# PHYS 512 Assignment 1

Liam Fitzpatrick

September 2022

## 1 Question 1

a)

$$D_\delta = \frac{f(x+\delta) - f(x-\delta)}{2\delta} = \frac{\big(f(x) + f'(x)\delta + f''(x)\delta^2/2 + ...\big) - \big(f(x) - f'(x)\delta + f''(x)\delta^2/2 + ..\big)}{2\delta}$$

$$= f'(x) + f'''(x)\delta^2/3!...$$

$$\Rightarrow Err(D_\delta) \sim \frac{1}{6} f'''(x)\delta^2$$

$$D_{2\delta} = \frac{f(x+2\delta) - f(x-2\delta)}{4\delta} = \frac{\big(f(x) + f'(x)2\delta + f''(x)4\delta^2/2 + ...\big) - \big(f(x) - f'(x)2\delta + f''(x)4\delta^2/2 + ...\big)}{4\delta}$$

$$= f'(x) + f'''(x)4\delta^2/3! + ...$$

$$\Rightarrow Err(D_{2\delta}) \sim \frac{2}{3} f'''(x)\delta^2$$

In order to cancel errors of order $\delta^2$ for the numerical derivative, we can combine the two derivatives taken above as follows into a new derivative operator:

$$D_{new} = \frac{4D_\delta - D_{2\delta}}{3} = \frac{4f'(x) - f'(x) + \frac{4}{6}f'''(x)\delta^2 - \frac{2}{3}f'''(x)\delta^2}{3} + O(\delta^4) = f'(x) + O(\delta^4)$$

The operator is therefore:

$$D_{new} = \frac{4}{3}\frac{f(x+\delta) - f(x-\delta)}{2\delta} - \frac{1}{3}\frac{f(x+2\delta) - f(x-2\delta)}{4\delta}$$

b)
As in the lectures, the operator is modified a factor of $(1 + g\epsilon)$ multiplying each function call (where $\epsilon$ is the machine precision error of $10^{-16}$). Due to alternating signs, odd powers of $\delta$ cancel and leave only $O(\delta^4)$ terms, therefore the error of the combined derivative operator goes like:

$$Err \sim \frac{\epsilon f(x)}{\delta} + f^{(5)}(x)\delta^4$$

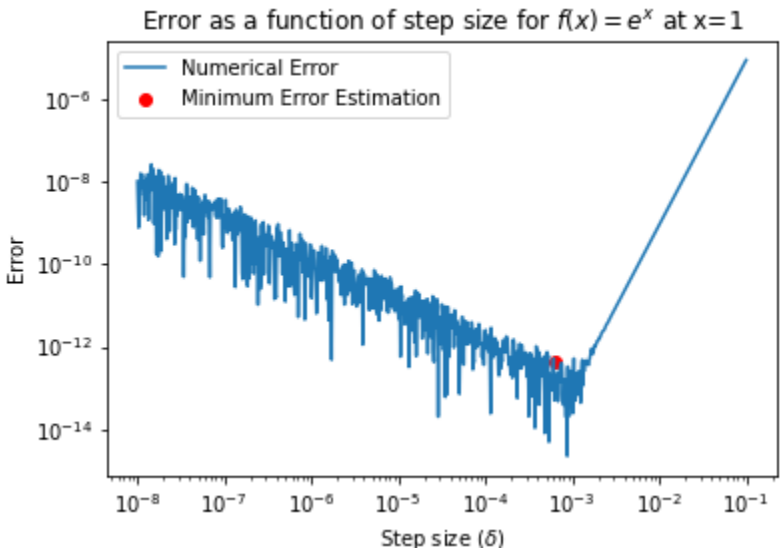
Setting the derivative w.r.t. $\delta$ equal to 0:

$$-\frac{\epsilon f(x)}{\delta^2} + 4f^{(5)}(x)\delta^3 = 0 \Rightarrow \delta = \left(\frac{\epsilon f(x)}{4f^{(5)}(x)}\right)^{\frac{1}{5}}$$

