MA 402: Project 1

Matthew Murray, Johnathan Rhyne

Fall 2019

Instructions:

- Detailed instructions regarding submission are available on the class website[1].

1 Pen-and-paper exercises

1) (5 points) Let $x, \tilde{x} \neq 0$. We have given two definitions of relative error

$$e_1 = \frac{|x - \tilde{x}|}{|x|} \qquad e_2 = \frac{|x - \tilde{x}|}{|\tilde{x}|}.$$

Obtain inequalities between e_1 and e_2 of the form (assuming $e_1 \neq 0$)

$$\frac{e_1}{1 + e_1} \leq e_2 \leq \frac{e_1}{1 - e_1}.$$

We know the following,

$$e_1 = \frac{|x - \tilde{x}|}{|x|} \qquad e_2 = \frac{|x - \tilde{x}|}{|\tilde{x}|}.$$

where $e_1 \neq 0 \wedge x, \tilde{x} \neq 0$.

We'll prove this inequality by showing that the left and right parts are true separately.

(a) First, we show that:

$$e_2 \geq \frac{e_1}{1 + e_1}.$$

$$e_2 = \frac{|x - \tilde{x}|}{|\tilde{x}|} = \frac{|\tilde{x} - x|}{|\tilde{x}|} = \frac{|\tilde{x} - x|}{|\tilde{x} + x - x|}$$

By the triangle inequality,

$$|(\tilde{x} - x) + x| \leq |\tilde{x} - x| + |x|$$

Given two positive numbers a, b with $a \geq b$. We know that

$$\frac{1}{a} \leq \frac{1}{b}.$$

Using the facts, we get the following...

$$\frac{|\tilde{x} - x|}{|\tilde{x} + x - x|} \geq \frac{|\tilde{x} - x|}{|\tilde{x} - x| + |x|} = \frac{\frac{|\tilde{x} - x|}{|x|}}{1 + \frac{|\tilde{x} - x|}{|x|}} = \frac{e_1}{1 + e_1}$$

$\therefore e_2 \geq \frac{e_1}{1+e_1}$.

[1] https://github.ncsu.edu/asaibab/ma402_fall_2019/blob/master/projects.md

(b) Next, we show that:

$$e_2 \leq \frac{e_1}{1-e_1}.$$

Since $\tilde{x} \approx x$, we have two cases:

i.

$$|\tilde{x}| > |x|$$
$$\implies |x| - |x - \tilde{x}| < |\tilde{x}|$$

Interpretation: Suppose we have two positive numbers $|x|$ and $|\tilde{x}|$, with $|\tilde{x}| > |x|$. If we subtract the distance($|x - \tilde{x}|$) between the two numbers from the smaller number($|x|$), the result will always be less than the larger number($|\tilde{x}|$).

ii.

$$|\tilde{x}| < |x|$$
$$\implies |x| - |x - \tilde{x}| = |\tilde{x}|$$

Interpretation: Suppose we have two positive numbers $|x|$ and $|\tilde{x}|$, with $|\tilde{x}| < |x|$. If we subtract the distance($|x - \tilde{x}|$) between the two numbers from the larger number($|x|$), the result will always be equal to the smaller number($|\tilde{x}|$).

We can combine the two inequalities into one:

$$|\tilde{x}| \geq |x| - |x - \tilde{x}|$$

We know that:

$$|\tilde{x}| \geq |x| - |x - \tilde{x}|$$
$$\implies \frac{1}{|\tilde{x}|} \leq \frac{1}{|x| - |x - \tilde{x}|}$$
$$e_2 = \frac{|x - \tilde{x}|}{|\tilde{x}|} \leq \frac{|x - \tilde{x}|}{|x| - |x - \tilde{x}|} = \frac{\frac{|x-\tilde{x}|}{|x|}}{1 - \frac{|x-\tilde{x}|}{|x|}} = \frac{e_1}{1-e_1}$$

$\therefore \frac{e_1}{1+e_1} \leq e_2 \leq \frac{e_1}{1-e_1}$.

Using this relation show that the two definitions give similar estimates of error if $e_1 < 0.01$.

