, or emergency response. Stream processing frameworks, coupled with HPC backends, enable rapid model updates, adaptive inference, and low-latency decision-making. Biostatistics presents further applications: continuous patient monitoring can detect subtle physiological changes by analyzing high-volume sensor data from ICU devices (Etli et al. 2024), while real-time surveillance models track infectious disease outbreaks using hospital records, mobile apps, and public health databases (Desai et al. 2019).

Together, these developments underscore the growing importance of HPC in both batch and streaming statistical analytics. By enabling large-scale, low-latency computation, HPC is reshaping the boundaries of what is computationally feasible, expanding the reach, speed, and impact of modern statistical science.

5.4 | Energy-Efficient Statistical Computing

As HPC systems scale in processor count, memory capacity, and accelerator usage, energy consumption has become an increasingly pressing concern, both environmentally and economically. Large-scale statistical applications, including simulation-based inference, bootstrapping, and iterative optimization, often incur high computational costs that translate directly into significant energy usage. Designing energy-efficient statistical workflows is therefore critical, particularly as sustainability becomes a central consideration in scientific computing. One promising approach is mixed-precision computation, where select operations are executed in lower-precision formats (e.g., FP16 or bfloat16), reducing power consumption without compromising statistical accuracy. This strategy is especially effective on GPUs and AI accelerators optimized for low-precision arithmetic (Dongarra et al. 2024). Mixed-precision methods have proven successful in accelerating iterative algorithms such as MLE, achieving faster convergence and reduced energy footprints (Cao et al. 2023).

Beyond precision tuning, algorithmic approximation techniques offer further gains in energy efficiency. Randomized numerical linear algebra methods, such as sketching, random projections, and low-rank matrix approximations, can significantly reduce computational load while maintaining acceptable accuracy (Baumann et al. 2024; Abdulah et al. 2018b). Likewise, methods like approximate Bayesian computation (ABC) (Csilléry et al. 2010) and variational inference serve as scalable alternatives to traditional MCMC, enabling faster inference under limited resource budgets. These approximations are especially valuable in high-dimensional or streaming settings, where evaluating the whole model is often infeasible. As discussed throughout this review, such strategies are already making statistical computing more tractable across domains ranging from spatial modeling to genomics.

Hardware-aware optimization further amplifies these benefits. Adapting statistical algorithms to energy-efficient architectures, such as ARM processors (Ou et al. 2012) or vectorized instruction sets (Jakobs et al. 2016), can yield substantial savings without

sacrificing performance. Minimizing data movement is another key priority, as communication across memory hierarchies or networked systems often accounts for a significant portion of energy costs. Techniques such as communication-avoiding algorithms and asynchronous execution can mitigate these bottlenecks, enabling more sustainable computation at scale.

As demand grows for large-scale statistical analysis in fields like epidemiology, climate science, and biomedicine, incorporating energy efficiency into both algorithm and system design is no longer optional. It is a necessary step toward building a statistical computing infrastructure that is not only fast and accurate but also environmentally and economically sustainable.

6 | Future Directions

As scientific, engineering, and industrial applications generate ever more complex and massive datasets, the statistical community faces both pressing challenges and unprecedented opportunities: to scale existing methods, design new algorithms, and reimagine statistical inference at the extremes of scale. Meeting these demands will require more than technical progress alone; it calls for deep, sustained collaboration between statisticians, computer scientists, and HPC practitioners. It also demands the development of new tools, standards, and methodologies that bridge disciplinary divides and make HPC accessible, robust, and reproducible for statistical applications. This section highlights several new directions shaping the future of HPSC, including the integration of specialized hardware, the rise of federated statistical computing, the support for standardization and interoperability, and the creation of statistical methods for modern high-performance environments. While many opportunities exist for advancing HPSC, it is helpful to distinguish between directions that are achievable in the near term with relatively low risk and those that are longer term and more speculative, involving higher uncertainty but potentially transformative impact.

6.1 | Statistical Computing on Specialized Hardware (GPU, TPU, IPU, DPUs)

Most existing R libraries and statistical software are optimized for CPU execution and offer limited GPU acceleration, typically via low-level C/C++ bindings. As datasets rapidly grow and models become more complex, this CPU-centric approach is increasingly insufficient. Tasks such as large-scale simulation, Bayesian inference, and high-dimensional modeling often exceed the capabilities of a single-node CPU, resulting in significant performance bottlenecks. Modern accelerator hardware, initially designed for machine learning, offers massive parallelism and high memory bandwidth that can dramatically speed up statistical workloads. GPUs, in particular, enable large-scale matrix operations and sampling routines to be executed orders of magnitude faster. By employing low-precision formats such as FP16 or BF16 where acceptable, and reserving FP32 or FP64 for numerically sensitive operations, modern platforms can cut memory usage, boost throughput, and reduce energy consumption (Ltaief et al. 2023; Cao et al. 2023). However, statistical computing

often requires high numerical accuracy, especially for tasks such as variance estimation, likelihood evaluation, and posterior sampling. Thus, future work must focus on adaptive mixed-precision strategies that dynamically adjust precision to preserve stability and avoid bias in intermediate computations (Abdullah, Baker, et al. 2024).

The shift toward specialized hardware also opens the door to domain-specific accelerators. GPU tensor cores (Markidis et al. 2018), Intelligence Processing Units (IPUs, Louw and McIntosh-Smith 2021), and Data Processing Units (DPUs, Barsellotti et al. 2022) offer compelling advantages for different classes of statistical computation. IPUs are well-suited for fine-grained parallelism in matrix-intensive tasks such as GPs and hierarchical Bayesian models. DPUs, on the other hand, can offload data movement and preprocessing operations, such as shuffling, sorting, and filtering, freeing up the main processor for model training or inference. Looking further ahead, quantum computing presents a radically different paradigm. While still in its infancy, quantum accelerators have the potential to offer exponential speedups for certain statistical tasks (Yazdi 2024). Early exploration into quantum algorithms, such as quantum Monte Carlo methods, variational quantum eigensolvers (Tilly et al. 2022), and quantum-enhanced optimization, could unlock new approaches to Bayesian inference, MCMC, and high-dimensional clustering. Major challenges remain, including hardware instability, high error rates, and the need for hybrid quantum–classical workflows. However, investing in quantum circuit design, error correction techniques, and algorithmic robustness could now position the statistical computing community to fully leverage the benefits as quantum technologies mature. This is an achievable future direction for statisticians that requires understanding modern hardware architectures and optimizing existing statistical algorithms to fit these architectures well.

6.2 | Federated Statistical Computing and Privacy-Preserving Inference

Initially developed in the machine learning community (McMahan et al. 2017), federated learning enables decentralized model training across multiple data sources without requiring raw data to be centrally aggregated. Building on this foundation, the emerging paradigm of Federated Statistical Computing (FSC) extends these principles to traditional statistical modeling. FSC provides a framework for performing inference across distributed datasets, including health records, financial transactions, and climate measurements, while respecting privacy constraints and regulatory boundaries (Jordan et al. 2019). Instead of pooling raw data, federated statistical workflows compute local sufficient statistics, parameter estimates, or posterior approximations at each institution and securely aggregate them to form a global model. This preserves privacy while enabling reproducible, interpretable, and statistically valid inference. Active areas of research include the integration of privacy-preserving technologies such as differential privacy (Dwork and Lei 2009) and homomorphic encryption (Lu et al. 2016), which provide strong formal guarantees for secure computation. Recent advances in federated Bayesian modeling (Zhang et al. 2022) and federated GPs (Yue and Kontar 2024) demonstrate how complex probabilistic models can be trained without exposing sensitive data or intermediate computations.

However, FSC introduces challenges distinct from those in federated learning. Statistical inference demands accurate uncertainty quantification, posterior consistency, and asymptotic guarantees—all of which must be revisited in settings characterized by communication bottlenecks, heterogeneous data distributions, and asynchronous updates across institutions. Novel algorithms, such as federated stochastic gradient Langevin dynamics (El Mekkaoui et al. 2021) and hierarchical model partitioning (Marion et al. 2018), aim to minimize communication overhead while maintaining statistical integrity. Applications of FSC are rapidly expanding into domains such as survival analysis, spatial modeling, and causal inference—especially in contexts where collaborative analysis is essential but centralized data collection is infeasible, as seen in healthcare, finance, and environmental science. As FSC evolves, it is poised to become a cornerstone of scalable, privacy-aware statistical computing. Future directions include developing federated analogs to classical statistical procedures, formalizing uncertainty-aware aggregation protocols, and integrating FSC into HPC workflows. In doing so, FSC closes a crucial gap between statistical methodology and the decentralized, privacy-constrained reality of modern data science. Figure 3 illustrates a typical FSC architecture, where multiple institutions compute and securely transmit local summaries, such as histograms, regression coefficients, or scatter plots, to a central server for privacy-preserving model integration. FSC is an achievable, low-risk direction. Recent studies are tackling privacy challenges, and the mature ML literature on federated learning offers clear guidelines for researchers.

6.3 | Standardization and Interoperability

As HPSC continues to expand, the need for standardization and interoperability has become increasingly critical. The

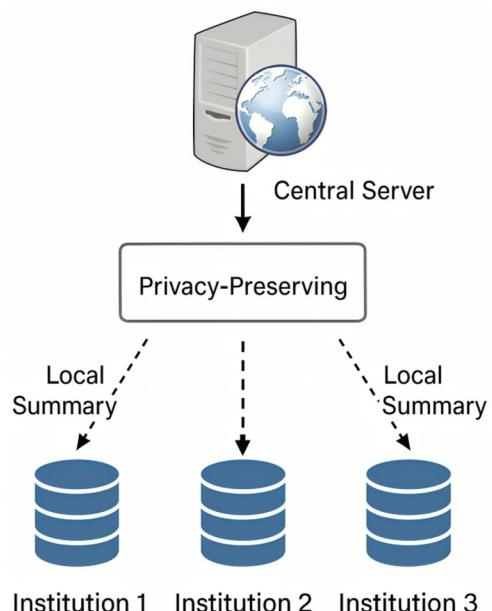


FIGURE 3 | Federated statistical computing: Institutions share local statistical summaries with a central server to enable privacy-preserving model aggregation.

current statistical computing ecosystem remains highly fragmented, composed of specialized libraries, programming languages, and toolchains optimized for narrow domains, specific hardware architectures, or bespoke workflows. This heterogeneity creates substantial barriers to reproducibility, portability, and collaborative development. Looking forward, one of the most pressing priorities is the development of standardized interfaces for distributed statistical computing—interfaces that enable algorithms to operate seamlessly across platforms, from local machines to large-scale HPC clusters. Such standards would define consistent protocols for data exchange, task coordination, and metadata management, allowing statistical workflows to scale without the need for extensive code rewrites. Advances in distributed computing frameworks—from MPI-based systems to modern, task-parallel runtimes—are already laying the foundation for this interoperability.

