Palmer Edholm
Dr. Joe Koebbe
MATH 4610
September 27, 2021

Tasksheet 3

Task 1 The following is a table of the h values used, the error values from the approximations, and the calculated values using the centered difference approximation.

```
|   iter |            h |            error |           dfVal |
|--------+--------------+------------------+-----------------|
|      1 | 0.5          |      0.0085978   |      0.407549   |
|      2 | 0.25         |      0.00216292  |      0.413984   |
|      3 | 0.125        |      0.000541576 |      0.415605   |
|      4 | 0.0625       |      0.000135447 |      0.416011   |
|      5 | 0.03125      |      3.3865e-05  |      0.416113   |
|      6 | 0.015625     |      8.46646e-06 |      0.416138   |
|      7 | 0.0078125    |      2.11663e-06 |      0.416145   |
|      8 | 0.00390625   |      5.29157e-07 |      0.416146   |
|      9 | 0.00195312   |      1.32289e-07 |      0.416147   |
|     10 | 0.000976562  |      3.30163e-08 |      0.416147   |
|     11 | 0.000488281  |      7.98698e-09 |      0.416147   |
|     12 | 0.000244141  |      1.2349e-09  |      0.416147   |
|     13 | 0.00012207   |      4.35304e-09 |      0.416147   |
|     14 | 6.10352e-05  |      8.07833e-09 |      0.416147   |
|     15 | 3.05176e-05  |      3.78807e-08 |      0.416147   |
|     16 | 1.52588e-05  |      1.5709e-07  |      0.416147   |
|     17 | 7.62939e-06  |      1.34918e-06 |      0.416148   |
|     18 | 3.8147e-06   |      2.30286e-06 |      0.416149   |
|     19 | 1.90735e-06  |      6.11755e-06 |      0.416153   |
|     20 | 9.53674e-07  |      5.18939e-05 |      0.416199   |
|     21 | 4.76837e-07  |      0.000112929 |      0.41626    |
|     22 | 2.38419e-07  |      0.000845351 |      0.416992   |
|     23 | 1.19209e-07  |      0.00182191  |      0.417969   |
|     24 | 5.96046e-08  |      0.00572816  |      0.421875   |
|     25 | 2.98023e-08  |      0.0213532   |      0.4375     |
|     26 | 1.49012e-08  |      0.333853    |      0.75       |
|     27 | 7.45058e-09  |      0.583853    |      1          |
|     28 | 3.72529e-09  |      3.58385     |      4          |
|     29 | 1.86265e-09  |     15.5839      |     16          |
|     30 | 9.31323e-10  |     63.5839      |     64          |
|     31 | 4.65661e-10  |    255.584       |    256          |
|     32 | 2.32831e-10  |   1023.58        |   1024          |
|     33 | 1.16415e-10  |      0.416147    |      0          |
|     34 | 5.82077e-11  | 16383.6          | 16384           |
|     35 | 2.91038e-11  | 65535.6          | 65536           |
|     36 | 1.45519e-11  |      0.416147    |      0          |
|      37  | 7.27596e-12 |      1.04858e+06 |      1.04858e+06 |
|      38  | 3.63798e-12 |      4.1943e+06  |      4.1943e+06  |
|      39  | 1.81899e-12 |      1.67772e+07 |      1.67772e+07 |
|      40  | 9.09495e-13 |      0.416147    |      0           |
|      41  | 4.54747e-13 |      0.416147    |      0           |
|      42  | 2.27374e-13 |      1.07374e+09 |      1.07374e+09 |
|      43  | 1.13687e-13 |      4.29497e+09 |      4.29497e+09 |
```

Task 2 Figure 1 shows the log-log plot created from the table above. The figure shows how the error

