Project 1 Report - John Akujobi

Numerical Differentiation Error Analysis Web App

Author: John Akujobi

Course: Math 374 – Scientific Computing

Date: Spring 2024

Project Question:

Here we consider two numerical differentiation formulas,

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

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

Study and compare the two formulas for $f(x) = \sin x$ and $x = 1$ as $h \to 0$.

Tasks:

1. Find truncation error bounds for (1) and (2).
2. Find rounding error bounds for (1) and (2).
3. On the two graphs for (1) and (2), plot truncation error bound, rounding error bound, and total error using a log-scale;
   The axes in the plot should be $\log_{10} |\text{error}|$ versus $\log_{10} h$ as $h = 10^{-k}, k = 1, \ldots, 16$.
4. Discuss the optimal values of (h) and the relations between errors.
5. Compare (1) and (2) for your conclusion.

---

1. Introduction

Through a DuckDuckGo search, I found out that the two numerical differentiation methods were called:

1. the forward difference formula
2. the central difference formula.

First we need to explore them and approximate the derivative of the function f(x)=sin(x) at x = 1 and to analyze the errors associated with each method.

Specifically, we calculate:

- Truncation Error: Error from neglecting higher order terms in the Taylor series.
- Rounding Error: Error due to the finite precision (machine epsilon) of computer arithmetic.
- Total Error: The combination of truncation and rounding errors.

How it is presented.

I presented the project as an interractive web app built with Streamlit. THere, you can adjust parameters (like the range of h values and machine epsilon). And you can see log-log plots that show how the errors change as the step size hh varies.

---

2. Background and Theory

2.1. Numerical Differentiation

Numerical differentiation involves approximating the derivative of a function using its values at nearby points. Two widely used methods are:

- Forward Difference Formula:

  $f'(x) \approx \frac{f(x+h)-f(x)}{h}$
- Central Difference Formula:

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

2.2. Error Analysis

Here are the errors in numerical differentiation

Truncation Error:

Derived by expanding f(x+h) (and f(x−h)f(x-h) for the central difference) in a Taylor series.

- For the forward difference , the expansion gives:

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

  The leading error term is proportional to h (i.e., $O(h)$).
- For the central difference , the expansion gives us:

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

  The truncation error is proportional to h^2 (i.e., $O(h^2)$). This means a higher accuracy.

Rounding Error:

Due to the limitations of finite precision arithmetic (quantified by machine epsilon, ϵ). When we subtract numbers that are nearly equal, (this happens in both formulas), the relative error is increased roughly by a factor of 1/h.

Total Error:

This is sum of the truncation error and rounding error.

As h decreases, truncation error reduces while rounding error increases. There is a sweet spot, an optimal value of h where these competing effects balance to minimize the total error.

---

3. Project Objectives

The key objectives of the project are:

1. Find the theoretical truncation and rounding error bounds for both differentiation formulas.
2. **Compute Errors Numerically:**

   Use Python to calculate the actual error (the difference between the numerical approximation and the true derivative), the truncation error, and the rounding error over a range of hh values.
3. **Visualization:**

   Plot the errors on a log-log scale to clearly observe the relationship between hh and the different types of errors.
4. **Interactivity:**(Self assigned)

   Build an interactive web app with Streamlit where users can adjust input parameters (such as the range of h values and the machine epsilon) and seeupdated plots and optimal hh values.
5. **Comparison and Conclusion:**

   Compare the forward and central difference methods. Then discuss the optimal h for each, and conclude which method is more accurate and stable.

---

4. Implementation Details

4.1. Configuration & Constants

- **Page Setup:**

  The `configure_page()` function sets up the Streamlit page. It includes the title, layout, and some custom CSS for styling. I also added MathJax support for properly rendering LaTeX formulas.

4.2. Theoretical Background

- Theory Display:

  The `show_theory()` function uses an expandable section to present the theoretical background of the forward and central difference methods. This includes the order of truncation and rounding errors for each method.

4.3. Controls & Inputs

- User Input Controls:

  `get_user_inputs()` function makesa sidebar in the Streamlit app where users can select:
  - The minimum and maximum exponents for h (e.g., $h = 10^{-k}$).
  - The number of points to generate between these values.
  - The machine epsilon (ϵ). This is is used in the rounding error calculation.

4.4. Error Calculations

- Calculating Errors:

  The `calculate_errors()` function iterates over an array of h values to compute:
  - Actual error: Difference between the numerical derivative and the exact derivative ($\cos(1)$).
  - Truncation error bounds: Based on the theoretical derivations:
    - For forward difference: h/2
    - For central difference: $h^2/6$
  - Rounding error bounds: Estimated as:
    - For forward difference: $2\epsilon/h$
    - For central difference: ϵ/h

4.5. Visualization

- Plotting the Data:

  The `create_error_plot()` function makesa log-log plot for either the forward or central difference method. It plots:

  - The actual error.
  - The truncation error bound.
  - The rounding error bound.

  Both plots are integrated into the Streamlit app, allowing for side-by-side comparisons.

4.6. Optimal Values Calculation

- Finding the Optimal hh:

The `calculate_optimal_values()` function computes the optimal step sizes for both methods:

- **Forward difference optimal h:** $h_{\text{opt}} \approx \sqrt{2\epsilon}$
- **Central difference optimal h:** $h_{\text{opt}} \approx \sqrt[3]{3\epsilon}$

These values represent the balance point where the total error (truncation plus rounding) is minimized.

4.7. Main App Flow

- **Orchestrating the App:**

  The `main()` function ties together all of the components:
  - Configures the page and sets up the title.
  - Displays the theoretical background.
  - Collects user inputs.
  - Generates hh values, calculates errors, and displays the visualizations.
  - Presents the computed optimal hh values and a comparison table that summarizes the differences between the two methods.

---

5. Running the App

Prerequisites

- **Python:** Version 3.13
- **Libraries:** NumPy, Matplotlib, and Streamlit

  (Install them with pip: `pip install numpy matplotlib streamlit`)

Execution

1. Save the code in a file (e.g., `streamlit_app.py`).
2. In your terminal or command prompt, navigate to the directory containing the file.
3. Run the command:

   ```
   streamlit run streamlit_app.py
   ```

4. Your browser will open an interactive page where you can explore the error analysis.

---

6. Results & Discussion

- **Visualizations:**

  The app showstwo side-by-side plots (one for each differentiation method) on a log-log scale. This makes it easier to understand how the actual error, truncation error, and rounding error changewith h.
- **Error Behavior:**
  - For larger h, the truncation error dominates.
  - As h decreases, rounding error becomes more significant.
  - There is an optimal h where the total error (sum of truncation and rounding errors) is minimized.
- **Method Comparison:**

  The central difference method, with a truncation error of $O(h^2)$, generally offers better accuracy and stability than the forward difference method (with O(h) truncation error).
- **Optimal hh Values:**

  The app calculates and displays the optimal hh for both methods, demonstrating the balance between decreasing truncation error and increasing rounding error.

---

7. Conclusion

This project successfully demonstrates the key concepts in numerical differentiation error analysis. By:

- Finding the theoretical error bounds,
- Implementing the numerical computations in Python,
- Visualizing the results interactively with Streamlit,

I could easily understand the trade-offs between truncation and rounding errors.

The project shows that the central difference method is more accurate for the same hh values. THis shows why it is important to choosean optimal h to reduce totalerror.
