Lab 7 Report

Neo Nguyen

December 2020

Abstract

This lab focuses on differential equations and methods we use to solve them. In particular, we studied Euler's Method, Runge-Kutta, Heuns Method, and some modifications to these methods. They we compared the errors between them.

Introduction

When dealing with an in ital value problem, we know at least one solution to the differential equation. From here, we want to find more. Some Initial Value Problems, or IVPs, are easy to solve analytically but some are impossible to solve with current methods and require a numerical approximation. The methods described in the report are possible ways to gain accurate approximations that could be used in real life like modeling fluid flow or rocket flight.

Euler's Method

Euler's Method is a fairly simple but extremely powerful method used to solve differential equations numerically rather than analytically. It is not used in practice much anymore but nonetheless should be studied. The formal definition of the Euler's Method algorithm is a recurisive function as follows:

$$x_{k+1} = x_k + hf(t_k, x_k)$$

The way it works is by first dividing the whole domain into N pieces, each of length h. For example, if the domain is the interval $[a, b]$, then we have $h = \frac{b-a}{N}$. From here, we can redefine the algorithm as such:

$$x(0) = x_a$$

$$s_{i+1} = s_i + hf(t_i, s_i), fori \in [0, N)$$

From here, we can iterate through t and evaluate the function $f(t, x)$ to get some approximation for $x_i \approx x(t)$ given our starting point.
From here we wrote the algorithm in python and plotted the difference to the real solution. The approximation is so good that only for values of n greater than 1.75 can one even tell that there is a difference between the two.

Modified Euler's

The natural approach to increase accuracy would be to make h smaller, thus reducing the step size. This would surely reduce the error but would also drastically increase the computational time and power needed. Our goal would be to increase accuracy while maintaing a large enough time step h. One idea would be to define x_{k+1} as follows:

$$x_{k+1} = x_k + \frac{1}{2}h[f(t_k, x_k) + f(t_{k+1}, x_{k+1})]$$

From here we can use the trapizoid rule for integration along with an approximation of $f(t_{k+1}, x_{k+1})$ to get

$$x_{k+1} = x_k + \frac{1}{2}h[f(t_k, x_k) + f(t_k + h, x_k f(t_k, x_k))]$$

Finally we can apply the midpoint method and get the following algorithm to impliment into python:

$$x(0) = x_a$$

$$x_{i+1} = x_i + hf(t_i + \frac{h}{2}, x_i + \frac{h}{2}f(t_i, x_i))$$

Essentially this method is the combination of the midpoint method and Euler's method.

Huen's Method

This method takes the Modifieid Euler's Method and weighs it more towords the midpoint evaluation. The system is defined as follows:

$$x(0) = x_a$$

$$x_{i+1} = x_i + \frac{h}{4}[f(t_i, x_i) + 3f(t_i + \frac{2}{3}h, x_i + \frac{2}{3}hf(t_i, x_i))]$$

Runge- Kutta

This method utilizes Taylor's method and modifies it such that we don't actually need to compute higher order derivatives. The most common version of RK is the 4th order RK4, which is defined as follows:

$$x_o = a$$

$$k_1 = hf(t_i, x_i)$$

$$k_2 = hf(t_i + \frac{h}{2}, x_i + \frac{k_1}{2})$$

$$k_3 = hf(t_i + \frac{h}{2}, x_i + \frac{k_2}{2})$$

$$k_4 = hf(t_{i+1}, x_i + k_3)$$

$$x_{i+1} = x_i + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4)$$

Essentially, k_1 is the slope at the start point, approximated by Euler's Method. k_2 is the slope at the midpoint of the interval approximated by y and k_1.k_3 is still the slope at the midpoint of the interval but now approximated by y and k_2. k_4 is the slope at the end point approximated by y and k_3. When weighing the 4, clearly the two midpoint approximations are weighed most heavily.

Error Analysis

The chart below displays all the methods we tried and their error from the actual solution. This was done by plotting each of the solutions and computing the deviation from the real plot. As you one can see, Runge-Kutta had the smallest error, meaning it was the best choice. There does exist a trade off however, as the higher accuracy requires some more computation and in some cases one might prefer a looser approximation that comes much quicker.

METHOD	RELATIVE ERROR
Euler	0.722245
Midpoint	0.00268413
Modified	0.00921544
Heun	0.00486124
4th-order R-K	1.40903e-07

Conclusion

In conclusion, approximations are used extremely often when computing the solutions to ODE's and IVP's. These methods of solution might seem to appear out of no where but all build off one another and often can be extremely accurate. Some methods, like RK4, can be accurate up to 8 decimal digits! The trade off between accuracy and computational time is one that must be managed but with proper implementation of the algorithms, one can speed up the process.