September 28, 2021
Task Sheet 1
David Merkley
A02204704

Tasks

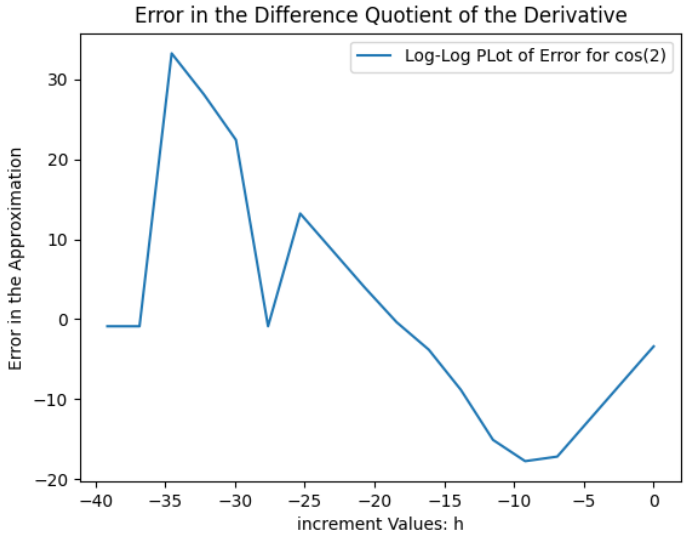
1. I calculated the central difference approximation in python. I have attached code below to show the table and code used.

```
The exact derivative is:  0.4161468365471424
This is the list of differences between exact value and the approximate value:
h-value:  1
0.38260348236197905
h-value:  0.5
0.4075490368602161
h-value:  0.01
0.4161433686711291
h-value:  0.001
0.4161468019070469
h-value:  0.0001
0.41614681700608
h-value:  1e-05
0.4161465563279882
h-value:  1e-06
0.41600056732704616
h-value:  1e-07
0.3937555783673479
h-value:  1e-08
-0.27792935153087156
h-value:  1e-09
-54.678857558163536
h-value:  1e-10
-5550.282829452687
h-value:  1e-11
-555110.6800189051
h-value:  1e-12
0.0
h-value:  1e-13
-5551115122.293488
h-value:  1e-14
-1665334536937.7349
h-value:  1e-15
-277555756156288.3
h-value:  1e-16
0.0
h-value:  1e-17
0.0
```

