Leveraging HPC in University Courses: A Cross-Disciplinary Workforce Development Model

F. Alex Feltus
Clemson University
ffeltus@clemson.edu

Jon C. Calhoun
Clemson University
jonccal@clemson.edu

Rong Ge
Clemson University
rge@clemson.edu

Jill Gemmill
Clemson University
gemmill@clemson.edu

Carlos Toxtli Hernandez
Clemson University
ctoxtli@clemson.edu

Bing Li
Clemson University
bli4@clemson.edu

Todd Allen Price
Clemson University
tprice2@clemson.edu

Justin Talbot
Clemson University
jjtalbo@clemson.edu

MinJae Woo
Clemson University
mwoo@clemson.edu

ABSTRACT

High-performance computing (HPC) is an increasingly essential skill across science, engineering, and data-intensive industries, yet most university students and faculty lack prior HPC experience. This paper presents a cross-disciplinary workforce development model in which nine faculty members at Clemson University integrated the Palmetto2 supercomputer (3.0 PFlops; 54,476 CPU cores; 300+ GPU nodes) into seven courses spanning Bioinformatics and Computational Biology, Business Data Analytics, Computational Chemistry, Computer Science and Parallel Architectures, Automotive Engineering and AI Applications, Healthcare Data Science, and HPC Systems Hardware. A democratized access policy—removing departmental and financial barriers to cluster accounts—enabled broad adoption across student populations ranging from wet-lab biologists to working business professionals. Common pedagogical challenges included HPC onboarding, software dependency management, and large-dataset transfer; shared solutions included priority course scheduling, scaffolded desktop-to-cluster exercises, and modular training materials developed by Clemson's Research Computing and Data Engagement (RCDE) team. Discipline-specific computational profiles varied from large-dataset I/O in bioinformatics and healthcare to GPU-intensive deep learning in automotive AI and intensive floating-point computation in chemistry and parallel architectures. Three transferable principles emerge: democratized access, guaranteed priority scheduling for courses, and domain-contextualized scaffolded exercises. Students completing these courses gain Linux proficiency, job scheduling, parallel programming, and large-scale data analysis skills directly applicable to research and industry careers in an economy increasingly dependent on data-intensive computation.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page, along with the DOI information below. To copy otherwise, republish, post on servers or redistribute to lists requires prior specific permission from the first author.

© 2026 Journal of Computational Science Education
https://doi.org/10.22369/issn.2153-4136/17/2/9

KEYWORDS

High-performance Computing, Workforce Development, Interdisciplinary Education, University Curriculum, Research Computing

1 INTRODUCTION

In the Fall 2025 semester at Clemson University, a diverse group of professors shared their experiences teaching undergraduate and graduate students using a campus cluster called Palmetto2 [3] in their university courses. Course topics included Bioinformatics and Computational Biology, Business Data Analytics, Computational Chemistry, Computer Science and Parallel Architectures, Automotive Engineering and AI Applications, Healthcare Data Science and Public Health. After Clemson's HPC Days workshop, the group saw value in presenting to a wider audience.

In this report, an overview of each course is presented along with a hardware training course called "Constructing HPC Systems: The Hardware Perspective." Then we provide a comparative analysis along with challenges and education solutions that are common for most, if not all disciplines. Finally we discuss workforce development opportunities.

1.1 Current State of HPC Education in Universities

High-performance computing (HPC) has become a strategic resource for scientific discovery, economic competitiveness, and national security, yet its integration into university curricula remains uneven. A survey of 133 domestic and international university HPC centers found that while HPC use in undergraduate education has grown significantly across STEM disciplines, the majority of students (63.5%) and faculty (69.0%) surveyed reported no prior HPC experience [4]. Key barriers include the absence of dedicated curricular materials, insufficient access to HPC hardware, and a lack of technical support staff. HPC-related skills are rarely embedded in standard undergraduate programs, and research opportunities are largely confined to campuses with existing HPC infrastructure [4].

1.2 National HPC Resources

Several United States initiatives have been established to address the HPC access and workforce development gap, providing training

materials, allocations, and programmatic support to researchers and educators across the United States.

1.2.1 National Science Foundation Cyberinfrastructure Worker Training Programs. The National Science Foundation (NSF) Training-based Workforce Development for Advanced Cyberinfrastructure (CyberTraining) program (NSF 23-520) is the flagship federal initiative for cyberinfrastructure (CI) workforce development [6]. The program pursues two primary goals: (i) ensuring broad adoption of CI tools, methods, and resources by the research community to catalyze major research advances; and (ii) integrating core literacy and discipline-appropriate advanced skills in CI and computational methods into undergraduate and graduate educational curricula. In 2021, NSF published the *Blueprint for Cyberinfrastructure Learning and Workforce Development* [7]. The blueprint defines three communities of concern: CI Contributors, CI Professionals, and CI Users and articulates a three-pillar architecture for sustainable CI community building: (1) promoting community vision and organization, (2) fostering sustainability through professional development and career pathways, and (3) measuring and communicating the impact of CI investments. The blueprint calls for curriculum development, career recognition, and the nurturing of regional and domain-centered CI communities.

