# MA539 Course Syllabus

Methods of Scientific Computing
T/TH 11AM-12:15PM, CAS-314

**Instructor Information:**
Samuel Isaacson (isaacsas@bu.edu)
Professor
Department of Mathematics and Statistics
Office: CDS 442 (where office hours are held).

**Course Webpage**:
We will use the course Blackboard site available at https://learn.bu.edu for all course material, and Gradescope for submitting homework assignments.

**Office Hours:**
To be determined based on student schedules once registration is settled. For the first week of classes I will have an office hour on Wednesday 9/3 from 2-3pm in CDS 442.

**Description and Goals:**
Scientific Computing is a broad subject, covering topics such as the accuracy, efficiency and optimized implementation of numerical methods, along with investigating how to select appropriate methods for specific problem types. Topics covered in this class will include matrix factorizations for solving linear systems and least squares problems involving different types of matrices; methods for numerical differentiation, including modern automatic differentiation (AD) approaches that are compatible with machine learning libraries; and optimization methods, with discussion of how linear algebra methods and AD can be exploited to improve their performance and how optimization can be used in parametrizing neural networks.

(*Time-permitting)* As they underpin widely-used Monte Carlo methods, we'll also study pseudo-random number generators, with a focus on basic types of pseudo-random number generation methods and understanding how/when they can fail reasonable statistical tests for randomness. We will conclude by exploring methods for approximating functions by a basis of polynomial or trigonometric functions, techniques which underly methods for integrating functions and solving differential equations.

Applications of the various numerical methods will be demonstrated to problems arising across engineering and the sciences, and a key focus of the class will be in understanding performance and memory tradeoffs when actually implementing numerical algorithms (i.e. in vs. out of place functions, arrays vs. loop performance, etc).

**Prerequisites:**
Linear algebra (MA242 or equivalent), multivariable calculus (MA225 or equivalent), and previous programming experience in some language (Julia, Python, C/C++, Java, MATLAB, or R). Prior Julia knowledge is not required.

**Textbooks:**
We will use material and assign problems from the following two books, along with potentially covering some material from the third. The first book is more elementary, and we will follow it closely at the start of the semester. As it is (currently) less comprehensive we will take material from the second as the semester continues. All are available online for free.

**1.** *Fundamentals of Numerical Computation*, Driscoll and Braun.
We will follow the online Julia-based version available at:
https://tobydriscoll.net/fnc-julia/home.html
Complementary video lectures by Driscoll are available at:
https://www.youtube.com/playlist?list=PLvUvOH0OYx3BcZivtXMIwP6hKoYv0YvGn

**2.** *Scientific Computing: An Introductory Survey*, Heath, SIAM Classics (2008).
ebook freely available from the BU library at:
https://epubs-siam-org.ezproxy.bu.edu/doi/book/10.1137/1.9781611975581

**3.** *Parallel Computing and Scientific Machine Learning (SciML): Methods and Applications,*
ebook freely available online at:

https://book.sciml.ai/

**Please download the full pdf eBook of Heath (2) as soon as possible.** SIAM eBook access often breaks / disappears at times during the course of the semester, so continued access cannot be guaranteed. Lack of access to the eBook later in the semester will not be an acceptable reason for late submission of an assignment.

**Programming:**
Computer assignments will use the Julia programming language.

Instructions will be available on Blackboard for setting up Julia and Visual Studio Code for editing / writing Julia code. If you'd like to get started before we use it in class please install Julia 1.11.6 from https://julialang.org/downloads/. You can start learning about the language by going through the "Introduction to Julia" tutorial at https://julialang.org/learning/, and/or watching the associated Julia Youtube tutorial linked in the instructions on the course webpage. We will go through a Julia for scientific computing tutorial during the second week of class.

**Grading:**
5% - Class participation and engagement
- Attending and participating in class.
- Asking questions in office hours.
- ***Keeping phones, computers, and tablets put away during lecture.***

35% - Computational homework assignments.
60% - Three in class exams.

**Lecture Information:**
- You are generally expected to attend class and take notes – I do not normally have nice lecture notes that can be distributed. If you miss class due to illness or other reasons, please get a copy of any lecture material you missed from a fellow student.
- During lectures, cell phones and computers must be put away. Tablets are allowed only if being used for note taking.

**Homework Information:**
- You may work together with other students on completing the homework, or use AI tools such as GPT and Claude Code to assist you in writing code. However, the final pdf report you submit must be written in your own words and demonstrate *your* understanding of the problem solutions.
- *If you worked with another student or used AI tools for any homework problems, please include a short statement on how they were used at the start of your assignment.*
- Computational assignment solutions should be submitted on Gradescope as one pdf file with the problems written up neatly, in order, with answers clearly indicated. This should include clear labeling and captioning of any figures or tables that are reported as part of the assignment. More detailed instructions will be provided as part of each assignment.
- *Homework will not be accepted late except in exceptional circumstances (illness), however, the lowest homework grade will be dropped.*
- Homework assignments will include analytical (i.e. pen + paper) problems that *will not* be collected or graded. It is expected that you will work on these as they will be covered on exams.
- Homework will generally be posted on Thursday or Friday, and due at 11:59pm one or two weeks later.
- I will post homework solutions shortly after the due date (for both computational and non-collected analytical problems).

**Exam Dates and Information:**
***Exams will be held on 10/2, 11/6, and 12/9.***

- There will be three in-class exams, covering all lecture and homework material (both graded computational portions and non-graded analytical portions).
- The lowest exam grade will be replaced by its average with your highest exam grade.
- Students are expected to be able to attend all lectures for the course, and as such, make up exams are only given under exceptional circumstances (e.g. illness). Approved make up exams will generally be oral exams answering questions on a blackboard. If you feel you have an exceptional reason you will miss one of these exam dates please notify me the first week of classes.

**Academic Conduct:**

All students are expected to know and abide by BU's code of academic conduct, see:
http://www.bu.edu/academics/policies/academic-conduct-code/

**Tentative Course Outline (subject to change, * denotes a topic that is time-permitting):**

1. Introduction to Julia
    - performance considerations
2. Basic background material
    - floating point arithmetic
    - conditioning and stability
3. Numerical Differentiation
    - Finite differences
    - Automatic Differentiation
        - forward mode
        - reverse mode
4. Numerical Zero Finding and Optimization in One Dimension
    - bisection
    - steepest descent in 1D
    - Newton's method
        - Inexact methods
5. Numerical Linear Algebra:
    - Solving linear systems:
        - LU factorization
        - Cholesky factorization
        - Methods for banded and sparse matrices
        - Matrix norms, stability and conditioning
    - Least squares systems:
        - Normal equations
        - QR factorization
        - Singular value decomposition (SVD)
            - SVD applications such as image compression.
    - * Iterative methods for solving linear systems.
6. Numerical Zero Finding and Optimization in Multiple Dimensions
    - steepest descent
        - variants and neural network parametrization.
    - Newton's method
        - Inexact methods
    - * Nonlinear least squares (Gauss-Newton)
7. * Random Number Generation
    - Classical random number generation methods.
    - Statistical tests, failures and gotchas.
    - Modern, robust random number generation libraries.
    - Applications to Monte Carlo simulation.
8. * Function Approximation
    - Polynomial interpolation
        - Classical and Barycentric Lagrange interpolation
        - Chebyshev Polynomial Approximation
    - Trigonometric Approximation
        - FFT
        - Spectral Differentiation
    - Neural Network Approximation