Equally important is the development of portable statistical software that performs reliably across a range of HPC architectures, including multicore CPUs, GPUs, and emerging accelerators, each with its own programming model and performance characteristics. High-level abstraction layers, such as Kokkos and SYCL, allow developers to write performance-portable code. Meanwhile, containerization tools such as Docker and Singularity help encapsulate statistical environments, ensuring reproducibility and simplifying deployment across diverse systems. Integrating these technologies into mainstream statistical tools is crucial for their widespread adoption. For example, while R remains a cornerstone of statistical practice due to its accessible syntax and rich package ecosystem, it lacks native support for distributed or GPU computing. Bridging this gap often requires wrapping low-level HPC libraries using tools such as Rcpp or exposing high-performance backends through user-friendly interfaces. Similarly, efforts to integrate machine learning frameworks, such as TensorFlow, with R and Python demonstrate how statistical computing can benefit from shared computational infrastructure. These integrations not only accelerate performance but also enable new hybrid workflows that combine traditional statistical modeling with scalable, learning-based methods. Standardization and interoperability are thus not peripheral concerns—they are foundational to making HPSC broadly usable, maintainable, and impactful in real-world applications.

In the near term, low-risk progress in this direction lies in standardizing distributed interfaces and adopting performance-portable layers and containers. Longer-term, higher-uncertainty goals include a cross-language intermediate representation for statistical kernels, automated cross-platform autotuning, and end-to-end standards across HPC–cloud–edge and federated settings, all of which demand substantial community coordination and new tooling.

6.4 | Novel Statistical Methods

The future of HPSC is not only about retrofitting existing algorithms to run on faster hardware; it increasingly demands the creation of new statistical methods explicitly designed for parallel, distributed, and heterogeneous computing environments.

Many classical techniques were built for sequential execution, with tight interdependencies between computational steps. As data volumes grow exponentially and hardware becomes more parallel, these assumptions no longer hold. This shift presents both a challenge and an opportunity: to rethink core statistical approaches from the ground up for modern computing architectures.

One promising direction involves methods that are inherently parallel, rather than sequential algorithms awkwardly adapted to parallel settings. In scalable Bayesian inference, for instance, techniques like Consensus Monte Carlo (Scott 2017) compute independent posterior samples across data subsets and combine them post hoc, enabling efficient inference across distributed datasets. Similarly, divide-and-conquer strategies for likelihood estimation and optimization (Zhang et al. 2013) partition computations into smaller, parallelizable subtasks, yielding dramatic runtime reductions for large-scale problems.

Approximation methods also play an important role. Methods such as variational inference, randomized sketching, and low-rank matrix approximations help to strike a balance between computational efficiency and statistical accuracy. Algorithms such as randomized singular value decomposition (Halko et al. 2011) and sketching-based regression (Woodruff 2014) make it feasible to analyze massive datasets within memory and time constraints. Although approximations can introduce bias or affect uncertainty quantification, they are often essential to enable inference at scale. The growing convergence of statistical modeling and machine learning within HPC contexts is also driving hybrid methodologies that fuse the interpretability of statistical inference with the representational power of deep learning. Examples include deep Bayesian neural networks, probabilistic graphical models on GPUs, and statistical layers embedded within neural architectures—tools increasingly deployed in scientific domains where uncertainty and explainability are paramount.

At scale, communication costs become a significant bottleneck. Efficient statistical computing requires novel methods to manage this overhead, particularly for iterative or gradient-based algorithms. Communication-avoiding techniques, asynchronous updates, and decentralized learning strategies are actively being developed to mitigate these constraints (Ballard et al. 2014). In tandem, hierarchical modeling frameworks that map naturally onto HPC hardware hierarchies—nodes, sockets, cores—can minimize costly inter-node data movement.

Emerging hardware platforms, such as GPUs, TPUs, and IPUs, offer new opportunities for designing statistical algorithms. Tools such as *ExaGeoStat* (Abdullah et al. 2018a) and *pbdR* (Ostroumov et al. 2020) show how high-performance linear algebra and GPU acceleration can power scalable spatial and multivariate inference. Future research must continue to develop statistical kernels optimized for tensor cores, mixed-precision arithmetic, and hardware-specific instructions to ensure performance and accuracy on next-generation systems.

Finally, debugging and validation of such methods become more complex as they scale. Ensuring correctness, numerical stability, and reproducibility across distributed architectures remains

a nontrivial task. Building robust diagnostic tools, test suites, and reproducible benchmarking frameworks will be essential to gaining user trust and accelerating adoption. Continued co-design between statistical methodology and HPC will shape the next generation of tools and theory, keeping statistical science not only relevant but central to the future of large-scale, data-driven discovery.

7 | A Roadmap for the HPSC Community

The convergence of HPC and statistical science holds transformative potential for modern data analysis. However, realizing this potential will require more than technical innovation; it demands the cultivation of a dedicated, interdisciplinary community. Building such a community means bridging cultural and technical divides, fostering sustained collaboration, and training a new generation of researchers with deep expertise in both advanced statistical modeling and HPC technologies. These researchers will play a crucial role in developing novel methods that can handle the scale, complexity, and computational demands of contemporary scientific and industrial data. As data-intensive applications across genomics, climate modeling, remote sensing, finance, and public health continue to outpace the capabilities of traditional statistical workflows, the need for scalable, high-performance statistical inference has never been more urgent. Sophisticated models, such as high-dimensional hierarchical Bayes, and real-time applications, including anomaly detection and epidemiological forecasting, increasingly rely on the computational power and architectural diversity that HPC environments provide.

Encouragingly, progress is already underway. Efforts such as the *pbdR* suite (Hasan et al. 2019), *ExaGeoStat* (Abdullah et al. 2018a), and other high-performance libraries demonstrate the viability of scalable statistical software that harnesses distributed-memory systems and accelerators. Interdisciplinary sessions at conferences such as Supercomputing (SC), Joint Statistical Meetings (JSM), and SIAM are also fostering dialogue between statisticians and computational scientists. However, these efforts remain fragmented—often limited to isolated projects without shared standards, cohesive tooling, or broad adoption. For HPSC to mature into a unified ecosystem, the community must move from isolated success stories to an integrated framework that supports sustainable growth and wide applicability.

HPC offers unparalleled capabilities, including high-throughput computation, access to GPUs, TPUs, and IPUs, and powerful tools for profiling and performance optimization. Statistical science, in turn, offers essential contributions to uncertainty quantification, dimensionality reduction, reproducibility, and interpretability—core elements often lacking in traditional HPC applications. Yet integration is still hindered by several structural barriers. HPC resources are often inaccessible to statisticians due to administrative overhead, institutional gatekeeping, and unfamiliar software stacks. Most statisticians work in interactive, high-level languages such as R or Python, while HPC workflows tend to rely on batch processing, lower-level languages, and rigid scheduling systems. Additionally, a few individuals possess

deep expertise in both domains, creating knowledge silos that slow collaboration and limit methodological innovation. A key step toward overcoming these barriers is the formation of a dedicated community around HPSC. To this end, the recently launched webpage² serves as a central hub for building this community. The platform aims to connect statisticians, data scientists, and HPC scientists by organizing workshops, short courses, collaborative projects, and online discussions. This should provide visibility for emerging tools, encourage mentorship across career stages, and promote interdisciplinary engagement. By establishing a common identity and shared resources, this community-driven effort will accelerate the maturation of HPSC from scattered initiatives into a cohesive field with global impact.

Herein, we also propose a roadmap to address these challenges centered around five pillars: dialogue, education, software, reproducibility, and funding.

- 1. Dialogue:** Establish sustained communication channels between HPC and statistical communities through interdisciplinary panels, workshops, and joint research initiatives. These forums can surface shared challenges, accelerate collaboration, and catalyze co-designed solutions.
- 2. Education:** Develop interdisciplinary training programs that equip researchers with foundational HPC knowledge along with statistical theory. Curriculum modules, summer schools, and online courses can prepare the next generation to work fluently across both domains.
- 3. Software and Standards:** Promote open-source tools and standardized APIs that seamlessly integrate statistical methods into HPC workflows. Create shared repositories of reproducible case studies, scalable datasets, and benchmark metrics to guide adoption and experimentation.
- 4. Reproducibility and Accessibility:** Encourage containerized deployments, high-level interfaces, and performance-portable frameworks that reduce the barrier to entry for statisticians. HPC researchers should develop interactive tools, robust documentation, and reusable modules that align with statistical workflows and prioritize user experience.
- 5. Funding:** Advocate for funding mechanisms that explicitly support interdisciplinary HPSC efforts. Agencies should issue targeted calls that value not only scientific outcomes, but also software infrastructure, community building, and education. Projects at the intersection of statistical methodology and HPC infrastructure must be recognized as central to scientific innovation, not peripheral.

A successful HPSC ecosystem requires shared responsibility. HPC scientists must focus on usability, abstraction, and system scalability, as well as designing software that lowers the barrier to entry while maximizing performance across various architectures. Statisticians should refine their existing and future methods to facilitate parallelism and distributed execution, adopt new paradigms, and promote reproducibility in HPC contexts. Both groups demonstrate a commitment to collaborating to promote diversity, equity, and inclusion in their training, outreach, and hiring practices.

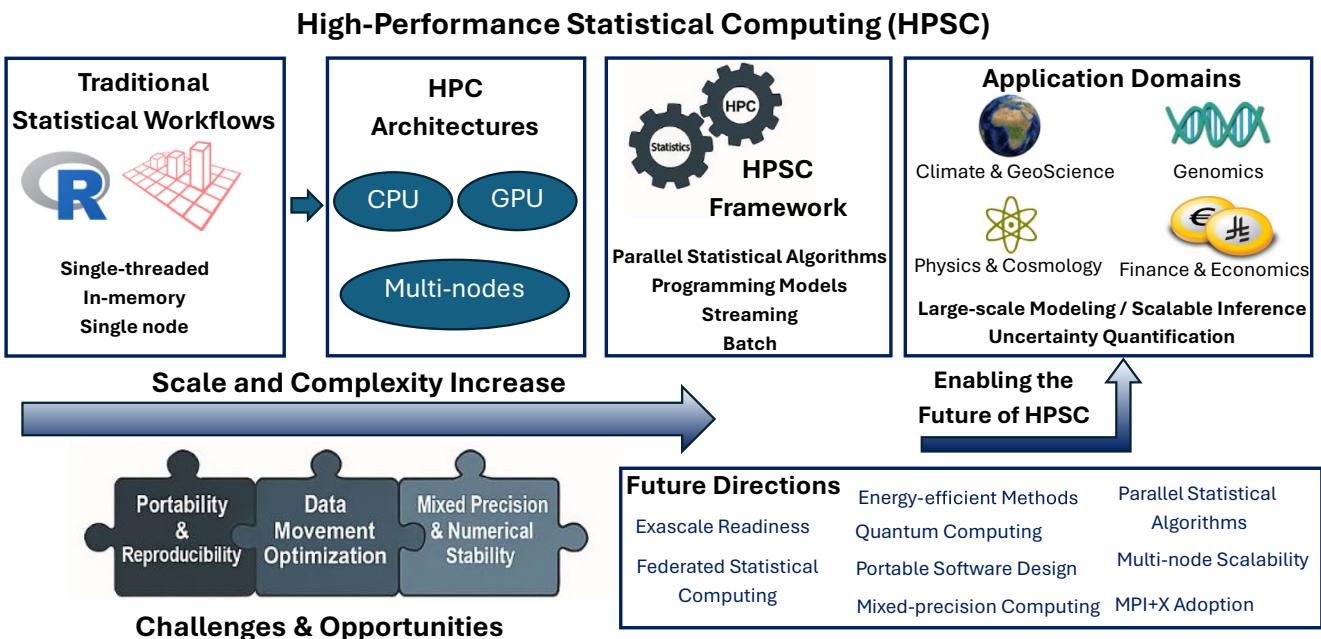


FIGURE 4 | The evolution of High-Performance Statistical Computing (HPSC): Transitioning from traditional, single-threaded R workflows to scalable, multi-node architectures empowered by parallel statistical algorithms, HPC programming models, and emerging technologies such as federated computing, quantum acceleration, and mixed-precision methods.