We start with the following $e_1 < 0.01$. Next, we divide both sides of the inequality by $1 - e_1$ to get, $\frac{e_1}{1-e_1} < \frac{0.01}{1-0.01}$.

$$\implies e_2 \leq 0.\overline{01}$$

Again, we start with $e_1 < 0.01$. Next, we divide both sides of the inequality by $1 + e_1$ to get, $\frac{e_1}{1+e_1} < \frac{0.01}{1+0.01}$

$$\implies e_2 \geq 0.\overline{0099}$$

$$\implies 0.\overline{0099} \leq e_2 \leq 0.\overline{01}$$

$\therefore e_1 \approx e_2$ for $e_1 < 0.01$.

2) (10 points) (Thanks to I. Ipsen) Chip manufacturers are branching out from the traditional double-precision floating point (fp) format for high-performance computing, and are targeting the artificial intelligence and machine learning communities with processors that trade off precision for computational speed up.

Intel's Knights Mill and AMD's Radeon Vega Frontier processors can operate in a low-precision fp format consisting of 16 binary bits, which allocates 1 bit for the sign s, 5 bits for the exponent field e, and 10 bits for the fraction field f. Normalized floating point numbers are represented as $(-1)^s \cdot (1.f)_2 \cdot 2^{e-b}$, and all other representations are in analogy with the conventions of IEEE double precision arithmetic.

Determine exact values for the following, and justify your answers:

(a) Bias b, and range of biased exponent $e - b$.

There are 5 bits for the biased exponent in low-precision fp format. This means that there are $2^5 = 32$ possible exponents. The maximum number that can be represented with 5 bits is $(11111)_2 = 2^5 - 1 = 31$ and the minimum number is $(00000)_2 = 0$. By IEEE convention, we reserve these two of these two exponents for special cases(overflow, underflow). This leaves us with $32 - 2 = 30$ possible exponents. We divide this number by 2 to get the bias, b.

$$\implies b = 15$$

$$\implies 1 \leq e \leq 30$$

$$\implies -14 \leq e - b \leq 15$$

(b) Smallest normalized positive and largest fp numbers.

The smallest positive normalized fp number is of the form:

$$n = (-1)^0 \cdot (1.0000000000)_2 \cdot 2^{-14}$$

$$\implies n = 2^{-14}$$

The largest positive normalized fp number is of the form:

$$N = (-1)^0 \cdot (1.1111111111)_2 \cdot 2^{15}$$

$$\implies N = (1 + 2^{-1} + 2^{-2} + ... + 2^{-9} + 2^{-10}) \cdot 2^{15}$$

(c) Machine epsilon (distance from 1 to next larger fp number).

Machine epsilon,ϵ, is the distance between 1 and the next larger fp number.

$$\epsilon = (-1)^0 \cdot (1.0000000001)_2 \cdot 2^0 - (-1)^0 \cdot (1.0000000001)_2 \cdot 2^0$$

$$\implies \epsilon = (0.0000000001)_2 = 2^{-10}$$

(d) Smallest and largest positive denormalized fp numbers.

The smallest positive denormalized fp number is of the form:

$$n_d = (-1)^0 \cdot (0.0000000001)_2 \cdot 2^{-14}$$

$$\implies n_d = 2^{-10} \cdot 2^{-14} = 2^{-14}$$

The largest positive denormalized fp number is of the form:

$$N_d = (-1)^0 \cdot (0.1111111111)_2 \cdot 2^{-14}$$

$$\implies N_d = (2^{-1} + 2^{-2} + ... + 2^{-9} + 2^{-10}) \cdot 2^{-14}$$

3) (6 points) Let $x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \in \mathbb{R}^2$ and let $\|x\| = |x_1 - x_2| + |x_2|$. Is $\|\cdot\|$ a norm? Give either a proof or a counterexample.