For $f(x) = e^x$, $f(x)$ is of the same order of magnitude as $f^{(5)}(x)$,

$$\delta \sim \epsilon^{\frac{1}{5}} = (10^{-16})^{\frac{1}{5}} = 10^{-3.2}$$

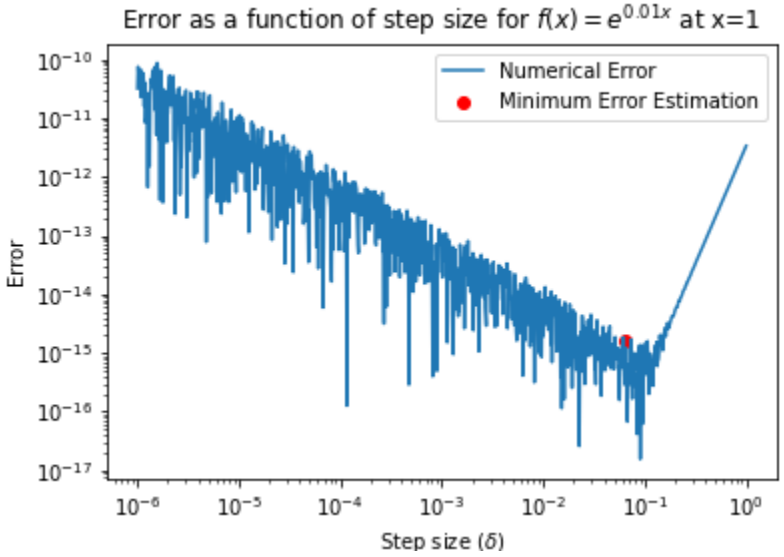
This result is confirmed in the graph below:




For $f(x) = e^{0.01x}$,

$$\frac{f(x)}{f^{(5)}(x)} = 10^{10} \Rightarrow \delta \sim (10^{-16} \cdot 10^{10})^{\frac{1}{5}} = 10^{-1.2}$$

This result is also confirmed in the graph below:



Here is the block of code for $f(x) = e^{0.01x}$ :

```
import numpy as np
from matplotlib import pyplot as plt

logdelta=np.linspace(-6, 0, 1001)
delta=10**logdelta #Create points space apart exponentially

def fun(x):
    return np.exp(0.01*x)

x0=1
eps=10**(-16)  #Machine precision error

y0_1=fun(x0-delta)
y1_1=fun(x0+delta)
d_1=(y1_1-y0_1)/(2*delta) #Derivative operator from +/-delta

y0_2=fun(x0-2*delta)
y1_2=fun(x0+2*delta)
d_2=(y1_2-y0_2)/(4*delta) #Derivative operator from +/-2*delta

d=4/3*d_1-1/3*d_2  #Combined derivative operator cancels delta^2 terms

delta_est=10**(-1.2)  #Estimate of delta with minimum error
approx_err=0.01**5*fun(x0)*delta_est**4
# ^ Approximate error arising from delta (0.01^5 from fifth derivtive)

plt.loglog(delta, np.abs(d-0.01*fun(x0)), label='Numerical Error')
plt.scatter(delta_est, approx_err, color='red', label='Minimum Error Estimation')
plt.xlabel(r'Step size ($\delta$)')
plt.ylabel('Error')
plt.title(r'Error as a function of step size for $f(x)=e^{0.01x}$ at x=1')
plt.legend()
```

## 2 Question 2

As was derived in lecture 1, the order of magnitude estimate for the error on the centered derivative is:

$$Error = \frac{\epsilon f(x)}{dx} + f'''(x)dx^2$$

Setting the derivative equal to 0,

$$dx = \left(\frac{\epsilon f(x)}{f'''(x)}\right)^{\frac{1}{3}}$$

Therefore in order to find the optimal $dx$, we must first calculate the third derivative numerically which is done as such:

$$f'''(x) \simeq \frac{f(x+2dx) - 2f(x+dx) + 2f(x-dx) - f(x-2dx)}{2dx^3}$$

To calculate this third derivative, a rough estimate of $dx = 10^{-5}$ was used (not important to find optimal $dx$ in this case since it is the "error of the error"). With the optimal $dx$ calculated, the numerical derivative can be evaluated as such:

```python
import numpy as np
from matplotlib import pyplot as plt

eps=10**(-16)  #Machine precision error

def ndiff(fun, x, full=False):
    dx=10**-5

    y2=fun(x+2*dx)
    y1=fun(x+dx)
    y_1=fun(x-dx)
    y_2=fun(x-2*dx)
    d3=(y2-2*y1+2*y_1-y_2)/(2*dx**3) #3rd num. deriv. with dx rough estimate

    dx=np.abs(eps*fun(x)/d3)**(1/3) #Estimate for optimal dx using 3rd deriv.
    d=(fun(x+dx)-fun(x-dx))/(2*dx)
    err=eps*fun(x)/dx+d3*dx**2

    if full==True:
        return d, dx, err
    return d
```

As an example to show that the function works when x is an array, here is the plot of the numerical derivative for $f(x) = x^3$: