Syllabus

Production Quality Computational Multiphysics Software Development

A University of Colorado Boulder Distance Learning Course

Summer Session D (June 1 - August 6, 2021)
Monday-Thursday 2:50pm-3:50pm
All classes are held via Zoom

Scott R. Runnels, Ph.D. (Instructor) ECOT 411
scott.runnels@colorado.edu 505-695-9241

Course No. CVEN 5838-200B (for credit), NCEN 5838-570 (for non-credit)

Course Description

Multiphysics code development is a multidisciplinary endeavor, which by its nature requires contributors who have different strengths. At the same time, each contributor is eventually required to perform beyond their trained area of expertise. In this course, you will be exposed to some of the most essential elements of multiphysics code development, in such a way that builds your knowledge, skills, and confidence both for your current research efforts and your possible future role in the field of computational physics. The key multidisciplinary components include numerical methods for discretizing time-dependent partial differential equations, numerical analysis, linear and nonlinear solvers, computer programming, software verification, version control, production process control, testing, and the underlying physics being simulated. This course provides an in-depth survey of examples across all of these areas and provides a level of assurance to the successful student that they have enough knowledge to confidently participate in a multiphysics software endeavor in the future while also accelerating their own research now.

The course uses a combination of technical homework problems, software development assignments, and a project to instill understanding and develop skills. There is considerable flexibility in project choices. You may work alone or on a small team. You may start a project from scratch or use what you learn in this course to advance an existing code, e.g., your research. For every project choice, you will be required to implement multiple components, depending on their complexity and state of the code with which you start, that were not pre-existing in your work. Examples include: (1) A new discretization method, (2) A new solver method, (3) Self-documenting regression tests, (4) A test harness, (5) Verification testing, or (6) Repository management. You will be required to write a project pre-proposal, proposal, update, and final report. The goal here is to learn by doing, and to make the “doing” part as relevant as possible to your current research or production work.

Learning Outcomes

1. Experiencing live code development for these common methods:
    - Finite element method for elliptic and parabolic systems
    - Finite volume method hyperbolic systems, shock physics

- Finite difference method for diffusion and electromagnetics
- Time marching schemes
- Nonlinear solvers
- Iterative linear solvers

2. Hands-on experience in augmenting codes for:

   - Verification testing using the method of manufactured solutions
   - Multiphysics nonlinear coupling
   - Boundary fitting, curvilinear discretizations

3. Understanding of common analytics for numerical methods, including:

   - Theory of spatial and temporal convergence
   - Stability of diffusive systems
   - Stability of hyperbolic systems

4. Experience in basic tools of production code development, including:

   - Verification using the method of manufactured solutions
   - Version control using Git
   - C++ object-oriented programming
   - Python
   - Developing a regression testing system
   - Developing automatic documentation for testing
   - Introducing unit testing into a multiphysics code

5. Career enhancement through these motivational lectures:

   - Using communication to stabilize and advance one's career
   - Conducting effective peer reviews
   - The philosophy of test-driven code development
   - Time estimation

Prerequisites

This is intended to be a first-year graduate level course, and does not have any formal prerequisites. The field of computational multiphysics is multidisciplinary, and people who enter it have different strengths and weaknesses depending on their backgrounds. Those types of differences among the participants in this course are expected. If you have questions about your ability to participate in this course productively, please feel free to contact the instructor to discuss it. The following is provided as a guideline.

The theory lectures are designed such that they do not assume much knowledge beyond undergraduate mathematics for technical degrees. However, the lectures do quickly build upon that knowledge and so you should be firmly grounded in your undergraduate mathematics, including Calculus (I-III), with vector calculus, and you should understand what differential equations are. At least some previous exposure to partial differential equations is also very valuable, and can possibly be considered required. If you have not had any exposure to partial differential equations, you may not be ready to take or audit this course. Contact the instructor to discuss it first. Exposure to linear algebra will be very helpful. You should also have some experience programming

(in any language); without it, the assignments will take you a lot longer. It is still doable as a new programmer because C++ and Python tutorials are included, but only if you are willing to take on the task of learning how to program while taking this course.

This course will deal with physical laws, in particular, the idea of "conservation", e.g., conservation of mass, momentum, and energy. Physics and engineering courses that provide students with knowledge of these laws are not required as prerequisites. But if you have not taken those types of courses, you will benefit from doing some background reading on those topics first, especially relative to partial differential equations. Also, you would benefit from discussing your participation in this course with the instructor for a more specific evaluation and recommendation.

How to Enroll with a Non-Credit Option

Students, faculty, post-docs, and practicing professionals are all welcome to participate in this course under a non-credit option. The instructions for doing that are provided here. If you, instead, desire to register for credit, please follow the standard enrollment procedures at www.colorado.edu.

Technically, there is not an audit option *per se* for this course. Rather, it is a non-credit, pass/fail course where the requirements for passing is to attend or listen to 30% of the lectures. Participants will self-report at the end of the course regarding which lectures they saw or attended and will be on the honor system for that. To enroll with this non-credit pass/fail option, follow these steps:

1. Email Yen-Shi.Li@colorado.edu expressing your desire to enroll as a non-credit, pass/fail participant in Course Number NCEN 5838-570, "Production Quality Computational Multiphysics Software Development" provided through CU Boulder's Continuing Education Extraordinary Program.
2. She will email you a form that you can complete and sign electronically, returning it to her.
3. In response, she will provide instructions for how to log into your "Buff Portal" ("Buff" is in reference to the CU Boulder Buffaloes).
4. Through that portal, you can provide payment. You might consider contacting your administrative assistant to see if they can provide payment directly on behalf of your employer, if policy allows, using a corporate purchase card.

If you run into any problems, feel free to contact the instructor at scott.runnels@colorado.edu.

Estimated Schedule and Course Content

Note that the schedule and content are subject to change.

Week #1 Basic Software Tools for the Course (6/1)

In this first week, we establish common ground, e.g., a common language for the various in-class demonstrations as well as systems for managing the development process. Toward the end of the week, we begin with some basic theory of continuum equations.

C++ Tutorial: In-class object-oriented code development tutorial providing all syntax needed for the course

Python Tutorial: In-class code development tutorial demonstrating tools needed for the course

git Tutorial: In-class demonstration (creation, branching, merging, local/master, difftools)

Review of Continuum Equations (Reynold's Transport Theorem)

Homework #1 (5%): Programming and git basics

Week #2 A Basic Single-Physics Production-Oriented Code (6/8)

This week we develop enough theory to write our first code, while immediately developing a regression test system to go with it. The purpose will be to demonstrate the great power and confidence that accompanies having a regression test system operating in coordination with a version control system from the very beginning of development.

Applications of Continuum Equations to Mechanics (Solids/Fluids)

Electromagnetics (Maxwell's Equations)

Physical Constitutive Models, Phenomenological Models

Finite Difference Method

Gauss-Seidel Linear Solver

In-Class Demonstration: Development of a steady-state 2-D finite difference code

Introduction to Regression Testing

In-Class Demonstration: Development of a regression test system

Production Essentials Lecture: Development Planning and Time Estimation

Homework #2 (5%): Rudimentary discretization and linear solver methods

Week #3 Adding Capability Under Regression Testing and Peer Review (6/15)

This week we add more theory and apply that to the code being built in class. Along the way, we expand our regression test system to automatically produce documentation that reports on the code's progress in a shareable, reader-friendly format. We also discuss how to participate in and manage peer review in the context of a production code environment.

Temporal Discretization (Backward Euler, Forward Euler, Crank-Nicholson, Runge-Kutta)

In-Class Demonstration: Development of an implicit transient 2-D finite difference code

Using Latex and Python for Automatic Documentation Deployment

In-Class Demonstration: Development of a self-documenting regression test system

Production Essentials Lecture: Effective Peer Review Methods

Project Milestone #1 (10%): Students will provide a written project pre-proposal describing what they plan on doing either in a from-scratch code or a code that is part of their current work. The project must be focused on adding capabilities or components taught in this class, and must include aspects related to improving code quality (e.g., adding regression testing, a programmer's guide, a User's Manual, or methods of manufactured solutions).

Week #4 Adding Multi-Physics Under Regression Testing (6/22)

This week we add the equation for voltage, electric current, and Joule heating to that of heat conduction to form a basic multi-physics code. In the process, we discuss multiple aspects that arise in a multi-physics environment, including differences in geometric scale, temporal scale, meshing requirements, along with the nonlinearties of coupled physics, and the pathways for solving those non-linear equations.

Voltage equation, resistive heating

Thermal-electric coupling, non-linear systems

Non-Linear Solvers: Successive Iteration, Non-Linear Lagging, Newton-Raphson, relaxation

Inter-relationship of numerical tolerances for non-linear coupled systems (linear solver, non-linear solver, temporal solver)

In-Class Demonstration: Adding Multi-Physics

Homework #3 (5%): Adding a coupling term to a multi-physics code

Project Milestone #2 (10%): Students will submit a revised project proposal based on instructor comments provided on the pre-proposal from the previous week.

