Mathematics & Statistics Research Competition

Linge SHI and Yenuli De Coonghe
Box Hill High School

1. Abstract
As part of the 2025 Mathematics and Statistics Research Challenge. This research investigates numerical methods for approximating the first derivative of a function using finite difference techniques. The study explores the mathematical basis of these approximations through Taylor expansions. A range of test functions is used to compare the accuracy and behavior of each method, including smooth, polynomial, and non-differentiable functions. Results demonstrate how approximation performance varies depending on function smoothness and the step size h.

Contents

2. Restatement and interpretation of topic Topic S–06

Derivatives are used to calculate the rate of change of a graph. In maths, the value of *h* is often reduced to 0 to calculate the instantaneous rate of change. But in the real world, we are restricted by physical properties, where h cannot equal zero. This is when we rely on approximations.
According to the difference formulas:
forward difference $f'(a) \approx \frac{f(a+h)-f(a)}{h}$
backward difference
and centred difference $f'(a) \approx \frac{f(a)-f(a-h)}{h}$
$f'(a) \approx \frac{f(a+h)-f(a-h)}{2h}$
Here, *h* is a small positive number that represents the discrete interval used in place of an infinitesimal change. Each of these approximations assumes a different approach to estimating the slope near the point *a*.

The forward and backward difference formulas are categorized as first-order approximations, meaning their error decreases linearly with smaller *h*. The centered difference formula, being a second-order approximation, has a faster rate of error reduction — the error decreases quadratically as *h* becomes smaller.

This topic encourages deeper exploration into how different approximation methods perform under varying conditions. Some functions yield accurate and stable derivative estimates with these methods, while others — due to properties such as discontinuity, sharp curvature, or oscillation — may cause significant inaccuracies.

Further investigations include:

- Comparing the accuracy of these approximations using custom-defined functions.
- Constructing a function that highlights the failure or instability of these methods.
- Extending the investigation to include approximations of second derivatives and functions with multiple variables.

This exploration combines core calculus ideas with numerical methods, leading into broader applications in computational mathematics, physics, and data science.

3.practical applications

In practical scenarios, derivative values must often be estimated from discrete measurements rather than continuous functions. A typical example is estimating the instantaneous speed of a moving object when only discrete position data points are available.

Consider a vehicle traveling along a straight path, whose position is recorded every second by a GPS device. At discrete time intervals *t = 1, 2, 3* seconds, the recorded positions (in meters) are as follows:

Time *t* (s)	Position *f(t)* (m)

1	4.2
2	5.8
3	7.1

The objective is to estimate the instantaneous velocity at *t = 2* seconds, i.e., the derivative of position with respect to time, *f'(2)*.

Using a time step h = 1 second, the derivative can be approximated by applying finite difference formulas:

Forward difference approximation:

$$f'(2) \approx \frac{f(3)-f(2)}{h} = \frac{7.1-5.8}{1} = 1.3\,\mathrm{m/s}$$

Backward difference approximation:

$$f'(2) \approx \frac{f(2)-f(1)}{h} = \frac{5.8-4.2}{1} = 1.6\,\mathrm{m/s}$$

Centered difference approximation:

$$f'(2) \approx \frac{f(3)-f(1)}{2h} = \frac{7.1-4.2}{2} = 1.45\,\mathrm{m/s}$$

To verify and illustrate the use of finite difference approximations computationally, a simple Python simulation was developed. This code implements the forward, backward, and centered difference formulas to estimate the instantaneous velocity of an object at *t = 2* seconds, based on the discrete position data. The numerical implementation reflects a common approach in scientific and engineering disciplines where data is collected at regular time intervals and continuous models are unavailable.

```
# Simulated GPS data (time in seconds, position in meters)
times = [1, 2, 3]
positions = [4.2, 5.8, 7.1]

# h is the time interval (assumed uniform here)
h = times[1] - times[0]

# Forward Difference (at t = 2s)
forward_diff = (positions[2] - positions[1]) / h

# Backward Difference (at t = 2s)
backward_diff = (positions[1] - positions[0]) / h

# Centered Difference (more accurate)
centered_diff = (positions[2] - positions[0]) / (2 * h)
```

```
# Print results
print("Estimated speed at t = 2 seconds:")
print(f"Forward Difference:  {forward_diff:.2f} m/s")
print(f"Backward Difference: {backward_diff:.2f} m/s")
print(f"Centered Difference: {centered_diff:.2f} m/s")
```

Output:
Estimated speed at t = 2 seconds:
Forward Difference: 1.30 m/s
Backward Difference: 1.60 m/s
Centered Difference: 1.45 m/s

In real life situations like this, the data is discrete, therefore the calculations rely on approximations.