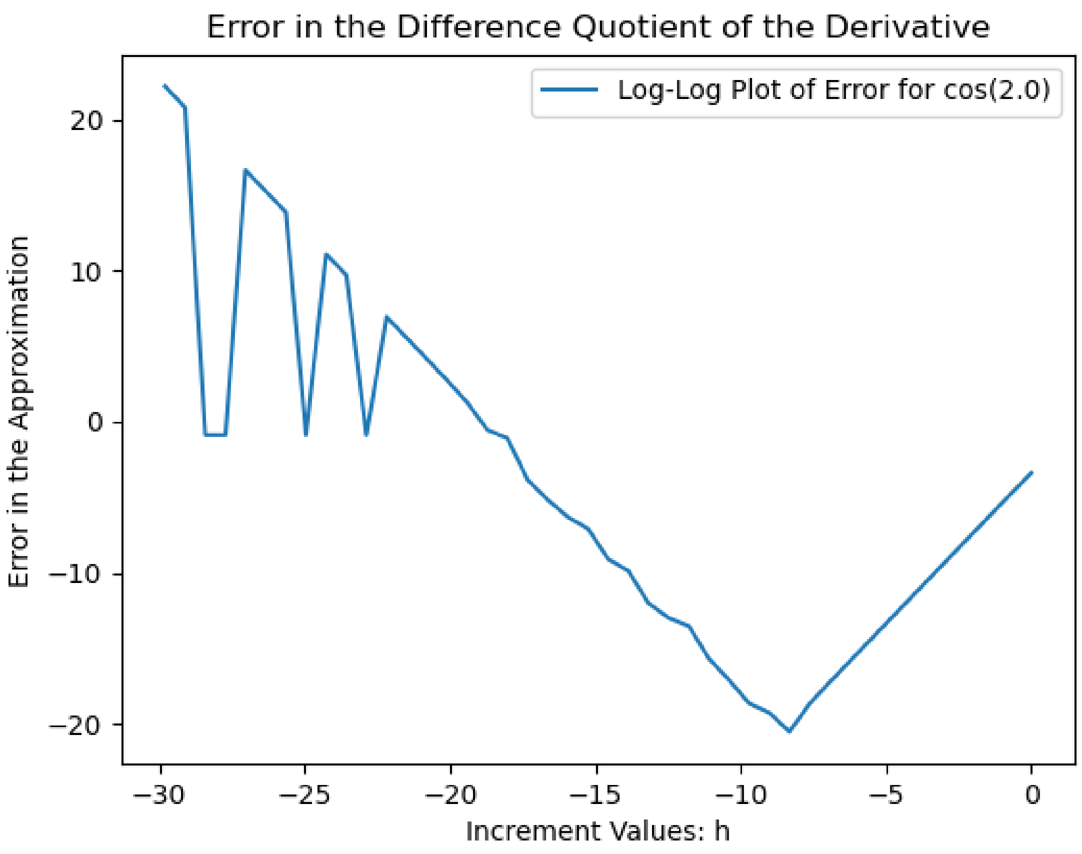


Figure 1: log-log plot

is reduced as h approaches zero until the value is too small. Before the graph explodes, the slope is approximately -2. Since this is a log-log plot, what that effectively means is that the error is being reduced at a rate of h^2. Or, in other words, the approximation is second order accurate. At a value close to $\log(h) = -10$, the approximation begins to fail. The Python code used to generate the table and figure 1 is as follows:

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

```python
#
# initialize the exact value of the derivative
# --------------------------------------------
#
list=[]
aval = 2.0
exactVal = -np.cos(aval)
#
# set up the arrays for plotting the log-log plot we need
# -------------------------------------------------------
#
x = []
y = []
#
# initialize the array for the increment size and error in the
# finite difference approximation
# ------------------------------------------------------------
#
h = []
error = []
#
# append the initial increment with a starting value - in this
# case, 1.0
# ------------------------------------------------------------
#
h.append(1.0)
#
# compute the difference quotient for the increment value
# -------------------------------------------------------
#
dfVal = ( np.cos(aval + h[0]) - 2*np.cos(aval) +
          np.cos(aval - h[0]) ) / (h[0]**2)
error.append(np.abs(exactVal - dfVal))
#
# append the log-log point for plotting at the end
# ------------------------------------------------
#
x.append(np.log(h[0]))
y.append(np.log(error[0]))
#
# print the exact value for sanity
# --------------------------------
#
print('The exact derivative value is: ', exactVal)
#
# set a loop counter
# ------------------
#
```

```
l=1
#
# the loop over ndiv increments
# -----------------------------
#
ndiv = 44
while l<44:
    # print(dfVal)
    #
    # append the next increment of h
    # ------------------------------
    #
    h.append(0.5 * h[l-1])
    #
    # compute the numerator and denominator for the difference
    # approximation and compute the approximation from these
    # --------------------------------------------------------
    #
    numval = np.cos(aval + h[l]) - 2*np.cos(aval) + \
             np.cos(aval - h[l])
    denom = (h[l]**2)
    dfVal = numval / denom
    #
    # compute the error in the approximation
    # --------------------------------------
    #
    error.append(np.abs( dfVal - exactVal ))
    #
    # append the log-log point to the arrays for plotting below
    # ---------------------------------------------------------
    #
    x.append(np.log(h[l]))
    y.append(np.log(error[l]))
    #
    # update the loop iterator
    # ------------------------
    #
    list.append([l,h[l],error[l],dfVal])
    l += 1
#
# set up a plot for the data generated
# ------------------------------------
#
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(2.0)')
plt.legend()
plt.show()
table = tabulate(list, headers=['iter', 'h', 'error', 'dfVal'],
                 tablefmt='orgtbl')
print(table)
```