1.2.2 XSEDE and ACCESS Programs. The Extreme Science and Engineering Discovery Environment (XSEDE), funded by NSF from 2011 to 2022, provided a national network of advanced computing resources, training, and user support to the U.S. research community [8]. XSEDE offered online lectures, workshops, and training distributed across its partner sites, and served as a model for multi-institutional HPC education. In 2022, XSEDE was succeeded by the Advanced Cyberinfrastructure Coordination Ecosystem: Services & Support (ACCESS) [1], which continues to provide NSF-funded advanced computing allocations and support services at no cost to eligible U.S.-based researchers and educators.

1.2.3 The Carpentries HPC Training Materials. HPC Carpentry is a community-driven initiative that develops and delivers open, peer-reviewed training materials for high-performance computing, modeled on the pedagogy of The Carpentries (Software Carpentry, Data Carpentry, Library Carpentry) [5]. The core HPC Carpentry curriculum bridges the gap between laptop-scale computing and supercomputer use through three lessons: the Software Carpentry Unix Shell lesson, an *Introduction to High Performance Computing* lesson covering job scheduling, module systems, file transfer, and parallel execution, and a lesson on HPC Workflow Management using Snakemake. Materials are freely available, instructor-tested, and designed for two-day workshop delivery. The mission of HPC Carpentry is to lower the barrier to entry for HPC operations across a wide range of users, particularly those without formal computational training.

2 PALMETTO2: DEMOCRATIZING HPC AS A TRAINING RESOURCE

2.1 Infrastructure Overview

Named for South Carolina's state tree, the Palmetto2 Cluster was designed to suit many different research applications, with many powerful multi-core nodes, each with a significant amount of memory. Currently, Palmetto2 is comprised of 1202 compute nodes (totaling 54,476 CPU cores), and features:

- over 300 nodes are equipped with 2x or more NVIDIA Tesla GPUs (P100, V100, A100, H100, and H200)
- 20 large-memory nodes with greater than 1.5 TB of memory
- 250 GB home directory space for each user
- 5 TB scratch storage space for each user
- 10 and 25 Gbps Ethernet, and 56, 100, 200 and 400 Gbps Infiniband networks
- benchmarked at 3.0 PFlops

GPU-accelerated nodes are interconnected with high-speed Infiniband networking. Palmetto2 also contains special resources such as large-memory nodes featuring 1.5TB of RAM or more with the largest containing 6TB, and NVIDIA DGX and HGX systems for large-scale AI/ML workloads.

The Indigo Data Lake storage system is a research data repository and processing platform backed by an all-flash scalable storage system provided by VAST Data. Indigo has 5.5PB of data leveraging data reduction and de-duplication technologies. The data lake can support large numbers of simultaneous clients by leveraging an aggregate of 1.6 Tbps of network throughput over InfiniBand and 400 Gbps over ethernet. Indigo is accessible via NFS, SMB, or S3 protocols. Students, faculty, and staff can access their storage via the SMB protocol over the campus network. Faculty can purchase persistent storage on Indigo that also features backups and snapshots. The Palmetto2 HPC cluster is housed at Clemson's Information Technology Center (ITC). The ITC is a 24/7 monitored environment with proper power, cooling, and physical security. The Palmetto Cluster is UPS and generator backed to prevent unexpected interruptions to compute jobs.

2.2 Democratized Access Model and User Experience

All Clemson HPC resources (compute, storage, and staff support) are available across all departments, to all faculty and students. Faculty and graduate students simply need to request an account; undergraduate students are given accounts through faculty request, such as for a course or supervised research work. This democratized access model makes computing available to everyone, from engineering to humanities and the social sciences. User jobs are scheduled on the Slurm scheduler's fair-share algorithm. A shared ownership model (condominium model) is available for faculty who require on-demand compute resources. Compute node owners have priority access to queue jobs on their nodes, and other users on the cluster may take advantage of idle time on the node to run their jobs.

Courses using HPC are assigned priority scheduling, so that resources are guaranteed to be available during class hours if needed,

and the user support and training infrastructure provides the assistance users need to become proficient in their computing needs. This has been especially important for disciplines that do not typically include training in computing as part of their core curriculum. Integration with Clemson's Canvas course management system vastly simplifies account management for these students.

2.3 Clemson HPC Workshop Training Series

The Research Computing and Data Engagement (RCDE) team at Clemson University provides training for the HPC community. Most workshops are available as an online course, which supplements live training sessions. The online course allows people to review the materials at their own pace. Course topics include introductory materials (Intro to Linux, Intro to HPC); advanced HPC topics (Containers for HPC; Kubernetes; parallel methods); materials simulation codes (LAMMPS, VASP); software development (Git; CI/CD; AI code assistance); Python (Spark; Deep Learning; Machine Learning; Visualization); R (programming, Machine Learning/caret); Nextflow workflow manager; and Deep Learning workshops (running and fine-tuning LLMs on Palmetto; PyTorch; RAGs; Attention & Transformers).

