Scientific Publications

Journal of Artificial Intelligence and Big Data, 2024, 4, 1695
www.scipublications.com/journal/index.php/jaibd
DOI: 10.31586/jaibd.2024.1695

Review Article

AI-Powered Optimization for High-Performance Computing in Scientific Simulations

Shubham Gupta [1,*]

[1] Noblesoft Solutions, San Antonio, Texas, USA

*Correspondence: Shubham Gupta (shubhamgupta133@gmail.com)

Abstract: High-Performance Computing (HPC) is indispensable for large-scale scientific simulations, but achieving optimal performance on modern supercomputers is increasingly challenging. As HPC systems scale toward exascale, they face escalating complexity in hardware, software, and workloads. Traditional optimization methods (manual tuning and heuristic algorithms) struggle to cope with dynamic workloads and intricate system behaviors. Artificial Intelligence (AI) techniques offer a promising approach to address these challenges. This article provides an overview of AI-powered optimization in HPC, focusing on how machine learning and related AI methods enhance performance, efficiency, and scalability of scientific simulations. We survey key AI techniques applied to HPC optimization including machine learning for performance modeling, deep reinforcement learning for resource management, and AI-driven surrogate models for accelerating simulations and illustrate their impact through case studies in domains such as job scheduling and fluid dynamics. We discuss the practical applications of these techniques, highlighting reported performance gains (e.g., substantial reductions in simulation run time and improved resource utilization). We also examine the challenges in integrating AI with HPC (such as training overhead, data movement, and reliability concerns) and outline future directions for research. The convergence of AI and HPC is poised to produce "smart" simulation workflows that intelligently adapt and optimize in real time, pushing the frontiers of scientific computing.

Keywords: High-Performance Computing (HPC); Artificial Intelligence; Machine Learning; Deep Reinforcement Learning; Surrogate Modeling; Bayesian Optimization; Scientific Simulations; Exascale Computing; Job Scheduling; Computational Fluid Dynamics; Autotuning; Neural Networks; Resource Management; Performance Optimization; In-Situ Analytics

How to cite this paper:
Gupta, S. (2024). AI-Powered Optimization for High-Performance Computing in Scientific Simulations. *Journal of Artificial Intelligence and Big Data, 4*(1), 1-8. DOI: 10.31586/jaibd.2024.1695

Received: February 12, 2024
Revised: May 21, 2024
Accepted: July 26, 2024
Published: July 30, 2024

1. Introduction

High-Performance Computing has revolutionized scientific research by enabling simulations of complex physical phenomena from climate models and astrophysics to molecular dynamics at unprecedented scale and fidelity [9]. Modern HPC systems consist of tens of thousands of processors (often augmented with GPUs or other accelerators) capable of peta scale and, in the near term, exascale performance. This immense scale introduces new challenges in reliability, energy consumption, and software complexity [1]. As systems grow larger, HPC researchers must cope with higher failure rates, power demands measured in tens of megawatts, and extreme code complexity [2]. In essence, relying solely on traditional HPC optimization techniques is no longer sufficient to fully exploit these cutting-edge machines [3].

A critical aspect of HPC is performance optimization: ensuring that vast computing resources are efficiently utilized to minimize simulation time and maximize throughput [1]. Historically, HPC optimization has relied on expert-driven tuning and relatively

simple heuristics. For example, batch job schedulers have used fixed priority rules (e.g., first-come-first-served or shortest-job-first) to allocate resources [4]. While effective in earlier systems, such manually crafted heuristics struggle to adapt to the scale and variability of modern workloads, which can be highly diverse and dynamic [2]. Moreover, HPC applications often have numerous tunable parameters; algorithmic settings, mesh resolution, and communication topology that interact in complex ways with hardware performance [1]. The resulting optimization problem, whether scheduling jobs, tuning application parameters, or managing I/O and memory, has become extraordinarily complex and often NP-hard [3]. This complexity has motivated researchers to explore AI-powered methods that can automatically learn from data and experience to make better optimization decisions than static rules [5].

