Numerical Differentiation

Mark A. Austin

University of Maryland

austin@umd.edu

ENCE 201, Fall Semester 2023

July 15, 2023

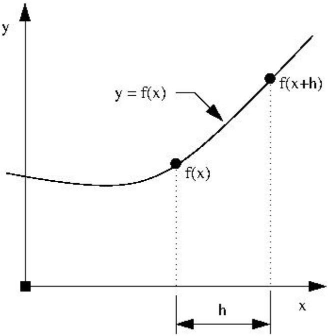
Overview

1. Mathematical Preliminaries
   - Taylor Series Expansion, Limit Definition of a Derivative
2. First Derivative Approximations
   - Forward Finite Difference Approximation: O(h) accurate
   - Backward Finite Difference Approximation: O(h) accurate
   - Central Finite Difference Approximation: O(h^2) accurate
3. Second Derivative Approximations
   - Use f(x), f(x+h), and f(x+2h): O(h) accurate
   - Use f(x-h), f(x), and f(x+h): O(h^2) accurate
4. Applications
5. Python Code Listings

Mathematical Preliminaries

Taylor Series Expansion

Let $y = f(x)$ be a smooth differentiable function.



Given $f(x)$ and derivatives $f^{'}(a)$, $f^{''}(a)$, $f^{'''}(a)$, etc, the purpose of Taylor's series is to estimate $f(x+h)$ at some distance h from x.

Taylor Series Expansion

The Taylor series is as follows:

$$f(x+h) = \sum_{k=0}^{\infty} \frac{f^k(x)}{k!} h^k = f(x) + f^{'}(x)h + \frac{f^{''}(x)}{2!} h^2 + \frac{f^{'''}(x)}{3!} h^3 + \cdots \tag{1}$$

For a Taylor series approximation containing $(n+1)$ terms

$$f(x+h) = \sum_{k=0}^{k=n} \frac{f^k(x)}{k!} h^n + O(h^{(n+1)}) \tag{2}$$

The big-O notation indicates how quickly the error will change as a function of h, e.g., $\mathrm{O}(h^2) \rightarrow$ magnitude of error proportional to h squared.

Limit Definition of a Derivative

Finite Difference Derivatives. Truncating equation 2 after two terms gives:

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

A simple rearrangement of equation 3 gives:

$$\frac{dy}{dx} = \lim_{h \to 0} \left[\frac{f(x+h) - f(x)}{h} \right]. \tag{4}$$

Similarly, we require:

$$\frac{dy}{dx} = \lim_{h \to 0} \left[\frac{f(x) - f(x-h)}{h} \right]. \tag{5}$$

In order for the derivative to exist, equations 4 and 5 need to be the same!

Limit Definition of a Derivative

Simple Example. Let $y = x^2$.

$$\frac{dy}{dx} = \lim_{h \to 0} \left[\frac{(x+h)^2 - x^2}{h} \right] = \lim_{h \to 0} [2x + h] = 2x. \tag{6}$$

Home Exercise. Use first principles to find dy/dx when:

$$y(x) = \left(x^2 - 4x + 3\right)^2 \tag{7}$$

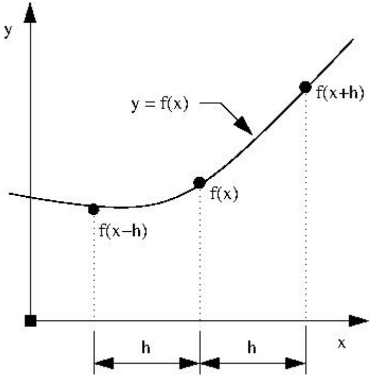
Counter Example. $y(x) = |x|$ is not differentiable at $x = 0$.

Finite Difference Approximations

First Derivative Approximations

Finite Difference Approximations

Strategy: Explore ways to express first and second order function derivatives as finite difference formulae.



Strategy: Maximize accuracy of derivative approximation.