Task 3 The routine smaceps that produces single precision machine epsilon is

```
import numpy as np

def smaceps():
    # Initialize variables to compute the machine value near 1
    one = np.single(1)
    seps = np.single(1)
    appone = np.single(one + seps)
    # Loop, dividing by 2 each time to determine when the difference
    # between one and the approximation is zero in single precision
    ipow = 0
    # Break out of the loop if the comparison is small enough
    while abs(appone - one) != np.single(0):
        ipow += 1
        # Update the perturbation and compute the approximation to
        # one
        seps = np.single(seps / 2)
        appone = one + seps
    return f'Bits: {ipow}, Value: {seps}'
```

and returns the following output:

```
Bits: 24, Value: 5.960464477539063e-08
```

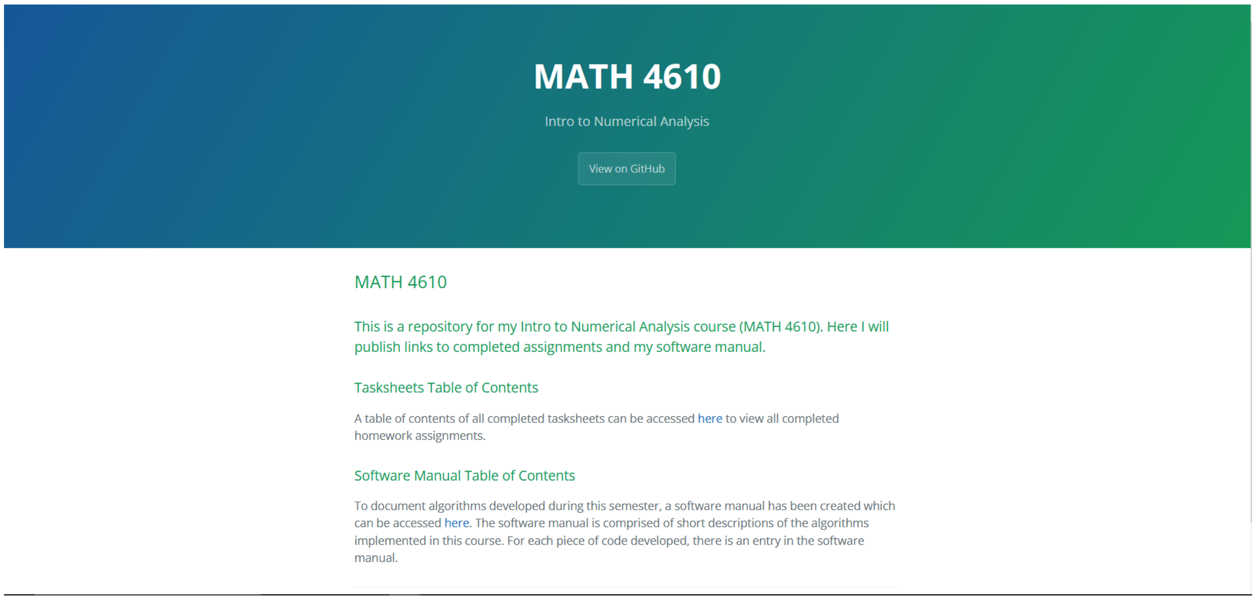
The routine dmaceps that produces double precision machine epsilon is

```
def dmaceps():
    # Initialize variables to compute the machine value near 1.0
    one = 1.0
    seps = 1.0
    appone = one + seps
    # Loop, dividing by 2 each time to determine when the difference
    # between one and the approximation is zero in double precision
    ipow = 0
    # Break out of the loop if the comparison is small enough
    while abs(appone - one) != 0.0:
        ipow += 1
        # Update the perturbation and compute the approximation to
        # one
        seps = seps / 2
        appone = one + seps
    return f'Bits: {ipow}, Value: {seps}'
```

and returns the following output:

```
Bits: 53, Value: 1.1102230246251565e-16
```

Task 4 A table of contents link was added to the home README.md document as seen in figure 2.