Week #5 Production Quality Verification and Additional Linear Solver Options (6/29)

This week we introduce and apply the powerful analytical verification method called the "method of manufactured solutions" (MMS), an invaluable tool for providing high quality verification of a production code. Under the growing sophistication of our in-class production-oriented code development, we will also expand the solver capability by introducing the multi-grid linear solver technique.

Method of Manufactured Solutions, or "MMS" for single-physics applications

MMS for multiphysics applications

In-Class Demonstration: Addition of MMS to an implicit 2-D finite difference code with automation regression documentation

Introduction to Multi-Grid, Algebraic Multi-Grid solver methods

In-Class Demonstration: Development of a multi-grid implicit 2-D finite difference code

Homework #4 (5%): Adding MMS to a multi-physics code

Week #6 Conjugate Gradient Method and the Philosophy of Testing (7/6)

This week we derive the conjugate gradient linear solver. The goal is to give production code developers some understanding of linear solver theory so they can better appreciate this complex and influential branch of the field. The lecture is fairly intense and the week is rounded out with a non-technical lecture describing how test driven code development elevates the field of computational physics into its own distinct profession, with more predictability, accountability, and sustainability from a business and career management perspective.

Conjugate Gradient Linear Solver

Production Essentials Lecture: Test Driven Code Development

Project Milestone #3 (10%): Students will present their project plan for a peer review. Students will be graded on participation, style of offering criticism, style of receiving criticism. This aspect of the course is aimed at developing and exercising the "soft skills" essential on a production team and with client interaction.

Week #7 Other Discretization Methods – Finite Elements (7/13)

This week is dedicated to explaining the theory of the Galerkin finite element method, and developing a standalone 2D, transient code demonstrating it.

Finite Element Method

In-Class Demonstration: Development of an implicit 2-D finite element code

Homework #5 (5%): Finite elements

Week #8 Other Types of PDEs – Hyperbolic Equations with the Finite Volume Method (7/20)

This week we introduce the finite volume method as applied to shock physics, writing a new, self-contained 1-D code so that students can see how different hyperbolic PDEs are from elliptic/parabolic PDEs.

Finite Volume Method on Regular Grids

Mimetic Method for Diffusion on Irregular Grids

Shock Physics and Artificial Viscosity

In-Class Demonstration: Development of a 1-D shock physics code using the finite volume method

Production Essentials Lecture: Communication (Style Guides, Programmer's Manuals, Programmer Training)

Week #9 Advanced Testing and Verification Methods (7/27)

This week focuses on two more verification testing topics. "Unit testing" is introduced and added as another aspect of regression testing. Also, the idea of integrating performance aspects into testing will be demonstrated, including convergence analysis and convergence testing. Stability analysis will also be covered.

Advanced Testing: Software Test Harnesses, Methods of Unit Testing, Unit Test Platforms

In-Class Demonstration: Adding a unit test to the regression test suite

Numerics-based Verification: Stabiity, spatial and temporal convergence analysis and testing

Numerics-based Verification: Matrix condition, performance-based testing

In-Class Demonstration: Adding a convergence regression test

Production Essentials Lecture: Documenting and Using the Release Process

Homework #6 (10%): Students will be given the finite difference solver developed in class with the same regression test system, except the regression test system will not be complete. Students will add a convergence test, based on the in-class demonstration.

Project Milestone #4 (10%): Project status update (1 page) describing progress towards goals.

Week #10 Parallel Computing (8/3)

This week is dedicated to conveying and demonstrating the basics of parallel computing, including the in-class development of a simple parallel diffusion solver.

Essentials of parallel computing

Parallel file I/O

Measuring performance

In-Class Demonstration: Parallel diffusion solver

Production Essentials Lecture: User Support, Bug Reporting, Issue Tracking

Course Recapitulation

Review for the final exam will be highly detailed. All material to be covered in the exam will be discussed in class during this one-session review.

Project Milestone #5 (10%): Students will provide a report on their project commensurate with the expectations set forth and approved in the project proposal.

Exam (15%): Final Exam, in class, closed book

Policies

Classroom Behavior

Both students and faculty are responsible for maintaining an appropriate learning environment in all instructional settings, whether in person, remote or online. Those who fail to adhere to such behavioral standards may be subject to discipline. Professional courtesy and sensitivity are especially important with respect to individuals and topics dealing with race, color, national origin, sex, pregnancy, age, disability, creed, religion, sexual orientation, gender identity, gender expression, veteran status, political affiliation or political philosophy. For more information, see the policies on classroom behavior and the Student Code of Conduct.