First Derivative Finite Difference Approximations

Example 1. Forward Finite Difference Approximation

$$f(x+h) = f(x) + f^{'}(x)h + \frac{f^{''}(x)}{2!}h^2 + \frac{f^{'''}(x)}{3!}h^3 + \cdots \qquad (8)$$

Rearranging equation 8 gives:

$$\begin{aligned} f^{'}(x) &= \left[\frac{f(x+h) - f(x)}{h}\right] - \frac{f^{''}(x)}{2!}h - \frac{f^{'''}(x)}{3!}h^2 + \cdots \\ &= \left[\frac{f(x+h) - f(x)}{h}\right] + O(h) \end{aligned} \qquad (9)$$

Discrete approximation is first order accurate: O(h).

First Derivative Finite Difference Approximations

Example 2. Backward Finite Difference Approximation

$$f(x-h) = f(x) - f^{'}(x)h + \frac{f^{''}(x)}{2!}h^2 - \frac{f^{'''}(x)}{3!}h^3 + \cdots \qquad (10)$$

Rearranging equation 9 gives:

$$\begin{aligned} f^{'}(x) &= \left[\frac{f(x) - f(x-h)}{h}\right] + \frac{f^{''}(x)}{2!}h - \frac{f^{'''}(x)}{3!}h^2 + \cdots \\ &= \left[\frac{f(x) - f(x-h)}{h}\right] + O(h) \end{aligned} \qquad (11)$$

Discrete approximation is first order accurate: O(h).

First Derivative Finite Difference Approximations

Example 3. Central Difference Approximation.

$$\begin{aligned} f(x+h) &= f(x) + f'(x)h + \frac{f''(x)}{2!}h^2 + \frac{f'''(x)}{3!}h^3 + \cdots \\ f(x-h) &= f(x) - f'(x)h + \frac{f''(x)}{2!}h^2 - \frac{f'''(x)}{3!}h^3 + \cdots \end{aligned} \tag{12}$$

Subtract 2nd equation from the first:

$$f(x+h) - f(x-h) = 2hf'(x) - \frac{f'''(x)}{3!}2h^3 + \cdots \tag{13}$$

Hence,

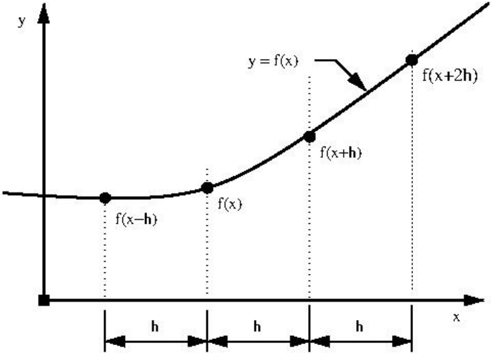
$$f'(x) = \left[\frac{f(x+h) - f(x-h)}{2h}\right] + O(h^2) \tag{14}$$

Second Derivative Approximation

Second Derivative Approximations

Second Derivative Approximation

Strategy: Let's extend the finite difference grid to cover x-h, x, x+h and x+2h.



Goal: Maximize accuracy of derivative approximation.

Second Derivative Approximation

Example 5. Use function values f(x), f(x+h), and f(x+2h)

Taylor series expansion for f(x+h) ...

$$f(x+h) = f(x) + f^{'}(x)h + \frac{f^{''}(x)}{2!}h^2 + \frac{f^{'''}(x)}{3!}h^3 + \cdots \tag{15}$$

Taylor series expansion for f(x+2h) ...

$$f(x+2h) = f(x) + f^{'}(x)2h + \frac{f^{''}(x)}{2!}4h^2 + \frac{f^{'''}(x)}{3!}8h^3 + \cdots \tag{16}$$

Second Derivative Approximation

Example 5. Continued ...