MATH 4610

This is a repository for my Intro to Numerical Analysis course (MATH 4610). Here I will publish links to completed assignments and my software manual.

Taskshеets Table of Contents

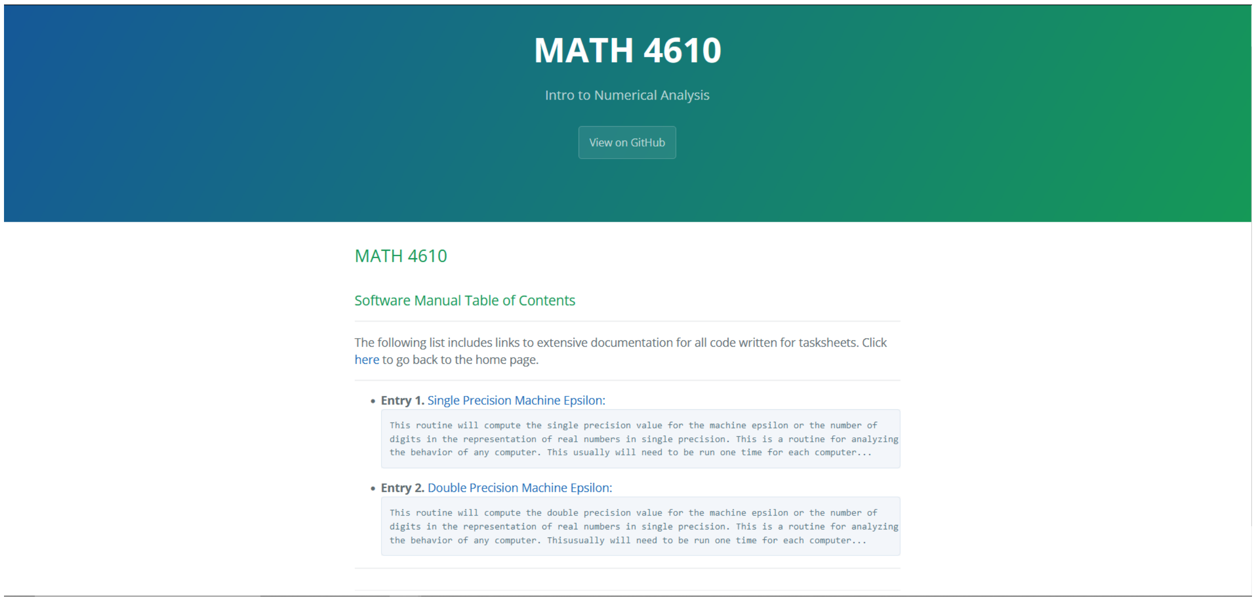
A table of contents of all completed tasksheets can be accessed here to view all completed homework assignments.

Software Manual Table of Contents

To document algorithms developed during this semester, a software manual has been created which can be accessed here. The software manual is comprised of short descriptions of the algorithms implemented in this course. For each piece of code developed, there is an entry in the software manual.

Figure 2: Table of contents link on home page

Software manual entries were created for both single and double precision machine epsilon as seen in figure 3.



MATH 4610

Software Manual Table of Contents

The following list includes links to extensive documentation for all code written for tasksheets. Click here to go back to the home page.

