Course Information

Scientific and technical computing covers an increasingly wide range of disciplines. Traditionally dominated by the solution of ordinary and partial differential equations, linear algebra, and optimization, it now also encompasses many other analysis domains including informatics, pattern matching and large-scale data analysis, image processing, statistical uncertainty quantification and more.

To effectively leverage available scientific/high-performance computing (HPC) hardware, the modern computational scientist must have a practical understanding of underlying HPC architecture design, operating system tools and behavior, scientific programming, compiler transformations, performance optimization, and basic software engineering and verification procedures. This course provides an introductory perspective on these topics, with the goal of enabling participants to leverage Unix workstations and supercomputers effectively for developing scientific research applications.

We begin with an introduction to the traditional mainstays of scientific and technical computing: computer architecture and hardware principles, followed by operating systems and environments. These topics will be covered in the context of computational science, so much attention will be paid to floating point arithmetic and tools for developing and managing scientific code (including source code control and build systems). We then discuss the most common program languages in technical computing, including common data structures and interoperability issues. Finally, we discuss practical tips for developing effective, high-performance scientific technical applications and the associated research documentation required.

Note that this course is designed to provide practical exposure to scientific computing in order to develop a minimum research computing skill-set, and as such, hands-on computer exercises and individual programming assignments are a fundamental learning component of the class.

Lecture Times

Lectures will be held Tuesday and Thursday, 12:30pm-2:00pm, RLM 6.112

Required Labs or Discussion Sections

None

Course Prerequisites

Students must have basic Unix/Linux command-line experience, prior programming experience in C/C++ or Fortran, mathematics through differential equations, and numerical analysis (e.g. CSE 383C). Experience with developing scientific codes is helpful but not required.
Course Topics

- **Hardware principles**: processors, caches, memory, I/O, networking principles, binary representations of numbers, round-off errors, integer and floating point computations.

- **Programming Languages**: common scientific computing languages (C/C++, F90): memory allocation/management, array operations, and common data structures. Will also discuss interoperability issues for codes using routines from multiple languages.

- **Operating Systems and Unix Environments**: features of UNIX/Linux for scientific and technical computing; languages, compilers, debuggers, performance tools, make files, build systems, shell scripting, file management, source code control.

- **Performance Programming and Debugging**: best practices for developing, testing and debugging codes; performance measurement and code optimization.

- **Research Documentation and Simple Data Visualization**: tools for generating research and code documentation: \LaTeX, Doxygen, plotting tools.

- **Software Best Practices**: software design cycle, regression testing, defensive programming, verification, code coverage.

- **Scientific Libraries**: availability of common math libraries and usage for scientific computing.

Course Policies

**Grading Policy**

There will be one midterm exam, periodic homework assignments, and two programming projects with in-class presentations and reports to document the implementations and runtime results. Grades will be based on the midterm exam (30%), periodic homework assignments (30%), and two programming projects (20% each).

**Computing Resources**

Assignments will be performed on ICES Unix workstations and one or more of TACCs high performance computing resources. Account information will be handed out during the first week of class. Students will use their own PCs/Macs/workstations to access the machines through SSH. Note that free SSH clients for PCs and Macs are available through BevoWare.

**Incomplete Grade Policy**

As per university policy, incomplete grades will be granted only for work unavoidably missed at the semesters end and only if 70% of the course work has been completed. An incomplete grade must be resolved within eight (8) weeks from the first day of the subsequent long semester. If the required work to complete the course and to remove the incomplete grade is not submitted by the specified deadline, the incomplete grade is changed automatically to a grade of F.
Class Attendance
Students are highly encouraged to attend class and ask questions.

Instructors
This class is co-taught by Dr. Karl W. Schulz and Dr. Chris Simmons. Both instructors are active researchers in the field of high-performance computing with significant experience in developing parallel codes for scientific and engineering applications, and in training users of supercomputing resources.

Contact Information and Office Hours
Karl W. Schulz
Email: karl@tacc.utexas.edu
Phone: 512-475-9411
Office Hours: ACES 6.332 after class, by appointment.

Chris Simmons
Email: csim@ices.utexas.edu
Phone: 512-232-2881
Office Hours: ACES 6.340 after class, by appointment.

Please address all inquiries to both instructors, and include the course # in the subject line.

University Policies

Disability Statement
The University of Texas at Austin provides upon request appropriate academic accommodations for qualified students with disabilities. For more information, contact Services for Students with Disabilities at 471-6259 (voice) or 232-2937 (video phone).

University of Texas Honor Code
The core values of The University of Texas at Austin are learning, discovery, freedom, leadership, individual opportunity, and responsibility. Each member of the university is expected to uphold these values through integrity, honesty, trust, fairness, and respect toward peers and community.