CSE3365 - Project 1
Analysis of roundoff and truncation errors
through derivative approximation

Eric Smith
Bobby B. Lyle School of Engineering,
Computer Science

February 2017

I. Project Description

This article attempts to demonstrate the significance of *roundoff errors* and *truncation errors* through the processes of forward and central difference approximation. First, given the function:

$$f(x) = sin(x) \quad (1)$$

we will approximate its derivative, $f'(x)$, where $x = \pi/4$. Our two difference approximation techniques will provide a sample of data that is slightly different from the true value of $f'(x)$ due to truncation errors.

Next, by better approximating $f'(x)$, we will minimize this truncation error to the point that the roundoff error produced by Python will be more prominent in our graph. *See Figure 1*

II. Concepts and Theory

Roundoff errors occur because computers have limited memory space to represent numbers and, thus, can only approximate so many decimal values. Meanwhile, *truncation errors* occur when a function's value is approximated in an effort to reserve computational power. In both cases.

$$Error = True\,value - Approximate\,value \quad (2)$$

Our first approximation method, forward difference, is defined as

$$D_+f(x) = \frac{f(x+h) - f(x)}{h} \quad (3)$$

and central difference is defined as:

$$D_0f(x) = \frac{f(x+h) - f(x-h)}{2h} \quad (4)$$

While both equations will approximate $f'(x)$, the central difference method is considered more accurate because its error is described as $O(h^2)$ compared to forward difference approximation's $O(h)$, thus, its error more quickly converges to zero.

III. Numerical Implementation

In Part I, the derivative of $f(x)$ was approximated using both methods over a small set of h, where h = 0.1, 0.05, 0.025, 0.0125. The results of the approximations can be seen in Tables 1 and 2.

Then for Part II, the errors of $D_+f(x)$ and $D_0f(x)$ were graphed on a logarithmic scale with step size $h = 1/2^2$ where $j = 0 : 64$.

IV. Numerical Results and Discussion

i. Parts A&B

h	$D_+f(x)$	$f'(x) - D_+f(x)$
0.100000	0.670603	0.036504
0.050000	0.689138	0.017969
0.025000	0.698195	0.008912
0.012500	0.702669	0.004438
$\frac{f'(x)-D_+f(x)}{h}$	$\frac{f'(x)-D_+f(x)}{h^2}$	$\frac{f'(x)-D_+f(x)}{h^3}$
0.365038	3.650381	36.503808
0.359372	7.187431	143.748624
0.356481	14.259247	570.369861
0.355022	28.401753	2272.140254

Table 1: *Forward Difference Approximation*

Table 1 describes results from running our forward difference approximation. The first three columns show: h, the approximated derivative, and the approximation's distance from the true value.

The third, fourth, and fifth columns exist to find the constant C that's part of the approximation's error, $O(h)$:

$$O(h) = f'(x) - D_+f(x) = Ch \quad (5)$$

In the fourth column specifically, as h approaches zero, the value approaches some constant, C. This is because the h's on both sides of the equation cancel out. Meanwhile, the values in the last two columns are increasing as the value of the denominator approaches zero.

h	$D_0 f(x)$	$f'(x) - D_0 f(x)$
0.100000	0.705929	0.001178
0.050000	0.706812	0.000295
0.025000	0.707033	0.000074
0.012500	0.707088	0.000018
$\frac{f'(x)-D_0 f(x)}{h}$	$\frac{f'(x)-D_0 f(x)}{h^2}$	$\frac{f'(x)-D_0 f(x)}{h^3}$
0.011779	0.117792	1.177922
0.005892	0.117836	2.356728
0.002946	0.117847	4.713898
0.001473	0.117850	9.428017

Table 2: *Central Difference Approximation*

Table 2 describes results from running our central difference approximation. Again, the first three columns show: h, the approximated derivative, and the approximation's distance from the true value. Notice here that the errors in column three is less than those in Table 1.

The third, fourth, and fifth columns once again exist to find the constant C that's part of the approximation's error. But this time, because central difference is a second order method, it's error is defined as $O(h^2)$:

$$O(h^2) = f'(x) - D_0 f(x) = Ch^2 \tag{6}$$

This time, it's the fifth column where the values converge to some constant, C. This is because the h^2's on both sides of the equation cancel out. Meanwhile, the values in the fourth and sixth columns are approaching zero and infinity, respectfully.

ii. Part C

Figure 1 shows the errors of $D_+ f(x)$ and $D_0 f(x)$ plotted on log scales with step size h. Similar to Parts A and B, the graph shows the errors of each function decreasing as the value of h increases. At a certain point, however, the errors of both functions start to increase. This is because as h approaches zero, the truncation error produced by Python dominates the roundoff error.

