Numerical Methods: Implementation and Analysis

Your Name

September 20, 2024

1 Introduction

This report presents the implementation of three numerical methods: Bisection Method, Newton's Method, and Secant Method. Each method is tested on various functions and scenarios, as per the programming assignment requirements.

2 A. Abstract Base Class and Derived Classes

2.1 Problem Statement

Implement an abstract base class `EquationSolver` with a pure virtual method `solve`, and derive classes for each numerical method.

2.2 Analysis

The abstract class `EquationSolver` provides a common interface for all derived classes. Each derived class implements the specific logic required for the corresponding numerical method.

2.3 Code Explanation

The following code snippet shows the base class and derived classes for the methods:

```
class EquationSolver {
public:
    virtual double solve() = 0; // Pure virtual function
};

class Bisection_Method : public EquationSolver {
    // Implementation of the Bisection Method
};
```

```
class Newton_Method : public EquationSolver {
    // Implementation of Newton's Method
};

class Secant_Method : public EquationSolver {
    // Implementation of the Secant Method
};
```

3 B. Bisection Method Testing

3.1 Problem Statement

Test the Bisection Method on the following functions:

- $f(x) = \frac{1}{x} - \tan(x)$ on $[0, \frac{\pi}{2}]$
- $f(x) = \frac{1}{x} - 2^x$ on $[0, 1]$
- $f(x) = 2^{-x} + e^x + 2\cos(x) - 6$ on $[1, 3]$
- $f(x) = \frac{x^3+4x^2+3x+5}{2x^3-9x^2+18x-2}$ on $[0, 4]$

3.2 Analysis

The Bisection Method is a root-finding method that repeatedly bisects an interval and selects a subinterval in which a root must lie. It is effective for continuous functions where the values at the endpoints have opposite signs.

3.3 Code Explanation

The implementation of the Bisection Method involves defining a class that overrides the `solve` method. The function evaluates the midpoint and checks for the root condition.

```
// Bisection Method Implementation
double Bisection_Method::solve() {
    // Bisection logic
}
```

4 C. Newton's Method Testing

4.1 Problem Statement

Use Newton's Method to solve $x = \tan(x)$ near the roots 4.5 and 7.7.

4.2 Analysis

Newton's Method uses the derivative to approximate the root. It converges rapidly when close to the actual root but can diverge if the initial guess is not suitable.

4.3 Code Explanation

The derived class for Newton's Method implements the `solve` method by iterating until convergence.

```
// Newton's Method Implementation
double Newton_Method::solve() {
    // Newton's iteration logic
}
```

5 D. Secant Method Testing

5.1 Problem Statement

Test the Secant Method on the following functions with specified initial values:

- $f(x) = \sin\left(\frac{x}{2}\right) - 1$ with $x_0 = 0, x_1 = \frac{\pi}{2}$
- $f(x) = e^x - \tan(x)$ with $x_0 = 1, x_1 = 1.4$
- $f(x) = x^3 - 12x^2 + 3x + 1$ with $x_0 = 0, x_1 = -0.5$

5.2 Analysis

The Secant Method approximates the root using two initial guesses and is generally faster than the Bisection Method but may fail to converge.

5.3 Code Explanation

The Secant Method is implemented by evaluating the function at the initial guesses and iterating based on the secant line.

```
// Secant Method Implementation
double Secant_Method::solve() {
    // Secant iteration logic
}
```

6 E. Trough Volume Problem

6.1 Problem Statement

Given a trough with length $L = 10$, radius $r = 1$, and volume $V = 12.4$, find the depth of water using all three methods.

6.2 Analysis

The volume formula involves trigonometric and square root calculations. Each numerical method is applied to solve for the depth h.

6.3 Code Explanation

The implementation involves defining a class for the volume function and invoking the methods accordingly.

```
// Trough Volume Implementation
class TroughVolume : public Function {
    // Volume function implementation
};
```

7 F. Vehicle Failure Analysis

7.1 Problem Statement

Using Newton's Method and the Secant Method, find the maximum angle α under specified conditions.

7.2 Analysis

This section involves understanding the implications of vehicle dynamics in terrain negotiation. Different initial values can lead to different outcomes, highlighting the importance of the choice of initial guess in numerical methods.

7.3 Code Explanation

The angle calculation uses trigonometric relationships defined in the function class for vehicle failure.

```
// AlphaEquation Implementation
class AlphaEquation : public Function {
    // Function to evaluate alpha based on parameters
};
```

8 Conclusion

This report demonstrated the implementation and testing of Bisection, Newton's, and Secant Methods. Each method was evaluated on various functions, providing insight into their strengths and weaknesses. Future work may include improving convergence rates and handling edge cases more robustly.