The future of HPSC lies in this collective effort. As illustrated in Figure 4, the HPSC landscape aims to change traditional single-threaded, in-memory workflows in existing statistical code powered by languages such as R with scalable, multi-node execution models powered by HPC architectures, parallel statistical algorithms, and emerging technologies such as federated computing, mixed-precision methods, and quantum acceleration. These advances support applications ranging from climate science to finance, driven by the growing scale of data and increasing computational complexity. The community can push the boundaries of what is statistically and computationally possible by combining the algorithmic depth of statistics with the computational power of HPC.

Author Contributions

Sameh Abdullah: conceptualization (equal), supervision (equal), writing – original draft (equal), writing – review and editing (equal). **Mary Lai O. Salvaña:** conceptualization (equal), supervision (equal), writing – original draft (equal), writing – review and editing (equal). **Ying Sun:** conceptualization (equal), supervision (equal), writing – original draft (equal), writing – review and editing (equal). **David E. Keyes:** conceptualization (equal), supervision (equal), writing – original draft (equal), writing – review and editing (equal). **David E. Keyes:** conceptualization (equal), supervision (equal), writing – original draft (equal), writing – review and editing (equal). **Marc G. Genton:** conceptualization (equal), supervision (equal), writing – original draft (equal), writing – review and editing (equal).

Funding

This work was supported by King Abdullah University of Science and Technology.

Conflicts of Interest

The authors declare no conflicts of interest.

Data Availability Statement

Data sharing is not applicable to this article as no new data were created or analyzed in this study.

Related WIREs Articles

[Parallel computing with R: A brief review](#)

Endnotes

¹Referred to as machine learning algorithms in the source, reflecting their computer science perspective.

²<https://hpsc4science.org>.

References

- Abadi, M., P. Barham, J. Chen, et al. 2016. “Tensorflow: A System for Large-Scale Machine Learning.” In *12th USENIX symposium on operating systems design and implementation (OSDI 16)*, 265–283.
- Abdelfattah, A., M. Baboulin, V. Dobrev, et al. 2016. “High-Performance Tensor Contractions for Gpus.” *Procedia Computer Science* 80: 108–118.
- Abdul Fattah, E., H. Ltaief, H. Rue, and D. Keyes. 2025. “Stiles: An Accelerated Computational Framework for Sparse Factorizations of Structured Matrices.” In *ISC High Performance 2025 Research Paper Proceedings (40th International Conference)*, 1–14. Prometheus GmbH.
- Abdullah, S., F. Alamri, P. Nag, et al. 2022. “The Second Competition on Spatial Statistics for Large Datasets.” *Journal of Data Science* 20: 439–460.
- Abdullah, S., A. H. Baker, G. Bosilca, et al. 2024. “Boosting Earth System Model Outputs and Saving Petabytes in Their Storage Using Exascale Climate Emulators.” In *SC24: International Conference for High Performance Computing, Networking, Storage and Analysis*, 1–12. IEEE.
- Abdullah, S., Q. Cao, Y. Pei, et al. 2021. “Accelerating Geostatistical Modeling and Prediction With Mixed-Precision Computations: A

- High-Productivity Approach With PaRSEC.” *IEEE Transactions on Parallel and Distributed Systems* 33: 964–976.
- Abdulah, S., J. Ejarque, O. Marzouk, et al. 2024. “Portability and Scalability Evaluation of Large-Scale Statistical Modeling and Prediction Software Through HPC-Ready Containers.” *Future Generation Computer Systems* 161: 248–258.
- Abdulah, S., Y. Li, J. Cao, et al. 2023. “Large-Scale Environmental Data Science With ExaGeoStatR.” *Environmetrics* 34: e2770.
- Abdulah, S., H. Ltaief, Y. Sun, M. G. Genton, and D. E. Keyes. 2018a. “ExaGeoStat: A High Performance Unified Software for Geostatistics on Manycore Systems.” *IEEE Transactions on Parallel and Distributed Systems* 29: 2771–2784.
- Abdulah, S., H. Ltaief, Y. Sun, M. G. Genton, and D. E. Keyes. 2018b. “Parallel Approximation of the Maximum Likelihood Estimation for the Prediction of Large-Scale Geostatistics Simulations.” In *2018 IEEE International Conference on Cluster Computing (CLUSTER)*, 98–108. IEEE.
- Abdulah, S., H. Ltaief, Y. Sun, M. G. Genton, and D. E. Keyes. 2019. “Geostatistical Modeling and Prediction Using Mixed Precision Tile Cholesky Factorization.” In *2019 IEEE 26th International Conference on High Performance Computing, Data, and Analytics (HiPC)*, 152–162. IEEE.
- Aldrich, E. M. 2014. “GPU Computing in Economics.” In *Handbook of Computational Economics*, vol. 3, 557–598. Elsevier.
- Aldrich, E. M., J. Fernández-Villaverde, A. R. Gallant, and J. F. Rubio-Ramírez. 2011. “Tapping the Supercomputer Under Your Desk: Solving Dynamic Equilibrium Models With Graphics Processors.” *Journal of Economic Dynamics & Control* 35: 386–393.
- Allombert, V., D. Michea, F. Dupros, et al. 2014. “An Out-Of-Core GPU Approach for Accelerating Geostatistical Interpolation.” *Procedia Computer Science* 29: 888–896.
- Altman, M., J. Gill, and M. P. McDonald. 2004. *Numerical Issues in Statistical Computing for the Social Scientist*. John Wiley & Sons.
- Amadio, G., A. Ananya, J. Apostolakis, et al. 2016. “Electromagnetic Physics Models for Parallel Computing Architectures.” In *Journal of Physics: Conference Series*, vol. 762, 012014. IOP Publishing.
- AMD. 2025. “AMD BLIS.” Accessed 2025-07-03. <https://github.com/flame/blis>.
- Amsler, C., Y. Y. Chen, P. Schmidt, and H. J. Wang. 2023. “A Hierarchical Panel Data Model for the Estimation of Stochastic Metafrontiers: Computational Issues and an Empirical Application.” In *Advanced Mathematical Methods for Economic Efficiency Analysis: Theory and Empirical Applications*, 183–195. Springer.
- Ashari, A., N. Sedaghati, J. Eisenlohr, S. Parthasarath, and P. Sadayappan. 2014. “Fast Sparse Matrix-Vector Multiplication on Gpus for Graph Applications.” In *SC’14: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, 781–792. IEEE.
- Ayres, D. L., and M. P. Cummings. 2017. “Heterogeneous Hardware Support in BEAGLE, a High-Performance Computing Library for Statistical Phylogenetics.” In *2017 46th International Conference on Parallel Processing Workshops (ICPPW)*, 23–32. IEEE.
- Ayres, D. L., M. P. Cummings, G. Baele, et al. 2019. “BEAGLE 3: Improved Performance, Scaling, and Usability for a High-Performance Computing Library for Statistical Phylogenetics.” *Systematic Biology* 68: 1052–1061.
- Ayres, D. L., A. Darling, D. J. Zwickl, et al. 2012. “BEAGLE: An Application Programming Interface and High-Performance Computing Library for Statistical Phylogenetics.” *Systematic Biology* 61: 170–173.
- Bai, Y., R. V. Craiu, and A. F. Di Narzo. 2011. “Divide and Conquer: A Mixture-Based Approach to Regional Adaptation for MCMC.” *Journal of Computational and Graphical Statistics* 20: 63–79.
- Balay, S., S. Abhyankar, M. Adams, et al. 2019. “PETSc Users Manual.”
- Ballard, G., J. Demmel, O. Holtz, and O. Schwartz. 2014. “Communication-Avoiding Algorithms.” *Acta Numerica* 23: 1–155.
- Bandopadhyay, D., and C. J. Moore. 2024. “GPU-Accelerated Semicoherent Hierarchical Search for Stellar-Mass Binary Inspiral Signals in LISA.” *Physical Review D* 110: 103026.
- Barash, L. Y., M. Weigel, M. Borovský, W. Janke, and L. N. Shchur. 2017. “GPU Accelerated Population Annealing Algorithm.” *Computer Physics Communications* 220: 341–350.
- Barsellotti, L., F. Alhamed, J. J. V. Olmos, F. Paolucci, P. Castoldi, and F. Cugini. 2022. “Introducing Data Processing Units (Dpu) at the Edge.” In *In 2022 International Conference on Computer Communications and Networks (ICCCN)*, 1–6. IEEE.
- Baştürk, N., S. Grassi, L. Hoogerheide, and H. K. Van Dijk. 2016. “Parallelization Experience With Four Canonical Econometric Models Using ParMitISEM.” *Econometrics* 4: 11.
- Baumann, L., L. Einkemmer, C. Klingenberg, and J. Kusch. 2024. “Energy Stable and Conservative Dynamical Low-Rank Approximation for the Su-Olson Problem.” *SIAM Journal on Scientific Computing* 46: B137–B158.
- Baydin, A. G., L. Shao, W. Bhimji, et al. 2019. “Etalumis: Bringing Probabilistic Programming to Scientific Simulators at Scale.” In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, 1–24.
- Bengtsson, H. 2020. “A Unifying Framework for Parallel and Distributed Processing in R Using Futures.” *arXiv Preprint arXiv:2008.00553*.
- Benson, S., L. C. McInnes, J. J. More, and J. Sarich. 2003. *Tao Users Manual*. Technical Report, Argonne National Lab.
- Bernholdt, D. E., S. Boehm, G. Bosilca, et al. 2020. “A Survey of MPI Usage in the US Exascale Computing Project.” *Concurrency and Computation: Practice and Experience* 32: e4851.
- Bezanson, J., A. Edelman, S. Karpinski, and V. B. Shah. 2017. “Julia: A Fresh Approach to Numerical Computing.” *SIAM Review* 59: 65–98.
- Bluhm, B., and J. Cutura. 2020. *Econometrics at Scale: Spark Up Big Data in Economics*. SAFE Working Paper. https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3226976.
- Blumofe, R. D., and C. E. Leiserson. 1999. “Scheduling Multithreaded Computations by Work Stealing.” In *Proceedings 40th Annual Symposium on Foundations of Computer Science (Cat. No. 99CB37039)*, 356–368. IEEE.
- Bottou, L., F. E. Curtis, and J. Nocedal. 2018. “Optimization Methods for Large-Scale Machine Learning.” *SIAM Review* 60: 223–311.
- Boyle, P., M. Chuvelev, G. Cossu, C. Kelly, C. Lehner, and L. Meadows. 2017. “Accelerating HPC Codes on Intel Omni-Path Architecture Networks: From Particle Physics to Machine Learning.” *arXiv Preprint arXiv:1711.04883*.
- Brunsdon, C., and A. Comber. 2020. “Big Issues for Big Data: Challenges for Critical Spatial Data Analytics.” *arXiv Preprint arXiv:2007.11281*.
- Buttari, A., J. Langou, J. Kurzak, and J. Dongarra. 2009. “A Class of Parallel Tiled Linear Algebra Algorithms for Multicore Architectures.” *Parallel Computing* 35: 38–53.
- Campagne, J.-E., F. Lanusse, J. Zuntz, et al. 2023. “Jax-Cosmo: An End-To-End Differentiable and GPU Accelerated Cosmology Library.” *arXiv Preprint arXiv:2302.05163*.

