MATH 450: Numerical Analysis for functions and differential equations

— Syllabus —

Course Topics *Numerical methods* are the foundation for all approaches by which we *solve mathematical or engineering problems on computers*. They are typically formulated in terms of algorithms that can be implemented on computers, and include things like solving linear systems or nonlinear equations, but also how to solve ordinary or partial differential equations such as those in fluid dynamics, elasticity, and many other areas.

This course provides an introduction to the basic techniques in this area, covering specifically "continuous problems" such as approximating and interpolating functions by polynomials or piecewise polynomial functions, computing integrals, computing derivatives, and solving differential equations.

In contrast to a course on numerical *methods*, this class on *numerical analysis* is concerned not only with devising methods, but also analyzing their performance. For example, we will consider how *expensive* particular algorithms are, so that we can understand which ones can be implemented efficiently. We will also consider how *accurate* algorithms are – this is important because in most cases, algorithms for non-trivial problems can only give us *approximate* solutions, and we need to know how close these approximations are to the exact solution.

This course will cover the following, specific topics:

1. **Approximation:** A function $f(x)$ is, in general, an infinite-dimensional object – it has a value for each of the infinitely many points x in its domain. "Infinite-dimensional" does not go well with computers: We only have finitely much memory, and finitely much patience to wait for computations. As a consequence, we often find ourselves in need to *approximate* functions $f(x)$ by functions $\tilde{f}(x)$ that can be represented by *finitely much data* – for example, $\tilde{f}(x)$ could a polynomial of degree 4 that can be represented and stored using only the five coefficients in $\tilde{f}(x) = a + bx + cx^2 + dx^3 + ex^4$. In this part of the semester, we will discuss how to compute such approximations, and how large the difference between f and $\tilde{f}$ can be.

2. **Interpolation:** A specific form of approximation if *interpolation* where f and $\tilde{f}$ are equal at a set of points chosen up front. Such functions are much easier to compute than general approximations.

3. **Computing integrals via "quadrature":** Both approximation and interpolation can be used to approximate integrals of the form $\int_a^b f(x)\ dx$. This is because in general we can not compute antiderivatives of functions f, but we can compute antiderivatives of the approximation $\tilde{f}$ because it is a polynomial. Using this observation, we can compute approximate values for the integral through a process called *quadrature*. We will discuss the accuracy of using this approach for both "global" and "piecewise" approximations.

4. **Computing derivatives:** Approximation and interpolation can also be used to compute approximations to the derivative of a function at a point, $f'(x_0)$, for functions for which we do not have a formula representation.

5. **Numerical solution of ordinary differential equations:** All of the ideas above can be used to solve ordinary differential equations, that is equations of the form $x'(t) = f(x(t), t)$. These equations

are the basic building blocks of modeling the world around us mathematically. You will see the many approaches one can bring to solving differential equations, such as ones that approximate the term $x'(t)$, ones that use approximations of integrals, and the Runge-Kutta approach.

6. **Numerical solution of partial differential equations:** More complex models of the world lead to *partial* differential equations. This part of the semester will give a very brief introduction to how one should approach the solution of such models.

In concrete terms, the following is an approximate weekly schedule:

- **Week 1:** Taylor's theorem as the foundation of approximation of functions.
- **Week 2:** Approximation and interpolation of a set of data points. Least-squares approximation.
- **Week 3:** Approximation of functions via polynomials and trigonometric series. Least-squares approximation.
- **Week 4:** Interpolation via global and piecewise polynomials. Accuracy of global and piecewise interpolation.
- **Week 5:** Interpolation for functions in two and more dimensions.
- **Week 6:** Quadrature via global and piecewise interpolation with polynomials.
- **Week 7:** Gauss quadrature.
- **Week 8:** Approximating derivatives.
- **Week 9:** Introduction to ordinary differential equations (ODEs). Scalar ODEs and systems of ODEs.
- **Week 10:** Solving ODEs by approximating the time derivative.
- **Week 11:** Solving ODEs by integral approximation.
- **Week 12:** Runge-Kutta methods.
- **Week 13:** Introduction to partial differential equations (PDEs), categorization into elliptic, parabolic, and hyperbolic equations.
- **Week 14:** An outline of finite-difference approximation of the Laplace equation; outline of a finite-volume approximation of the transport equation.
- **Week 15:** Student project presentations.

Textbook I do not require you to get any particular book, and in particular will not pose homework that references a book. That said, if you want to read up on some of the material we discuss in class, the following books (in any edition you can find) cover essentially everything we do over the course of this semester:

- D. Kincaid and W. Cheney: *Numerical Analysis*, Brooks & Cole Publishing Co.
- E. Hairer, S. P. Norsett, G. Wanner: *Solving Ordinary Differential Equations I: Nonstiff Problems*, Springer.

Prerequisites (CS 150A or CS 150B or CS 152 or CS 163 or CS 164 or CS 165 or CS 253 or MATH 151) and (MATH 256 or MATH 261 or MATH 340 or MATH 345)
Many of the homework assignments will require you to write small programs – say, in the range of 20–100 lines of code. In general, I leave the choice of programming language to you, but if your choice is somewhat exotic or outside the realm of what a typical programmer can be expected to read, you will need to provide sufficient commentary to make the code understandable.

Webpage Homework assignments and other course information will be posted on Canvas.

Exams + Grading Final grades will be determined based on the following components:

- Homework and programming assignments: 60%
- Midterm: 20%
- Final project: 20%

Your minimum grade will be A, B, C, or D, for a score of 90%, 80%, 70%, and 60% over the course of the semester, respectively.
You must make arrangements in advance if you expect to miss an exam or quiz. Exam absences due to recognized University-related activities, religious holidays, verifiable illness, and family/medical emergencies will be dealt with on an individual basis. In all cases of absence from exams a written excuse is required. Ignorance of the time and place of an exam will not be accepted as an excuse for absence.

Learning Outcomes and Course Objectives Numerical methods are the foundation of computer simulations in all fields of the sciences and engineering. But they are also important for solving all sorts of equations you find in mathematical biology, chemistry, physics, or finance: Whenever you end up with an equation that cannot be solved on a piece of paper (because you can't find the right approach to solving it, because there just isn't a way to solve the equation on a piece of paper, or because you find yourself not in the mood to solve the problem because it would take too long), then you need a computer to do it; this class teaches you how computers do that.
The goal of this class is to (i) provide a basic level of literacy in numerical methods, as well as (ii) to learn about their analysis. At the end of the semester, you will be able to identify and understand what methods to use depending on the situation; how they will likely perform; and analyze these methods in terms of properties such as approximation quality or speed of convergence. You will also have practice in implementing these methods on computers.

Policies As always, university policies apply – see this link for the details.