4.What is meant by first- and second-approximation

In numerical analysis, an approximation is said to be first-order if its error is proportional to the step size h, and second-order if the error is proportional to h^2. This classification describes how rapidly the approximation error decreases as *h* becomes smaller.

A first-order approximation means the error shrinks linearly as h becomes smaller. That is, halving h will roughly halve the error.

A second-order approximation means the error shrinks quadratically as h decreases. Halving h will reduce the error by a factor of four.

This is a direct result of the taylor series expansion of a function.

For example, the forward difference formula:

$$f'(2) \approx \frac{f(3) - f(2)}{h} = \frac{7.1 - 5.8}{1} = 1.3\,\mathrm{m/s}$$

has a leading error term of $h/2f''(a)$, so the total error is $O(h)$.

Meanwhile, the centered difference formula:

$$f'(2) \approx \frac{f(3) - f(1)}{2h} = \frac{7.1 - 4.2}{2} = 1.45\,\mathrm{m/s}$$

has a leading error term of $h^2/6\, f'''(a)$, which gives it a smaller total error of Oh^2.

In summary:

First-order = error proportional to $h = \propto\ h$

Second-order = error proportional to $h^2 = \propto\ h^2$

Second-order methods are more accurate, especially when h is small.

5.Create functions to compare performance of the approximations

The accuracy of approximate calculations can vary depending on the type of function it is used on.

Step one, create some functions to test approximate derivatives on.

Here's a list of functions that will be used to test this on.

$f_1(x) = sin(x)$, periodic function, smooth.

$f_2(x) = x^2 - 2x + 1$, polynomial, easy calculation of derivative

$f_3(x) = \sqrt{x}$, this function has a sharp curve at $x=0$, ideal for testing approximation

Step two, choose a point a.

$f_1(x) = sin(x)$ *$\rightarrow a=\pi/4\approx0.7854$*

$f_2(x) = x^2 - 2x + 1$ *$\rightarrow a=1$ (where it's flat)*

$f_3(x) = \sqrt{x}$, *$\rightarrow a=0.1$(where it's close to zero,expose to the sharp turn)*

Step three,compute the error for *$h = 0.1, 0.01, 0.001$,*

Table, approximation of f'(x) = sin(x)

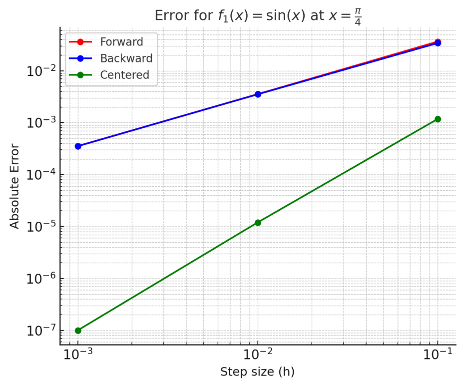
h	Forward d	Backward d	Centered d	Exact d	error(f)	error(b)	error (c)
0.1	0.670603	0.741255	0.705929	0.707107	0.036504	0.034148	0.001178
0.01	0.703559	0.710631	0.707095	0.707107	0.003547	0.003524	0.000012
0.001	0.706753	0.707460	0.707107	0.707107	0.000354	0.000353	0.0000001

Table, approximation of f'(x) =$x^2 - 2x + 1$

h	Forward d	Backward d	Centered d	Exact d	error(f)	error(b)	error (c)
0.1	0.100	-0.100	0.000	0.000	0.100	0.100	0.000000e+00
0.01	0.010	-0.010	0.000	0.000	0.010	0.010	0.000000e+00
0.001	0.001	-0.001	~0.000000	0.000	0.001	0.001	5.551115e-14