- Canales, R., E. Peise, and P. Bientinesi. 2016. "Large Scale Parallel Computations in r Through Elemental." *arXiv Preprint arXiv:1610.07310*.
- Cao, Q., S. Abdulah, R. Alomairy, et al. 2022. "Reshaping Geostatistical Modeling and Prediction for Extreme-Scale Environmental Applications." In *SC22: International Conference for High Performance Computing, Networking, Storage and Analysis*, 1–12. IEEE.
- Cao, Q., S. Abdulah, H. Ltaief, M. G. Genton, D. Keyes, and G. Bosilca. 2023. "Reducing Data Motion and Energy Consumption of Geospatial Modeling Applications Using Automated Precision Conversion." In *In 2023 IEEE International Conference on Cluster Computing (CLUSTER)*, 330–342. IEEE.
- Casarini, R., R. V. Craiu, and F. Leisen. 2016. "Embarrassingly Parallel Sequential Markov Chain Monte Carlo for Large Sets of Time Series." *Statistics and Its Interface* 9: 497–508.
- Castrillon-Candás, J. E., M. G. Genton, and R. Yokota. 2016. "Multi-Level Restricted Maximum Likelihood Covariance Estimation and Kriging for Large Non-Gridded Spatial Datasets." *Spatial Statistics* 18: 105–124.
- Castruccio, S., R. Huser, and M. G. Genton. 2016. "High-Order Composite Likelihood Inference for Max-Stable Distributions and Processes." *Journal of Computational and Graphical Statistics* 25: 1212–1229.
- Chambers, J. M. 2008. *Software for Data Analysis: Programming With R*. Springer.
- Chandra, R. 2001. *Parallel Programming in OpenMP*. Morgan kaufmann.
- Chang, W.-Y., M.-C. Wu, Y.-L. Chang, S.-Y. Shih, and B. Huang. 2018. "GPU Acceleration of Adaptive Local Kriging Applied to Retrieving Slant-Range Surface Motion Maps." *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* 11: 4317–4325.
- Chen, L., J. Su, A. T. Wan, and Y. Zhou. 2024. "A Simple Divide-And-Conquer-Based Distributed Method for the Accelerated Failure Time Model." *Journal of Computational and Graphical Statistics* 33: 681–698.
- Chen, T., and C. Guestrin. 2016. "Xgboost: A Scalable Tree Boosting System." In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 785–794.
- Chib, S., M. Shin, and F. Tan. 2023. "DSGE-SVt: An Econometric Toolkit for High-Dimensional DSGE Models With SV and t Errors." *Computational Economics* 61: 69–111.
- Chung, D. J., K. Seo, and R. Song. 2023. "Efficient Computation of Discrete Games: Estimating the Effect of Apple on Market Structure." *Production and Operations Management* 32: 2245–2263.
- Creel, M. 2008. "Using Parallelization to Solve a Macroeconomic Model: A Parallel Parameterized Expectations Algorithm." *Computational Economics* 32: 343–352.
- Creel, M., and W. L. Goffe. 2008. "Multi-Core CPUs, Clusters, and Grid Computing: A Tutorial." *Computational Economics* 32: 353–382.
- Creel, M., and M. Zubair. 2012. "High Performance Implementation of an Econometrics and Financial Application on GPUs." In *2012 SC Companion: High Performance Computing, Networking Storage and Analysis*, 1147–1152. IEEE.
- Croci, M., M. Fasi, N. J. Higham, T. Mary, and M. Mikaitis. 2022. "Stochastic Rounding: Implementation, Error Analysis and Applications." *Royal Society Open Science* 9: 211631.
- Csilléry, K., M. G. Blum, O. E. Gaggiotti, and O. François. 2010. "Approximate Bayesian Computation (Abc) in Practice." *Trends in Ecology & Evolution* 25: 410–418.
- Danner, A., A. Breslow, J. Baskin, and D. Wilikofsky. 2012. "Hybrid MPI/GPU Interpolation for grid DEM Construction." In *Proceedings of the 20th International Conference on Advances in Geographic Information Systems*, 299–308.
- Davis, T. A. 2019. "Algorithm 1000: Suitesparse: Graphblas: Graph Algorithms in the Language of Sparse Linear Algebra." *ACM Transactions on Mathematical Software* 45: 1–25.
- De Luca, P., D. Di Luccio, A. Galletti, G. Giunta, L. Marcellino, and R. Montella. 2022. "Towards a GPU Parallel Software for Environmental Data Fitting." In *Proceedings of the 15th International Conference on PErvasive Technologies Related to Assistive Environments*, 469–472.
- De Luca, P., A. Galletti, and L. Marcellino. 2022. "A GPU-Based Algorithm for Environmental Data Filtering." In *International Conference on Computational Science*, 45–52. Springer.
- De Sensi, D., S. Di Girolamo, K. H. McMahon, D. Roweth, and T. Hoeferl. 2020. "An In-Depth Analysis of the Slingshot Interconnect." In *SC20: International Conference for High Performance Computing, Networking, Storage and Analysis*, 1–14. IEEE.
- Delgado, M. S., and C. F. Parmeter. 2013. "Embarrassingly Easy Embarrassingly Parallel Processing in R." *Journal of Applied Econometrics* 28: 1224–1230.
- Deng, M., W. Yang, Q. Liu, and Y. Zhang. 2017. "A Divide-And-Conquer Method for Space-Time Series Prediction." *Journal of Geographical Systems* 19: 1–19.
- Desai, A. N., M. U. Kraemer, S. Bhatia, et al. 2019. "Real-Time Epidemic Forecasting: Challenges and Opportunities." *Health Security* 17: 268–275.
- Dixon, M., S. A. Khan, and M. Zubair. 2014. "Accelerating Option Risk Analytics in R Using GPUs." In *HPC'14: Proceedings of the High Performance Computing Symposium*, 1–7. Elsevier.
- Dixon, M. F., T. Bradley, J. Chong, and K. Keutzer. 2012. "Monte Carlo-Based Financial Market Value-At-Risk Estimation on GPUs." In *GPU Computing Gems Jade Edition*, 337–353. Elsevier.
- Dixon, W. J. 1983. *BMDP Statistical Software*. Vol. 1. Univ of California Press.
- Dong, B., Y.-W. Hsieh, and X. Zhang. 2022. "Implementing Maximum Likelihood Estimation of Empirical Matching Models: MPEC Versus NFXP." *Computational Economics* 59: 1–32.
- Dongarra, J., J. Gunnels, H. Bayraktar, A. Haidar, and D. Ernst. 2024. "Hardware Trends Impacting Floating-Point Computations in Scientific Applications." *arXiv Preprint arXiv:2411.12090*.
- Dongarra, J. J., J. Du Croz, S. Hammarling, and I. S. Duff. 1990. "A Set of Level 3 Basic Linear Algebra Subprograms." *ACM Transactions on Mathematical Software* 16: 1–17.
- Dunn, L., P. Clearwater, A. Melatos, and K. Wette. 2022. "Graphics Processing Unit Implementation of the \mathcal{F} -Statistic for Continuous Gravitational Wave Searches." *Classical and Quantum Gravity* 39: 045003.
- Dwork, C., and J. Lei. 2009. "Differential Privacy and Robust Statistics." In *Proceedings of the Forty-First Annual ACM Symposium on Theory of Computing*, 371–380.
- El Mekkaoui, K., D. Mesquita, P. Blomstedt, and S. Kaski. 2021. "Federated Stochastic Gradient Langevin Dynamics." In *Uncertainty in Artificial Intelligence*, 1703–1712. PMLR.
- Elemental Project Contributors. 2025. "Elemental." Accessed 2025-07-03. <https://github.com/elemental/Elemental>.
- Elgamal, T., and M. Hefeeda. 2015. "Analysis of Pca Algorithms in Distributed Environments." *arXiv Preprint arXiv:1503.05214*.
- Etli, D., A. Djurovic, and J. Lark. 2024. "The Future of Personalized Healthcare: Ai-Driven Wearables for Real-Time Health Monitoring and Predictive Analytics." *Current Research in Health Sciences* 2: 10–14.
- Fernández-Villaverde, J., and D. Zarruk Valencia. 2018. *A Practical Guide to Parallelization in Economics*. NBER Working Paper. <https://ssrn.com/abstract=3170815>.