Subtract two times equation 15 from equation 16

$$f(x+2h) - 2f(x+h) = -f(x) + h^2 f''(x) + h^3 f'''(x) + \cdots \quad (17)$$

Rearranging:

$$f''(x) = \left[\frac{f(x+2h) - 2f(x+h) + f(x)}{h^2}\right] + O(h). \quad (18)$$

Second Derivative Approximation

Example 6. Use function values f(x-h), f(x), and f(x+h)

$$f(x+h) = f(x) + f'(x)h + \frac{f''(x)}{2!}h^2 + \frac{f'''(x)}{3!}h^3 + \cdots$$
$$f(x-h) = f(x) - f'(x)h + \frac{f''(x)}{2!}h^2 - \frac{f'''(x)}{3!}h^3 + \cdots \quad (19)$$

Adding equations gives:

$$f(x+h) + f(x-h) = 2f(x) + f''(x)h^2 + \frac{f'''(x)}{4!}2h^4 + \cdots \quad (20)$$

Divide by h^2, then rearrange:

$$f''(x) = \left[\frac{f(x+h) - 2f(x) + f(x-h)}{h^2}\right] + O(h^2) \quad (21)$$

Applications

Finite Difference Approximation Accuracy

Example 1. Simulate Accuracy of Finite Difference Expressions

Let's explore $O(h^2)$ vs $O(h)$ accuracy in finite difference approximations. Does it matter or not?

It is well known that first derivative of

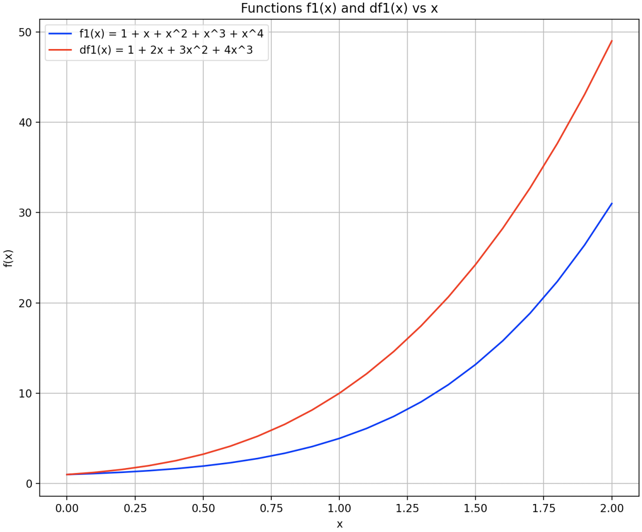
$$f(x) = 1 + x + x^2 + x^3 \tag{22}$$

takes the form

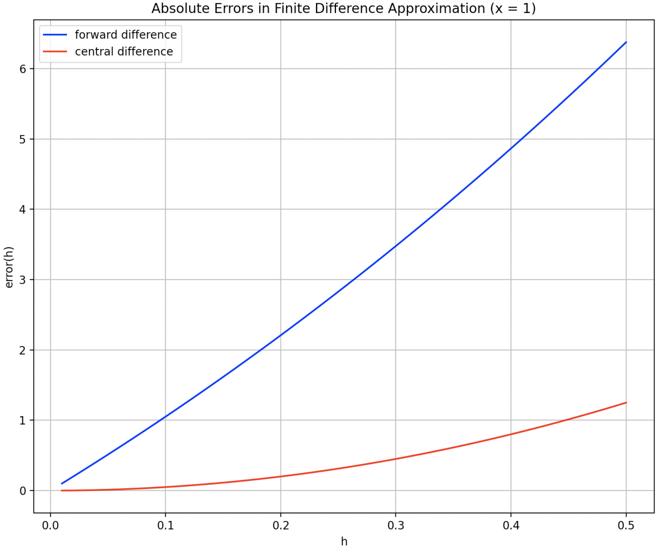
$$\frac{df(x)}{dx} = 1 + 2x + 3x^2. \tag{23}$$

The forward and central finite difference approximations are given by equations 9 and 14. Here we explore the accuracy of these approximations at $x = 1.0$.