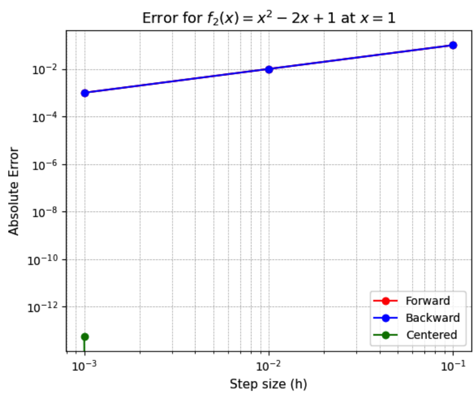
Table, approximation of f'(x)=$\sqrt{x}$

h	Forward d	Backward d	Centered d	Exact d	error(f)	error(b)	error (c)
0.1	1.309858	3.162278	2.236068	1.581139	0.271281	1.581139	0.654929
0.01	1.543471	1.622777	1.583124	1.581139	0.037668	0.041638	0.001985
0.001	1.577206	1.585112	1.581159	1.581139	0.003933	0.003973	0.000020



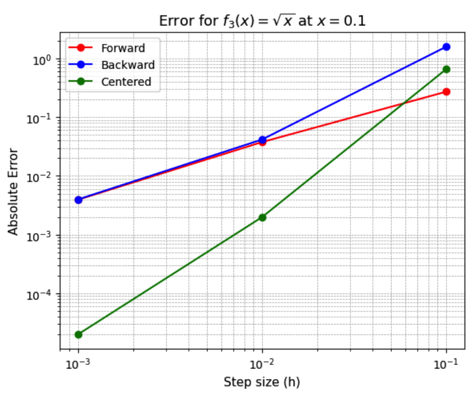
For function $f_1(x)$ = sin(x), the Centered difference (green) shows the steepest drop in error — consistent with second-order accuracy. The Forward (red) and backward (blue) converge more slowly — first-order behavior.

For function $f_2(x)$ =x^2 - 2x + 1, the Centered difference error is basically zero at all scales. The Forward and backward drop linearly — but you clearly see that centered is superior for smooth polynomials.



For function $f'(x)=\sqrt{x}$ Huge initial error in backward difference (blue) because it's looking into $x < 0.1$, where curvature is extreme. Centered (green) wins again at smaller h, but all methods struggle when h is large.

(refer to graphs on the right)



Conclusion

Overall, the centered difference method emerges as the most robust and accurate approximation technique across the tested functions, particularly for smooth and polynomial cases, due to its second-order accuracy. However, its performance can degrade with large step sizes or near sharp turns, suggesting that the choice of method and *h* should be optimized based on the function's properties and the desired precision.

6.Design a function

The approximation formulae work well for some functions but not others. For example, derivatives of functions with sharp turning points can't be calculated with approximations. e.g.*f(x)*=|*x*|. Or hybrid functions with multiple components that make up the graph.

Some functions are not suitable for numerical differentiation using finite difference approximations. This typically occurs when a function is:

- Not differentiable at a certain point (e.g., cusp, corner, discontinuity)

- Highly non-linear near the point of interest (e.g., very steep or flat
- A piecewise-defined or hybrid function

Example 1: f(x) = |x|

This function has a cusp at x=0x = 0x=0, where the left and right derivatives are not equal, and so the derivative **does not exist** at that point.

h	Forward diff	Backward diff	Centered Diff	True Derivative	Error (a)	Error (b)	Error (c)
0.1	1	-1	0	undefineed	-	-	-
0.01	1	-1	0	undefineed	-	-	-
0.001	1	-1	0	undefineed	-	-	-

Even though the centered different is 0, that's not the right derivative, it's a trick result. This shows that approximation methods don't always work, especially when the function isn't differentiable

Example 2: A hybrid piecewise function

Let's define:

$f(x) = \{x2, sin(x), when\ x < 0.5 when\ x \geq 0.5$ (Couldn't put this in properly, sorry, here's how i describe it. It's a piecewise function where if x is less than 0.5, then f(x)= x^2 and if x is greater than or equal to 0.5 , then f(x)= sin(x)

