190503

MTH2008

Scientific computing

Second semester

5th lecture

Danilo Roccatano - Marco Pinna

Office : INB3129

Email: Mpinna@lincoln.ac.uk

Padlet

https://uol.padlet.org/mpinna2/scientific-computing-l5c5cjv5fmvv72l0

Logbook

- The logbook structure will follow the same structure as previously explained in the first semester
- Only the part of the second semester topics should be sumbitted.

Notes

For each lecture, I have created a program folder containing all the relevant code. Please avoid cutting and pasting directly from the PowerPoint slides, as this may result in errors (not always). Instead, ensure you copy the code directly from the program folder.

Overview Semester B

NUMERICAL METHODS AND ALGORITHMS

- Random Numbers
- Sorting algorithm
- Numerical Differentiation
- Root of a equations
  - Bisection Method
  - Newton Method
  - Secant Method
- Numerical Integration
- Numerical integration of Ordinary Differential Equation (first order)
- Linear Algebra
  - Product of scalar with vectors/matrix
  - Product of two matrix

C++ Coding

- Open, read and write a file in C++
- Dynamical arrays

Task 4.2: Differentiate sin (x)

1. User inputs
   - The value of x_0 (the point where the derivative is approssimate)
   - The starting value of h
   - The number of iterations N=500

task4_2.cpp

2. Output
   - Display each iteration with approximation from all three methods
   - Their corresponding errors, value of h.
   - Absolute and relative error
   - Write interation numbers and approximation to a file (difference_exercise_results.txt).

Task 4.2: Differentiate sin (x)

Extra tasks-

- Identify which method provides the smallest error
- Determine how the error decreases as h becomes smaller

Modify h and N.

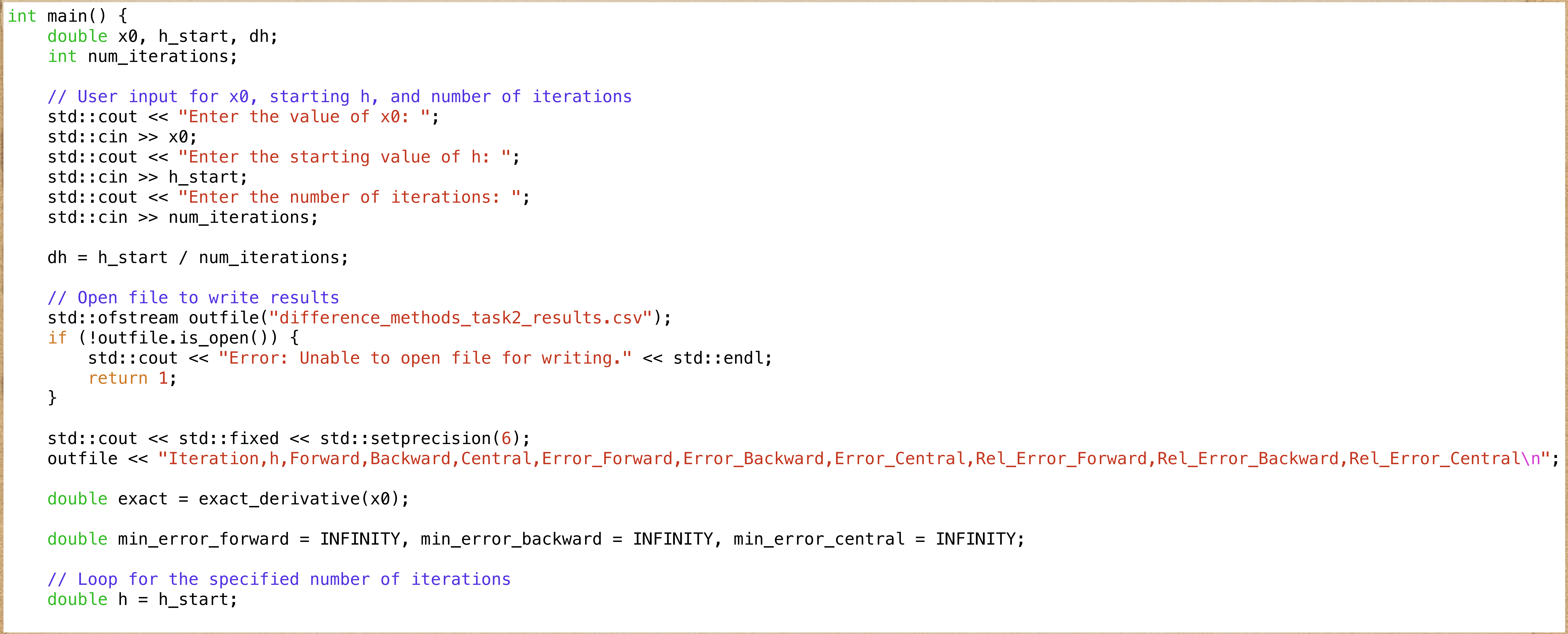
- h=0.5 and N=500
- h=0.1 and N=2000
- Observe how the results change.