Recent years have seen a convergence of HPC with advances in artificial intelligence and machine learning. AI techniques (including classical machine learning, deep learning, and reinforcement learning) have demonstrated remarkable success in pattern recognition and decision-making problems, suggesting they could assist in navigating HPC optimization spaces that elude manual reasoning [4]. The HPC community has begun adopting AI for tasks such as performance modeling, job scheduling, fault prediction, and augmenting simulations with learned surrogate models [6,7]. Early results are promising: machine learning models can predict program runtimes or resource needs more accurately than user estimates [2], and learned scheduling policies can outperform human-designed heuristics [5]. Meanwhile, deep neural networks are being used as surrogate models to replace expensive physics calculations, dramatically speeding up simulations with minimal loss of accuracy [8]. These successes indicate that AI-driven approaches can unlock new levels of efficiency in HPC environments [3].

This article provides a comprehensive overview of how AI is powering optimization in HPC for scientific simulations. Section 2 surveys major AI techniques for HPC optimization, covering machine learning for performance prediction, deep reinforcement learning for resource management, and neural network surrogates for simulation acceleration [1,4]. Section 3 presents case studies in AI-enhanced job scheduling and AI-accelerated computational fluid dynamics [5,6]. Section 4 examines integration challenges, data requirements, trust, overhead, and generalization concerns together with ongoing research directions [7]. Section 5 concludes by summarizing how AI-powered optimization is shaping the future of high-performance scientific computing [9].

2. AI Techniques for Optimization in HPC

AI techniques are being leveraged at multiple levels of the HPC stack to improve performance and efficiency. Figure 1 depicts key areas where AI can be applied to HPC workflows, spanning input preprocessing, simulation execution, performance monitoring, and machine learning-driven feedback loops. The following figure depicts the AI in HPC workflows.