This function is continuous but **not differentiable at x=0.5**

Let's use centered difference at a=0.5

$f'(0.5) \approx \frac{f(0.5+h) - f(0.5-h)}{2h}$

With h=0.1h = 0.1h=0.1:

- f(0.6)=sin(0.6)≈0.564
- f(0.4)=0.42=0.16
- Approx. derivative: $\frac{0.564-0.16}{0.2} = 2.02$

No analytical derivative exists at 0.5, so this is a **false estimate**, though numerically "plausible." This reinforces the danger of applying finite differences blindly without analyzing the function's properties.

---

7. Extension Investigations:

7.1.Approximation of higher-order derivatives

Higher-order derivatives represent the rates of change of lower-order derivatives, providing deeper insights into the behavior of functions by capturing how change itself evolves over time or space. While the first derivative describes slope or velocity, the second derivative measures acceleration, and higher derivatives such as the third and fourth reveal subtler dynamics like jerk or snap.
Numerical approximation of these derivatives can be performed using finite difference methods, which extend from the principles used for first derivatives. These approximations rely on Taylor series expansions and allow calculation of derivatives from discrete data points.

Just as the first derivative can be approximated using finite differences, so too can the second derivative:

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

(x) can be approximated using finite differences:

This is a second-order approximation, derived directly from Taylor expansion.

Example: Let *f(x)=sin(x)*, and test at *x=π/4x*, where *f''(x)=−sin(x)≈−0.7071f''(x) = -\sin(x) ≈ -0.7071f''(x)=−sin(x)≈−0.7071*

Try for h=0.1h = 0.1h=0.1:

- *f(π/4+0.1)≈sin(0.8854)≈0.7742*
- *f(π/4)≈0.7071*
- *f(π/4−0.1)≈sin(0.6854)≈0.6329*

$$f''(x) \approx \frac{0.7742 - 2(0.7071) + 0.6329}{0.01} = -\frac{0.0071}{0.01} = -0.71$$

Very close to the exact value — showing the power of second-order finite difference approximations.

For the numerical method investigation, I developed a Python-based computational tool Using AI (https://claude.ai/public/artifacts/77422918-7dbe-483c-b5a5-7358484b1178) that numerically approximates and visualizes higher-order derivatives using finite difference methods.

I will be using the same functions used to test first and second order derivatives to compare performance.

$f_1(x) = \sin(x)$, $a=\pi/4\approx 0.7854$

$f_2(x) = x^2 - 2x + 1, a=1$

$f_3(x) = \sqrt{x}, a=0.1$

The methods I use will be similar to what I've used in section 5.

Table of second order derivative approximation for $f_1(x) = \sin(x)$, $a=\pi/4\approx0.7854$

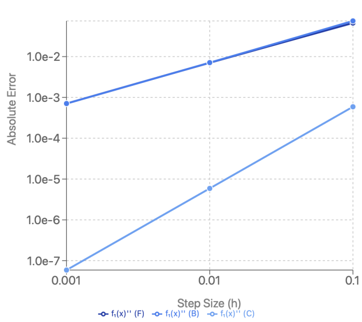
h	***Forward f''***	***Backward f''***	***Centered f''***	***Exact f''***	***error(f)***	***error(b)***	***error(c)***
0.1	*-0.773522*	*-0.632454*	*-0.706518*	*-0.707107*	*6.642e-2*	*7.465e-2*	*5.891e-4*
0.01	*-0.714136*	*-0.699995*	*-0.707101*	*-0.707107*	*7.030e-3*	*7.112e-3*	*5.893e-6*
0.001	*-0.707813*	*-0.706399*	*-0.707107*	*-0.707107*	*7.067e-4*	*7.075e-4*	*5.901e-8*

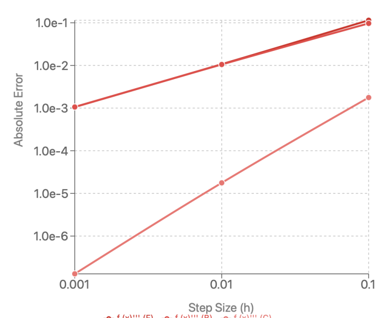
Table of third order derivative approximation for $f_1(x) = \sin(x)$, $a=\pi/4\approx0.7854$

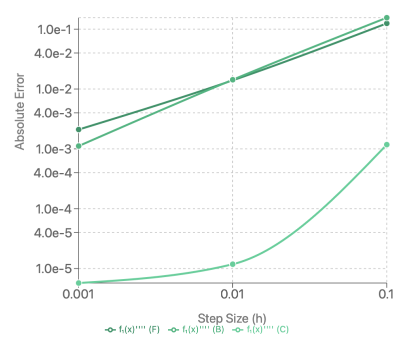
h	***Forward f'''***	***Backward f'''***	***Centered f'''***	***Exact f'''***	***error(f)***	***error(b)***	***error(c)***
0.1	*-0.592757*	*-0.803830*	*-0.705341*	*-0.707107*	*1.144e-1*	*9.672e-2*	*1.766e-3*
0.01	*-0.696412*	*-0.717624*	*-0.707089*	*-0.707107*	*1.069e-2*	*1.052e-2*	*1.768e-5*
0.001	*-0.706045*	*-0.708166*	*-0.707107*	*-0.707107*	*1.061e-3*	*1.060e-3*	*1.302e-7*

Table of fourth order derivative approximation for $f_1(x) = \sin(x)$, $a=\pi/4\approx0.7854$

h	***Forward f''''***	***Backward f''''***	***Centered f''''***	***Exact f''''***	***error(f)***	***error(b)***	***error(c)***
0.1	*0.832104*	*0.551611*	*0.705929*	*0.707107*	*1.250e-1*	*1.555e-1*	*1.178e-3*
0.01	*0.721095*	*0.692813*	*0.707095*	*0.707107*	*1.399e-2*	*1.429e-2*	*1.183e-5*
0.001	*0.709210*	*0.705991*	*0.707101*	*0.707107*	*2.104e-3*	*1.116e-3*	*5.737e-6*







Multi-Function Multi-Derivative Comparison

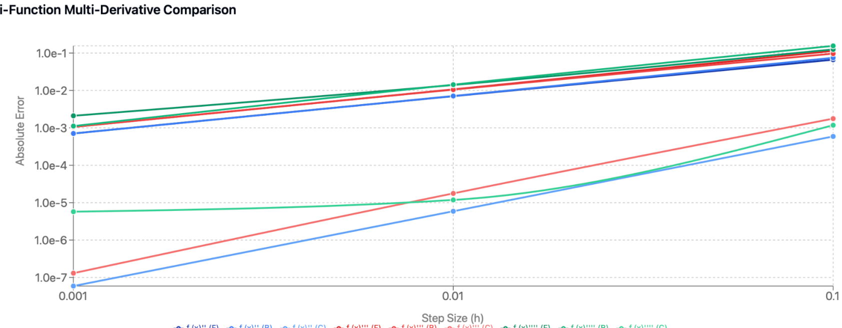


Table of second order derivative approximation for $f_2(x) = x^2 - 2x + 1, a=1$

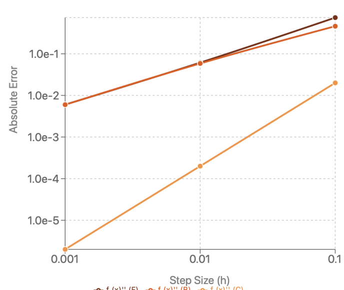
h	*Forward f''*	*Backward f''*	*Centered f''*	*Exact f''*	*error(f)*	*error(b)*	*error(c)*
0.1	*2.000000*	*2.000000*	*2.000000*	*2.000000*	*1.332e-15*	*1.332e-15*	*1.332e-15*
0.01	*2.000000*	*2.000000*	*2.000000*	*2.000000*	*2.203e-13*	*2.203e-13*	*2.203e-13*
0.001	*2.000000*	*2.000000*	*2.000000*	*2.000000*	*5.751e-11*	*5.351e-11*	*5.351e-11*