3 INTERDISCIPLINARY HPC IMPLEMENTATION: DISCIPLINE-SPECIFIC UNIVERSITY COURSES

3.1 Bioinformatics and Computational Biology

3.1.1 Course Structure and HPC Integration. The Genetics and Biochemistry department at Clemson University offers multiple bioinformatics courses that demonstrates a scalable, multi-level approach to integrating HPC into life sciences education using the Palmetto2 HPC cluster. *Advanced Medical Bioinformatics* (GEN 8450) is a graduate course, delivered in a hybrid format over 15 weeks to 10-20 students, emphasizing applied genomics workflows for translational research applications. *Foundations of Genetics and Bioinformatics* (GEN 8410) provides an intensive 5-week summer foundation course for 10 students in a fully asynchronous online format and establishes core computational biology competencies. A *Bioinformatics* course (BIOCH/GEN 4400/6400) accommodates 100-150 undergraduate and graduate students annually in a hybrid self-paced format, serving as a workforce development pipeline for computational biology skills across multiple disciplines.

3.1.2 Learning Objectives and HPC Skill Development. All three courses utilize Palmetto2's computational resources to achieve progressive learning objectives spanning theoretical knowledge and hands-on technical skills. Students master Linux command-line operations for bioinformatics workflows, develop proficiency in Python and R programming for genomic data analysis, and learn to navigate high-performance computing environments including Slurm job scheduling, resource allocation, and parallel processing concepts.

The curriculum emphasizes real-world dataset manipulation using major biological databases (SRA, ENSEMBL, GTEx, NCI GDC), implementation of sequence alignment algorithms, and systems biology approaches for network construction and pathway analysis. Students engage in biomarker discovery workflows and population genetics analyses requiring substantial computational resources unavailable on personal workstations.

3.1.3 Educational Outcomes and Workforce Impact. The integration of Palmetto2 infrastructure enables students to tackle authentic research problems with real-world computational complexity, bridging the gap between classroom theory and professional practice. Students demonstrate measurable competency gains in computational thinking, scalable data analysis, and interdisciplinary problem-solving essential for modern life sciences careers.

The complexity across the three courses—from 10-student intensive workshops to 150-student large-enrollment implementations—establishes a replicable model for HPC integration that addresses diverse institutional needs and student populations. Assessment outcomes show consistent achievement of technical proficiency in genome-scale data processing, statistical analysis programming, and collaborative computational workflows that directly translate to research productivity and industry readiness in bioinformatics, biotechnology, and healthcare sectors. This multi-tiered approach demonstrates how HPC resources can be effectively scaled from specialized graduate training through broad undergraduate workforce development, providing a sustainable model for other institutions seeking to integrate computational infrastructure into life sciences curricula.

3.1.4 Example Computational Exercise: Sequence Alignment on Palmetto2. Students complete a comprehensive hands-on laboratory exercise demonstrating core computational biology workflows on high-performance computing infrastructure [2]. The exercise encompasses ten sequential steps that progressively build HPC competency: (1) secure cluster access and authentication, (2) interactive resource allocation using Slurm job scheduling, (3) Jupyter interface navigation for terminal access, (4) scratch filesystem utilization for temporary storage, (5) conda environment management for bioinformatics software installation, (6) large-scale genomic dataset acquisition from public repositories, (7) file compression and decompression operations, (8) BLAST database construction using `makeblastdb`, (9) comparative sequence alignment execution using both global (Needleman-Wunsch) and local (Smith-Waterman) algorithms, and (10) quantitative analysis of alignment results including statistical significance assessment.

This exercise directly addresses authentic research scenarios by utilizing human genome and cDNA datasets, requiring computational resources exceeding typical desktop capabilities. Students gain practical experience with Linux command-line operations, bioinformatics software deployment, parallel processing concepts, and large-scale data managemen that are essential skills for modern genomics research. The progressive complexity from basic cluster access to advanced alignment analysis demonstrates how HPC infrastructure enables sophisticated biological analyses previously accessible only to specialized research groups, effectively democratizing computational biology education across diverse student populations.

3.2 Business Data Analytics

3.2.1 Course Structure and HPC Integration. Effective management in the information age relies on the proper use and interpretation of data. Virtually every functional area of management relies on the statistical treatment of data and information. Knowing when and how to use statistics as a tool to make decisions in an uncertain business environment is an essential skill of a successful manager. *Business Data Analytics and Statistical Computing* (MBA 8040) teaches modern statistical inference in business operations. Topics include testing statistical hypotheses, consequences of making decisions with incomplete information, univariate and multivariate regression with emphasis on business applications and design of experiments and analysis of variance. Special attention is given to efficient and relevant data collection and interpretation.