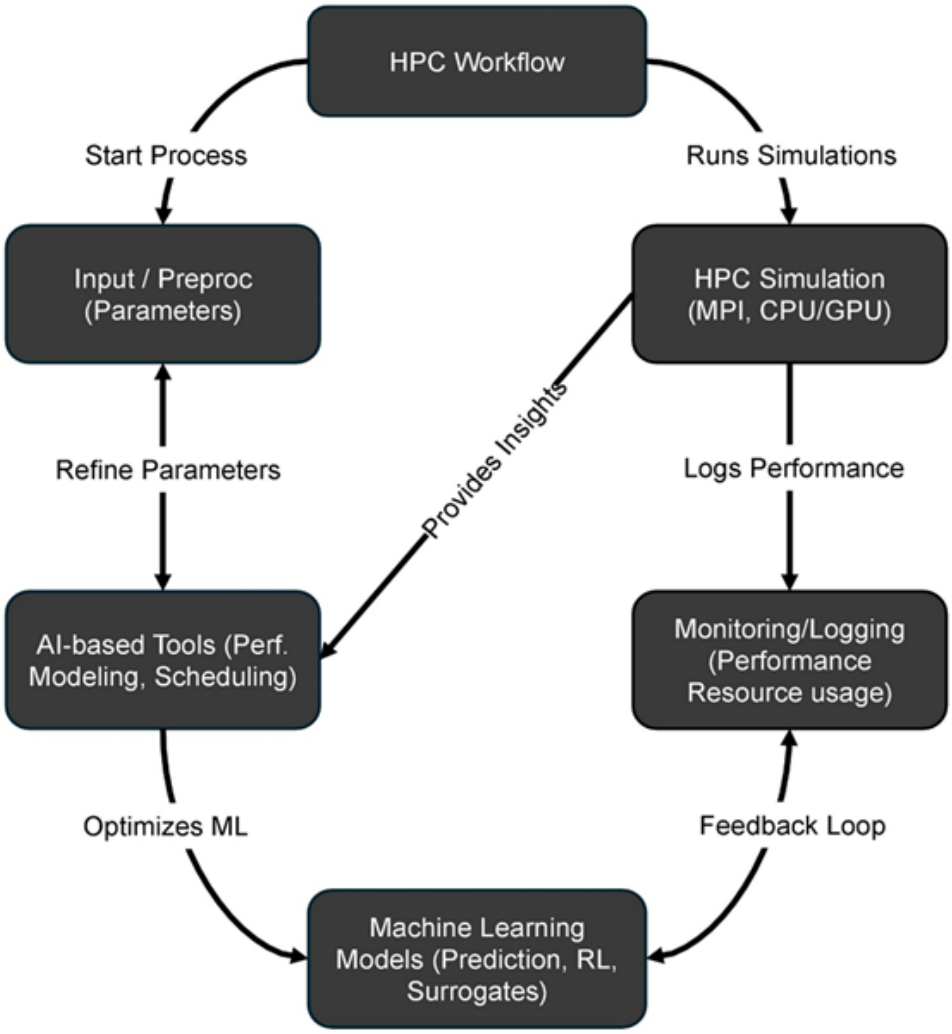

Figure 1. Conceptual Diagram of AI in HPC Workflows

2.1. Machine Learning for Performance Modeling and Prediction

Supervised learning algorithms can be trained on historical performance data of HPC applications to predict future behavior [2]. For example, regression or neural network models can predict run time or resource usage of a job based on input parameters and past performance logs [2,4]. Such predictions enable smarter scheduling and resource allocation: if a model foresees that a particular simulation will run long, the scheduler can plan accordingly or allocate more nodes [5]. Machine learning-based performance models can adapt as applications or systems change, unlike static analytical models [9]. Researchers have reported that machine learning predictions can significantly outperform user-provided estimates or simple heuristics, reducing scheduling wait times and improving overall system utilization [2]. Machine learning has also been applied to predictive maintenance in HPC, analyzing system sensor data and logs to predict node failures or performance degradation before they occur [6], allowing proactive mitigation.

2.2. Deep Reinforcement Learning for Scheduling and Resource Management

Reinforcement learning (RL) offers a framework for an AI agent to learn optimal decisions through trial-and-error interactions with the environment [3]. In HPC, the

environment is the cluster scheduler: at each time step, the agent decides which job to run or which resources to allocate, receiving a reward signal related to throughput and wait times [5]. Deep RL, which uses neural networks as function approximators, has been explored for HPC job scheduling and can learn complex policies that adapt to workload changes [2]. Unlike fixed algorithms, an RL-based scheduler improves over time by observing the outcomes of its decisions [5]. Studies have shown that RL-based schedulers outperform manual heuristics and earlier ML-based approaches, especially under diverse and changing workloads [4,5]. However, RL presents challenges: learning can be unstable or sample-inefficient, and naive online training may waste resources while the agent explores suboptimal actions [3]. To address this, researchers have proposed offline RL and imitation learning, which pre-train on historical job logs before live deployment [4,5].

2.3. Autotuning and Bayesian Optimization

HPC applications typically expose dozens of tunable parameters; block sizes, compiler flags, communication buffer depths that can dramatically affect performance. Searching the high-dimensional configuration space manually is impractical [1]. AI-driven autotuning frameworks address this with intelligent search strategies [9]. Bayesian Optimization (BO) is the leading approach: it models the performance metric as a black-box function of the parameter vector and uses a probabilistic surrogate to decide which configuration to evaluate next [3]. BO-based auto tuners have been shown to find near-optimal HPC configurations with far fewer experiments than brute-force or grid search, saving significant developer effort and uncovering non-intuitive settings [1,2].

2.4. Surrogate Modeling and AI-Accelerated Simulations

Surrogate modeling is among the most impactful developments at the intersection of AI and HPC [8]. Many simulations such as fluid dynamics, climate modeling, plasma physics require expensive numerical integration of differential equations. AI surrogates, typically deep neural networks, approximate these computations at a fraction of the cost [7,8]. A neural network trained on high-fidelity run data can predict the outcome of a fluid flow time-step, effectively emulating the full solver but orders of magnitude faster [8]. By substituting the surrogate for expensive kernels at selected steps, HPC workflows can reduce total runtime dramatically [4]. For example, Kochkov et al. demonstrated 40–80× wall-clock speedups in 2D turbulence simulations with errors below 1% [8]. Developing reliable surrogates requires substantial training data and rigorous validation; physics-informed neural networks partially address the data burden by incorporating domain equations directly into the training loss function [7].