Showing that $\|x\|$ is a norm:

(a) Positivity:

i. $\|x\| \geq 0 \ \forall x \in \mathbb{R}^2$

By definition, we know that $|x| \geq 0$. The sum of two absolute numbers($|x_1 - x_2|$,$|x_2|$) is also at least 0. $\therefore \|x\| = |x_1 - x_2| + |x_2| \geq 0 \ \forall x \in \mathbb{R}^2$.

ii. $\|x\| = 0$ iff $x = \underline{0}$

i. $x = \underline{0} \implies \|x\| = 0$

If $x = \underline{0} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$, $\|x\| = |0 - 0| + |0| = 0$

ii. $\|x\| = 0 \implies x = \underline{0}$

If $\|x\| = 0 \implies |x_1 - x_2| + |x_2| = 0$. If two non-negative numbers ($|x_1 - x_2|$,$|x_2|$) sum to zero then both numbers must be zero. $\implies |x_1 - x_2| = 0$ and $\implies |x_2| = 0$. If $|x_2| = 0$, then $x_2 = 0$. And if $x_2 = 0$, then $|x_1 - 0| = |x_1| = x_1 = 0$. If $x_1 = 0$ and $x_2 = 0$, then $x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$.

(b) Scaling:

Show that $\|\alpha x\| = |\alpha|\|x\| \ \forall x \in \mathbb{R}^2$, and $\alpha \in \mathbb{R}$.

$$\alpha x = \begin{bmatrix} \alpha x_1 \\ \alpha x_2 \end{bmatrix}$$

$$\implies \|\alpha x\| = |\alpha x_1 - \alpha x_2| + |\alpha x_2| = |\alpha(x_1 - x_2)| + |\alpha||x_2| = |\alpha||x_1 - x_2| + |\alpha||x_2|$$

$$\implies \|\alpha x\| = |\alpha|(|x_1 - x_2| + |x_2|) = |\alpha|\|x\|$$

(c) Triangle Inequality:

Show that for any two vectors $x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$,$y = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} \in \mathbb{R}^2$:

$$\|x + y\| \leq \|x\| + \|y\|.$$

$$x + y = \begin{bmatrix} x_1 + y_1 \\ x_2 + y_2 \end{bmatrix}$$

$$\implies \|x + y\| = |(x_1 + y_1) - (x_2 + y_2)| + |x_2 + y_2| = |(x_1 - x_2) + (y_1 - y_2)| + |x_2 + y_2|$$

$$\implies \|x + y\| \leq |x_1 - x_2| + |y_1 - y_2| + |x_2| + |y_2| = |x_1 - x_2| + |x_2| + |y_1 - y_2| + |y_2|$$

$$\implies \|x + y\| \leq \|x\| + \|y\|$$

$\therefore \|\cdot\|$ is a norm.

4) (9 points) Conditioning of vector addition

(a) (5 points) Let $x, y \in \mathbb{R}^n$ be nonzero vectors and let $\tilde{x}, \tilde{y} \in \mathbb{R}^n$ respectively. Derive the following bound for the relative error in the addition (provided $x + y \neq 0$)

$$\frac{\|(\tilde{x}+\tilde{y})-(x+y)\|_2}{\|x+y\|_2} \leq \frac{\|x\|_2+\|y\|_2}{\|x+y\|_2} \max\left\{\frac{\|x-\tilde{x}\|_2}{\|x\|_2}, \frac{\|y-\tilde{y}\|_2}{\|y\|_2}\right\}.$$

Let $\tilde{x} = x + e$ and $\tilde{y} = y + f$.

$$\implies \frac{\|(\tilde{x}+\tilde{y})-(x+y)\|_2}{\|x+y\|_2} = \frac{\|(x+e+y+f)-(x+y)\|_2}{\|x+y\|_2} = \frac{\|e+f\|_2}{\|x+y\|_2}$$

By the Triangle Inequality,

$$\implies \frac{\|(\tilde{x}+\tilde{y})-(x+y)\|_2}{\|x+y\|_2} \leq \frac{\|e\|_2+\|f\|_2}{\|x+y\|_2} = \frac{\|\tilde{x}-x\|_2+\|\tilde{y}-y\|_2}{\|x+y\|_2}$$