Graphical representation

- Import the data from the file to Excel or using Matlab,Python.
- Plot error vs iteration number for the three different methods.

Solution: task 4.2

```cpp
#include <iostream>
#include <fstream>
#include <cmath>    // For sin() function
#include <iomanip>  // For setting precision

// Function to differentiate (changeable)
double function(double x) {
    return sin(x);  // You can change this to any differentiable function
}

// Exact derivative of the function (changeable)
double exact_derivative(double x) {
    return cos(x);  // Derivative of sin(x)
}

// Forward difference method
double forward_difference(double x0, double h) {
    return (function(x0 + h) - function(x0)) / h;
}

// Backward difference method
double backward_difference(double x0, double h) {
    return (function(x0) - function(x0 - h)) / h;
}

// Central difference method
double central_difference(double x0, double h) {
    return (function(x0 + h) - function(x0 - h)) / (2 * h);
}

// Function to calculate relative error
double relative_error(double approx, double exact) {
    return fabs((approx - exact) / exact);
}
```



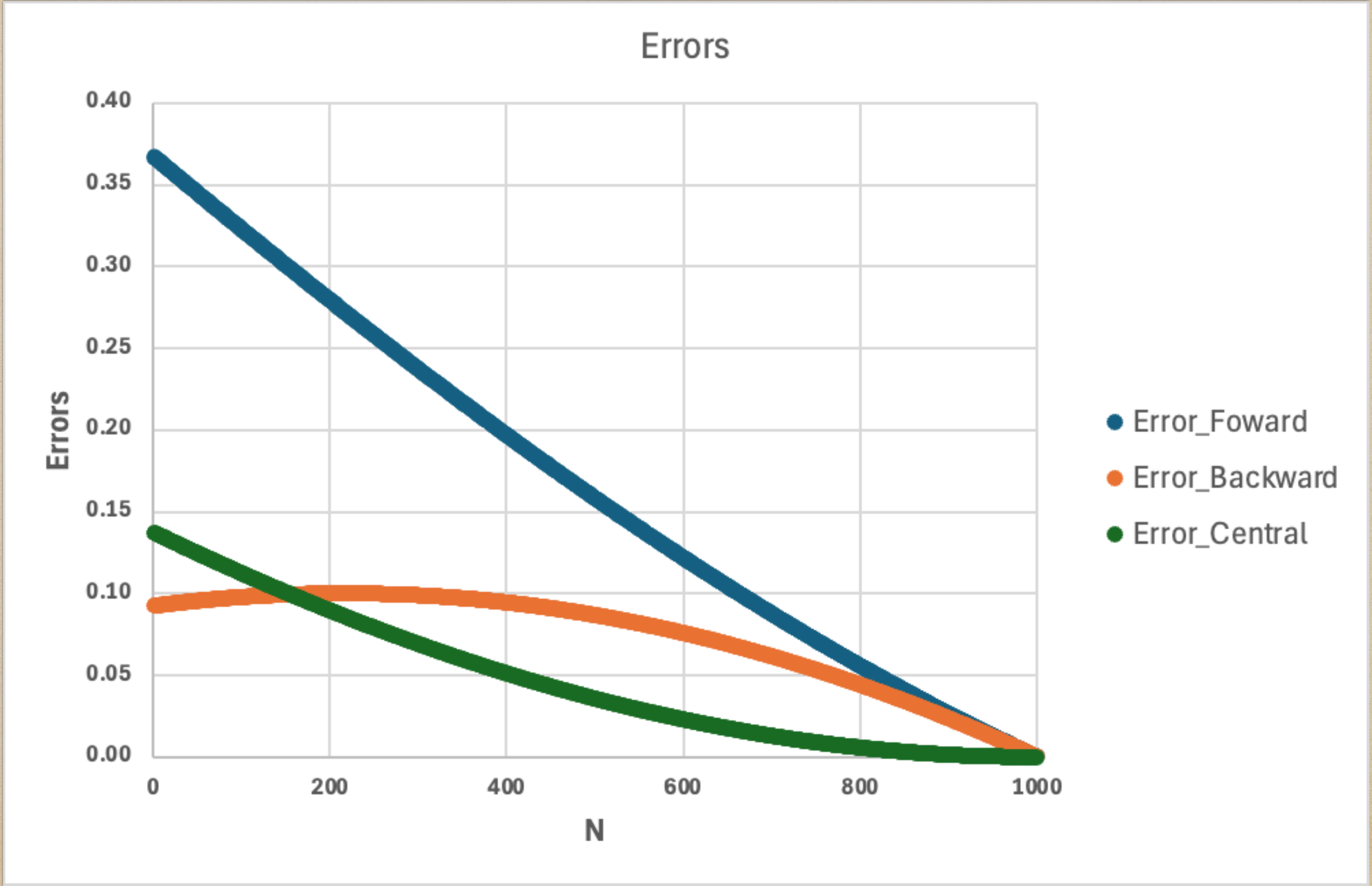
```cpp
int main() {
    double x0, h_start, dh;
    int num_iterations;

    // User input for x0, starting h, and number of iterations
    std::cout << "Enter the value of x0: ";
    std::cin >> x0;
    std::cout << "Enter the starting value of h: ";
    std::cin >> h_start;
    std::cout << "Enter the number of iterations: ";
    std::cin >> num_iterations;

    dh = h_start / num_iterations;

    // Open file to write results
    std::ofstream outfile("difference_methods_task2_results.csv");
    if (!outfile.is_open()) {
        std::cout << "Error: Unable to open file for writing." << std::endl;
        return 1;
    }

    std::cout << std::fixed << std::setprecision(6);
    outfile << "Iteration,h,Forward,Backward,Central,Error_Forward,Error_Backward,Error_Central,Rel_Error_Forward,Rel_Error_Backward,Rel_Error_Central\n";

    double exact = exact_derivative(x0);

    double min_error_forward = INFINITY, min_error_backward = INFINITY, min_error_central = INFINITY;

    // Loop for the specified number of iterations
    double h = h_start;
    for (int i = 1; i <= num_iterations; i++) {