- Ferrall, C. 2005. "Solving Finite Mixture Models: Efficient Computation in Economics Under Serial and Parallel Execution." *Computational Economics* 25: 343–379.
- Ferraro Petrillo, U., M. Sorella, G. Cattaneo, R. Giancarlo, and S. E. Rombo. 2019. "Analyzing Big Datasets of Genomic Sequences: Fast and Scalable Collection of k-Mer Statistics." *BMC Bioinformatics* 20: 1–14.
- Fluke, C. J., D. G. Barnes, B. R. Barsdell, and A. H. Hassan. 2011. "Astrophysical Supercomputing With GPUs: Critical Decisions for Early Adopters." *Publications of the Astronomical Society of Australia* 28: 15–27.
- Frontiere, N., K. Heitmann, E. Rangel, et al. 2022. "Farpoint: A High-Resolution Cosmology Simulation at the Gigaparsec Scale." *Astrophysical Journal Supplement Series* 259: 15.
- Geng, Z., S. Abdulah, H. Ltaief, Y. Sun, M. G. Genton, and D. E. Keyes. 2023. "GPU-Accelerated Dense Covariance Matrix Generation for Spatial Statistics Applications." *Poster Presented at SuperComputing Research Poster'23, New York, NY, USA*.
- Geng, Z., S. Abdulah, Y. Sun, H. Ltaief, D. E. Keyes, and M. G. Genton. 2025. "GPU-Accelerated Modified Bessel Function of the Second Kind for Gaussian Processes." In *ISC High Performance 2025 Research Paper Proceedings (40th International Conference)*, 1–12. Prometheus GmbH.
- Gerber, F., and D. Nychka. 2021. "Fast Covariance Parameter Estimation of Spatial Gaussian Process Models Using Neural Networks." *Stat* 10: e382.
- Givens, G. H., and J. A. Hoeting. 2012. *Computational Statistics*. John Wiley & Sons.
- Gopinathan, D., M. Heidarzadeh, and S. Guillas. 2021. "Probabilistic Quantification of Tsunami Current Hazard Using Statistical Emulation." *Proceedings of the Royal Society A* 477: 20210180.
- Gramacy, R. B., J. Niemi, and R. M. Weiss. 2014. "Massively parallel approximate Gaussian process regression." *SIAM/ASA Journal on Uncertainty Quantification* 2: 564–584.
- Gropp, W., E. Lusk, and A. Skjellum. 1999. *Using MPI: Portable Parallel Programming With the Message-Passing Interface*. MIT press.
- Grun, P. 2010. *Introduction to Infiniband for End Users*. Vol. 55, 11. White Paper, InfiniBand Trade Association.
- Guhaniyogi, R., C. Li, T. D. Savitsky, and S. Srivastava. 2017. "A Divide-And-Conquer Bayesian Approach to Large-Scale Kriging." *arXiv Preprint arXiv:1712.09767*.
- Günther, S., L. Balkenhol, C. Fidler, et al. 2025. "OLÉ - Online Learning Emulation in Cosmology." *Journal of Cosmology and Astroparticle Physics* 2025: 059.
- Guo, J., J. Peng, H. Yuan, and L. M.-s. Ni. 2023. "HXPY: A High-Performance Data Processing Package for Financial Time-Series Data." *Journal of Computer Science and Technology* 38: 3–24.
- Gustavson, F. G. 2001. "New Generalized Matrix Data Structures Lead to a Variety of High-Performance Algorithms." In *The Architecture of Scientific Software: IFIP TC2/WG2. 5 Working Conference on the Architecture of Scientific Software October 2–4, 2000, Ottawa, Canada*, 211–234. Springer.
- Hadjidoukas, P. E., P. Angelopoulos, L. Kulakova, C. Papadimitriou, and P. Koumoutsakos. 2015. "Exploiting Task-Based Parallelism in Bayesian Uncertainty Quantification." In *Euro-Par 2015: Parallel Processing: 21st International Conference on Parallel and Distributed Computing, Vienna, Austria, August 24–28, 2015, Proceedings* 21, 532–544. Springer.
- Hadjidoukas, P. E., P. Angelopoulos, C. Papadimitriou, and P. Koumoutsakos. 2015. "Π4U: A High Performance Computing Framework for Bayesian Uncertainty Quantification of Complex Models." *Journal of Computational Physics* 284: 1–21.
- Halko, N., P.-G. Martinsson, and J. A. Tropp. 2011. "Finding Structure With Randomness: Probabilistic Algorithms for Constructing Approximate Matrix Decompositions." *SIAM Review* 53: 217–288.
- Hariadi, M., A. A. Muhammad, and S. M. S. Nugroho. 2020. "Prediction of Stock Prices Using Markov Chain Monte Carlo." In *2020 International Conference on Computer Engineering, Network, and Intelligent Multimedia (CENIM)*, 385–390. IEEE.
- Hasan, S. S., D. Schmidt, R. Kannan, and N. Imam. 2019. "A Scalable Graph Analytics Framework for Programming With Big Data in r (Pbdr)." In *2019 IEEE International Conference on Big Data (Big Data)*, 4783–4792. IEEE.
- Hayfield, T., and J. S. Racine. 2008. "Nonparametric Econometrics: The Np Package." *Journal of Statistical Software* 27: 1–32.
- He, K. Y., D. Ge, and M. M. He. 2017. "Big Data Analytics for Genomic Medicine." *International Journal of Molecular Sciences* 18: 412.
- Heitzler, M., J. C. Lam, J. Hackl, B. T. Adey, and L. Hurni. 2017. "GPU-Accelerated Rendering Methods to Visually Analyze Large-Scale Disaster Simulation Data." *Journal of Geovisualization and Spatial Analysis* 1: 1–18.
- Hentosh, L., Y. I. Tsikalos, N. Kustra, and H. Kutucu. 2022. "ML-Based Approach for Credit Risk Assessment Using Parallel Calculations." In *CITRisk*, 161–173.
- Herbst, E., and F. Schorfheide. 2014. "Sequential Monte Carlo Sampling for DSGE Models." *Journal of Applied Econometrics* 29: 1073–1098.
- Heroux, M. A., R. A. Bartlett, V. E. Howle, et al. 2005. "An Overview of the Trilinos Project." *ACM Transactions on Mathematical Software* 31: 397–423.
- Higham, N. J. 2002. *Accuracy and Stability of Numerical Algorithms*. SIAM.
- Ho, M., D. J. Bartlett, N. Chartier, et al. 2024. "LTU-ILI: An All-In-One Framework for Implicit Inference in Astrophysics and Cosmology." *arXiv Preprint arXiv:2402.05137*.
- Holbrook, A. J., X. Ji, and M. A. Suchard. 2022. "Bayesian Mitigation of Spatial Coarsening for a Hawkes Model Applied to Gunfire, Wildfire and Viral Contagion." *Annals of Applied Statistics* 16: 573–595.
- Holbrook, A. J., C. E. Loeffler, S. R. Flaxman, and M. A. Suchard. 2021. "Scalable Bayesian Inference for Self-Excitatory Stochastic Processes Applied to Big American Gunfire Data." *Statistics and Computing* 31: 4.
- Homrighausen, D., and D. J. McDonald. 2016. "On the nyström and Column-Sampling Methods for the Approximate Principal Components Analysis of Large Datasets." *Journal of Computational and Graphical Statistics* 25: 344–362.
- Hong, C., Y. Wang, and T. Cai. 2022. "A Divide-And-Conquer Method for Sparse Risk Prediction and Evaluation." *Biostatistics* 23: 397–411.
- Hong, Y., S. Abdulah, M. G. Genton, and Y. Sun. 2021. "Efficiency Assessment of Approximated Spatial Predictions for Large Datasets." *Spatial Statistics* 43: 100517.
- Hong, Y., Y. Song, S. Abdulah, et al. 2023. "The Third Competition on Spatial Statistics for Large Datasets." *Journal of Agricultural, Biological and Environmental Statistics* 28: 618–635.
- Hori, A., E. Jeannot, G. Bosilca, et al. 2021. "An International Survey on MPI Users." *Parallel Computing* 108: 102853.
- Hornung, R. D., and J. A. Keasler. 2014. "The RAJA Portability Layer: Overview and Status."
- Hu, H., H. Shu, Z. Hu, and J. Xu. 2016. "Using Compute Unified Device Architecture-Enabled Graphic Processing Unit to Accelerate Fast Fourier Transform-Based Regression Kriging Interpolation on a MODIS Land Surface Temperature Image." *Journal of Applied Remote Sensing* 10: 26036.

- Hu, Y., R. Harris, R. Timmerman, and B. Lu. 2024. "A Backfitting Maximum Likelihood Estimator for Hierarchical and Geographically Weighted Regression Modelling, With a Case Study of House Prices in Beijing." *International Journal of Geographical Information Science* 38: 2458–2491.
- Huang, F., S. Bu, J. Tao, and X. Tan. 2016. "OpenCL Implementation of a Parallel Universal Kriging Algorithm for Massive Spatial Data Interpolation on Heterogeneous Systems." *ISPRS International Journal of Geo-Information* 5: 96.
- Huang, H., S. Abdulah, Y. Sun, H. Ltaief, D. E. Keyes, and M. G. Genton. 2021. "Competition on Spatial Statistics for Large Datasets." *Journal of Agricultural, Biological and Environmental Statistics* 26: 580–595.
- Ihaka, R., and R. Gentleman. 1996. "R: A Language for Data Analysis and Graphics." *Journal of Computational and Graphical Statistics* 5: 299–314.
- Ings, G., D. Thomas, and W. Luk. 2013. "A Heterogeneous Computing Framework for Computational Finance." In *2013 42nd International Conference on Parallel Processing*, 688–697. IEEE.
- Innovative Computing Laboratory, University of Tennessee. 2025a. "Dplasma." Accessed 2025a-07-03. <https://icl.utk.edu/dplasma/>.
- Innovative Computing Laboratory, University of Tennessee. 2025b. "MAGMA: Matrix Algebra on GPU and Multicore Architectures." Accessed 2025b-07-03. <https://icl.utk.edu/magma/>.
- Intel Corporation. 2025. "Intel Math Kernel Library (MKL)." Accessed 2025-07-03. <https://www.intel.com/content/www/us/en/developer/tools/oneapi/onemkl.html>.
- Jain, A., M. Bakshi, A. Kalele, and E. Subramanian. 2015. "On Accelerating Concurrent PCA Computations for Financial Risk Applications." In *2015 IEEE 22nd International Conference on High Performance Computing (HiPC)*, 175–184. IEEE.
- Jakobs, T., M. Hofmann, and G. Rünger. 2016. "Reducing the Power Consumption of Matrix Multiplications by Vectorization." In *2016 IEEE Intl Conference on Computational Science and Engineering (CSE) and IEEE Intl Conference on Embedded and Ubiquitous Computing (EUC) and 15th Intl Symposium on Distributed Computing and Applications for Business Engineering (DCABES)*, 213–220. IEEE.
- James, Z., and J. Guinness. 2024. "Implementation and Analysis of GPU Algorithms for Vecchia Approximation." *Statistics and Computing* 34: 207.
- Jan, A., R. O'Shaughnessy, D. Shoemaker, and J. Lange. 2025. "Adapting a Novel Framework for Rapid Inference of Massive Black Hole Binaries for LISA." *Physical Review D* 111: 064079.
- Jarp, S., A. Lazzaro, J. Leduc, A. Nowak, and Y. S. Lindal. 2012. "Evaluation of Likelihood Functions on CPU and GPU Devices." *Journal of Physics: Conference Series* 368: 012023.
- Jiang, J. 2024. "MPH: Fast REML for Large-Scale Genome Partitioning of Quantitative Genetic Variation." *Bioinformatics* 40: btae298.
- Jordan, M. I., J. D. Lee, and Y. Yang. 2019. "Communication-Efficient Distributed Statistical Inference." *Journal of the American Statistical Association* 114: 668–681.
- Jouppi, N. P., C. Young, N. Patil, et al. 2017. "In-Datacenter Performance Analysis of a Tensor Processing Unit." *Proceedings of the 44th Annual International Symposium on Computer Architecture*, 1–12.
- Kamburugamuve, S., P. Wickramasinghe, S. Ekanayake, and G. C. Fox. 2018. "Anatomy of Machine Learning Algorithm Implementations in MPI, Spark, and Flink." *International Journal of High Performance Computing Applications* 32: 61–73.
- Katagiri, T., J. Iwata, and K. Uchida. 2024. "A Communication Avoiding and Reducing Algorithm for Symmetric Eigenproblem for Very Small Matrices." *arXiv Preprint arXiv:2405.00326*.
- Katzfuss, M., and D. Hammerling. 2017. "Parallel Inference for Massive Distributed Spatial Data Using Low-Rank Models." *Statistics and Computing* 27: 363–375.
- Ke, G., Q. Meng, T. Finley, et al. 2017. "Lightgbm: A Highly Efficient Gradient Boosting Decision Tree." *Advances in Neural Information Processing Systems* 30: 52.
- Kennedy, W. J., and J. E. Gentle. 2021. *Statistical Computing*. Routledge.
- Khanthaporn, R., and N. Wichitaksorn. 2023. "Modelling and Forecasting COVID-19 Stock Returns Using Asymmetric GARCH-ICAPM With Mixture and Heavy-Tailed Distributions." *Applied Economics* 55: 6042–6061.
- Kim, S. 2016. "Parameter Estimation Using Divide-And-Conquer Methods for Differential Equation Models." *Journal of Biometrics & Biostatistics* 7, no. 3: 305.
- Ko, S., G. X. Li, H. Choi, and J.-H. Won. 2021. "Computationally Scalable Regression Modeling for Ultrahigh-Dimensional Omics Data With ParProx." *Briefings in Bioinformatics* 22: bbab256.
- Ko, S., H. Zhou, J. Zhou, and J.-H. Won. 2020. "Diststat. Jl: Towards Unified Programming for High-Performance Statistical Computing Environments in Julia." *arXiv Preprint arXiv:2010.16114*.
- Krasnosky, K. E., and C. Roman. 2022. "A Massively Parallel Implementation of Gaussian Process Regression for Real Time Bathymetric Modeling and Simultaneous Localization and Mapping." *Field Robotics* 2: 940–970.
- Kriemann, R., H. Ltaief, M. B. Luong, F. E. H. Pérez, H. G. Im, and D. Keyes. 2022. "High-Performance Spatial Data Compression for Scientific Applications." In *European Conference on Parallel Processing*, 403–418. Springer.
- Krupa, J., K. Lin, M. A. Flechas, et al. 2021. "GPU Coprocessors as a Service for Deep Learning Inference in High Energy Physics." *Machine Learning: Science and Technology* 2: 035005.
- Kucherenko, S., P. Bilokon, and C. Williams. 2023. "Quasi-Monte Carlo Methods for Calculating Derivatives Sensitivities on the GPU." *Wilmott* 127: 62–77.
- Kumari, S., N. Patil, P. Nankar, and M. Kulkarni. 2020. "CUDA Parallel Computing Framework for Stock Market Prediction Using K-Means Clustering." In *2020 International Conference on Smart Electronics and Communication (ICOSEC)*, 467–473. IEEE.
- Kůš, P., A. Marek, S. S. Köcher, et al. 2019. "Optimizations of the Eigensolvers in the ELPA Library." *Parallel Computing* 85: 167–177.
- Lach, M., and M. Žukovič. 2024. "Fast Gap-Filling of Massive Data by Local-Equilibrium Conditional Simulations on GPU." *Mathematical Geosciences* 56: 573–603.
- Leclercq, F., B. Faure, G. Lavaux, et al. 2020. "Perfectly Parallel Cosmological Simulations Using Spatial Comoving Lagrangian Acceleration." *Astronomy & Astrophysics* 639: A91.
- Leon-Gonzalez, R., and B. Majoni. 2025. "Exact Likelihood for Inverse Gamma Stochastic Volatility Models." *Journal of Time Series Analysis* 46: 774–795.
- Li, H.-F., J.-T. Wang, Q. Zhao, and Y.-M. Zhang. 2024. "BLUPmrMLM: A Fast mrMLM Algorithm in Genome-Wide Association Studies." *Genomics, Proteomics & Bioinformatics* 22: qzae020.
- Li, M. 2024. "Exploring the Future of Biostatistics in Genomic Research: Opportunities and Challenges." *Genomics and Applied Biology* 15, no. 4: 172–181.
- Li, X., S. Gao, R. Gao, and S. Xu. 2025. "Spatially-Heterogeneous Causal Bayesian Networks for Seismic Multi-Hazard Estimation: A Variational Approach With Gaussian Processes and Normalizing Flows." *arXiv Preprint arXiv:2504.04013*.

