This implementation demonstrates numerical differentiation using the central finite difference method for both 1D functions (tangent lines) and 2D functions (tangent planes with normal vectors). The code successfully compares numerical approximations against exact analytical derivatives, analyzes truncation and roundoff errors across step sizes $h \in [10^{-8}, 10^{-1}]$, and visualizes the characteristic $O(h^2)$ error behavior. All mathematical implementations are correct and numerically stable.

Mathematical Foundation

Test Functions (Lines 8-24)

1D Function:

- $f(x) = x^2e^x$ at $x_0 = 0.5$
- Exact derivative: $f'(x) = (x^2 + 2x)e^x$ ✓
- Verification: Product rule correctly applied

2D Function:

- $g(x,y) = xy + x^2 + y^2$ at (1.5, 1.0)
- Exact gradient: $\nabla g = (y + 2x, x + 2y)$ ✓
- Verification: Partial derivatives correctly computed

Analytical Correctness: Both derivatives manually verified - no computational shortcuts taken.

Core Algorithm Implementation

1. Central Finite Difference (Lines 30-36)

1D Central Difference:

```
f'(x) ≈ [f(x+h) - f(x-h)] / (2h)
```

- **Truncation Error:** $O(h^2)$ - superior to forward/backward differences $O(h)$
- **Symmetry:** Uses points equidistant on both sides
- **Stability:** Better conditioned than one-sided formulas

2D Partial Derivatives:

- Computed independently for $\partial f/\partial x$ and $\partial f/\partial y$
- Same central difference formula applied in each direction
- Gradient vector assembled from partial derivatives

Implementation Quality: Clean, minimal code with no unnecessary complexity. Correct application of numerical differentiation theory.

Error Analysis Framework

2. Step Size Sweep (Lines 40-48, 87-103)

Methodology:

- Tests 15 logarithmically-spaced h values: 10^{-8} to 10^{-1}
- Computes absolute error: |f'*FD(x) - f'_exact(x)*|
- Stores results in pandas DataFrame for analysis

Error Components Captured:

1. **Truncation Error** (large h): Dominates for h > 10^{-4}
- Theoretically $O(h^2)$ for central differences
- Code plots h^2 reference line for comparison (lines 75, 151)
1. **Roundoff Error** (small h): Dominates for h < 10^{-6}
- Caused by catastrophic cancellation in subtraction
- Amplified by division by small 2h

Sweet Spot Detection: Code identifies optimal h via `df['Error'].idxmin()` - typically around 10^{-5}**Scientific Rigor:** Demonstrates understanding that numerical differentiation is a balancing act between competing error sources.

Visualization & Interpretation

3. 1D Analysis (Lines 38-83)

Dual-Plot Design:Plot 1 - Tangent Line Visualization (Lines 54-71):

- Function curve in blue
- Exact tangent line in red (using analytical derivative)
- FD tangent line in green (using optimal h)
- Visual verification that tangent slopes match function behavior

Pedagogical Value: Makes abstract derivative concept concrete - "the line that just touches the curve."**Plot 2 - Error Behavior (Lines 74-80):**

- Log-log plot reveals power-law relationships
- FD error shows characteristic "U" shape
- $O(h^2)$ reference line validates truncation error theory
- Left side upturn shows roundoff error onset

Insight: The "sweet spot" at the valley bottom is where numerical differentiation works best.

4. 2D Analysis (Lines 85-159)

Advanced 3D Visualization:Plot 1 - Surface with Tangent Plane (Lines 120-146):

- Original surface rendered in viridis colormap

- Tangent plane overlaid in semi-transparent red
- Point of tangency marked prominently
- **Normal vectors:** Exact (red) vs FD (green) shown as arrows
- Direction vectors normalized for visual comparison

Mathematical Sophistication:

- Normal vector computed as $N = (-\partial f/\partial x, -\partial f/\partial y, 1) / ||...||$
- Correctly perpendicular to tangent plane
- Arrow scaling for visibility (scale=1.5)

Plot 2 - Gradient Error (Lines 149-156):

- L2 norm of gradient error: $||\nabla g\_FD - \nabla g\_exact||$
- Same $O(h^2)$ behavior as 1D case
- Confirms method consistency across dimensions