        double forward_approx = forward_difference(x0, h);
        double backward_approx = backward_difference(x0, h);
        double central_approx = central_difference(x0, h);

        double error_forward = fabs(forward_approx - exact);
        double error_backward = fabs(backward_approx - exact);
        double error_central = fabs(central_approx - exact);

        double rel_error_forward = relative_error(forward_approx, exact);
        double rel_error_backward = relative_error(backward_approx, exact);
        double rel_error_central = relative_error(central_approx, exact);

        // Track minimum errors
        if (error_forward < min_error_forward) min_error_forward = error_forward;
        if (error_backward < min_error_backward) min_error_backward = error_backward;
        if (error_central < min_error_central) min_error_central = error_central;

        // Display the results on the screen
        std::cout << "Iteration: " << i << ", h: " << h << "\n"
                  << "  Forward Approximate: " << forward_approx << ", Error: " << error_forward << "\n"
                  << "  Backward Approximate: " << backward_approx << ", Error: " << error_backward << "\n"
                  << "  Central Approximate: " << central_approx << ", Error: " << error_central << "\n";

        // Write the approximations and errors to the CSV file
        outfile << i << "," << h << "," << forward_approx << "," << backward_approx << "," << central_approx
                << "," << error_forward << "," << error_backward << "," << error_central
                << "," << rel_error_forward << "," << rel_error_backward << "," << rel_error_central << "\n";

        h -= dh; // Decrease h by dh for the next iteration
    }

    outfile.close();

    // Summary of minimum errors
    std::cout << "\nSummary of Minimum Errors:" << std::endl;
    std::cout << "  Minimum Forward Error: " << min_error_forward << std::endl;
    std::cout << "  Minimum Backward Error: " << min_error_backward << std::endl;
    std::cout << "  Minimum Central Error: " << min_error_central << std::endl;

    std::cout << "Results have been written to difference_methods_task2_results.csv" << std::endl;

    return 0;
}
```

Have you spotted any mistake?