3.2.2 Learning Objectives and HPC Skill Development. The objectives of this 3 credit course are to present the most important statistical techniques and illustrate their applications. Emphasis is placed on appropriate formation of problems, proper choice of statistical techniques, and on effective interpretation and communication of results. Conceptual understanding is facilitated by the use of technology to show how each of the statistical techniques can be implemented in spreadsheet (MS Excel) or HPC environments. Using Palmetto2, students install R and Python working environments, and explore calculation structures. By the end of the course, students recognize how to deal with situations that involve uncertainty using local and HPC technologies as well as understand how to apply and to interpret regression analysis and other managerially important analytical models.

3.2.3 Educational Outcomes and Workforce Impact. This class provides an entrance into HPC where most students do not have not have any experience. By allowing students to see what could be done with compute capabilities, they can then use HPC not only for their assignments, but also discuss with their current employers on how HPC can benefit their current work areas.

3.2.4 Example Computational Exercise. This course gives students access to Python and R notebook environments where they can apply statistical methodologies to business cases with provided datasets for business decision-making. An example teaching workflow is to (1) Access Palmetto2 using Open OnDemand, (3) open a terminal application, (3) create a working directory in scratch space, (4) install Python, (5) download case datasets into working directory on scratch, (6) uncompress data files, and (7) apply numpy and other libraries to analyze the dataset.

3.3 Computational Chemistry

3.3.1 Course Structure and HPC Integration. Computational Chemistry (CHE 8380) is a mixed lecture/lab course that introduces graduate students to various aspects of computational chemistry including fundamental computational techniques such as Hartree-Fock and density functional theory, energy decomposition analysis, ab initio molecular dynamics, and computational thermochemistry and spectroscopy. The course is explicitly designed to train students in the use of HPC resources and architectures, including remote environments, Linux-based workflows, batch scheduling, resource allocation, and basic concepts of scalability. While representative calculations in laboratory activities are executed on students' personal computers, the exercises are framed as small-scale analogs of workflows that are later deployed on HPC systems. The final project consists of a student-led computational project carried out entirely using HPC resources in which an insightful chemical question is investigated. The final project is intended to simulate a graduate-level research environment in computational chemistry.

3.3.2 Learning Objectives and HPC Skill Development. The primary learning objective of CHE 8380 is to equip students with the tools needed to translate abstract chemical concepts and hypotheses into computable quantities. Students develop a practical competency in HPC-based scientific computing and learn to interact with Linux-based systems, prepare and submit batch jobs, manage computational output, and construct reproducible computational workflows. The computational skills gained in this course are introduced incrementally through a progression of in-class lab activities which emphasize the scientific method: students learn how to formulate chemical questions, construct physical models to gain relevant insights, and implement those models computationally, while maintaining a conceptual understanding of the underlying mathematical formalism.

3.3.3 Educational Outcomes and Workforce Impact. Students completing the course demonstrate the ability to translate chemical research questions into HPC workflows. In addition to domain-specific knowledge in electronic structure theory and molecular modeling, students gain experience with technical practices common in HPC environments, including remote job execution, file system management, and post-processing of large datasets. The course therefore primarily prepares students to engage productively with HPC resources in graduate research, although the skills obtained are applicable to employment in industry.

3.3.4 Example Computational Exercise: Potential Energy Surfaces. An example of how HPC-oriented workflows are introduced at smaller scale: students complete a laboratory exercise in which they construct small molecules and perform electronic structure scans of potential energy surfaces by varying a single internal coordinate. These calculations are executed on a personal computer, but the workflow mirrors that used in HPC environments: students generate multiple independent calculations, organize a set of output files, and analyze trends in energy and molecular properties. This exercise prepares students for the final HPC project, where similar analyses are performed at a greater computational scale.

3.4 Computer Science and Parallel Architectures

3.4.1 Course Structure and HPC Integration. Parallel Architectures (CPSC8200) is a graduate-level course that focuses on the core elements of high-performance computing. Parallel computing has long been indispensable, enabling scientific discovery and engineering innovation, and it is also critical to the rapidly growing field of deep learning and its applications. Parallel computer architectures are ubiquitous today, spanning systems from mobile devices (e.g., smartphones and laptops) to large-scale data centers. The course covers modern parallel computer architectures, including multicore and manycore CPUs and GPUs, networking, system software,

parallel programming paradigms and libraries, algorithm design, and the measurement, analysis, evaluation, and optimization of performance and scalability.

3.4.2 Learning Objectives and HPC Skill Development. Programming assignments and course projects are performed on the Palmetto2 cluster. Upon completion of the course, students master the principles and design methods for parallel system from hardware to applications across the software stack. They become familiar with state-of-the-art computer architecture, systems, and applications, and they understand the technical details of software for parallel computing. Students can apply a range of parallel programming models—including SIMD, OpenMP, GPU programming with CUDA, and the Message Passing Interface (MPI)—to solve important problems. They also learn performance-analysis methodologies such as Amdahl's law, strong scaling, and weak scaling, as well as tools such as hardware performance counters, Nsight Systems (nsys), and Nsight Compute (ncu), to profile and evaluate performance and scalability and to identify inefficiencies and bottlenecks. In practice, they gain experience with batch scheduling systems to request resources and to submit interactive and batch jobs on compute clusters.