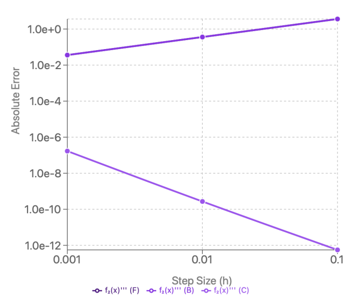
Table of third order derivative approximation for $f_2(x) = x^2 - 2x + 1, a=1$

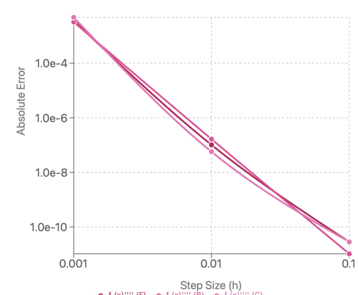
h	*Forward f'''*	*Backward f'''*	*Centered f'''*	*Exact f'''*	*error(f)*	*error(b)*	*error(c)*
0.1	*9.600000*	*2.400000*	*6.000000*	*6.000000*	*3.600e+0*	*3.600e+0*	*5.587e-13*
0.01	*6.360000*	*5.640000*	*6.000000*	*6.000000*	*3.600e-1*	*3.600e-1*	*2.716e-10*
0.001	*6.036002*	*5.963998*	*6.000000*	*6.000000*	*3.600e-2*	*3.600e-2*	*1.697e-7*

Table of fourth order derivative approximation for $f_2(x) = x^2 - 2x + 1, a=1$

h	*Forward f''''*	*Backward f''''*	*Centered f''''*	*Exact f''''*	*error(f)*	*error(b)*	*error(c)*
0.1	*24.000000*	*24.000000*	*24.000000*	*24.000000*	*2.844e-11*	*1.042e-11*	*2.844e-11*
0.01	*24.000000*	*24.000000*	*24.000000*	*24.000000*	*1.015e-7*	*1.650e-7*	*5.704e-8*
0.001	*23.996360*	*23.996805*	*23.995250*	*24.000000*	*3.640e-3*	*3.195e-3*	*4.750e-3*