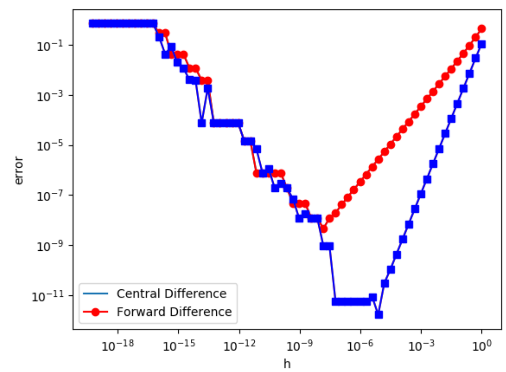


Figure 1: *Forward vs Central Difference Approximation*

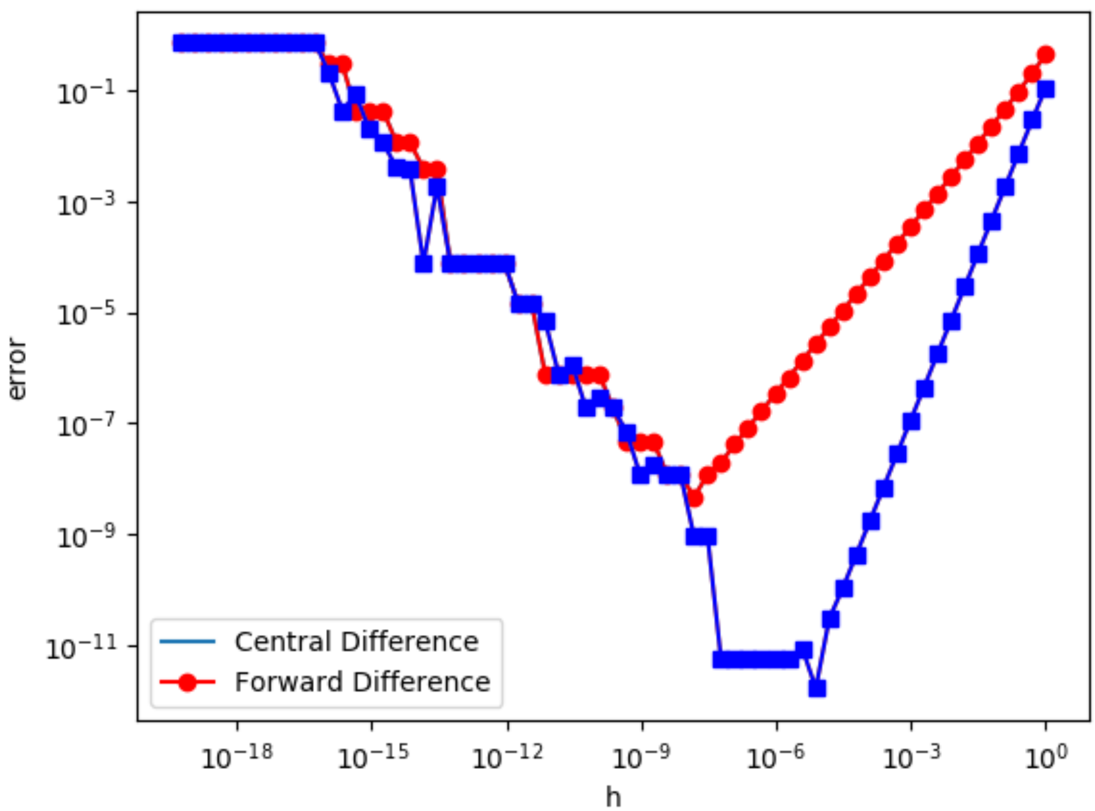
V. Conclusion

In this article, we looked at two different methods of approximating the derivative of *f(x)*: forward and central difference approximation. In Parts A and B, we saw how the difference between the true value of *f′(x)* and each approximation decreased as the value of h approached zero. Also, we saw that the method of central difference approximation was a more accurate way to approximate our value of *f′(x)*. In Part C, we demonstrated the prominence of roundoff and truncation errors in our data at different values of h.

Attached you will find the Python code used to calculate the data sets in this article.

```python
"""
Eric Smith
Southern Methodist University
CSE3365
February 8th, 2017
"""
from math import sin, cos, pi

f_out = open('output.csv', 'w')
x = pi/4
h_array = [0.1, 0.05, 0.025, 0.0125]


# math functions
def f(x):
    return sin(x)


def f_prime(x):
    return cos(x)


def forward_difference(x, h):
    return (f(x + h) - f(x))/h


def center_difference(x, h):
    return (f(x + h) - f(x - h))/(2*h)


# do calculations, output to file
def build_table(approximation_type):
    for h in h_array:
        a = approximation_type(x, h)
        b = f_prime(x) - a

        f_out.write('{0:.6f}'.format(h) + ',')
        f_out.write('{0:.6f}'.format(a) + ',')

        for i in range(4):
            f_out.write('{0:.6f}'.format(b / (h ** i)) + ',')

        f_out.write('\n')
    f_out.write('\n\n')

build_table(forward_difference)
build_table(center_difference)

f_out.close()
```

```python
"""
Eric Smith
Southern Methodist University
CSE3365
February 8th, 2017
"""

from math import sin, fabs, pi
from numpy import zeros
import matplotlib.pyplot as plt
x = pi/4
nmax = 65
h = zeros((65, 1), 'float')
errorForward = zeros((nmax, 1), 'float')
errorCentral = zeros((nmax, 1), 'float')
exact_value = sin(x)

for j in range(nmax):
    h[j] = (1 / 2.0) ** j

    # forward difference
    computed_value = (sin(x + h[j]) - sin(x)) / h[j]
    errorForward[j] = fabs(computed_value - exact_value)

    # central difference
    computed_value = (sin(x + h[j]) - sin(x - h[j])) / (2*h[j])
    errorCentral[j] = fabs(computed_value - exact_value)

plt.loglog(h, errorForward, h, errorForward, '-ro')
plt.loglog(h, errorCentral, h, errorCentral, '-bs')

plt.xlabel('h')
plt.ylabel('error')
plt.legend(('Central Difference', 'Forward Difference'), loc = 0)
plt.savefig('ForwardvsCentralDifference.png', format='png')
plt.show()
```



error
h
10^{-1}
10^{-3}
10^{-5}
10^{-7}
10^{-9}
10^{-11}
10^{-18}
10^{-15}
10^{-12}
10^{-9}
10^{-6}
10^{-3}
10^{0}
Central Difference
Forward Difference