Racket Numerical Methods Solver

Project Description

Racket programs designed to solve different types of mathematical problems using numerical methods. Each type has a different method implemented that will be described in the "Methods Included" section. The problems solved are the following:

- Non-Linear Equations
- Numerical Integration
- Differential Equations

Use Cases

Modern-day problems require modern-day solutions. Many engineering-related problems consist of systems involving complex equations or tedious formulas that represent a bottleneck in an engineer's workflow. There are areas such as process engineering that would benefit from this system, because, with fast algorithms and the capability to input a lot of data from files the time consumed in calculations becomes minimal.

Run

Once the progam is complete, in this section there will be a detailed description on how to run the program and input the required data.

Methods Included

Non-Linear Equations

- ☐ Secant Method

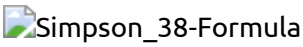
Root calculation on equations is commonly used to determine different behaviors of a system. In most cases the quadratic general formula is used, but more often than not, equations can be very complicated to calculate in this fashion; hence, the use of numerical methods.
The secant method is a modification of the Newton-Raphson method that lacks derivation calculation in the iterating function; this can be represented as a formula:

Although closed methods such as bisection or false position are quite simpler to implement, these tend to be less optimal than the open methods and they have to receive two values to calculate the estimate. The main difference between this method and other closed methods is the lack of derivative calculations that are a bit more complex to implement.

Numerical Integration

- ☐ Simpson ⅜

In engineering or process related fields, calculus is necessary to measure the changes within a system. Numerical methods are commonly frowned upon mathematicians due to their approximation-related nature; but the truth is that some systems require results in a quick fashion from difficult functions that take a lot of time to integrate, hence, the existence of integration algorithms based on numerical methods. Simpson methods consist in the calculation of higher degree polynomials that pass through points within a function. The rule can be expressed by the following expression:

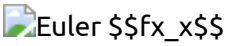


There are other methods that are quite simple derived from the Newton-Cotes method. These are fairly simple algorithms to develop but tend to be less precise per cycle than Simpson based methods. The selection of ⅜ instead of ⅓ is also related to precision terms.

Differential Equations

- ☐ Euler

The real life usage of the method is identical from the Integration method, but applicable to derivative calculations.
Euler's method consists of an estimated value that works with the slope of two points within a function to extrapolate the value every "h" units represented as the "step". The representation is the following:



Due to the complex nature of differential equations calculations, Euler's method will be the function to implement due to it's acceptable precision and easy-to-implement nature.

Topics that will be used

- **Recursion.** This topic will be used extensively in the project as all the iterations will be done using recursion
- **Lists.** Lists will be used to store the initial information of the target function to apply the method, as well as a possible way to store results after each method iteration
- **File I/O.** The files will be used to store initial information of the functions, meaning that calculations can be done by uploading a single document that was prepared beforehand
- **Functional programming.** as the language that was chosen for this project is Racket, it leans towards accepted functional programming 'good practices'