- **Entry 1.** Single Precision Machine Epsilon:

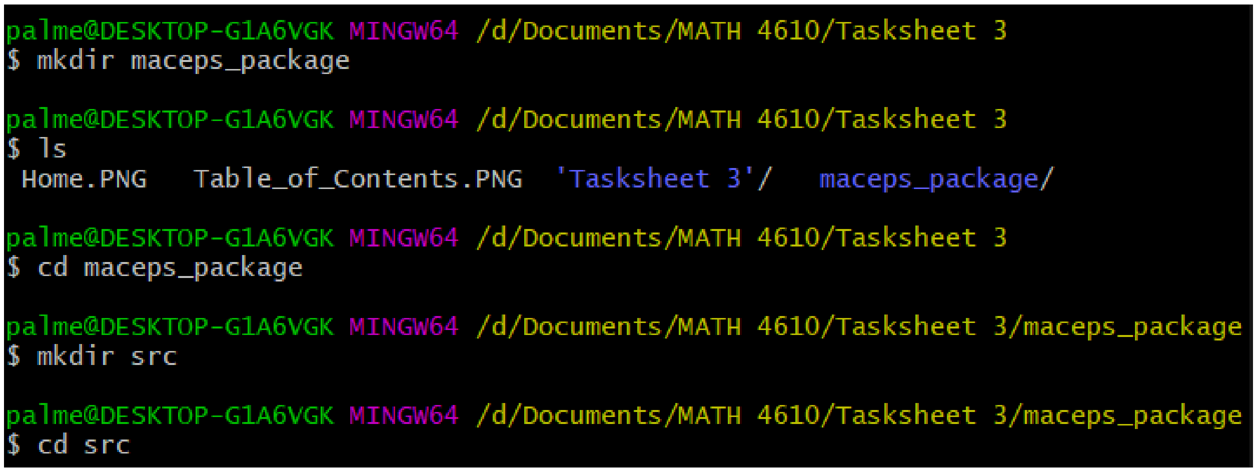
```
This routine will compute the single precision value for the machine epsilon or the number of
digits in the representation of real numbers in single precision. This is a routine for analyzing
the behavior of any computer. This usually will need to be run one time for each computer...
```

- **Entry 2.** Double Precision Machine Epsilon:

```
This routine will compute the double precision value for the machine epsilon or the number of
digits in the representation of real numbers in single precision. This is a routine for analyzing
the behavior of any computer. Thisusually will need to be run one time for each computer...
```

Figure 3: Table of contents

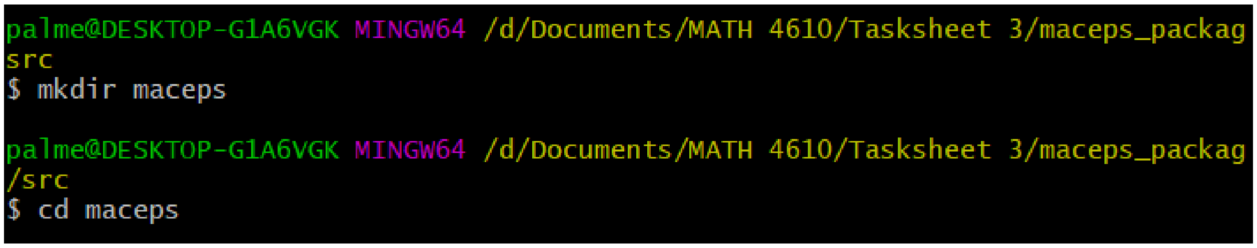
Task 5 Since I've been using Python, I will create a python package which is analogous to creating a shared library. I'll be referencing [3] to create this. First, we'll create a package directory called maceps_package. Within the package directory, we'll create a src directory. Both of these steps are seen in figure 4.



```
palme@DESKTOP-G1A6VGK MINGW64 /d/Documents/MATH 4610/Tasksheet 3
$ mkdir maceps_package

palme@DESKTOP-G1A6VGK MINGW64 /d/Documents/MATH 4610/Tasksheet 3
$ ls
 Home.PNG   Table_of_Contents.PNG  'Tasksheet 3'/   maceps_package/

palme@DESKTOP-G1A6VGK MINGW64 /d/Documents/MATH 4610/Tasksheet 3
$ cd maceps_package

palme@DESKTOP-G1A6VGK MINGW64 /d/Documents/MATH 4610/Tasksheet 3/maceps_package
$ mkdir src

palme@DESKTOP-G1A6VGK MINGW64 /d/Documents/MATH 4610/Tasksheet 3/maceps_package
$ cd src
```