Plugging in for e and f,

$$\implies \frac{\|(\tilde{x}+\tilde{y})-(x+y)\|_2}{\|x+y\|_2} \leq \frac{\|\tilde{x}-x\|_2\frac{\|x\|_2}{\|x\|_2}+\|\tilde{y}-y\|_2\frac{\|y\|_2}{\|y\|_2}}{\|x+y\|_2}$$

$$\implies \frac{\|\tilde{x}-x\|_2\frac{\|x\|_2}{\|x\|_2}+\|\tilde{y}-y\|_2\frac{\|y\|_2}{\|y\|_2}}{\|x+y\|_2} \leq \frac{\|x\|_2+\|y\|_2}{\|x+y\|_2} \max\left\{\frac{\|x-\tilde{x}\|_2}{\|x\|_2}, \frac{\|y-\tilde{y}\|_2}{\|y\|_2}\right\}$$

$$\therefore \frac{\|(\tilde{x}+\tilde{y})-(x+y)\|_2}{\|x+y\|_2} \leq \frac{\|x\|_2+\|y\|_2}{\|x+y\|_2} \max\left\{\frac{\|x-\tilde{x}\|_2}{\|x\|_2}, \frac{\|y-\tilde{y}\|_2}{\|y\|_2}\right\}.$$

(b) (2 points) Give an interpretation of this bound.

This means that the relative error in the output for the addition of two vectors is less than or equal to the relative error in the inputs times the condition number for this operation.

(c) (2 points) Identify the condition number with respect to addition. Show that the condition number is greater than or equal to 1.

The condition number w.r.t addtion is,

$$K = \frac{\|x\|_2+\|y\|_2}{\|x+y\|_2}$$

By the Triangle Inequality,

$$\|x+y\|_2 \leq \|x\|_2+\|y\|_2$$

$$\implies K = \frac{\|x\|_2+\|y\|_2}{\|x+y\|_2} \geq \frac{\|x\|_2+\|y\|_2}{\|x\|_2+\|y\|_2} = 1$$

$\therefore K \geq 1$.

2 Numerical exercises

5) (10 points) Consider the quadratic equation $ax^2 + bx + c$, with $a \neq 0$, which has roots

$$x_{\pm} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}.$$

Consider the values $a = c = 1$, $b = 10^8$.

(a) (2 points) Write a MATLAB/Python function `myroots` to implement the formula described above.

```
"""
@author: Matthew Murray
"""
##Imports
import math
import numpy as np
##Define the myroots function: gives the roots of a quadratic equation
def myroots(a,b,c):
    Roots = np.array([(-b-math.sqrt(b**2-4*a*c))/2*a,format((-b+math.
        sqrt(b**2-4*a*c))/2*a,'16')])
    return Roots
r = myroots(1,10**8,1)
print(r) ##Roots are given in the form [x-,x+]

##Numpy's default root finder
Roots_default = np.roots([1,10**8,1])
print(Roots_default) ##Roots are given in the form [x-,x+]

##Define the myrootsacc function: correct for catastrophic cancelation
def myrootsacc(a,b,c):
    Roots = np.array([(-b-math.sqrt(b**2-4*a*c))/2*a,-2*c/(b+math.sqrt
        (b**2-4*a*c))])
    return Roots
r_acc = myrootsacc(1,10**8,1)
print(r_acc) ##Roots are given in the form [x-,x+]

rel = (float(r_acc[1])-float(r[1]))/float(r_acc[1])##The relative
    error between the negative root(x+) of myroots and myrootsacc
print(rel)
```

(b) (2 points) How do the roots compare with the default implementation in MATLAB/Python? (For MATLAB use `roots`, for Python use `numpy.roots`).

Outputs of the program:

$$myroots(1, 10^8, 1) = \begin{bmatrix} x_- \\ x_+ \end{bmatrix} = \begin{bmatrix} -100000000.0 \\ -7.450580596923828e-09 \end{bmatrix}$$

$$numpy.roots(1, 10^8, 1) = \begin{bmatrix} x_- \\ x_+ \end{bmatrix} = \begin{bmatrix} -1.e+08 \\ -1.e-08 \end{bmatrix}$$

The relative error between the roots x_+ is 0.2549419403076172. The relative error is large, this makes sense because of the cancellation errors that occur in $myroots(1, 10^8, 1)$.

(c) (3 points) Suggest a different way to compute the roots that accounts for the cancellation errors.