Requirements for COVID-19

As a matter of public health and safety due to the pandemic, all members of the CU Boulder community and all visitors to campus must follow university, department and building requirements, and public health orders in place to reduce the risk of spreading infectious disease. Required safety measures at CU Boulder relevant to the classroom setting include:

- maintain 6-foot distancing when possible,
- wear a face covering in public indoor spaces and outdoors while on campus consistent with state and county health orders,

- clean local work area,
- practice hand hygiene,
- follow public health orders, and
- if sick and you live off campus, do not come onto campus (unless instructed by a CU Healthcare professional), or ifyou live on-campus, please alert CU Boulder Medical Services.

Students who fail to adhere to these requirements will be asked to leave class, and students who do not leave class when asked or who refuse to comply with these requirements will be referred to Student Conduct and Conflict Resolution. For more information, see the policies on COVID-19 Health and Safety and classroom behavior and the Student Code of Conduct. If you require accommodation because a disability prevents you from fulfilling these safety measures, please see the "Accommodation for Disabilities" statement on this syllabus.

Before returning to campus, all students must complete the COVID-19 Student Health and Expectations Course. Before coming on to campus each day, all students are required to complete a Daily Health Form. Students who have tested positive for COVID-19, have symptoms of COVID-19, or have had close contact with someone who has tested positive for or had symptoms of COVID-19 must stay home and complete the Health Questionnaire and Illness Reporting Form remotely. In this class, if you are sick or quarantined, you can still participate in lectures since they are all remote, but only if you are feeling well enough to do so. If you are too sick to participate, please do your best to notify the instructor. However, do not feel any obligation to reveal the details of your illness (whether COVID-19 or otherwise). The main point will be to convey that you will not be able to participate.

Accommodation for Disabilities

If you qualify for accommodations because of a disability, please submit your accommodation letter from Disability Services to your faculty member in a timely manner so that your needs can be addressed. Disability Services determines accommodations based on documented disabilities in the academic environment. Information on requesting accommodations is located on the Disability Services website. Contact Disability Services at 303-492-8671 or dsinfo@colorado.edu for further assistance. If you have a temporary medical condition, see Temporary Medical Conditions on the Disability Services website.

Preferred Student Names and Pronouns

CU Boulder recognizes that students' legal information doesn't always align with how they identify. Students may update their preferred names and pronouns via the student portal; those preferred names and pronouns are listed on instructors' class rosters. In the absence of such updates, the name that appears on the class roster is the student's legal name.

Honor Code

All students enrolled in a University of Colorado Boulder course are responsible for knowing and adhering to the Honor Code. Violations of the policy may include: plagiarism, cheating, fabrication, lying, bribery, threat, unauthorized access to academic materials, clicker fraud, submitting the same or similar work in more than one course without permission from all course instructors involved, and aiding academic dishonesty. All incidents of academic misconduct will be reported to the Honor Code (honor@colorado.edu; 303-492-555). Students found responsible for violating the academic integrity policy will be subject to nonacademic sanctions from the Honor Code as well as academic sanctions from the faculty member. Additional information regarding the Honor Code academic integrity policy can be found at the Honor Code Office website.

Sexual Misconduct, Discrimination, Harassment and/or Related Retaliation

The University of Colorado Boulder (CU Boulder) is committed to fostering an inclusive and welcoming learning, working, and living environment. CU Boulder will not tolerate acts of sexual misconduct (harassment, exploitation, and assault), intimate partner violence (dating or domestic violence), stalking, or protected-class discrimination or harassment by members of our community. Individuals who believe they have been subject to misconduct or retaliatory actions for reporting a concern should contact the Office of Institutional Equity and Compliance (OIEC) at 303-492-2127 or cureport@colorado.edu. Information about the OIEC, university policies, anonymous reporting, and the campus resources can be found on the OIEC website.

Please know that faculty and instructors have a responsibility to inform OIEC when made aware of incidents of sexual misconduct, dating and domestic violence, stalking, discrimination, harassment and/or related retaliation, to ensure that individuals impacted receive information about options for reporting and support resources.

Religious Holidays

Campus policy regarding religious observances requires that faculty make every effort to deal reasonably and fairly with all students who, because of religious obligations, have conflicts with scheduled exams, assignments or required attendance. In this class, students are required to notify the instructor at the beginning of the semester so that proper accommodations can be made for all assignments and tests. See the campus policy regarding religious observances for full details.

Compiled Sunday 28th February, 2021at 13:50