Figure 4: File structure

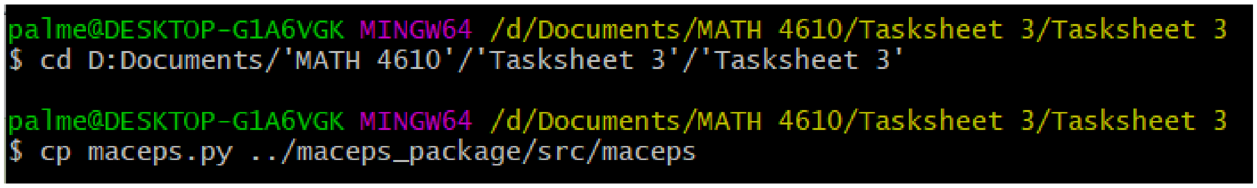
Next, we'll create our package directory maceps inside of our src directory and cd into maceps as in figure 5.



```
palme@DESKTOP-G1A6VGK MINGW64 /d/Documents/MATH 4610/Tasksheet 3/maceps_packag
src
$ mkdir maceps

palme@DESKTOP-G1A6VGK MINGW64 /d/Documents/MATH 4610/Tasksheet 3/maceps_packag
/src
$ cd maceps
```

Figure 5: Package directory

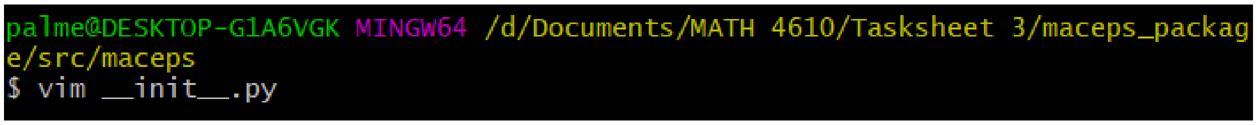
Once there, we'll copy our python file into our package directory as in figure 6.



```
palme@DESKTOP-G1A6VGK MINGW64 /d/Documents/MATH 4610/Tasksheet 3/Tasksheet 3
$ cd D:Documents/'MATH 4610'/'Tasksheet 3'/'Tasksheet 3'

palme@DESKTOP-G1A6VGK MINGW64 /d/Documents/MATH 4610/Tasksheet 3/Tasksheet 3
$ cp maceps.py ../maceps_package/src/maceps
```

Figure 6: Move python file

Within our maceps directory, we'll also create an empty __init__.py file as in figure 7.



```
palme@DESKTOP-G1A6VGK MINGW64 /d/Documents/MATH 4610/Tasksheet 3/maceps_packag
e/src/maceps
$ vim __init__.py
```

Figure 7: init file

We now have the basic file structure needed to create our python package and can start creating the necessary package files in our maceps_package directory.

We'll create an empty tests directory and a pyproject.toml file. The pyproject file contains the following specifications:

```
[build-system]
requires = [
    "setuptools>=42",
    "wheel"
]
build-backend = "setuptools.build_meta"
```

Now, we need to configure our metadata. We do that by creating a static metadata file setup.cfg. The setup.cfg file contains the following specifications:

```
[metadata]
name = maceps_package-palmeredholm
version = 0.0.1
author = Palmer Edholm
author_email = palmer.edholm@usu.edu
description = A machine epsilon package
long_description = file: README.md
long_description_content_type = text/markdown
url = https://github.com/palmeredholm/math4610
project_urls =
    Bug Tracker = https://github.com/palmeredholm/math4610
classifiers =
    Programming Language :: Python :: 3
    License :: OSI Approved :: MIT License
    Operating System :: OS Independent

[options]
package_dir =
    = src
packages = find:
python_requires = >= 3.6
[options.packages.find]
where = src
```

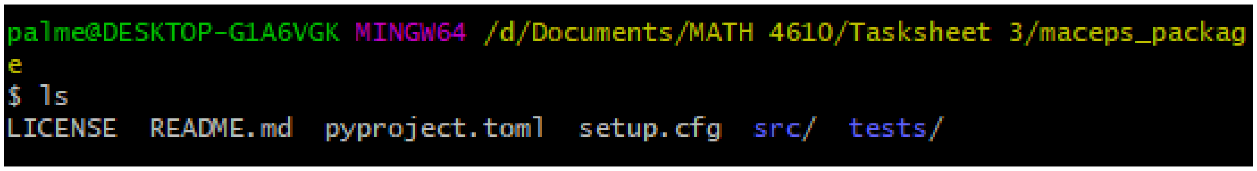
We specified that our long description would be specified in a README.md file. Therefore, our next step is to create said README.md file. The markdown file, for our purposes, will contain a succinct description. The following is what I included:

```
# Machine Epsilon Package

This package computes single and double precision value for the
machine epsilon.
```