The problem root is:

$$x_+ = \frac{-b + \sqrt{b^2 - 4ac}}{2a}$$

Multiply and divide by the conjugate of the numerator:

$$x_+ = \frac{-b + \sqrt{b^2 - 4ac}}{2a} \cdot \frac{(-b - \sqrt{b^2 - 4ac})}{(-b - \sqrt{b^2 - 4ac})}$$

$$\implies x_+ = \frac{-2c}{b + \sqrt{b^2 - 4ac}}$$

(d) (3 points) Implement this formula as a function `myroots_acc` and compare it with the default implementation. Comment on the accuracy.

Output of the program:

$$myrootsacc(1, 10^8, 1) = \begin{bmatrix} x_- \\ x_+ \end{bmatrix} = \begin{bmatrix} -1.e+08 \\ -1.e-08 \end{bmatrix}$$

These were the same answers that we got when we used Python's default root finder.

Note: MATLAB users should be a bit careful because the display can be a bit misleading. Use `format long` before performing calculations and print out each number to 15 digits after decimal point.

6) (10 points) Let $f(x)$ be a differential function on the real line. The derivative is defined as

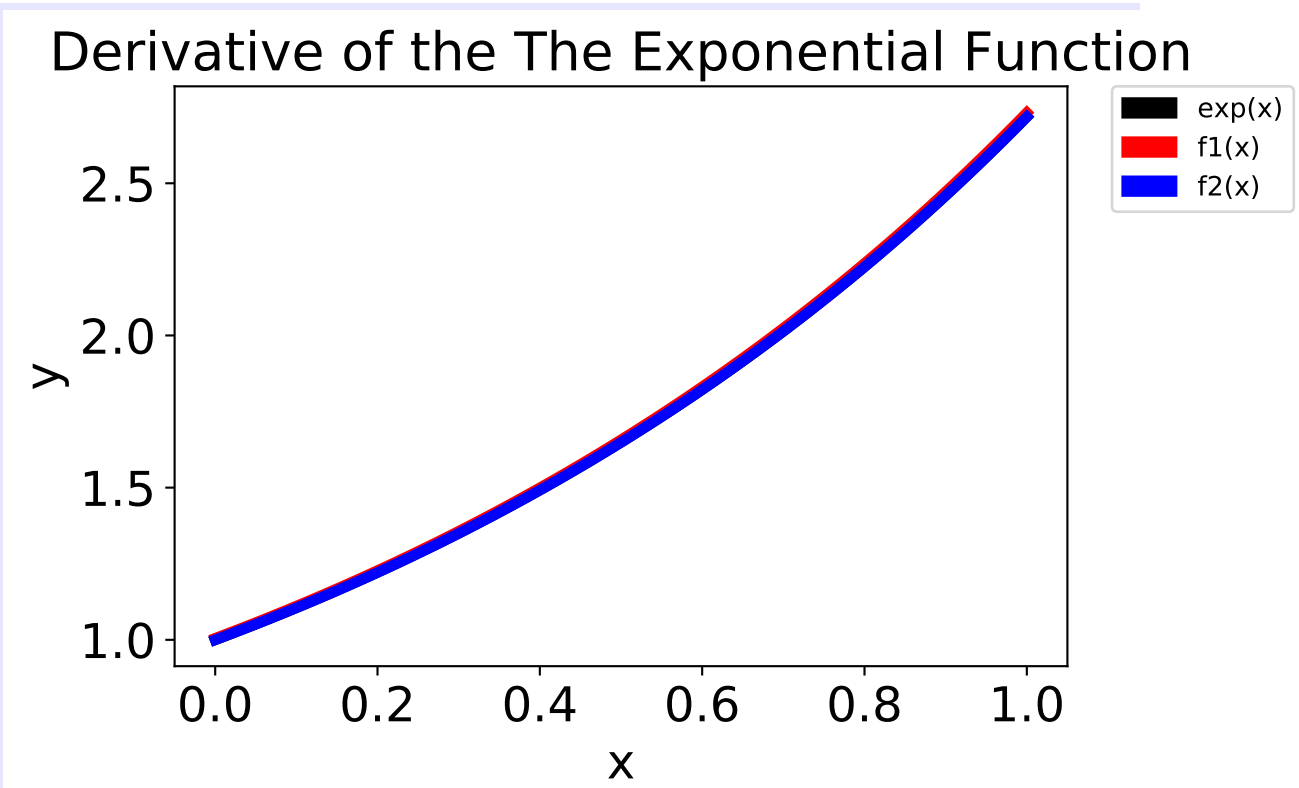
$$f'(x) = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h}.$$