2.5. Data Analytics and In-Situ Analysis

HPC simulations produce massive output datasets that challenge storage and post-processing pipelines. AI-based analytics such as clustering, anomaly detection, and dimensionality reduction can be deployed in situ (i.e., during the running simulation) to glean insights and reduce I/O volume [6]. Anomaly detection algorithms identify numerical instabilities or unexpected flow patterns, triggering adaptive refinement or checkpointing before errors propagate [3]. Neural network autoencoders compress high-dimensional field data into compact latent representations, reducing storage footprint significantly. This helps HPC workflows manage the data deluge from large-scale runs [9]. Coupled with simulation steering, in-situ AI analytics allow scientists to build adaptive workflows that respond to real-time results, focusing computational effort on the most informative regions of parameter space [1,6]. Table 1 summarizes all AI techniques discussed in this section.

Table 1. AI Techniques for HPC Optimization

AI Technique	HPC Use Case	Reported Benefit
Supervised ML (Regression)	Performance modeling (e.g., job run times)	More accurate runtime predictions ⇒ improved scheduling [2]
Deep Reinforcement Learning	Job scheduling & resource allocation	Learned policies adapt to workload changes ⇒ outperforms heuristics [5]
Bayesian Optimization	Auto-tuning HPC code parameters	Rapid convergence on near-optimal configs ⇒ fewer trials needed [3]
Surrogate Modeling (DNNs)	Accelerating computationally expensive solvers	Up to 40–80× speedups in fluid dynamics simulations [8]
Anomaly Detection (ML)	Predictive maintenance / fault tolerance	Early warning on node failures ⇒ proactive fault recovery [6]
Data Analytics (Clustering)	In-situ analysis, data reduction	Identifies patterns, reduces I/O overhead [9]

3. Case Studies and Applications

This section illustrates AI-powered optimization in HPC through concrete examples from recent research, showcasing improvements in scheduler performance, simulation speed, reliability, and workflow efficiency.

3.1. Deep Reinforcement Learning for HPC Job Scheduling

Intelligent job scheduling is a primary application of AI in HPC [2,5]. Wang et al. developed the RLSchert scheduler, which uses a deep RL agent trained with Proximal Policy Optimization (PPO) to make real-time scheduling decisions on a simulated cluster [5]. The agent's state encodes predicted job runtimes from a companion ML model, enabling resource-aware allocation [2]. In experiments on real job traces, RLSchert consistently reduced wait times and increased utilization compared with traditional backfilling heuristics [5]. Li et al. extended this line of work by training an RL scheduler entirely offline on historical logs, eliminating the instability of early online exploration [4]. The offline-trained scheduler achieved strong performance from the first day of deployment, demonstrating the practical viability of RL-based scheduling in production HPC environments [4].

3.2. AI-Accelerated Computational Fluid Dynamics

CFD simulations are among the most compute-intensive workloads in HPC, particularly for turbulent flows [1]. Kochkov et al. achieved a landmark result in 2021 by training a CNN-based surrogate to emulate fine-scale turbulence effects that normally require high-resolution grids [8]. The surrogate reduced simulation time by 40–80× while maintaining accuracy comparable to a full-resolution reference solver. This hybrid AI and HPC workflow has since inspired analogous surrogates in climate modeling, materials science, and combustion research [7]. The central lesson is that carefully validated neural networks can substitute for expensive numerical kernels, allowing HPC workflows to explore larger design spaces within the same compute budget [8].

3.3. Workflow Automation in Fusion Energy Simulations

Lawrence Livermore National Laboratory's Merlin workflow framework couples HPC simulations with machine learning for large ensemble studies [6]. In one deployment, Merlin coordinated tens of millions of small plasma physics simulations for inertial confinement fusion (ICF) optimization, with a machine learning model performing active learning to propose the next simulation parameters in real time [6]. This automated approach drastically reduced the time required to identify optimal fusion experiment conditions, while the per-simulation workflow overhead remained below 30 ms demonstrating that AI orchestration can be embedded in large-scale HPC pipelines at negligible cost [6].