3.4.3 Educational Outcomes and Workforce Impact. By the end of CPSC 8200, students demonstrate practical readiness for advanced research and professional practice in parallel and distributed systems and high-performance computing. Because all assignments and projects are executed on the Palmetto2 cluster, students gain hands-on experience with realistic constraints—resource allocation, queueing policies, job scheduling, and performance variability—mirroring both academic and industry HPC environments.

The training directly supports workforce development by preparing students for research careers (MS, PhD) in parallel and distributed systems, HPC architecture and software, and performance analysis. They also prepare students for industry roles that demand strong systems and performance expertise, including *performance engineer*, *HPC/software engineer*, *systems engineer*, *GPU/CUDA engineer*, and *computational scientist*. Graduates leave the course with demonstrable skills in parallel programming, profiling and optimization, scalability analysis, and cluster-based execution—competencies that are increasingly essential across scientific computing, engineering simulation, and large-scale machine learning.

3.4.4 Example Computational Exercise: Accelerating a Matrix-Multiply Kernel on GPUs. Students are provided with a naïve GPU implementation of matrix multiplication, along with relevant technical papers, and are asked to apply multiple optimization techniques to accelerate execution. Students submit the optimized source code and a report describing the techniques used, the corresponding speedups, and the cumulative speedup achieved. The final performance target is to be comparable to an optimized CUDA library call (e.g., cuBLAS). A successful solution requires exposing sufficient parallelism, choosing an effective thread/block organization, making optimal use of shared memory, coalescing global-memory accesses, and managing register usage. Students also use profiling tools to identify performance bottlenecks—such as high cache-miss rates, low occupancy, and memory stalls—and to guide further optimization.

3.5 Automotive Engineering and AI Applications

3.5.1 Course Structure and HPC Integration. With continued advances in computing capabilities and increasing automotive system complexity, the Department of Automotive Engineering incorporates Clemson's Palmetto2 Cluster platform for BS, MS, and PhD program curricula at Clemson University International Center for Automotive Research (CU-ICAR)

Machine Perception and Intelligence (AUE 8200) is a graduate-level course designed to introduce students in automotive engineering, rather than computing backgrounds to modern perception and AI methods for intelligent vehicles and manufacturing systems. The course is structured in three coherent themes: sensing, 2D and 3D machine perception and AI methods, and covers physical principles of sensing, geometric representations and data-driven perception approaches for automotive and robot systems. HPC resources are incorporated throughout live lab sessions, homework assignments and course projects to support computationally intensive perception workflows, such as AI model training and inference, sensor data processing, and simulation data generation.

By working with real-world automotive and robotic datasets such as KITTI, Waymo, and nuScenes, students develop and evaluate perception pipelines that closely align with current research and industry practice. Grounded in problem-driven pedagogy, the course integrates multi-modal sensing, geometric learning, AI training and inference, and edge-AI deployment with hands-on project development, visualization and validation to support student experiential learning (ExL) cycle.

3.5.2 Learning Objectives and HPC Skill Development. The learning objectives of this course emphasize the development of applied AI and analytical skills required for modern automotive and machine perception applications. Students are trained to formulate sensing and machine vision problems, select baseline models, design and validate AI-based perception methods leveraging HPC resources. Students develop applied HPC skills such as Linux-based system usage, Bash commands and coding, and Python programming. The course introduces deep learning concepts and model development using PyTorch, with an emphasis on training and inference for perceptive sensor data recognition. Students implement Python-based workflows for processing multimodal sensor data such as radar, 2D and 3D camera and LiDAR data. Widely adopted libraries for Python, 2D and 3D processing and visualization such as NumPy, Matplotlib, scikit-image, OpenCV, and Open3D are used to support these workflows. Students learn and consolidate these skills through course projects involving the design, implementation, and evaluation of AI-enabled perception methods using real-world datasets and HPC platforms.

3.5.3 Educational Outcomes and Workforce Impact. The educational outcomes are closely aligned with the technical and professional workforce training needs in the automotive and intelligent systems fields. Through experiential and project-based learning, students will develop the ability to integrate sensing, geometric perception, and AI methods under practical computational constraints. Students will gain hands-on experience using GPU-accelerated HPC environments, learning how to interpret complex sensor data and

evaluate data-driven perception methods. In addition to technical skills, the course also emphasizes professional engineering competencies, including teamwork, technical communication, and project report writing. These outcomes will enable students to effectively participate in interdisciplinary teams in autonomous vehicles, robotics in smart manufacturing and data-driven engineering problems, where scalable computing, data-centric reasoning and system-level understanding are essential.