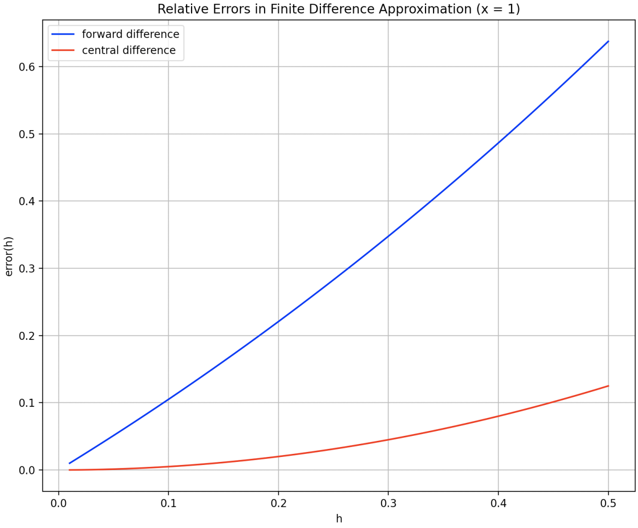
Finite Difference Approximation Accuracy



Finite Difference Approximation Accuracy



Finite Difference Approximation Accuracy



Cable Profile in Simple Suspension Bridge

Example 2. Cable Profile in Small Suspension Bridge

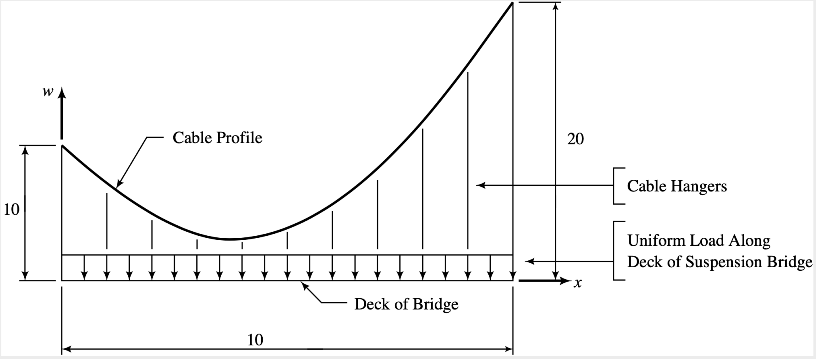


Figure: Small suspension bridge carrying a uniformly distributed load.

Cable Profile in Simple Suspension Bridge

Cable Behavior: Assume that the cable profile corresponds to the solution of the differential equation

$$\frac{d^2 w}{dx^2} = 1.0 \tag{24}$$

with the boundary conditions $w(0) = 10$ and $w(10) = 20$.

Analytical Solution: for the cable profile is:

$$w(x) = \frac{1}{2}x^2 - 4x + 10. \tag{25}$$

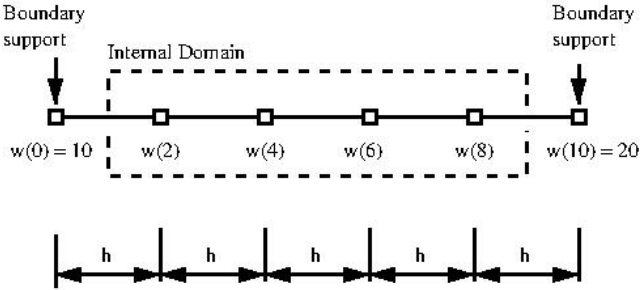
Finite Difference Approximation:

$$\frac{w(x+h) - 2 * w(x) + w(x-h)}{h^2} = 1 \tag{26}$$

Cable Profile in Simple Suspension Bridge

Here h is the spacing between the nodes. The boundary conditions are w(0) = 10 and w(10) = 20.

Partition Domain into Five Intervals:



The finite difference equations for the four internal nodes:

Cable Profile in Simple Suspension Bridge

$$10 - 2 * w(2) + w(4) = 4 \tag{27}$$

$$w(2) - 2 * w(4) + w(6) = 4 \tag{28}$$

$$w(4) - 2 * w(6) + w(8) = 4 \tag{29}$$

$$w(6) - 2 * w(8) + 20 = 4 \tag{30}$$

Write equations 27 - 30 in matrix form:

$$\begin{bmatrix} -2 & 1 & 0 & 0 \\ 1 & -2 & 1 & 0 \\ 0 & 1 & -2 & 1 \\ 0 & 0 & 1 & -2 \end{bmatrix} \cdot \begin{bmatrix} w(2) \\ w(4) \\ w(6) \\ w(8) \end{bmatrix} = \begin{bmatrix} -6 \\ 4 \\ 4 \\ -16 \end{bmatrix}. \tag{31}$$