3.4. Adaptive Mesh Refinement Guided by AI

Adaptive mesh refinement (AMR) concentrates computational effort in high-gradient regions, improving accuracy while conserving resources [9]. Traditional AMR relies on user-defined error indicators. Researchers have replaced these with ML models (e.g., autoencoders) trained to identify refinement-worthy regions from coarse-grid solutions [1]. In a 3D fluid flow study, the AI-guided refinement strategy reduced the number of refined cells by over 20% compared with a standard threshold approach, while preserving solution accuracy and saving substantial CPU hours [9]. This demonstrates that AI can be embedded directly inside numerical solvers to improve algorithmic decision-making.

3.5. Predictive Failure Management

Long-running simulations on large clusters are vulnerable to node failures that waste accumulated computation [6]. AI-based failure prediction mitigates this risk by training classifiers (e.g., random forests, RNNs) on system logs and hardware sensor data to anticipate which nodes are likely to fail [6]. With sufficient lead time, schedulers can proactively checkpoint jobs or migrate workloads to healthy nodes. A system at the National Center for Supercomputing Applications achieved 90% precision in predicting failures with tens of minutes of warning [6]. AI-driven reliability optimization therefore complements hardware-level fault tolerance mechanisms and boosts overall throughput by preventing unplanned downtime [9].

4. Challenges and Future Directions

Despite the promise of AI in HPC, several challenges must be addressed before these techniques achieve widespread production adoption.

4.1. Integration and Interoperability

HPC applications are typically written in C++/Fortran with MPI for parallelism, whereas AI frameworks rely on Python, TensorFlow, and PyTorch [4]. Bridging these ecosystems requires careful engineering. Frameworks such as SmartSim and DeepDriveMD address this by enabling in-memory data exchange between simulation and AI processes, eliminating costly file-based coupling [7]. Future HPC software stacks will likely embed AI inference runtimes natively, enabling seamless co-execution of simulations and ML models on the same nodes without resource contention [3,9].

4.2. Data Availability and Quality

Deep learning requires large, representative training datasets that are often expensive to generate in HPC contexts, where each data point may require hours of simulation time [2,5]. Online learning (updating models as new data arrives) and transfer learning (adapting pre-trained models to new systems) can reduce this burden [3]. Physics-informed neural networks incorporate governing equations into the training loss,

reducing the labeled-data requirement substantially [7]. Ensuring data diversity is critical: models trained on narrow workload distributions generalize poorly, so continuous data collection and model refresh pipelines will be necessary in production HPC environments [1,6,9].

4.3. Trust, Robustness, and Transparency

Scientific computing demands reproducibility and correctness. AI models, which often behave as black boxes, can be difficult to interpret and validate [1]. Surrogate models must be rigorously tested across the full input distribution to ensure they do not introduce hidden inaccuracies in critical simulation results [8]. RL-based schedulers require fairness audits, since learned policies can inadvertently deprioritize certain job classes [2]. Explainable AI methods, uncertainty quantification, and hybrid physics-ML architectures all contribute to building the trust required for production deployment [7,9].

4.4. Performance Overhead

AI inference and training consume CPU/GPU cycles and memory alongside the primary simulation workload. If overheads are too large, they erode the gains AI is meant to provide [3]. Co-designing HPC and AI algorithms is therefore essential: lightweight surrogate models, asynchronous inference pipelines, and dedicated AI accelerator partitions on HPC nodes can all limit this overhead [1]. Hardware trends favor this direction, as modern supercomputers already ship with GPU accelerators well-suited to both simulation and inference workloads [4].

4.5. Generality vs. Specialization

Most published AI solutions target specific codes or workloads; retraining whenever hardware or application changes limits practical adoption [5,9]. Building generalizable AI frameworks is an active research area: efforts include surrogates designed for broad families of PDEs, RL schedulers that transfer across cluster topologies, and meta-learning methods that adapt quickly to new HPC environments [3,7]. Shared benchmarks and community datasets will accelerate progress toward portable AI tools that HPC centers can adopt without extensive re-engineering [1,6].