3.5.4 Example Computational Exercise: Machine Visual Recognition. A typical machine vision exercise in the course involves the development, training and evaluation of a convolutional neural network (CNN) using PyTorch on the CIFAR-10 dataset. CIFAR-10 is a widely used benchmark dataset for image classification, comprising ten semantic categories: {automobile, truck, ship, airplane, bird, cat, deer, dog, frog, horseİn this exercise, students first examine the dataset structure and perform basic data preprocessing and visualization using Python-based tools. They then fine-tune or design a CNN architecture, implement it within the PyTorch framework, and train the model using GPU-accelerated HPC resources, employing cross-entropy loss in conjunction with a softmax non-linear function at the output layer. Model performance is evaluated using classification accuracy, confusion matrices, and loss convergence. Students iteratively refine network architectures and training parameters to improve performance. Through this exercise, students gain hands-on experience with deep learning–based visual recognition workflows, model evaluation, and scalable HPC practices relevant to machine perception and intelligence applications.

3.6 Healthcare Data Science and Public Health

3.6.1 Course Structure and HPC Integration. Instruction of healthcare data science is implemented through *Predictive Analytics in Healthcare* (HLTH 8900) , a graduate-level course serving doctoral students from Applied Health Research and Evaluation (AHRE) and related public health programs at Clemson University. The course was designed for health science students with no prior programming or computing background and delivered over a full semester with an application-driven structure. Instruction progressed from foundational Python programming to the development, evaluation, and interpretation of machine-learning models applied to real clinical and public health datasets. HPC resources were integrated through the Palmetto2 cluster to support scalable model training, reproducible experimentation, and exposure to research-grade computational environments. Rather than positioning HPC as a standalone technical topic, cluster resources were embedded into the analytic workflow once students moved beyond toy datasets. As project complexity increased, Palmetto2 enabled students to work with large national datasets, ensemble learning methods, and repeated model comparisons that exceed the limits of personal laptops. This approach ensured that HPC usage was introduced as a practical necessity tied directly to analytic objectives rather than as an abstract infrastructure concept.

3.6.2 Learning Objectives and HPC Skill Development. The primary learning objective of HLTH 8900 is to equip public health students with the ability to design, implement, and critically interpret predictive models in healthcare contexts. Students developed proficiency in Python-based data science workflows, including data cleaning, feature engineering, statistical modeling, and machine-learning pipelines using libraries such as pandas, scikit-learn, statsmodels, and XGBoost. HPC skill development focused on applied competency rather than systems-level optimization. Students learned to execute reproducible analyses in shared computing environments, manage computational experiments at scale, and reason about tradeoffs between model complexity, interpretability, and computational cost. Exposure to Palmetto2 enabled parallel model training, repeated cross-validation, and survival modeling workflows that would be impractical on local machines. Importantly, students learned to align computational choices with clinical and public health constraints, such as data availability, contextual missingness, and interpretability requirements.

3.6.3 Educational Outcomes and Workforce Impact. Student outcomes demonstrated that health science trainees can rapidly acquire advanced data science competencies when instruction is grounded in domain-relevant problems and supported by scalable computing infrastructure. Final projects and posters showed strong technical execution, including ensemble classification, survival analysis, and model comparison, along with sophisticated contextual reasoning about clinical workflows, population health implications, and ethical considerations. Projects included cardiovascular disease risk prediction, diabetes classification using national surveillance data, and cancer survival modeling using both traditional statistical methods and modern machine learning approaches. Across these domains, students demonstrated the ability to balance predictive performance with parsimony, interpret model outputs in clinically meaningful terms, and articulate how predictive tools could inform screening, prevention, and intervention strategies. From a workforce development perspective, this implementation highlights how HPC can lower barriers for non-traditional computing students entering health analytics roles. Graduates leave the course prepared to collaborate effectively with data scientists, informatics teams, and clinicians, with practical experience executing scalable analyses in shared computing environments.

3.6.4 Example Computational Exercise: Predictive Modeling for Population Health Outcomes. A representative computational exercise focused on developing and evaluating predictive models for chronic disease risk using large-scale public health data. Students worked with the CDC Behavioral Risk Factor Surveillance System (BRFSS) dataset, comprising over 70,000 records and multiple demographic, socioeconomic, and health status variables. Using Palmetto2 resources, students implemented logistic regression, random forest, and XGBoost models, conducted systematic train-test splits, and evaluated performance using accuracy, AUC, and various feature importance analyses. The exercise emphasized iterative experimentation and interpretability. Students compared full feature models with reduced variable models to assess how data availability constraints affect predictive performance, a common challenge in real world public health settings. Students concluded by translating model findings into population level insights, such as identifying actionable versus non-modifiable risk factors and discussing how

predictive analytics could support scalable screening and prevention strategies in the field.

