Tools and Techniques for Computational Science

CSE 380, Unique #65000
Course Syllabus, Fall 2024

Lecture Times

Lectures will be held (in person) Tuesday and Thursday, 12:30-2:00pm in PMA 5.112.

Instructor

Dr. Umberto Villa
Oden Institute for Computational Engineering and Sciences
Email: [redacted]
Office: POB 4.252
Office Hours: Tue 2:30-3:30pm, or by appointment.

Teaching Assistant

Mr Jenil Shah
Oden Institute for Computational Engineering and Sciences
Email: [redacted]
Office: POB 3rd floor, Desk 3SWo2C
Office Hours: TBD (by **appointment only**).

Email (either directly or via Canvas) is the best way to reach us – if corresponding by email, please include **CSE 380** in the subject line.

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 Linux 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 also 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.

Required Labs or Discussion Sections

None

Course Prerequisites

Students must have prior programming experience in C/C++ or Fortran and mathematics through differential equations. Experience with Linux command line interface (CLI) and the development of scientific codes is helpful but not required.

Course Topics

- Hardware principles: processors, caches, memory, I/O, networking principles, parallel architectures.
- Operating Systems and Unix Environments: Linux operating system, interacting with the shell, common command-line utilities, file management, shell scripting.
- Resource Management Systems on HPC Resources.
- Compiler Tools: invocation, common flags, static vs. dynamic linking, optimization.
- Software Build Systems and Revision Control: git, make, autotools, cmake.
- Performance Programming and Debugging: best practices for developing, testing and debugging codes; performance measurement and code optimization.
- Method of Manufactured Solutions: techniques for verifying scientific software.
- Virtualization and Containerization: overview of virtualization options, creation of Docker containers, execution on HPC systems.
- Software Engineering Best Practices: software design cycle, unit testing, regression testing, defensive programming, verification, code coverage.
- Scientific Libraries: availability and usage of common math and I/O libraries for scientific computing.
- Extra topics (if time allows): Introduction to GPU computing, Python bindings

Course Policies

Grading Policy

There will be four homework assignments throughout the semester, one personal website construction, and one small-groups (3 or 4 students per group) programming final project. The programming project will be divided into three milestones: a project proposal (milestone #1), in-class presentation (milestone #2) and report (milestone #3) to document the implementations and runtime results. All students in the same group will receive the same grade for the final project and presentation, provided they equally contributed to the project (See **Collaboration** section).

Final grades will be a weighted average of homeworks, personal website, and final project grades as follows

Task	**Percentage of final grade**
Homework assignments	60% (15% for each assignment)
Personal webpage	5%
Project proposal (milestone #1)	5%
Project presentation (milestone #2)	10%
Project report (milestone #3)	20%

Letter grades will be assigned as follows

Range	**Grade**	**Range**	**Grade**	**Range**	**Grade**
94 or above:	A	82–86:	B	65–70	C
90–94:	A-	76–82:	B-	60–65	C-
86–90:	B+	70–76	C+	Below 60:	D/F

Students taking the class on a Credit/No-Credit basis must maintain a 65% or higher average on the assignments to receive credit.

Late Assignment Deductions

Deadline extension requests can be granted on a case-by-case bases (e.g. conference travels). Note that if a deadline extension is granted all students in the class will be granted such extension. Late assignments are still accepted, but will incur the penalties below.

- Submission prior to next class meeting time: -5 pts
- Submission within 1 week of original deadline: -10 pts
- Submission more than 1 week after deadline: -20 pts

Note: The last allowed date to submit late homeworks or project reports is December 11, 2024. No assignment or project report will be accepted after that date.

Regrading requests

Regrading requests are accepted within a week from the date grade are posted. Please note that you are always welcome to ask questions and clarifications about the homework solutions at any time during the semester, but assignments will not be regraded past the deadline.

Computing Resources

Programming assignments will leverage one or more high performance computing resources at the Texas Advanced Computing Center (TACC) of the University of Texas. Students will use their own PCs/Macs/workstations to access the machines using a locally installed SSH client. Account information will be handed out during the first two weeks of class. To create an account with TACC, please register at portal.tacc.utexas.edu/utdr.

Class Attendance

Class attendance will not be used directly in grade calculations. However, students are highly encouraged to attend class presentations and ask questions.

All students **are required** to attend the Final Project Presentation that will be held during the last week of class (12/3/2024 and 12/5/2024).

Collaboration

All homeworks submissions are to be performed **individually**. Students are encouraged to ask each other questions related to the assignments but may not share code or other work related to class assignments. All students are expected to equally contribute to the final project and presentation. We will check on GitHub that all team members have contributed to the implementation/debugging of the code and regularly committed to the repository. A statement of contributions is also required as an appendix to the final report.

Required Materials

There are no textbooks that you are required to purchase for this course.

Class materials, supplemental resources, grades, and announcements will be posted on the course Canvas site:

[redacted]

Additional material for this class is available on GitHub: https://github.com/uvilla-teaching/cse380

Some of the topics presented in this class are also covered in the following on-line free (CC-BY 4.0) books:

- *The Science of Computing: The Art of High Performance Computing, volume 1*, Victor Eijkhout, 3rd edition 2023 (last formatted April 2, 2024). Available from https://theartofhpc.com/.

- *Tutorials for High Performance Scientific Computing: The Art of High Performance Computing, volume 4*, Victor Eijkhout, 2022 (last formatted March 28, 2024). Available from https://theartofhpc.com/.

The author of these books, Victor Eijkhout, is a Research Scientist at TACC.

Technologies and other communication tools

We will use GitHub Classroom to collect coding assignments and the final project. Please use the following link to register:
[redacted]

Finally, the productivity app *Slack* has grown in popularity for academic purposes and has become a key communication tools for several University research groups as well as open-source software developer communities. To give you a gist of it, we will use slack (instead of the Canvas discussion board) in this class. Please join the slack channel at
[redacted].

University Policies

Disability Statement

If you are a student with a disability, or think you may have a disability, and need accommodations please contact Services for Students with Disabilities (SSD). You may refer to SSD's website for contact and more information: http://diversity.utexas.edu/disability/. If you are already registered with SSD, please deliver your Accommodation Letter to me as early as possible in the semester so we can discuss your approved accommodations.

Class Recordings

Class Recordings: Class recordings are reserved only for students in this class for educational purposes and are protected under FERPA. The recordings should not be shared outside the class in any form. Violation of this restriction by a student could lead to Student Misconduct proceedings.

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.

Note: This syllabus is subject to change; students who miss class are responsible for learning about any changes to the syllabus.