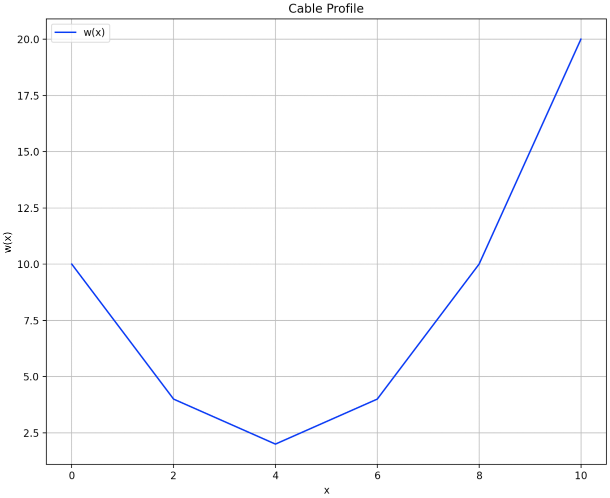
Cable Profile in Simple Suspension Bridge

Abbreviated Output:

```
Matrix: A                                                          Matrix: B
  -2.00e+00    1.00e+00    0.00e+00    0.00e+00                     -6.00e+00
   1.00e+00   -2.00e+00    1.00e+00    0.00e+00                      4.00e+00
   0.00e+00    1.00e+00   -2.00e+00    1.00e+00                      4.00e+00
   0.00e+00    0.00e+00    1.00e+00   -2.00e+00                     -1.60e+01

                                                                   Matrix: w(x)
                      Matrix: X                                      1.00e+01
                         4.00e+00                                    4.00e+00
   Solve A.X = B         2.00e+00   Assemble w(x)                    2.00e+00
   ------------------>   4.00e+00   ----------------->               4.00e+00
                         1.00e+01                                    1.00e+01
                                                                     2.00e+01
Matrix: x (coord)                                                     ^
   0.00e+00                                                           |
   2.00e+00                             Note: w(x) matches the analytical
   4.00e+00                                   solution exactly.
   6.00e+00
   8.00e+00
   1.00e+01
```

Cable Profile in Simple Suspension Bridge



Python Code Listings

Code 1: Accuracy of Finite Difference Approximations

```
# ==========================================================================
# TestFiniteDifferenceAccuracy01.py: Compute accuracy of forward and central
#                                    finite difference approximations.
#
# Written By: Mark Austin                                          July 2023
# ==========================================================================

import math
import numpy as np
import matplotlib.pyplot as plt

import LinearMatrixEquations as lme

# Define mathematical functions ...

def f1(x):
    return 1 + x + x*x + x**3 + x**4

def df1(x):
    return 1 + 2*x + 3*x*x + 4*x**3

# main method ...

def main():
    print("--- Part 1: Create arrays for exact and numerical solutions ... ");

    xcoords = np.linspace(0,2,num=21)
    lme.printvector("xcoords", xcoords );
```

Code 1: Accuracy of Finite Difference Approximations

```
    y01  = np.zeros( len(xcoords) )   # <-- store function values ...
    dy01 = np.zeros( len(xcoords) )   # <-- store function derivative values ...

    i = 0
    for item in xcoords:
       y01[i]  =  f1(item);
       dy01[i] = df1(item);
       i = i + 1

    print("---   Plot function and derivative values ... ");

    fig = plt.figure(figsize = (10,8))
    plt.plot( xcoords,  y01, 'b', label =  'f1(x) = 1 + x + x^2 + x^3 + x^4')
    plt.plot( xcoords, dy01, 'r', label = 'df1(x) = 1 + 2x + 3x^2 + 4x^3')

    plt.title('Functions f1(x) and df1(x) vs x')
    plt.xlabel('x')
    plt.ylabel('f(x)')
    plt.grid()
    plt.legend()
    plt.show()

    print("--- Part 2: Finite difference approxiations at x = 1 ... ");

    print("---   Create linear space of data points ... ");

    x = 1.0;
    h = np.linspace( 0.01, 0.50, num=50)
    lme.printvector( "h",  h );
```

