AMCS 252

Spring 2021

Time: 8:30–10:00 a.m. every Monday and Thursday
Location: https://kaust.zoom.us/j/93097661354

Instructor: David Ketcheson
david.ketcheson@kaust.edu.sa
Instructor's office: Al-Khawarizmi building, Room 4240
Office hour: Wednesday, 11–12 a.m. or by appointment

Teaching assistant: Jefferson RIera
Office hours: TBD

Textbook: "Finite Difference Methods for Ordinary and Partial Differential Equations", by R. J. LeVeque.

Additional references (not required):

- Hans P. Langtangen. A Primer on Scientific Programming with Python. Springer.
- Bertil Gustafsson, Heinz-Otto Kreiss, and Joseph Oliger. 1995. Time dependent problems and difference methods. Wiley-Interscience.
- Strikwerda, John. 2007. Finite Difference Schemes and Partial Differential Equations. Society for Industrial Mathematics.
- Thomas, J.W. 2010. Numerical Partial Differential Equations: Finite Difference Methods (Texts in Applied Mathematics). Springer.
- Trangenstein, John A. Numerical Solution of Hyperbolic Partial Differential Equations. Cambridge University Press.

Course resources: https://github.com/ketch/AMCS–252–2021

Overview

This course will introduce you to numerical methods for solving ordinary and partial differential equations, with a focus on finite difference methods. Your main goal should be to gain understanding of the methods through analysis of their accuracy

and stability, and also by implementing and experimenting with the methods. We will begin with finite difference solution of boundary value problems, followed by the solution of initial value ODEs, focusing on Runge-Kutta and linear multistep methods. The last part of the course will treat finite difference methods for initial value PDEs, with emphasis on the heat equation and the advection equation.

You should possess knowledge of linear algebra, differential equations, and advanced calculus (especially Taylor series), and at least some programming experience. Prior knowledge of numerical analysis and PDEs is very helpful, but not strictly necessary.

The instruction and homework will make use of the **Python** programming language, and especially of Python's numerical and mathematical packages (NumPy, SciPy, and Matplotlib). No prior experience with Python is required, although prior MATLAB or Python experience is helpful. If you have not used Python before – and especially, if you have done little programming before – you should expect to devote a significant amount of time to learning the basics of Python programming, in addition to the expected mathematical concepts, during the first few weeks of the course.

I recommend installing Python and the associated libraries on your own computer. Some suggestions on how to install the required tools can be found at https://www.scipy.org/install.html.

Alternatively, you can also use some cloud services that run Jupyter notebooks. I recommend either of the following: https://mybinder.org/ https://colab.research.google.com/notebooks/intro.ipynb

On Thursday, January 28th in class I will give a Python programming tutorial. Before then, you should make sure you are able to run the tutorial notebook found at https://github.com/ketch/AMCS–252–2021.

Evaluation

Homework (50%)

Homework will be assigned after most classes and will usually be due every second Thursday. You are allowed to discuss your solutions with other students but **all work that you turn in must be your own**, and you must understand it completely. If there is doubt, you may be asked to explain your solution in class or in an informal oral exam.

Since the course deals with mathematical theory and algorithms, homework problems will include proofs, derivations, calculations, and programming.

Computation

For homework problems that require programming, Python will be the language of the course. I will give an introduction to Python programming on the first Thursday of the course. If you want to get a head-start, or to supplement what you learn there, a good resource is https://scipy-lectures.github.io/.

For homework problems that require programming, you must turn in the code used to compute your solutions. However, **your code and plots are not the answer** and will not be graded; they are merely a supplement to your thoughtful written explanations. To be clear, if you turn in only code, you will receive NO credit.

Late Work

No credit will be given for late work. That includes work that is turned in even a few minutes after the deadline. If you have special circumstances that require an extension, talk to the instructor as soon as you know about them.

Exams (50%)

Two midterm exams will be given, in class; one approximately half way through the course and the other on the last day of the course. Each midterm will cover only half of the course and will be worth 1/4 of the total grade. There will be no final exam.

Tentative grading scale

90–100%	**A**
85–90%	A-
80–85%	B+
75–80%	B
70–75%	B-
60–70%	C
< 60%	F

Conduct

Plagiarism

Plagiarism is the act of taking credit for the words or ideas of someone else. Any plagiarism will be grounds for failure in this course. All material that you hand in must be your own work. If you are in doubt, ask the instructor.

Special Accommodations

If you have a personal activity, family, or religious conflict with the course schedule, you can expect to be heard sympathetically. Please contact me by the end of the second week of the term to discuss appropriate accommodations for any conflicts that can be foreseen. For illness-related absences, there are standard procedures to follow.