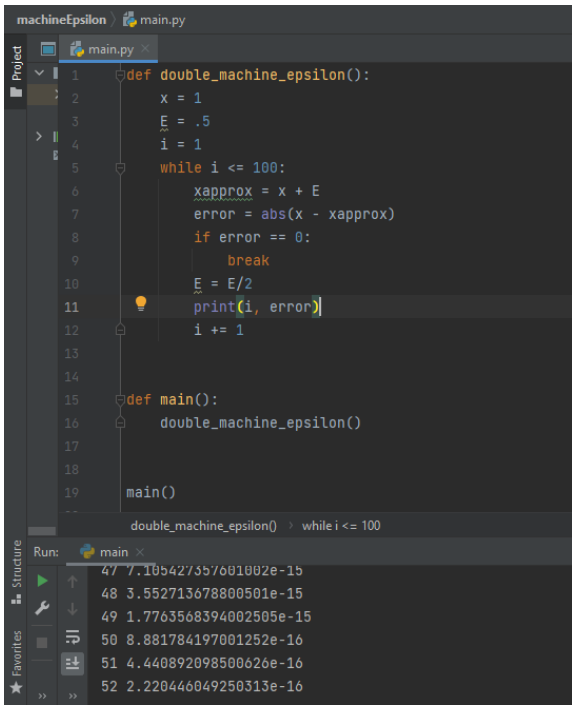
main.py

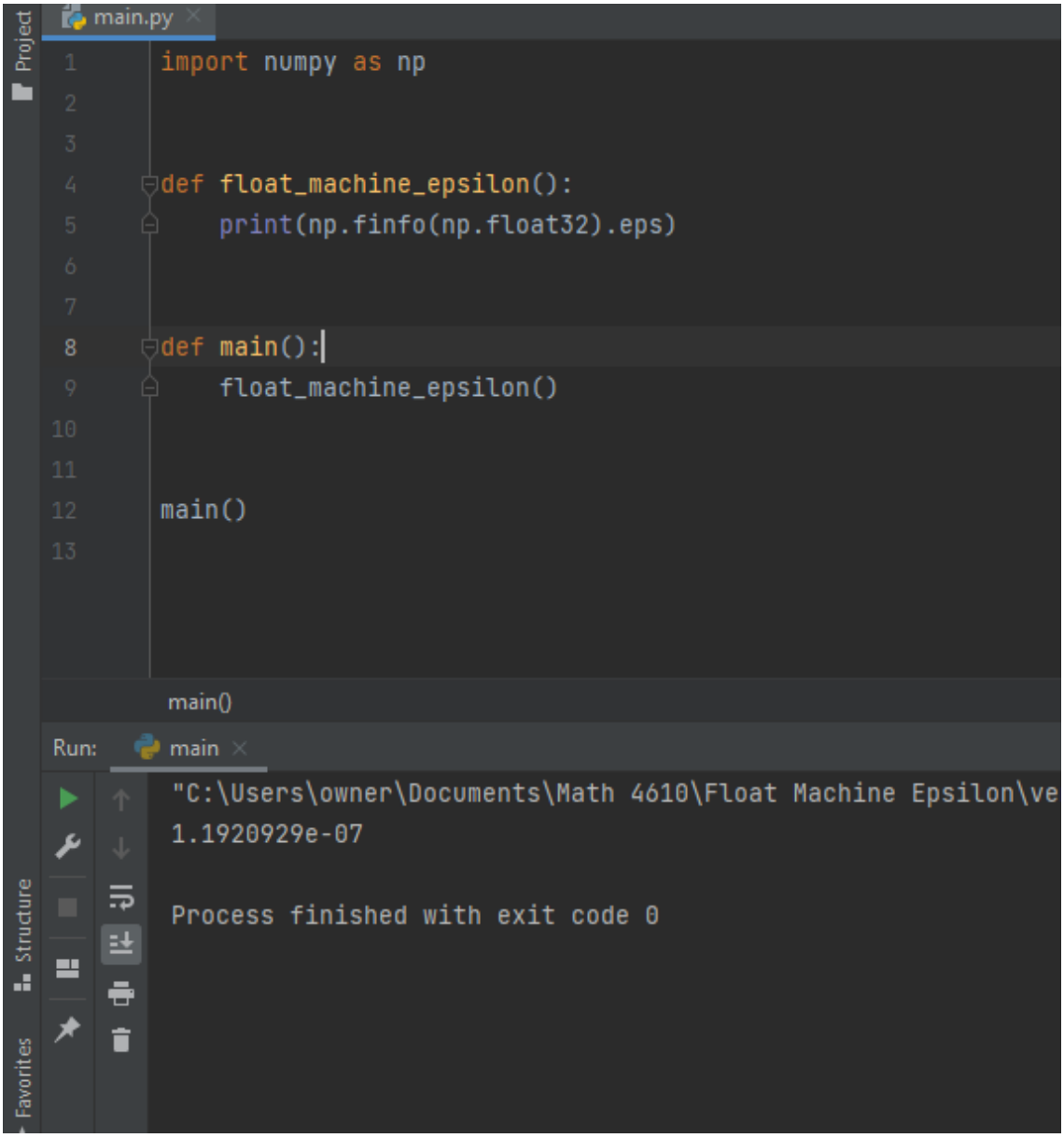
```python
import matplotlib.pyplot as plt
import numpy as np

aval = 2
exactVal = -np.cos(aval)

x = []
y = []

h = []
error = []

print("The exact derivative is: ", exactVal)

l = 0

print("This is the list of differences between exact value and the approximate value: ")

while l < 18:
    if l != 1:
        h.append(10**(-l))
    else:
        h.append(.5)
    print("h-value: ", h[l])
    numval = np.cos(aval + h[l]) - (2 * np.cos(aval)) + np.cos(aval - h[l])
    denom = h[l]**2
    dfVal = numval / denom

    error.append(np.abs(dfVal - exactVal))

    x.append(np.log(h[l]))
    y.append(np.log(error[l]))

    print(exactVal - error[l])

    l += 1

plt.title("Error in the Difference Quotient of the Derivative")
plt.xlabel("increment Values: h")
plt.ylabel("Error in the Approximation")
plt.plot(x, y, label="Log-Log PLot of Error for cos(2)")
plt.legend()
plt.show()
```

2. The graph shows how the error is reduced as h is approaching 0. It is around when h equals -10 when the approximation starts to fail. It goes all out of whack. As h goes above 0, we start to get a parabola, which shows that it is second order accurate.



3. I have attached pictures of my code for double machine epsilon and float machine epsilon. The first picture is the double machine epsilon, and the second is the float machine epsilon.




main.py
import numpy as np
def float_machine_epsilon():
print(np.finfo(np.float32).eps)
def main():
float_machine_epsilon()
main()
main()
Run:
main
"C:\Users\owner\Documents\Math 4610\Float Machine Epsilon\ve
1.1920929e-07
Process finished with exit code 0
Project
Structure
Favorites

4. I created the files to eventually be put into a library.

```
sweetbabydave@WINDOWS-FAUNA1D:~$ cd -
/home/sweetbabydave/software_manual
sweetbabydave@WINDOWS-FAUNA1D:~/software_manual$ ls
software_manual_toc.md
sweetbabydave@WINDOWS-FAUNA1D:~/software_manual$ cat software_manual_toc.md
# Math 4610 Fundamentals of Computational Mathematics Software Manual Template File
This is a template file for building an entry in the student software manual project. You should use the formatting below to
define an entry in your software manual.

**Routine Name:**           smaceps

**Author:** Joe Koebbe

**Language:** Fortran. The code can be compiled using the GNU Fortran compiler (gfortran).

For example,

    gfortran smaceps.f

will produce an executable **./a.exe** than can be executed. If you want a different name, the following will work a bit
better
```

5. I did my best to create a shared library picture below.

```
sweetbabydave@WINDOWS-FAUNA1D:~/math4610/software_manual$ ls
dmaceps.html  smaceps.html  software_manual_toc.md
sweetbabydave@WINDOWS-FAUNA1D:~/math4610/software_manual$ ar
> ls
-bash: syntax error near unexpected token `ls'
sweetbabydave@WINDOWS-FAUNA1D:~/math4610/software_manual$ ar
> smaceps.html dmaceps.html
-bash: syntax error near unexpected token `smaceps.html'
sweetbabydave@WINDOWS-FAUNA1D:~/math4610/software_manual$ ar
> smaceps.html
-bash: syntax error near unexpected token `smaceps.html'
sweetbabydave@WINDOWS-FAUNA1D:~/math4610/software_manual$
```

6. Shared libraries are very useful, they are libraries loaded by programs when they start. Every shared library has a real name and a "soname." It also saves memory and disk storage space. Along with making executable files easier to maintain. Updating a library updates all executable code that use the library.

Citations

https://tldp.org/HOWTO/Program-Library-HOWTO/shared-libraries.html
http://osr507doc.sco.com/en/tools/ShLib_WhatIs.html