4.6. Human Factors and Adoption

The best AI method remains unused if it disrupts established workflows or lacks interpretable outputs that scientists and administrators can trust [2,9]. Successful adoption requires investment in user-facing tooling, documentation, and training. Collaboration between HPC domain experts and AI researchers is essential: domain experts provide physical constraints and validation criteria, while AI researchers contribute algorithmic innovations. Demonstrated pilot successes such as production RL schedulers and accelerated simulation workflows build the organizational confidence needed for broader rollout [3,5].

5. Conclusion

The convergence of AI and High-Performance Computing is transforming large-scale scientific simulation. Machine learning models improve forecasting of performance and resource requirements; deep reinforcement learning agents discover adaptive scheduling strategies that outperform static heuristics [4,5]. Neural network surrogates replace computationally intensive kernels with fast approximations, delivering order-of-magnitude reductions in simulation time [7,8]. In-situ analytics and anomaly detection handle the data volumes produced by modern supercomputers, enabling real-time workflow steering [6,9].

Case studies in job scheduling, computational fluid dynamics, fusion energy, adaptive mesh refinement, and predictive failure management all confirm that AI techniques can deliver meaningful performance gains, reduce costs, and improve resource utilization [5,6,8]. The challenges that remain such as integration complexity, data scarcity, trust, overhead, and generalization are actively being addressed through co-designed algorithms, physics-informed models, and explainability tools [1,2,3,4,7].

Looking ahead, the trajectory points toward self-optimizing HPC platforms that continuously learn from operational telemetry and adapt scheduling, power profiles, and numerical resolution in real time [3]. AI-accelerated simulations that blend physics-informed surrogates with data-driven components will become standard practice, dramatically shortening time-to-solution for grand-challenge science [7,8]. As HPC centers enter the exascale era, the role of AI in managing system complexity and unlocking computational efficiency will only grow enabling scientific discoveries that would be out of reach with hand-tuned methods alone [1,4,9].

References

[1] A. Geist and D. A. Reed, "A survey of high-performance computing scaling challenges," International Journal of High Performance Computing Applications, vol. 31, no. 3, pp. 104–113, 2017.

[2] Q. Wang, H. Zhang, C. Qu, Y. Shen, X. Liu, and J. Li, "RLSchert: An HPC job scheduler using deep reinforcement learning and remaining time prediction," Applied Sciences, vol. 11, no. 20, Article 9448, Oct. 2021.

[3] S. Li, W. Dai, Y. Chen, and B. Liang, "Optimization of high-performance computing job scheduling based on offline reinforcement learning," Applied Sciences, vol. 14, no. 23, Article 11220, Dec. 2023.

[4] G. Fox and S. Jha, "Learning everywhere: A taxonomy for the integration of machine learning and simulations," Computing in Science & Engineering, vol. 22, no. 5, pp. 88–104, 2020.

[5] D. Kochkov, J. A. Smith, A. Alieva, Q. Wang, M. P. Brenner, and S. Hoyer, "Machine learning–accelerated computational fluid dynamics," Proceedings of the National Academy of Sciences, vol. 118, no. 21, e2101784118, 2021.

[6] S. A. Jacobs et al., "Enabling machine learning-ready HPC ensembles with Merlin," Future Generation Computer Systems, vol. 131, pp. 285–296, 2022.

[7] M. F. Kasim et al., "Building high accuracy emulators for scientific simulations with deep neural networks," Machine Learning: Science and Technology, vol. 2, no. 4, 045021, 2021.

[8] H. Wang, J. M. L. Ribeiro, and P. Tiwary, "Machine learning approaches for analyzing and enhancing molecular dynamics simulations," Current Opinion in Structural Biology, vol. 61, pp. 139–145, 2020.

[9] A. Gainaru, F. Cappello, M. Snir, and W. Kramer, "Failure prediction for HPC systems and applications: Current situation and open issues," International Journal of High Performance Computing Applications, vol. 27, no. 3, pp. 273–282, 2013.