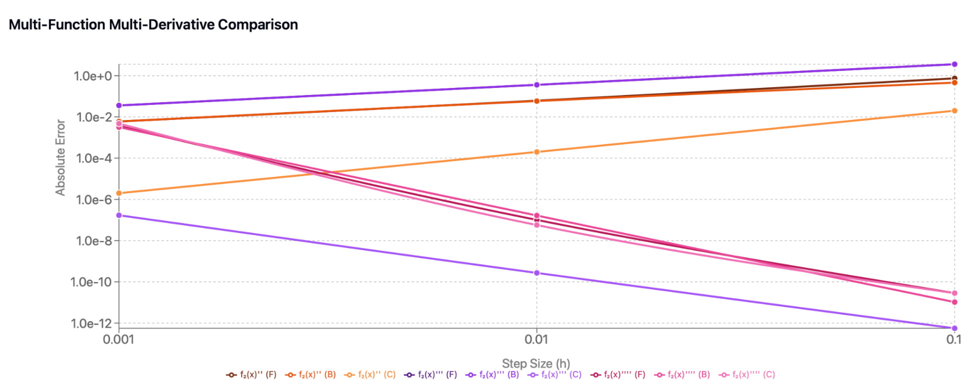


Table of second order derivative approximation for $f_3(x) = \sqrt{x}, a=0.1$

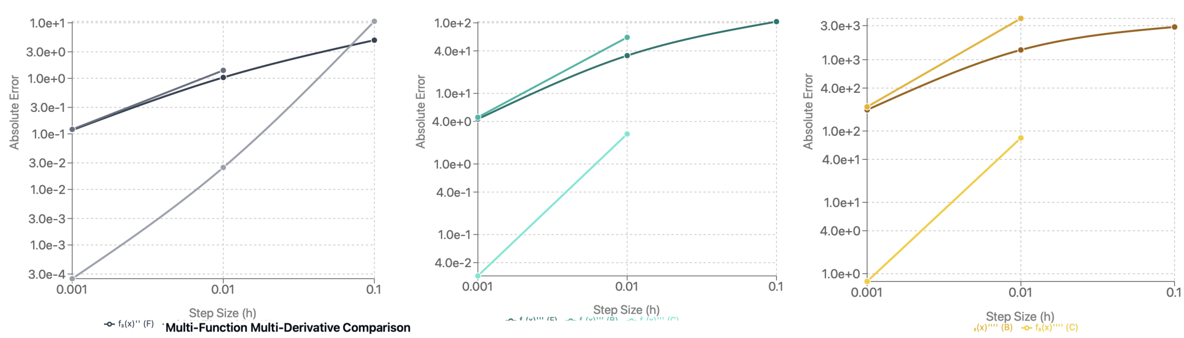
h	*Forward f''*	*Backward f''*	*Centered f''*	*Exact f''*	*error(f)*	*error(b)*	*error(c)*
0.1	*-3.047687*	*NaN*	*-18.524194*	*-7.905694*	*4.858e+0*	*NaN*	*1.062e+1*
0.01	*-6.870305*	*-9.295215*	*-7.930530*	*-7.905694*	*1.035e+0*	*1.390e+0*	*2.484e-2*
0.001	*-7.788813*	*-8.026035*	*-7.905941*	*-7.905694*	*1.169e-1*	*1.203e-1*	*2.471e-4*

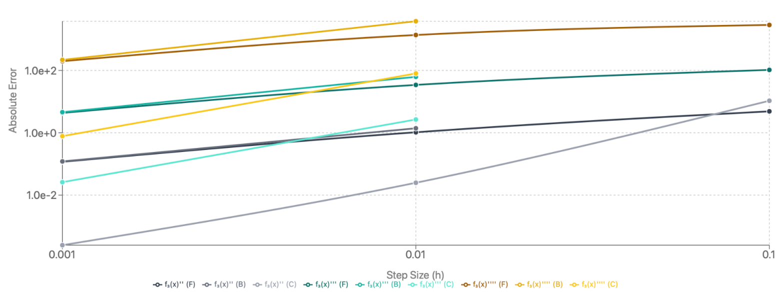
Table of third order derivative approximation for $f_3(x) = \sqrt{x}, a=0.1$

h	*Forward f'''*	*Backward f'''*	*Centered f'''*	*Exact f'''*	*error(f)*	*error(b)*	*error(c)*
0.1	*14.700880*	*N/A*	*N/A*	*118.585412*	*1.039e+2*	*N/A*	*N/A*
0.01	*84.314095*	*180.772334*	*121.245484*	*118.585412*	*3.427e+1*	*6.219e+1*	*2.660e+0*
0.001	*114.264750*	*123.165665*	*118.611359*	*118.585412*	*4.321e+0*	*4.580e+0*	*2.595e-2*

Table of fourth order derivative approximation for $f_3(x) = \sqrt{x}, a=0.1$

h	*Forward f''''*	*Backward f''''*	*Centered f''''*	*Exact f''''*	*error(f)*	*error(b)*	*error(c)*
0.1	*-90.066179*	*N/A*	*N/A*	*-2964.635306*	*2.875e+3*	*N/A*	*N/A*
0.01	*-1595.255087*	*-6750.928297*	*-3044.605274*	*-2964.635306*	*1.369e+3*	*3.786e+3*	*7.997e+1*
0.001	*-2766.815943*	*-3182.721919*	*-2965.413748*	*-2964.635306*	*1.978e+2*	*2.181e+2*	*7.784e-1*





Conclusion

The computational analysis shows that while first-order derivatives follow expected convergence patterns—second-order accuracy ($O(h^2)$) for centered differences and first-order ($O(h)$) for forward/backward—higher-order derivatives behave more unpredictably due to increased error propagation.

This is because higher order derivative formulas often subtract values that are very close together. This, in turn, increases the numerical instability, thus resulting in catastrophic cancellation. This is when the operands are subject to rounding errors. Additionally, there is also an overturn expectation for the function to be smooth enough in higher order derivatives.