Here are two different approximations to the derivative

$$f_1(x) = \frac{f(x+h) - f(x)}{h} \qquad f_2(x) = \frac{f(x+h) - f(x-h)}{2h}.$$

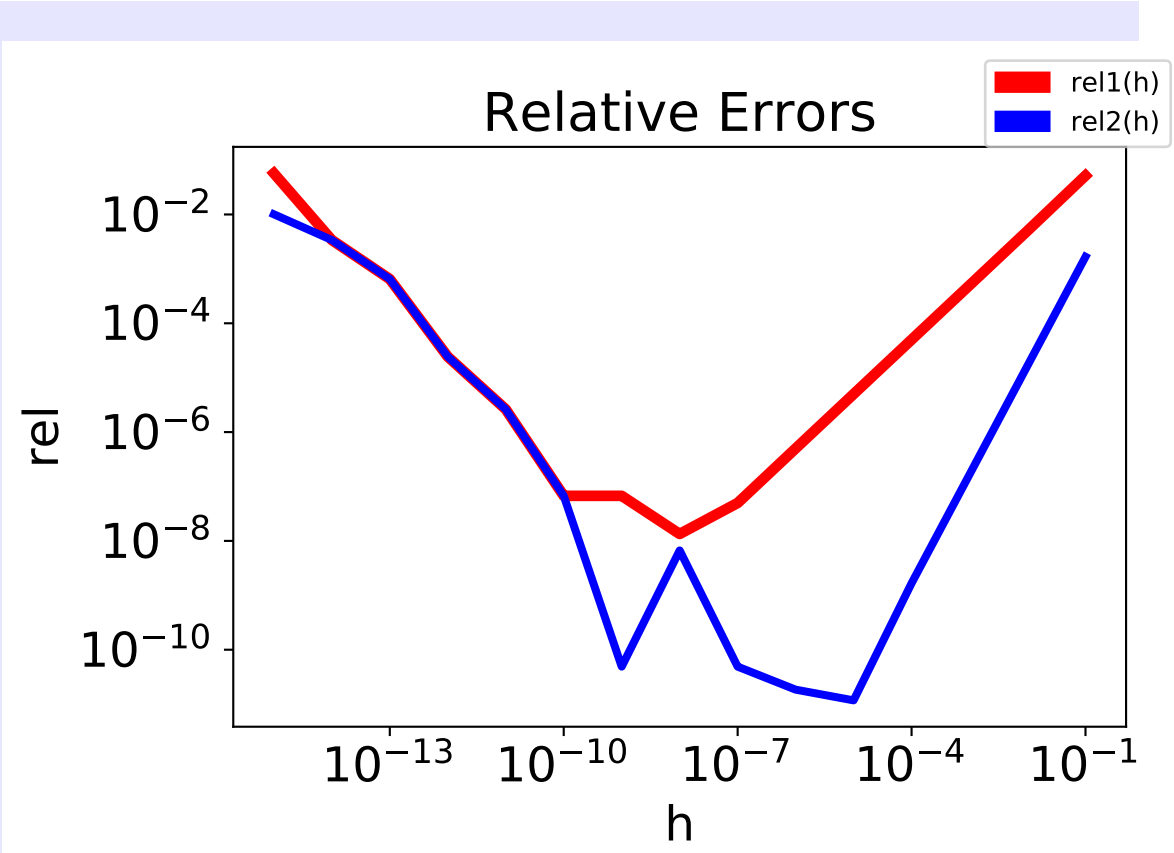
We anticipate that these approximations are accurate if $h \approx 0$. We investigate its accuracy numerically for the function $f(x) = \exp(x)$.

(a) (4 points) In the interval $[0, 1]$, construct 100 evenly spaced points. Plot the true derivative at these points as well as the two approximations for $h = 10^{-2}$. Comment on the accuracy of these two approximations.



The predictions appear to be very accurate. The two approximations to the derivative of the exponential function appear to hold up well.