- Liu, F., R. He, T. Sheeley, et al. 2025. "SPAC: A Scalable, Integrated Enterprise Platform for End-To-End Single Cell Spatial Analysis of Multiplexed Tissue Imaging." *bioRxiv*, 2025–04.
- Liu, S., C. Zhao, X. An, K. Gao, Y. Ji, and W. Wu. 2022. "Accelerating Ordinary Kriging Interpolation Algorithm on GPUs." In *2022 IEEE International Conference on Advances in Electrical Engineering and Computer Applications (AEECA)*, 225–231. IEEE.
- Liu, Y., K. Wu, S. Wang, Y. Zhao, and Q. Huang. 2010. "A Mapreduce Approach to G I*(D) Spatial Statistic." In *Proceedings of the ACM SIGSPATIAL International Workshop on High Performance and Distributed Geographic Information Systems*, 11–18.
- Liyanage, D., G. Fernando, D. Arachchi, R. Karunathilaka, and A. S. Perera. 2017. "Utilizing Intel Advanced Vector Extensions for Monte Carlo Simulation Based Value at Risk Computation." *Procedia Computer Science* 108: 626–634.
- Loh, W.-K., Y.-S. Moon, and W. Lee. 2011. "A Fast Divide-And-Conquer Algorithm for Indexing Human Genome Sequences." *IEICE Transactions on Information and Systems* 94: 1369–1377.
- Louw, T., and S. McIntosh-Smith. 2021. "Using the Graphcore IPU for Traditional HPC Applications." In *3rd Workshop on Accelerated Machine Learning (AccML)*.
- Loya, H., G. Kalantzis, F. Cooper, and P. F. Palamara. 2025. "A Scalable Variational Inference Approach for Increased Mixed-Model Association Power." *Nature Genetics* 57: 461–468.
- Ltaief, H., R. Alomairy, Q. Cao, et al. 2024. "Toward Capturing Genetic Epistasis From Multivariate Genome-Wide Association Studies Using Mixed-Precision Kernel Ridge Regression." In *SC24: International Conference for High Performance Computing, Networking, Storage and Analysis*, 1–12. IEEE.
- Ltaief, H., Y. Hong, A. Dabah, et al. 2023. "Steering Customized AI Architectures for Hpc Scientific Applications." In *International Conference on High Performance Computing*, 125–143. Springer.
- Lu, W.-j., S. Kawasaki, and J. Sakuma. 2016. *Using Fully Homomorphic Encryption for Statistical Analysis of Categorical, Ordinal and Numerical Data*. Cryptology ePrint Archive.
- Ly-Trong, N., C. Bielow, N. De Maio, and B. Q. Minh. 2024. "CMAPLE: Efficient Phylogenetic Inference in the Pandemic Era." *Molecular Biology and Evolution* 41: msae134.
- Makinen, T. L., T. Charnock, J. Alsing, and B. D. Wandelt. 2021. "Lossless, Scalable Implicit Likelihood Inference for Cosmological Fields." *Journal of Cosmology and Astroparticle Physics* 2021: 049.
- Mann, A. 2020. "Nascent Exascale Supercomputers Offer Promise, Present Challenges." *Proceedings of the National Academy of Sciences* 117: 22623–22625.
- Mano, O., and D. A. Clark. 2017. "Graphics Processing Unit-Accelerated Code for Computing Second-Order Wiener Kernels and Spike-Triggered Covariance." *PLoS One* 12: e0169842.
- Marion, Z. H., J. A. Fordyce, and B. M. Fitzpatrick. 2018. "A Hierarchical Bayesian Model to Incorporate Uncertainty Into Methods for Diversity Partitioning." *Ecology* 99: 947–956.
- Markidis, S., S. W. Der Chien, E. Laure, I. B. Peng, and J. S. Vetter. 2018. "Nvidia Tensor Core Programmability, Performance & Precision." In *In 2018 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*, 522–531. IEEE.
- McMahan, B., E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas. 2017. "Communication-Efficient Learning of Deep Networks From Decentralized Data." In *Artificial Intelligence and Statistics*, 1273–1282. PMLR.
- Michalak, S., A. DuBois, D. DuBois, S. V. Wiel, and J. Hogden. 2012. "Developing Systems for Real-Time Streaming Analysis." *Journal of Computational and Graphical Statistics* 21: 561–580.
- Minsker, S. 2019. "Distributed Statistical Estimation and Rates of Convergence in Normal Approximation."
- Mohammed, E. A., B. H. Far, and C. Naugler. 2014. "Applications of the Mapreduce Programming Framework to Clinical Big Data Analysis: Current Landscape and Future Trends." *Biodata Mining* 7: 1–23.
- Molnár, F., Jr., T. Szakály, R. Mészáros, and I. Lagzi. 2010. "Air Pollution Modelling Using a Graphics Processing Unit With CUDA." *Computer Physics Communications* 181: 105–112.
- Mondal, S., S. Abdulah, H. Ltaief, Y. Sun, M. G. Genton, and D. E. Keyes. 2022. "Parallel Approximations of the Tukey g-and-h Likelihoods and Predictions for Non-Gaussian Geostatistics." In *2022 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, 379–389. IEEE.
- Mondal, S., S. Abdulah, H. Ltaief, Y. Sun, M. G. Genton, and D. E. Keyes. 2023. "Tile Low-Rank Approximations of Non-Gaussian Space and Space-Time Tukey g-and-h Random Field Likelihoods and Predictions on Large-Scale Systems." *Journal of Parallel and Distributed Computing* 180: 104715.
- Monteiro, A. M., and A. A. Santos. 2023. "Parallel Computing in Finance for Estimating Risk-Neutral Densities Through Option Prices." *Journal of Parallel and Distributed Computing* 173: 61–69.
- Moore, G. 1965. "Moore's Law." *Electronics Magazine* 38, no. 8: 114.
- Moritz, P., R. Nishihara, S. Wang, et al. 2018. "Ray: A Distributed Framework for Emerging AI Applications." In *13th USENIX Symposium on Operating Systems Design and Implementation (OSDI 18)*, 561–577.
- Mudnic, E., S. Celar, Z. Seremet, and F. Matic. 2017. "European Swaption by Monte Carlo Estimation on Modern Intel Architectures." In *28th DAAAM International Symposium on Intelligent Manufacturing and Automation*, 351–356.
- Mudunuru, M. K., J. Ang, M. Halappanavar, et al. 2024. "Perspectives on AI Architectures and Codesign for Earth System Predictability." *Artificial Intelligence for the Earth Systems* 3: e230029.
- Müller, A., W. Deconinck, C. Kühnlein, et al. 2019. "The ESCAPE Project: Energy-Efficient Scalable Algorithms for Weather Prediction at Exascale." *Geoscientific Model Development* 12: 4425–4441.
- Mwamnyange, M., E. T. Luhanga, and S. R. Thodge. 2021. "Big Data Analytics Framework for Childhood Infectious Disease Surveillance and Response System Using Modified Mapreduce Algorithm."
- Nag, P., Y. Hong, S. Abdulah, G. A. Qadir, M. G. Genton, and Y. Sun. 2025. "Efficient Large-Scale Nonstationary Spatial Covariance Function Estimation Using Convolutional Neural Networks." *Journal of Computational and Graphical Statistics* 34: 683–696.
- Nesi, L. L., L. M. Schnorr, and A. Legrand. 2022. "Multi-Phase Task-Based HPC Applications: Quickly Learning How to Run Fast." In *2022 IEEE International Parallel and Distributed Processing Symposium*, 357–367. IEEE.
- Nguyen, H. D., L. R. Lloyd-Jones, and G. J. McLachlan. 2016. "A Block Minorization-Maximization Algorithm for Heteroscedastic Regression." *IEEE Signal Processing Letters* 23: 1131–1135.
- Nie, N. H., D. H. Bent, and C. H. Hull. 1970. *SPSS: Statistical Package for the Social Sciences*. McGraw-Hill.
- Noack, M. M., H. Krishnan, M. D. Risser, and K. G. Reyes. 2023. "Exact Gaussian Processes for Massive Datasets via Non-Stationary Sparsity-Discovering Kernels." *Scientific Reports* 13: 3155.
- NVIDIA. 1999. "Nvidia Introduces the Geforce 256." Press Release. <https://www.nvidia.com/object/geforce256.html>.
- NVIDIA. 2025. "Cuda Toolkit (Release 12.9.1)." <https://developer.nvidia.com/cuda-toolkit>.
- NVIDIA Corporation. 2025. "NVIDIA cuBLAS." Accessed: 2025-07-03. <https://developer.nvidia.com/cublas>.