3.7 Constructing HPC Systems: The Hardware Perspective

3.7.1 Course Structure and HPC Integration. *Creative Inquiry: High-Performance Cluster Computing* (ECE 4990) is designed to be an HPC introductory course approachable to freshman, including those with limited or no programming/technical experience. Each student in the course is provided with a *Cluster Kit* which contains all of the supplies necessary to construct a four node Raspberry Pi cluster. Over the course of the semester the students follow a provided tutorial walking them through cluster assembly and installation of the key cluster software. This culminates in the student benchmarking their cluster using the LINPACK benchmark (HPL), the same benchmark used in the Top 500 List[1]. As students work their way through the tutorial, direct comparisons are made between the supplied hardware and that on large-scale systems such as Palmetto2. The *why* and *what* each software package and terminal command does are presented using high-level explanations and layman real-world analogies. Each year, the six best students in the course write and submit an application in collaboration with a hardware vendor to the Student Cluster Competition[2] held each November at Supercomputing[3]. The vendor supplies the team a cluster, allowing them to repeat the Raspberry Pi assignment, but with production-level hardware. During the competition, they team must complete a series of challenges such running benchmarks, running scientific applications, and reproducing results from a paper selected from the prior year's Supercomputing conference.

3.7.2 Learning Objectives and HPC Skill Development. Successfully completing this course provides students with a strong foundation for working with HPC systems. This enables them to translate that knowledge into their discipline and answer *What is HPC?* and *Why is HPC important to my domain and society at large?* Furthermore, students are able to list key hardware elements inside a node/rack and describe their role. Through benchmarking their Raspberry Pi clusters, they are able to understand and quantify application speedup and parallel efficiency. Finally, students are able to identify well-known HPC software in their domain.

3.7.3 Educational Outcomes and Workforce Impact. Most students who complete this course go on to enroll in graduate school. For those who compete in the Student Cluster Competition, many companies and organizations approach them about internship and job opportunities. Multiple students have obtained internships at United States Department of Energy laboratories.

3.7.4 Example Computational Exercise: Raspberry Pi Cluster Benchmarking. Students benchmark their Raspberry Pi clusters using the LINPACK benchmark (HPL) to evaluate floating-point performance and to compute speedup as they use additional cores on their cluster. Through repeated testing, the students gain hands-on experience tuning HPL parameters — e.g., process grid dimensions, problem size — to understanding how performance changes with additional resources. Two key metrics (speedup and efficiency) are introduced. Students compete among themselves to obtain the highest HPL score. The students' performance results are extrapolated to large process sizes to introduce concepts such as over-provisioning, queue wait time, and total time-to-solution.

[1] https://top500.org/
[2] https://www.studentclustercompetition.us/
[3] https://supercomputing.org/

4 COMPARATIVE ANALYSIS: CROSS-DISCIPLINARY CHALLENGES AND SOLUTIONS

Across seven disciplines, several recurring patterns emerged in how faculty integrated Palmetto2 into their courses. Despite differences in domain, student background, and computational scale, some challenges and solutions were consistent.

4.1 Common Technical Challenges

The most universal barrier was *onboarding*: configuring accounts, navigating the Linux command line, file ingress & egress, and submitting Slurm jobs for the first time. Students in MBA 8040, with little prior computing experience, faced the steepest initial learning curve, while even CS graduate students in CPSC 8200 required orientation to cluster-specific workflows. Across all courses, instructors reported that the first one to two weeks were dominated by environment setup rather than domain content.

Software dependency management was a second shared challenge. Each discipline relies on a distinct software ecosystem—bioinformatics tools (BLAST, conda environments), quantum chemistry codes (Gaussian, ORCA), deep learning frameworks (PyTorch, TensorFlow), and statistical packages (R, Python)—requiring reproducible, containerized, or module-based deployment strategies to ensure consistent student environments.

Data transfer and storage management presented practical friction, particularly in bioinformatics (large genomic datasets from NCBI and ENSEMBL) and automotive AI (multi-gigabyte KITTI and Waymo sensor datasets). Priority scheduling for course allocations, provided by RCDE (the university provided "humanware") team), mitigated resource contention during class hours and was identified by multiple instructors as critical to pedagogical success.

4.2 Discipline-Specific Computational Requirements

Table 1 summarizes the primary computational profile of each course. Bioinformatics and healthcare data science are characterized by large dataset I/O and moderate parallelism; computational chemistry and parallel architectures demand intensive floating-point computation and strong scaling; automotive AI and computer science require GPU acceleration; and business analytics prioritizes interactive notebook environments over raw throughput.

