CSC349A Numerical Analysis

Lecture 1

George Tzanetakis

University of Victoria

2025

Table of Contents I

Course Information

- Course outline: https://course-outlines.uvic.ca/ban/course-outlines/public-outline?banner_term=202509&subject=CSC&course_number=349A
- **Optional** Textbook: Numerical Methods for Engineers (8th edition), S.C. Chapra and R.P. Canale,McGraw-Hill, ISBN: 978-0-07-3397924
- The new edition of this book is available as an e-book.
- MATLAB - https://matlab.engr.uvic.ca/student/
- Python - https://www.spyder-ide.org/

Grading

- 6 assignments worth 30% of final grade
  - No extensions unless the whole class gets one
- 2 Midterms 30% and Final Exam 40%.
- If you miss a midterm due to illness, there will be an alternative weighting
  - 1 Midterm 20% and Final Exam 60%
  - Note - You can't miss two midterms
- If you miss the final due to illness, apply for a Deferal
  - Note - will not be granted if you also missed a midterm
- You will be marked for both correctness and logic
- Early errors are worse than late errors
- Errors propogate, you do not get credit for wrong answers

Attendance

- Not compulsory - you are adults
- DO NOT email me when you will miss a lecture
- Handouts and associated coursepack
- Lecture notes, and slides
- I will NOT be recording or streaming the lectures
- **USE BRIGHTSPACE:**
  `https://bright.uvic.ca/d2l/home/314357`

Large scale challenges

- 200 students
- **USE BRIGHTSPACE and PRAIRIELEARN**
- If you e-mail me, be precise - always CSC349A in subject line, mention assignment etc
- Less flexibility and feedback on assignments

Assignments

- Online through **PRAIRIELEARN**
- Answers to be separated by question
- Any recycling of answers from previous years automatically zero grade on assignment and AIC case brought forward
- **NO LATE SUBMISSIONS**

Academic Integrity

- The CSC department has an Academic Integrity Committee (AIC)
- Any violations of Academic Integrity will be forwarded to the AIC
- UVic has strict policies against AI violations
  - https://www.uvic.ca/students/academics/academic-integrity/index.php
- Once a violation has be sent to the committee it is out of my hands

Highlights of UVic AI Policy

- Single or multiple instances of plagiarism should result in a failing grade for the work.
- A largely or fully plagiarized piece of work should result in a grade of F for the course.
- Isolated instances of copying during an exam should result in a grade of zero for the exam.
- Systematic copying during an exam should result in a grade of F for the course.
- Any instance of any of the violations committed by a repeat offender, should result in the student being placed on disciplinary probation.
- A letter of reprimand will be sent to the student and a copy shall be included in the record maintained by the Office of the Registrar.

Table of Contents I

Definition

Numerical analysis

Numerical analysis is concerned with accurate, efficient approximations of solutions to problems of continuous mathematics.

Rounding Errors

Real and complex numbers can not be represented exactly on computers therefore they are always **approximated**. This approximation introduces errors which, even though typically small, can have a significant effect in the accuracy of numerical computations especially when these computations involve lots of arithmetic operations.

Numerical analysis investigates the **stability** and **accuracy** of algorithms on these rounding errors.

Truncation Errors

University of Victoria Computer Science

Most continuous mathematics algorithms **cannot be solved by finite algorithms**. This is true even if we could somehow work with exact arithmetic. This is true for many problems of interest including zero finding, differential equations, and optimization.

Rapid **convergence of approximations** is the primary challenge that has been met successfully by numerical analysis.

Continuous mathematics applications

- Physics engines in computer games - Angry birds
- Simulating wind turbulance - wing design
- Filter design - audio effects
- Non-linear least squares - 3D models from photographs
- Optimization - a lot of data mining, machine learning
- Matrix computations - 3D graphics for games
- Differential equations - simulation in ECE, MEC
- Study of errors - major disasters due to numerical errors

Historical Origins of Numerical Analysis

- Computers were people before they were machines
- World War II was the catalyst for the development of computers
- Code breaking was the original discrete mathematics killer (literally) app
- Computation of ballistic tables and atomic bomb simulations were the original continuous mathematics killer (literally) apps

Importance in science and engineering

- Powerful problem solving tools that can handle large amounts of data, non-linearities and complex geometries that **cannot** be solved by any other means

- Under the hood knowledge of packaged software

- Not all real-world problems can be solved with packaged software

Book overview

- 8 parts and 32 chapters
- Each part begins with a Preface motivating each problem and some background mathematics
- Part 1: Modelling, computers and error analysis
- Part 2: Roots of equations
- Part 3: Linear algebraic equations
- Part 5: Curve fitting
- Part 6: Differentiation & integration
- Part 7: Ordinary differential equations
- Each part ends with Case Studies from Engineering to highlight the applications

Table of Contents I

Mathematical Models

These equations are typical of mathematical models of the physical world:

- Describes a natural process in mathematical terms
- It represents an idealization and simplification of reality
- It yields reproducible results and can be used for predictive purposes

A motivating example

Determine the terminal veloctiy of a free-falling body (a parachutist) near the earth's surface by using Newton's second law of motion and solving a differential equation by, (a) analytical methods, then (b) numerical methods, and (c) compare.