- Nygaard, A., E. B. Holm, S. Hannestad, and T. Tram. 2023. "CONNECT: A Neural Network Based Framework for Emulating Cosmological Observables and Cosmological Parameter Inference." *Journal of Cosmology and Astroparticle Physics* 2023: 025.
- Olman, V., F. Mao, H. Wu, and Y. Xu. 2008. "Parallel Clustering Algorithm for Large Data Sets With Applications in Bioinformatics." *IEEE/ACM Transactions on Computational Biology and Bioinformatics* 6: 344–352.
- OpenBLAS Project Contributors. 2025. "OpenBLAS." Accessed: 2025-07-03. <https://www.openblas.net/>.
- Ostrouchov, G., D. Maxwell, R. A. Ashraf, C. Engelmann, M. Shankar, and J. H. Rogers. 2020. "GPU Lifetimes on Titan Supercomputer: Survival Analysis and Reliability." In *SC20: International Conference for High Performance Computing, Networking, Storage and Analysis*, 1–14. IEEE.
- Ou, Z., B. Pang, Y. Deng, J. K. Nurminen, A. Ylä-Jääski, and P. Hui. 2012. "Energy-and Cost-Efficiency Analysis of Arm-Based Clusters." In *2012 12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (ccgrid 2012)*, 115–123. IEEE.
- Paciorek, C. J., B. Lipshitz, W. Zhuo, et al. 2015. "Parallelizing Gaussian Process Calculations in r." *Journal of Statistical Software* 63: 1–23.
- Pan, Q., S. Abdulah, M. Abduljabbar, et al. 2025. "Scaled Block Vecchia Approximation for High-Dimensional Gaussian Process Emulation on Gpus." *arXiv Preprint arXiv:2504.12004*.
- Pan, Q., S. Abdulah, M. G. Genton, D. E. Keyes, H. Ltaief, and Y. Sun. 2024. "GPU-Accelerated Vecchia Approximations of Gaussian Processes for Geospatial Data Using Batched Matrix Computations." In *ISC High Performance 2024 Research Paper Proceedings (39th International Conference)*, 1–12. Prometheus GmbH.
- Pan, Q., S. Abdulah, M. G. Genton, and Y. Sun. 2025. "Block Vecchia Approximation for Scalable and Efficient Gaussian Process Computations." *Technometrics* 67, no. 3: 1–558.
- Patil, S., C. P. Lora, and A. Prabhu. 2024. "Genome Analysis at Scale: Leveraging HPC for AI-Driven Genomics Research." In *2024 International Conference on Advances in Computing Research on Science Engineering and Technology (ACROSET)*, 1–6. IEEE.
- Piras, D., A. Polanska, A. S. Mancini, M. A. Price, and J. D. McEwen. 2024. "The Future of Cosmological Likelihood-Based Inference: Accelerated High-Dimensional Parameter Estimation and Model Comparison." *arXiv Preprint arXiv:2405.12965*.
- Prasad, S. K., D. Aghajarian, M. McDermott, et al. 2017. "Parallel Processing Over Spatial-Temporal Datasets From Geo, Bio, Climate and Social Science Communities: A Research Roadmap." In *2017 IEEE International Congress on Big Data (BigData Congress)*, 232–250. IEEE.
- Prasad, S. K., M. McDermott, S. Puri, et al. 2015. "A Vision for GPU-Accelerated Parallel Computation on Geo-Spatial Datasets." *SIGSPATIAL Special* 6: 19–26.
- Pratas, F., P. Trancoso, A. Stamatakis, and L. Sousa. 2009. "Fine-Grain Parallelism Using Multi-Core, Cell/BE, and GPU Systems: Accelerating the Phylogenetic Likelihood Function." In *2009 International Conference on Parallel Processing*, 9–17. IEEE.
- Putnam, A., A. M. Caulfield, E. S. Chung, et al. 2014. "A Reconfigurable Fabric for Accelerating Large-Scale Datacenter Services." *ACM SIGARCH Computer Architecture News* 42: 13–24.
- Qian, H. 2016. "A Computationally Efficient Method for Vector Autoregression With Mixed Frequency Data." *Journal of Econometrics* 193: 433–437.
- Que, X., C. Ma, X. Ma, and Q. Chen. 2021. "Parallel Computing for Fast Spatiotemporal Weighted Regression." *Computers & Geosciences* 150: 104723.
- Raim, A. M., M. K. Gobbert, N. K. Neerchal, and J. G. Morel. 2013. "Maximum-Likelihood Estimation of the Random-Clumped Multinomial Model as a Prototype Problem for Large-Scale Statistical Computing." *Journal of Statistical Computation and Simulation* 83, no. 12: 2178–2194.
- Rauber, T., and G. Rünger. 2013. *Parallel Programming: For Multicore and Cluster Systems*. Springer.
- Rocklin, M. 2015. "Dask: Parallel Computation With Blocked Algorithms and Task Scheduling." In *SciPy*, 126–132.
- Rodrigues, W., A. Pecchia, M. Lopez, M. A. der Maur, and A. Di Carlo. 2014. "Accelerating Atomistic Calculations of Quantum Energy Eigenstates on Graphic Cards." *Computer Physics Communications* 185: 2510–2518.
- Rodriguez, A., Y. Kim, T. N. Nandi, et al. 2024. "Accelerating Genome- and Phenome-Wide Association Studies Using GPUs – A Case Study Using Data From the Million Veteran Program." *bioRxiv*.
- Rouholahnejad, E., K. C. Abbaspour, M. Vejdani, R. Srinivasan, R. Schulin, and A. Lehmann. 2012. "A Parallelization Framework for Calibration of Hydrological Models." *Environmental Modelling & Software* 31: 28–36.
- Roy, S., Y. Atchadé, and G. Michailidis. 2019. "Likelihood Inference for Large Scale Stochastic Blockmodels With Covariates Based on a Divide-And-Conquer Parallelizable Algorithm With Communication." *Journal of Computational and Graphical Statistics* 28: 609–619.
- Russkova, T. 2018. "Monte Carlo Simulation of the Solar Radiation Transfer in a Cloudy Atmosphere With the Use of Graphic Processor and NVIDIA CUDA Technology." *Atmospheric and Oceanic Optics* 31: 119–130.
- Sabnis, G., D. Pati, B. Engelhardt, and N. Pillai. 2016. "A Divide and Conquer Strategy for High Dimensional Bayesian Factor Models." *arXiv Preprint arXiv:1612.02875*.
- Salloum, S., R. Dautov, X. Chen, P. X. Peng, and J. Z. Huang. 2016. "Big Data Analytics on Apache Spark." *International Journal of Data Science and Analytics* 1: 145–164.
- Saltas, I., and R. Oliveri. 2025. "EMRI_MC: A GPU-Based Code for Bayesian Inference of EMRI Waveforms." *SciPost Physics Codebases*, 044.
- Salváñia, M. L. O., S. Abdulah, H. Huang, et al. 2021. "High Performance Multivariate Geospatial Statistics on Manycore Systems." *IEEE Transactions on Parallel and Distributed Systems* 32: 2719–2733.
- Salváñia, M. L. O., S. Abdulah, M. Kim, D. Helmy, Y. Sun, and M. G. Genton. 2024. "MPCR: Multi-and Mixed-Precision Computations Package in R." *arXiv Preprint arXiv:2406.02701*.
- Salváñia, M. L. O., S. Abdulah, H. Ltaief, Y. Sun, M. G. Genton, and D. E. Keyes. 2022. "Parallel Space-Time Likelihood Optimization for Air Pollution Prediction on Large-Scale Systems." In *Proceedings of the Platform for Advanced Scientific Computing Conference*, 1–11.
- Santander-Jiménez, S., and M. A. Vega-Rodríguez. 2016. "Asynchronous Non-Generational Model to Parallelize Metaheuristics: A Bioinformatics Case Study." *IEEE Transactions on Parallel and Distributed Systems* 28: 1825–1838.
- Sarma, R., E. Inanc, M. Aach, and A. Lintermann. 2024. "Parallel and Scalable AI in HPC Systems for CFD Applications and Beyond." *Frontiers in High Performance Computing* 2: 1444337.
- ScaLAPACK Project. 2025. "ScaLAPACK." Accessed: 2025-07-03. <https://www.netlib.org/scalapack/>.
- Schmidt, H. A., K. Strimmer, M. Vingron, and A. Von Haeseler. 2002. "TREE-PUZZLE: Maximum Likelihood Phylogenetic Analysis Using Quartets and Parallel Computing." *Bioinformatics* 18: 502–504.