(b) (4 points) Now fix $x = 0.5$ and let $h \in \{10^{-1}, \ldots, 10^{-15}\}$. On a `log-log` scale plot the relative error of the two approximations as a function of h. Comment on the accuracy of these two approximations.



In our graph $rel1(x)$ and $rel2(x)$ are the relative errors between $f'(x)$ and $f1(x)$ and $f'(x)$ and $f2(x)$, respectively. The relative errors curves are high when h is large as one would expect. Also the $rel2(x)$ is beneath $rel1(x)$, which means that the derivative approximating function $f2(x)$ is does a better job at approximating the derivative of the exponential function than $f1(x)$

(c) (2 points) Can you explain the behavior of the error? Why is there large error when h is small?

The curves appear to reach a minimum and then rise again when h is very small. There must be a source of error that is becoming more noticeable as we decrease h. That source of error is dominating the error due to us approximating the derivative of the exponential function. This error could be round-off error.

Instructions: Each plot should have a title, all the axes labelled, a legible legend (for each curve in the plot). Failure to follow these instructions will result in points deduction.

Program for Both Graphs

```
"""
@author: Matthew Murray
"""
#Imports
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.patches as mpatches

##Variables
h = 1e-2
x = np.arange(0,1+h,h) ##Interval [0,1] w/ 100 evenly spaced points

##f'(x) Functions
def f_1(x):
    return (np.exp(x+h)-np.exp(x))/h
def f_2(x):
    return (np.exp(x+h)-np.exp(x-h))/(2*h)
def f_prime(x):
    return np.exp(x)

## Plot 1
f = plt.figure(1)
plt.plot(x,f_prime(x),color=(0,0,0),markersize=10,linewidth=4)
plt.plot(x,f_1(x),color=(1,0,0),markersize=10,linewidth=3)
plt.plot(x,f_2(x),color=(0,0,1),markersize=10,linewidth=2)
plt.title('Derivative of the The Exponential Function',fontsize=20)
plt.xlabel('x',fontsize=18)
plt.ylabel('y', fontsize = 18)
plt.rcParams['xtick.labelsize']= 18
plt.rcParams['ytick.labelsize']= 18
black_patch = mpatches.Patch(color='black', label='exp(x)')
red_patch = mpatches.Patch(color='red', label='f1(x)')
blue_patch = mpatches.Patch(color='blue', label = 'f2(x)')

plt.legend(handles=[black_patch,red_patch,blue_patch],bbox_to_anchor
    =(1.05,1),loc=2,borderaxespad=0.)

f.show()
# save as PDF
f.savefig("project1graph1.pdf", bbox_inches='tight')

##Relative Errors
hs = np.array([1e-15,1e-14,1e-13,1e-12,1e-11,1e-10,1e-9,1e-8,1e-7,1e-6,1e
    -5,1e-4,1e-3,1e-2,1e-1])
def rel_1(hs):
    return np.abs(f_prime(0.5)- np.divide((np.exp(0.5+hs)-np.exp(0.5)),hs)
        )/np.exp(0.5)
def rel_2(hs):
    return np.abs(f_prime(0.5)- np.divide((np.exp(0.5+hs)-np.exp(0.5-hs))
        ,2*hs))/np.exp(0.5)
```

```
##Plot 2
g = plt.figure(2)
plt.loglog(hs,rel_1(hs),color=(1,0,0),markersize=10,linewidth=4)
plt.loglog(hs,rel_2(hs),color=(0,0,1),markersize=10,linewidth=3)
plt.title('Relative Errors',fontsize=20)
plt.xlabel('h',fontsize=18)
plt.ylabel('rel', fontsize = 18)
plt.rcParams['xtick.labelsize']= 18
plt.rcParams['ytick.labelsize']= 18
red_patch = mpatches.Patch(color='red', label='rel1(h)')
blue_patch = mpatches.Patch(color='blue', label = 'rel2(h)')

plt.legend(handles=[red_patch,blue_patch],bbox_to_anchor=(1.05,1),loc=4,
    borderaxespad=0.)
g.show()
# save as PDF
g.savefig("project1graph2.pdf", bbox_inches='tight')
```