Solution: task 4.2

```cpp
int main() {
    double degrees, x0, h_start, dh;
    int num_iterations;

    // User input for angle in degrees
    std::cout << "Enter the angle in degrees: ";
    std::cin >> degrees;

    // Convert degrees to radians
    x0 = degrees * (M_PI / 180.0);
```

Solution: task 4.2



Overview Semester B

NUMERICAL METHODS AND ALGORITHMS

- Random Numbers
- Sorting algorithm
- Numerical Differentiation
- Root of a equations
  - ➢Bisection Method
  - ➢Newton Method
  - ➢Secant Method
- Numerical Integration
- Numerical integration of Ordinary Differential Equation (first order)
- Linear Algebra
  - ➢Product of scalar with vectors/matrix
  - ➢Product of two matrix

C++ Coding

- Open, read and write a file in C++
- Dynamical arrays

Topic for this session

NUMERICAL METHODS AND ALGORITHMS

- Random Numbers
- Sorting algorithm
- Numerical Differentiation (higher-order)
- Root of a equations
  - ➢Bisection Method
  - ➢Newton Method
  - ➢Secant Method
- Numerical Integration
- Numerical integration of Ordinary Differential Equation (first order)
- Linear Algebra
  - ➢Product of scalar with vectors/matrix
  - ➢Product of two matrix

C++ Coding

- Open, read and write a file in C++
- Dynamical arrays

Refresh previous session

Forward difference formula

$$f'(x) \approx \frac{f(x+h) - f(x)}{h}$$

Backward difference formula

$$f'(x) \approx \frac{f(x) - f(x-h)}{h}$$

Central difference formula

$$f'(x) \approx \frac{f(x+h) - f(x-h)}{2h}$$

Numerical differentiation

for 3 points $(x_0, x_0 + h, x_0 + 2h)$

$$f(x_0 + h) = f(x_0) + hf'(x_0) + \frac{h^2}{2}f''(x_0) + \frac{h^3}{6}f'''(x_0)$$

$$f(x_0 + 2h) = f(x_0) + 2hf'(x_0) + 2h^2f''(x_0) + \frac{4h^3}{3}f'''(x_0)$$

$$Af(x_0) + Bf(x_0 + h) + Cf(x_0 + 2h)$$

$$= Af(x_0) + B\left(f(x_0) + hf'(x_0) + \frac{h^2}{2}f''(x_0)\right) + C\left(f(x_0) + 2hf'(x_0) + 2h^2f''(x_0)\right)$$

Numerical differentiation

Central difference formula

$$= (A + B + C)f(x_0) + (B + 2C)hf'(x_0) + \left(\frac{B}{2} + 2C\right) h^2 f''(x_0)$$

$$\begin{cases} A + B + C = 0 & \text{(eliminate the constant term } f(x_0)\text{)} \\ B + 2C = 1 & \text{(ensure the coefficient of } f'(x_0) \text{ is correct)} \\ \frac{B}{2} + 2C = 0 & \text{(eliminate the second derivative term } f''(x_0)\text{)} \end{cases}$$

$$A = -\frac{3}{2}, \quad B = 2, \quad C = -\frac{1}{2}$$

$$f'(x_0) \approx \frac{-3f(x_0) + 4f(x_0 + h) - f(x_0 + 2h)}{2h}$$

2nd order

Forward difference formula

Let take again two points $(x_0, x_0 + h)$; we want to calculate $f'(x)$

eq. 1 $$f'(x) \approx \frac{f(x+h) - f(x)}{h}$$ and $f(x_0 + h)$ is given by:

eq. 2 $$f(x+h) = f(x) + hf'(x) + \frac{h^2}{2} f''(x) + O(h^3)$$

therefore:

eq. 3 $$f''(x) \approx \frac{2}{h^2} \big(f(x+h) - f(x) - hf'(x)\big)$$

2nd order

substituting eq.1 into eq. 2

eq. 4 $$f''(x) \approx \frac{2}{h^2}\left(f(x+h) - f(x) - h \cdot \frac{f(x+h) - f(x)}{h}\right)$$

which simplify to:

$$f''(x) \approx 0$$

This demonstrates that we need at least three points to estimate $f''(x)$ correctly.

Correct 2nd order calculation

Forward difference formula

Let take 3 points $(x_0, x_0 + h, x_0 + 2h)$

$$f'(x) \approx \frac{f(x_0 + h) - f(x_0)}{h}$$

$$f(x_0 + h) = f(x_0) + hf'(x_0) + \frac{h^2}{2} f''(x_0) + \frac{(h)^3}{6} f'''(x_0) + O(h^4)$$

$$f(x_0 + 2h) = f(x_0) + 2hf'(x_0) + \frac{(2h)^2}{2} f''(x_0) + \frac{(2h)^3}{6} f'''(x_0) + O(h^4)$$

we can write again

$$f''(x_0) \approx Af(x_0) + Bf(x_0 + h) + Cf(x_0 + 2h)$$

Correct 2nd order calculation

Forward difference formula

$$= Af(x_0) + B\left(f(x_0) + hf'(x_0) + \frac{h^2}{2}f''(x_0) + \frac{h^3}{6}f'''(x_0)\right)$$

$$+C\left(f(x_0) + 2hf'(x_0) + \frac{4h^2}{2}f''(x_0) + \frac{8h^3}{6}f'''(x_0)\right) + O(h^4)$$

Correct 2nd order calculation

Forward difference formula

Now regrouping the terms we have:

$$(A+B+C)f(x_0)+(B+2C)hf'(x_0)+\left(\frac{B}{2}+2C\right)h^2f''(x_0)$$

Solving for（A, B, C）

$$\begin{cases} A+B+C=0 & \text{(eliminate the constant term } f(x_0)) \\ B+2C=0 & \text{(eliminate } f'(x_0)) \\ \frac{B}{2}+2C=1 & \text{(ensure the coeefficient for } f''(x_0)) \end{cases}$$

Correct 2nd order calculation

Forward difference formula

$$\boxed{f''(x_0) \approx \frac{f(x_0) - 2f(x_0 + h) + f(x_0 + 2h)}{h^2}}$$

2nd order calculation

Central difference formula

Let take 3 points $(x_0 - h, x_0, x_0 + h)$

$$f(x+h) = f(x) + hf'(x) + \frac{(h)^2}{2}f''(x) + \frac{(h)^3}{6}f'''(x) + O(h^4)$$

$$f(x-h) = f(x) - hf'(x) + \frac{(h)^2}{2}f''(x) - \frac{(h)^3}{6}f'''(x) + O(h^4)$$

We can add the eq. 2 to the eq. 1:

$$f(x+h) + f(x-h) = 2f(x) + 2\frac{(h)^2}{2}f''(x) + O(h^4)$$

and obtain

$$2\frac{(h)^2}{2}f''(x) \approx -f(x+h) - f(x-h) + 2f(x)$$

2nd order calculation

Central difference formula

From the previous calculation we obtain

$$2\frac{(h)^2}{2}f''(x) \approx f(x+h) + f(x-h) - 2f(x)$$

$$f''(x) \approx \frac{f(x+h) - 2f(x) + f(x-h)}{h^2}$$

$$\boxed{f''(x) \approx \frac{f(x+h) - 2f(x) + f(x-h)}{h^2}}$$

2nd order calculation

Summary

Forward difference formula $\boxed{f''(x_0) \approx \frac{f(x_0) - 2f(x_0 + h) + f(x_0 + 2h)}{h^2}}$

Central difference formula $\boxed{f''(x) \approx \frac{f(x + h) - 2f(x) + f(x - h)}{h^2}}$

Backward difference formula $\boxed{f''(x_0) \approx \frac{f(x_0) - 2f(x_0 - h) + f(x_0 - 2h)}{h^2}}$

Task 5.1

Task 5.1: Deriving the Third-Order Finite Difference Formulas

- Backward difference formula - Approximate $f'''(x_0)$ using function values at $(x_0, x_0 - h, x_0 - 2h, x_0 - 3h)$.

- Forward difference formula - Approximate $f'''(x_0)$ using function values at $(x_0, x_0 + h, x_0 + 2h, x_0 + 3h)$

- Central difference formula - Approximate $f'''(x_0)$ using function values at $(x_0, x_0 - h, x_0 + h, x_0 + 2h)$

-

Task 4.2

Modify the numerical differentiation program to compute the **first-order, second-order, and third-order** derivatives of the sin(x) function using finite difference methods. The program should allow the user to define **step size reduction** dynamically.

Instructions:

1. User input
   - An angle in degree, which will be converted in radians
   - The starting step h=1.
   - The number of iterations N, determining how h will be reduced
2. Numerical implementation
   - 2nd order (central difference formula)
   - 2nd order (backward difference formula)

3. Computation details

- The program will reduce h at each step by a fraction determined by the user.
- The program should store the approximation values of the 3 methods, and the corresponding error

4. Error analysis

- The program should calculate the absolute error
- The program should calculate the relative error

5. Output

- The program should display the results on the screen and write these in .csv file
- The output should include: Iteration numbers, the step size h, approx for the order methods, absolute and relative errors.

Percentage error

$$\%\, err = \frac{|V_a - V_e|}{V_a}$$

Task 4.3

Extend the previous program and evaluate the derivative over an interval [a,b] using h=0.001.

- Instead of entering a value for x_0, the user should enter a range [a,b] in degrees.

- The program will compute the derivatives at every interation step (a-b)/N where N is the number of iteration or divisions.

- Repeat writing and output for all approximate methods.

- write the errors (relative and abolute values) for each approximation

Questions?