- Schulz, W. L., T. J. Durant, A. J. Siddon, and R. Torres. 2016. "Use of Application Containers and Workflows for Genomic Data Analysis." *Journal of Pathology Informatics* 7: 53.
- Schutte, J. F., J. A. Reinbolt, B. J. Fregly, R. T. Haftka, and A. D. George. 2004. "Parallel Global Optimization With the Particle Swarm Algorithm." *International Journal for Numerical Methods in Engineering* 61: 2296–2315.
- Scott, S. L. 2017. "Comparing Consensus Monte Carlo Strategies for Distributed Bayesian Computation."
- Shealy, B. T., J. J. Burns, M. C. Smith, F. A. Feltus, and S. P. Ficklin. 2019. "GPU Implementation of Pairwise Gaussian Mixture Models for Multi-Modal Gene Co-Expression Networks." *IEEE Access* 7: 160845–160857.
- Shi, J., S. Abdulah, Y. Sun, and M. G. Genton. 2025. "Decentralized Inference for Spatial Data Using Low-Rank Models." *arXiv Preprint arXiv:2502.00309*.
- Shohdy, S., A. Vishnu, and G. Agrawal. 2016. "Fault Tolerant Support Vector Machines." In *2016 45th International Conference on Parallel Processing (ICPP)*, 598–607. IEEE.
- Smith, K., D. Ayres, R. Neumaier, G. Wörheide, and S. Höhna. 2024. "Bayesian Phylogenetic Analysis on Multi-Core Compute Architectures: Implementation and Evaluation of BEAGLE in RevBayes With MPI." *Systematic Biology* 73: 455–469.
- Smith, R. J., G. Ashton, A. Vajpeyi, and C. Talbot. 2020. "Massively Parallel Bayesian Inference for Transient Gravitational-Wave Astronomy." *Monthly Notices of the Royal Astronomical Society* 498: 4492–4502.
- Solonen, A., P. Ollinaho, M. Laine, H. Haario, J. Tamminen, and H. Järvinen. 2012. "Efficient MCMC for Climate Model Parameter Estimation: Parallel Adaptive Chains and Early Rejection." *Bayesian Analysis* 7: 715–736.
- Spurio Mancini, A., D. Piras, J. Alsing, B. Joachimi, and M. P. Hobson. 2022. "CosmoPower: Emulating Cosmological Power Spectra for Accelerated Bayesian Inference From Next-Generation Surveys." *Monthly Notices of the Royal Astronomical Society* 511: 1771–1788.
- Stamatakis, A., P. Hoover, and J. Rougemont. 2008. "A Rapid Bootstrap Algorithm for the RAxML Web Servers." *Systematic Biology* 57: 758–771.
- Stamatakis, A., and M. Ott. 2008. "Efficient Computation of the Phylogenetic Likelihood Function on Multi-Gene Alignments and Multi-Core Architectures." *Philosophical Transactions of the Royal Society, B: Biological Sciences* 363: 3977–3984.
- Stewart, C. A., D. Hart, D. K. Berry, G. J. Olsen, E. A. Wernert, and W. Fischer. 2001. "Parallel Implementation and Performance of FastDNAml: A Program for Maximum Likelihood Phylogenetic Inference." In *Proceedings of the 2001 ACM/IEEE Conference on Supercomputing*, 20–20.
- Stewart, G. A., A. M. Briceño, P. Gras, et al. 2025. "Julia in Hep." *arXiv Preprint arXiv:2503.08184*.
- Su, Y. 2020. "A Divide and Conquer Algorithm of Bayesian Density Estimation." *arXiv Preprint arXiv:2002.07094*.
- Suchard, M. A., and A. Rambaut. 2009. "Many-Core Algorithms for Statistical Phylogenetics." *Bioinformatics* 25: 1370–1376.
- Suchard, M. A., Q. Wang, C. Chan, J. Frelinger, A. Cron, and M. West. 2010. "Understanding GPU Programming for Statistical Computation: Studies in Massively Parallel Massive Mixtures." *Journal of Computational and Graphical Statistics* 19: 419–438.
- Sukkuea, A., and A. Heednacram. 2022. "Practical Kriging Models With Divide and Conquer Algorithms for Spatial Heights Forecast." *Ecological Informatics* 70: 101756.
- Sunderland, D., B. Peterson, J. Schmidt, A. Humphrey, J. Thornock, and M. Berzins. 2016. "An Overview of Performance Portability in the Uintah Runtime System Through the Use of Kokkos." In *2016 Second International Workshop on Extreme Scale Programming Models and Middleware (ESPM2)*, 44–47. IEEE.
- Sutter, H. 2005. "The Free Lunch Is Over: A Fundamental Turn Toward Concurrency in Software." *Dr. Dobb's Journal* 30: 202–210.
- Szalai-Gindl, J. M., T. J. Loredo, B. C. Kelly, I. Csabai, T. Budavári, and L. Dobos. 2021. "GPU-Accelerated Hierarchical Bayesian Inference With Application to Modeling Cosmic Populations: CUDAHM." *arXiv Preprint arXiv:2105.08026*.
- Talbot, C., R. Smith, E. Thrane, and G. B. Poole. 2019. "Parallelized Inference for Gravitational-Wave Astronomy." *Physical Review D* 100: 043030.
- Thavapiragam, M., V. Kale, O. Hernandez, and A. Sedova. 2021. "Addressing Load Imbalance in Bioinformatics and Biomedical Applications: Efficient Scheduling Across Multiple GPUs." In *2021 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, 1992–1999. IEEE.
- Tilly, J., H. Chen, S. Cao, et al. 2022. "The Variational Quantum Eigensolver: A Review of Methods and Best Practices." *Physics Reports* 986: 1–128.
- Tiwari, D., S. Gupta, J. Rogers, et al. 2015. "Understanding GPU Errors on Large-Scale HPC Systems and the Implications for System Design and Operation." In *2015 IEEE 21st International Symposium on High Performance Computer Architecture (HPCA)*, 331–342. IEEE.
- TOP500. 2024. "Top500 the List." Accessed: February 7, 2025. <https://www.top500.org>.
- Van Rossum, G., and F. L. Drake Jr. 1995. *Python Tutorial*. Vol. 620. Centrum voor Wiskunde en Informatica Amsterdam.
- Vyner, C., C. Nemeth, and C. Sherlock. 2023. "Swiss: A Scalable Markov Chain Monte Carlo Divide-And-Conquer Strategy." *Stat* 12: e523.
- Wang, C., J. Ren, and P. Yang. 2025. "Alleviating Nonidentifiability: A High-Fidelity Calibration Objective for Financial Market Simulation With Multivariate Time Series Data." *IEEE Transactions on Computational Social Systems*: 1–13. <https://doi.org/10.1109/TCSS.2025.3574236>.
- Wang, H., S. H. Suyu, A. Galan, et al. 2025. "GPU-Accelerated Gravitational Lensing & Dynamical (GLaD) Modeling for Cosmology and Galaxies." *Astronomy and Astrophysics* 701: A280. *arXiv Preprint arXiv:2504.01302*.
- Wang, K., N. Holzer, Z. Xia, et al. 2025. "FinRL Contests: Benchmarking Data-Driven Financial Reinforcement Learning Agents." *arXiv Preprint arXiv:2504.02281*.
- Wang, L., Y. Zhu, X. Wan, et al. 2024. "Com-DNB: A Novel Method for Identifying Critical States of Complex Biological Processes and Its Parallelization." In *Proceedings of the 15th ACM International Conference on Bioinformatics, Computational Biology and Health Informatics*, 1–10.
- Wang, Y., C. Hong, N. Palmer, et al. 2021. "A Fast Divide-and-Conquer Sparse Cox Regression." *Biostatistics* 22: 381–401.
- Wei, H., Y. Du, F. Liang, et al. 2015. "A k-d Tree-Based Algorithm to Parallelize Kriging Interpolation of Big Spatial Data." *GIScience & Remote Sensing* 52: 40–57.
- Weine, E., P. Carbonetto, and M. Stephens. 2024. "Accelerated Dimensionality Reduction of Single-Cell RNA Sequencing Data With Fastglmpca." *Bioinformatics* 40: btae494.
- White, G., and M. D. Porter. 2014. "GPU Accelerated MCMC for Modeling Terrorist Activity." *Computational Statistics and Data Analysis* 71: 643–651.
- Williams, S., A. Waterman, and D. Patterson. 2009. "Roofline: An Insightful Visual Performance Model for Multicore Architectures." *Communications of the ACM* 52: 65–76.

- Woodruff, D. P. 2014. "Sketching as a Tool for Numerical Linear Algebra." *Foundations and Trends in Theoretical Computer Science* 10: 1–157.
- Wysocki, D., R. O'Shaughnessy, J. Lange, and Y.-L. L. Fang. 2019. "Accelerating Parameter Inference With Graphics Processing Units." *Physical Review D* 99: 084026.
- Xiao, N., J. Guo, Z. Kuang, and W. Wang. 2025. "Application of Data Partitioned Kriging Algorithm With GPU Acceleration in Real-Time and Refined Reconstruction of Three-Dimensional Radiation Fields." *Annals of Nuclear Energy* 212: 111047.
- Xu, G., Z. Shang, and G. Cheng. 2019. "Distributed Generalized Cross-Validation for Divide-and-Conquer Kernel Ridge Regression and Its Asymptotic Optimality." *Journal of Computational and Graphical Statistics* 28: 891–908.
- Xu, R., and P. Brown. 2024. "Profile Likelihoods for Parameters in Trans-Gaussian Geostatistical Models." *Spatial Statistics* 60: 100821.
- Xu, Z., J. J. Alonso, and E. Darve. 2024. "A Numerically Stable Communication-Avoiding-Step Gmres Algorithm." *SIAM Journal on Matrix Analysis and Applications* 45: 2039–2074.
- Yan, C., J. Liu, G. Zhao, C. Chen, and T. Yue. 2016. "A High Accuracy Surface Modeling Method Based on GPU Accelerated Multi-Grid Method." *Transactions in GIS* 20: 991–1003.
- Yan, C., G. Zhao, T. Yue, et al. 2015. "Speeding Up the High-Accuracy Surface Modelling Method With GPU." *Environmental Earth Sciences* 74: 6511–6523.
- Yang, G., W. Jiang, Q. Yang, and W. Yu. 2015. "PBOOST: A GPU-Based Tool for Parallel Permutation Tests in Genome-Wide Association Studies." *Bioinformatics* 31: 1460–1462.
- Yao, X., Q. Wang, Z. Liu, G. Hu, D. Huang, and W. Zou. 2015. "Fast 3D Kriging Interpolation Using Delaunay Tetrahedron With CUDA-Enabled GPU." In *SEG International Exposition and Annual Meeting, SEG-2015*. SEG.
- Yazdi, M. 2024. "Application of Quantum Computing in Reliability Analysis." In *Advances in Computational Mathematics for Industrial System Reliability and Maintainability*, 139–154. Springer.
- Yen, T.-J. 2020. "Solving Fused Penalty Estimation Problems via Block Splitting Algorithms." *Journal of Computational and Graphical Statistics* 29: 297–308.
- Yildirim, I., H. Devarajan, A. Kougkas, X.-H. Sun, and K. Mohror. 2023. "Iomax: Maximizing Out-of-Core i/o Analysis Performance on HPC Systems." In *Proceedings of the SC'23 Workshops of the International Conference on High Performance Computing, Network, Storage, and Analysis*, 1209–1215.
- Yin, L., H. Zhang, Z. Tang, et al. 2021. "rMVP: A Memory-Efficient, Visualization-Enhanced, and Parallel-Accelerated Tool for Genome-Wide Association Study." *Genomics, Proteomics & Bioinformatics* 19: 619–628.
- Yin, Z., W. Liao, X. Lei, and H. Wang. 2020. "Parallel Hydrological Model Parameter Uncertainty Analysis Based on Message-Passing Interface." *Water* 12: 2667.
- Yoginath, S. B., N. F. Samatova, D. Bauer, G. Kora, G. Fann, and A. Geist. 2005. "Rscalapack: High-Performance Parallel Statistical Computing With r and Scalapack." In *PDCS*, 61–67.
- Yue, X., and R. Kontar. 2024. "Federated Gaussian Process: Convergence, Automatic Personalization and Multi-Fidelity Modeling." *IEEE Transactions on Pattern Analysis and Machine Intelligence* 46: 4246–4261.
- Zenios, S. A. 1999. "High-Performance Computing in Finance: The Last 10 Years and the Next." *Parallel Computing* 25: 2149–2175.
- Zhang, X., J. Conejero, S. Abdulah, et al. 2025. "Rcompss: A Scalable Runtime System for r Code Execution on Manycore Systems." *arXiv Preprint arXiv:2505.06896*.
- Zhang, X., Y. Li, W. Li, K. Guo, and Y. Shao. 2022. "Personalized Federated Learning via Variational Bayesian Inference." In *International Conference on Machine Learning*, 26293–26310. PMLR.
- Zhang, Y., J. Duchi, and M. Wainwright. 2013. "Divide and conquer kernel ridge regression." In *Conference on Learning Theory*, 592–617. PMLR.