4.3 Student Background and Preparation Requirements

Student preparation varied widely. CPSC 8200 students typically hold CS bachelor's degrees and arrive with C-family and Linux familiarity. In contrast, MBA 8040 students are working professionals with no prior programming experience; the primary goal is computational exposure rather than proficiency. GEN 4400/6400

Table 1: Computational Profile by Discipline

Discipline	Primary Resource	Key Software
Bioinformatics	CPU + large storage	BLAST, conda, R/Python
Business Analytics	Interactive CPU	R, Python notebooks
Comp. Chemistry	CPU (multi-core)	Gaussian, ORCA
Computer Science	CPU + GPU (CUDA/MPI)	OpenMP, CUDA, MPI
Automotive AI	GPU (deep learning)	PyTorch, KITTI/Waymo
Healthcare Data Sci.	CPU + large storage	Python, R, SQL
HPC Systems (ECE)	Raspberry Pi cluster	HPL/LINPACK

students span a continuum from wet-lab biologists to computational scientists, requiring flexible, self-paced course designs. This diversity underscores the need for modular onboarding materials—such as the Clemson HPC Workshop Series—that can be tailored to discipline-specific entry points.

A common pedagogical solution across courses was the use of *scaffolded exercises*: small-scale desktop analogs that mirror HPC workflows before students transition to cluster execution. This approach, used explicitly in CHE 8380 and ECE 4990, reduced cognitive load and improved student confidence prior to full HPC deployment.

5 CONCLUSION

This paper presents a cross-disciplinary model for integrating HPC into university curricula, grounded in the shared experience of seven courses at Clemson University using the Palmetto2 HPC cluster The disciplines span life sciences, business, chemistry, computer science, engineering, and public health—demonstrating that HPC is not a niche resource but a broadly applicable educational infrastructure.

Three principles emerge as transferable to other institutions. First, *democratized access*—removing departmental and financial barriers to cluster accounts—is a prerequisite for broad adoption. Second, *priority scheduling for courses* ensures pedagogical reliability and instructor confidence. Third, *scaffolded, domain-contextualized exercises* are more effective than generic HPC tutorials for motivating diverse student populations.

The workforce development impact is substantial. Students completing these courses gain computational skills—Linux proficiency, job scheduling, parallel programming, large-scale data analysis—that are directly transferable to research careers and industry roles in an economy increasingly dependent on data-intensive computation [4, 7]. The Clemson model, supported by the RCDE team's training infrastructure and Palmetto2's democratized access policy, offers a replicable framework for institutions seeking to address the documented HPC skills gap [4].

Future work will focus on quantitative assessment of student learning outcomes across disciplines, expansion to humanities and social sciences, and development of open-source, discipline-specific HPC teaching modules for community adoption.

ACKNOWLEDGEMENTS

The authors thank the Clemson University Research Computing and Data Engagement (RCDE) team and the Palmetto2 support staff for their essential contributions to course delivery and student support. We also thank the students whose engagement and feedback continue to shape these courses.

REFERENCES

[1] ACCESS-CI. 2026. Advanced Cyberinfrastructure Coordination Ecosystem: Services & Support (ACCESS). https://access-ci.org. NSF-funded successor to XSEDE, providing advanced computing allocations and support services to U.S. researchers and educators.

[2] F. Alex Feltus. [n. d.]. Sequence Alignment Methods. https://github.com/feltus/Sequence_Alignment_Methods Accessed: November 9, 2026.

[3] Jill Gemmill. [n. d.]. Palmetto 2 Documentation. https://docs.rcd.clemson.edu/palmetto/ Accessed: November 9, 2026.

[4] Jordan Hebert, Ryan Hratisch, Rahul Gomes, William Kunkel, Daniel Marshall, Abhimanyu Ghosh, Isabella Doss, Ying Ma, Derek Stedman, Blake Stinson, et al. 2024. High-performance computing in undergraduate education at primarily undergraduate institutions in Wisconsin: Progress, challenges, and opportunities. *Education and Information Technologies* 29, 14 (2024), 18451–18475.

[5] HPC Carpentry Community. 2026. HPC Carpentry: Open Training Materials for High-Performance Computing. https://www.hpc-carpentry.org. Community-driven open curriculum for HPC training, modeled on The Carpentries pedagogy.

[6] National Science Foundation. 2023. Training-based Workforce Development for Advanced Cyberinfrastructure (CyberTraining), NSF 23-520. https://www.nsf.gov/funding/opportunities/training-based-workforce-development-advanced/nsf23-520/solicitation.

[7] National Science Foundation, Office of Advanced Cyberinfrastructure. 2021. *Blueprint for Cyberinfrastructure Learning and Workforce Development.* Technical Report. National Science Foundation. https://nsf-gov-resources.nsf.gov/files/CI-LWD%202.pdf Fifth in the NSF OAC strategic vision series.

[8] John Towns, Timothy Cockerill, Maytal Dahan, Ian Foster, Kelly Gaither, Andrew Grimshaw, Victor Hazlewood, Scott Lathrop, Dave Lifka, Gregory D. Peterson, Ralph Roskies, J. Ray Scott, and Nancy Wilkins-Diehr. 2014. XSEDE: Accelerating Scientific Discovery. *Computing in Science & Engineering* 16, 5 (2014), 62–74. https://doi.org/10.1109/MCSE.2014.80