Scientific Programming

Assignment: Polymorphism.

Anand Kamble

amk23j@fsu.edu

31th October 2023

This program utilizes two classes, namely Newton and Secant, both derived from the Solver class. As both classes inherit from Solver, they can be seamlessly managed using the common interface provided by the base class. This exemplifies the principle of polymorphism in C++.

Code Modules

The code is organized in the following modules:

- `Newton.cpp` : This class inherits from the Solver base class and provides an implementation of Newton's method for finding roots of a target function .
- `Secant.cpp` : This class inherits from the Solver base class and provides an implementation of the Secant method for finding roots of a target function.
- `Solver.cpp` : This class provides a common interface for solving equations using numerical methods, such as Newton's method. Derived classes must implement the ComputeRoot and Verify methods
- `Main.cpp` : Function to solve a system of equations using Newton's method and the Secant method.

Formatting Conventions:

a. Casing: Camel-Case
b. Indentation: 4 spaces
c. Line Break: CRLF

Functions

Using the program, we will be solving following functions using the Newton’s method and the Secant method,

$$f(x) = sin(x)$$
$$f(x) = x^3 - 6x^2 + 11x - 6$$
$$f(x) = log(x) + x^2 - 3$$

Testing and Verification

Verification tests ensure computed roots satisfy the equations $f(x) = 0$. Computed roots are tested against expected outputs.

Output

Computed roots, iterations, and verification errors are output for Newton's and Secant methods.

Execution

The Makefile is included with this code. You can run the command ‘make’ to compile the program. After successful compilation, you can find the executable named ‘a.out’ inside the bin folder. Run this executable by ‘./bin/a.out’. To clean the generated folders and files, use the command ‘make clean’.