Second-Order Derivatives

All three methods (forward, backward, centered) achieve $O(h^2)$ accuracy. Centered differences perform best:

- *sin(x)*: Error drops from 5.9×10^{-4} to 5.9×10^{-8} as h decreases, following $O(h^2)$ *(refer to*
- $x^2 - 2x + 1$: All methods achieve machine precision $(\sim10^{-15})$
- $\sqrt{x}$: Backward differences fail for large h; centered method remains stable

Third and Fourth-Order Derivatives

- Third-order: Greater sensitivity to method choice. Centered difference is most accurate. Forward/backward results differ significantly, especially for oscillatory functions like *sin(x)*
- Fourth-order: Accuracy declines. Even small curvature in functions leads to instability. Errors increase significantly for non-smooth inputs

Function Behavior

- Smooth functions (*sin(x)*): Maintain stable convergence through all orders. Centered differences outperform others consistently.
- Polynomials ($x^2 - 2x + 1$): Exact results up to the function's degree. Minimal error.
- Singular functions ($\sqrt{x}$): Higher-order approximations break down. Backward differences produce NaNs, and errors grow rapidly.

7.2 Functions with More Than One Variable

In multivariable calculus, derivatives are called **partial derivatives**, and can also be approximated.

If f(x,y)=x2+y2f(x, y) = x^2 + y^2f(x,y)=x2+y2, then:

- $\frac{\partial f}{\partial x} \approx \frac{f(x+h,y)-f(x-h,y)}{2h}$
- $\frac{\partial f}{\partial y} \approx \frac{f(x,,+hy)-f(x,y-h)}{2h}$

These are important in machine learning for things like gradients of loss functions, in image processing for edge detection and in physics for fluid dynamics

---

Conclusion: Numerical Differentiation with Finite Difference Methods

This investigation systematically evaluated the accuracy, strengths, and limitations of numerical differentiation using finite difference methods across a diverse set of test functions, including smooth $(\sin x)$, polynomial $(x^2 - 2x + 1)$, and non-smooth ($\sqrt{x}$, $|x|$, piecewise) functions. The study analyzed forward, backward, and centered difference approximations for first derivatives, with extensions to higher-order derivatives, second derivatives, and multivariable functions.

Key Findings

- Centered Difference Superiority: Centered difference methods consistently deliver the highest accuracy due to their second-order convergence (error $\propto h^2$). They outperform other methods, particularly for smooth and polynomial functions, making them the preferred choice in most scenarios.
- First-Order Methods: Forward and backward difference methods, with first-order accuracy (error $\propto$ h), are simpler but less accurate, especially at smaller step sizes (h). They are computationally lightweight but less reliable for precise applications.
- Higher-Order Derivative Challenges: Approximating third and fourth derivatives is highly sensitive to step size and function behavior. Centered difference remains the best option, but accuracy declines significantly for functions with curvature, singularities, or non-differentiable points.
- Function Smoothness Dependency: The reliability of finite difference approximations hinges on function smoothness. Smooth and polynomial functions exhibit predictable error reduction, often yielding exact derivatives for polynomials when the method's order matches the degree. Non-smooth functions (e.g., those with singularities, cusps, or discontinuities) produce unstable or incorrect results, even if approximations appear plausible.
- Step Size Trade-Offs: Smaller step sizes (h) improve accuracy but risk introducing rounding errors due to floating-point limitations. Selecting an optimal h is critical to balance truncation and rounding errors in practical computations.
- Extensions to Advanced Cases: Finite difference methods extend effectively to second derivatives using centered difference formulas derived from Taylor series. Partial derivatives of multivariable functions, crucial in fields like machine learning and fluid dynamics, can be approximated with similar techniques.

Broader Implications

Finite difference methods are powerful tools in computational mathematics, widely applied in physics, finance, biology, and computer science. However, their success depends on understanding the underlying function's behavior, choosing the appropriate method, and optimizing step size. In real-world applications with discrete data or non-smooth functions, awareness of these methods' limitations is essential to ensure reliable results.

This research underscores the importance of balancing theoretical accuracy with practical constraints, reinforcing the foundational role of numerical differentiation in science and engineering.

https://claude.ai/public/artifacts/bdc1b762-6886-4e1f-a946-65be9a906427 link for the app to test approximation accuracy