A motivating example continued

Using Newton's second law in our parachutist example, with downward pull of gravity given by mg, where m is mass and g is the gravity constant, and the upward force of air resistance given by $-cv$, where c is the drag coefficient and v is the velocity, we derive the following model to express the problem.

$$\frac{dv}{dt} = g - \frac{c}{m} v$$

Here, we have a differential equation in terms of the rate of change of velocity over time, given by t.

Analytical solution

It is possible to get an analytic solution to this differential equation using calculus. If the object is initially at rest i.e $v = 0$ at time $t = 0$ then:

$$v(t) = \frac{gm}{c}(1 - e^{-\frac{ct}{m}}) \tag{1}$$

This analytical solution can be used directly to solve problems.

For example, let $m = 68.1kg$, $g = 9.81m/s^2$, and $c = 12.5kg/s$.

Analytical solution - Example 1.1

For example a parachutist of mass $68.1kg$ jumps out of a stationary hot air balloon. The drag coefficient is equal to $12.5kg/sec$.
Inserting the parameters into the equation we get:

t	**v**
0	0.0
2	16.40
4	27.77
6	35.68
..	..
∞	53.44

Thus the *terminal velocity* is $53.44m/sec$ for this example.

Numerical solution

Numerical approach (in constrast to analytical) for solving this mathematical model. Not exact but we can get arbitrarily close. Why numerical methods (NM) ?

- NM can be applied to functions for which we can not easily find an analytical solution through Calculus
- As any equation is to some degree an approximation of reality and therefore errors are inevitable it is possible depending on the application that the errors introduced by NM are neglible in the context of the desired application
- It's extremely simple to write a computer program to do the job for us although the number of mathematical operations involved is much larger than the analytical solution

Numerical Solution

- The idea is rather simple. We will approximate the derivative of the function by a "finite divided difference" effectively discretizing time.
- If we denote the discrete time steps we take as t_i (we can decide what sampling rate is appropriate depending on the application) then we can approximate the derivative at time t_i as follows:

$$\frac{dv}{dt} \approx \frac{v(t_{i+1}) - v(t_i)}{t_{i+1} - t_i}$$

Numerical Solution continued

So, if we substitute our derivative approximation into our model differential equation, we can derive an approximation formula for the velcoity at time t_{i+1} given the velocity at time t_i, as follows.

$$v(t_{i+1}) \approx v(t_i) + \left[g - \frac{c}{m} v(t_i)\right] (t_{i+1} - t_i)$$

Numerical iteration

- Notice that this equation gives us a way to compute the velocity value at time t_{i+1} based on the previous value of the velocity.
- This approach is called Euler's method can be verbally expressed as: *New value* $=$ *old value* $+$ *slope x step size*.

Numerical solution - Example 1.2

- For example we can use it to compute the velocity values with a step size of 2 seconds.
- At the start of the computation we use an initial velocity value $v_0 = 0$ for $t_0 = 0$ plug the numbers into the equation and get the value of v_1. Then we can use the value at v_1 to compute the value at v_2 and so forth.

$$v(t_1) \approx v(t_0) + \left[g - \frac{c}{m} v(t_0)\right](t_1 - t_0)$$

Numerical solution - Example 1.2

- For example we can use it to compute the velocity values with a step size of 2 seconds.
- At the start of the computation we use an initial velocity value $v_0 = 0$ for $t_0 = 0$ plug the numbers into the equation and get the value of v_1. Then we can use the value at v_1 to compute the value at v_2 and so forth.

t	v
0	0.0
2	19.62
4	32.04
6	39.90
..	..
∞	53.44

Comparison

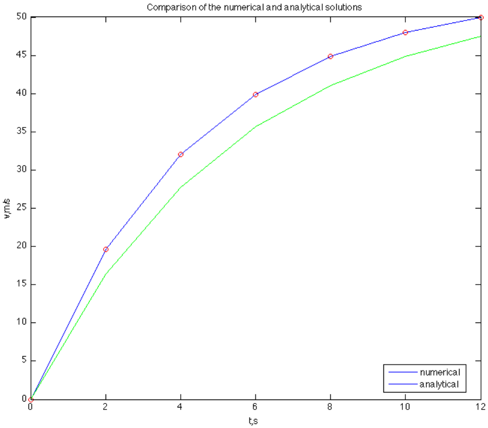

Figure: Comparison of numerical and analytical solution

Comparison of numeric and analytic solution

Figure 1 shows a plot comparing the results of the numerical approximation using Euler's method and the exact analytical solution. In engineering there is frequently a tradeoff between accuracy and computational resources.

Euler Method

University of Victoria Computer Science

The Euler method and more generally numerical methods allow you to control that tradeoff for example by adjusting the step size appropriately for the program at hand. There are several questions one could ask at this point that we will be exploring in the rest of the course. For example can we show that the error between the exact solution and the numerical approximate solution will always decrease with smaller step size? Can we compare how long different approaches to approximating the derivative of a function take? Are there any functions for which this would not work? Could we reduce the number of multiplication/additions needed to perform each step of the iteration?