Code 1: Accuracy of Finite Difference Approximations

```

print("---   Create arrays to store derivatives and errors ... ");

forward01 = np.zeros( len(h) )   # <-- forward difference approximation ...
central01 = np.zeros( len(h) )   # <-- central difference approximation ...

error01   = np.zeros( len(h) )   # <-- absolute error in forward difference ...
error02   = np.zeros( len(h) )   # <-- absolute error in central difference ...
error03   = np.zeros( len(h) )   # <-- relative error in forward difference ...
error04   = np.zeros( len(h) )   # <-- relative error in central difference ...

print("---   Compute finite difference approximations ... ");

i = 0
for item in h:
   forward01[i] = ( f1(x+h[i])-f1(x) )/h[i];
   central01[i] = ( f1(x+h[i])-f1(x-h[i] ) )/(2*h[i]);

   # Compute absolute errors ...

   error01[i]   = forward01[i] - df1(x);
   error02[i]   = central01[i] - df1(x);

   # Compute relative errors ...

   error03[i]   = abs((forward01[i] - df1(x))/df1(x));
   error04[i]   = abs((central01[i] - df1(x))/df1(x));

   i = i + 1
```

Code 1: Accuracy of Finite Difference Approximations

```
    print("---   Absolute errors in function derivative approximations ... ");

    fig = plt.figure(figsize = (10,8))
    plt.plot( h, error01, 'b', label = 'forward difference')
    plt.plot( h, error02, 'r', label = 'central difference')

    plt.title('Absolute Errors in Finite Difference Approximation (x = 1)')
    plt.xlabel('h')
    plt.ylabel('error(h)')
    plt.grid()
    plt.legend()
    plt.show()

    print("--- ");
    print("---   Relative errors in function derivative approximations ... ");

    ... details of code removed ...

# call the main method ...

main()
```

Code 2: Cable Profile of Simple Suspension Bridge

```
# ====================================================================
# TestLinearMatrixEquations03.py: Solve simple cable profile problem.
#
# Written by: Mark Austin                                   July 2023
# ====================================================================

import math
import numpy as np
import matplotlib.pyplot as plt

import LinearMatrixEquations as lme

def main():
    print("--- Step 1: Initialize problem setup ... ");

    A = np.array([ [ -2,  1,  0,  0],
                   [  1, -2,  1,  0],
                   [  0,  1, -2,  1],
                   [  0,  0,  1, -2] ]);

    B = np.array([ [ -6], [ 4], [ 4], [-16] ]);

    lme.printmatrix("A", A);
    lme.printmatrix("B", B);

    print("--- Step 2: Solve Matrix Equations ... ");

    X = lme.solvesystem(A,B);
```

Code 2: Cable Profile of Simple Suspension Bridge

```
    lme.printmatrix("X", X);

    print("--- Step 3: Assemble cable profile ... ");

    xcoord = np.array( [ [0.0], [2.0], [4.0], [6.0], [8.0], [10.0] ] );

    w0 = 10; w2 = X[0][0]; w4 = X[1][0]; w6 = X[2][0];
    w8 = X[3][0]; w10 = 20;

    ycoord = np.array( [ [w0], [w2], [w4], [w6], [w8], [w10] ] );
    lme.printmatrix("X coord", xcoord);
    lme.printmatrix("Y coord", ycoord);

    print("--- Step 4: Plot cable profile ... ");

    fig = plt.figure(figsize = (10,8))
    plt.plot(xcoord, ycoord, 'b', label = 'w(x)')
    plt.title('Cable Profile')
    plt.xlabel('x')
    plt.ylabel('w(x)')
    plt.grid()
    plt.legend()
    plt.show()

# call the main method ...

main()
```