Our last step is to create a license. We'll use the MIT license as it fits our purview best. The following is the MIT License that I included:

```
Copyright (c) 2018 The Python Packaging Authority

Permission is hereby granted, free of charge, to any person obtaining a copy
of this software and associated documentation files (the "Software"), to deal
in the Software without restriction, including without limitation the rights
to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
copies of the Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all
copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
SOFTWARE.
```

With a simple ls command, as in figure 8, I can see that I have everything I need to create this package.



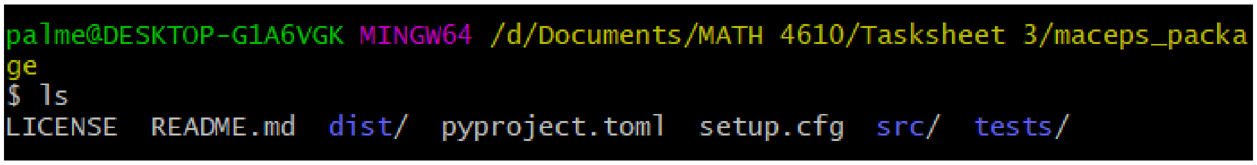
```
palme@DESKTOP-G1A6VGK MINGW64 /d/Documents/MATH 4610/Tasksheet 3/maceps_packag
e
$ ls
LICENSE  README.md  pyproject.toml  setup.cfg  src/  tests/
```

Figure 8: Package structure

All I need to do is run

```
py -m build
```

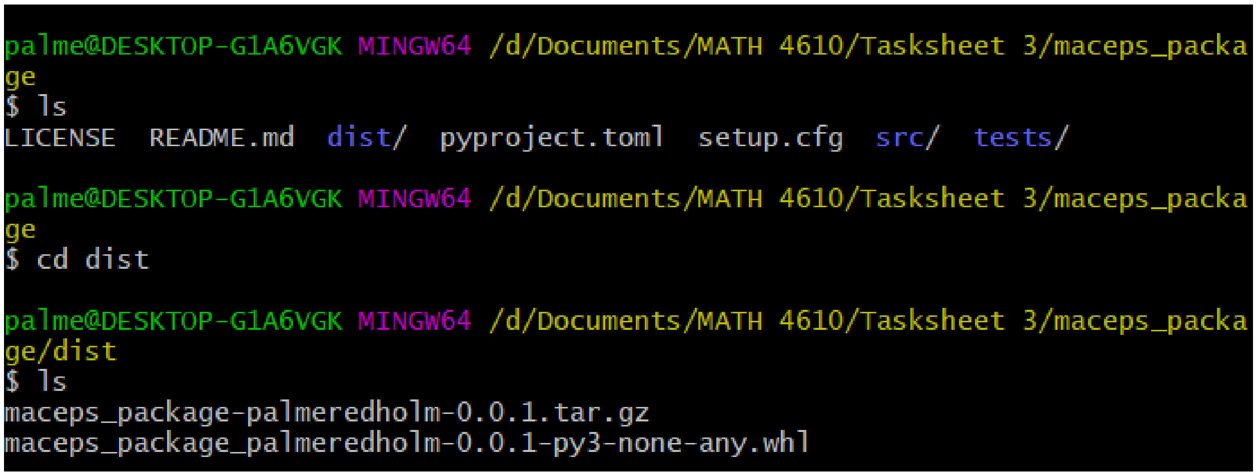
in my package directory and my python package will be created. After I run that command, I can run ls again (figure 9) and see that I now have a directory called dist.



```
palme@DESKTOP-G1A6VGK MINGW64 /d/Documents/MATH 4610/Tasksheet 3/maceps_packa
ge
$ ls
LICENSE  README.md  dist/  pyproject.toml  setup.cfg  src/  tests/
```

Figure 9: Directory after build

If I step into that directory (figure 10) I can see that it contains a tar.gz file which is a source archive and a .whl file which is a built distribution.



```
palme@DESKTOP-G1A6VGK MINGW64 /d/Documents/MATH 4610/Tasksheet 3/maceps_packa
ge
$ ls
LICENSE  README.md  dist/  pyproject.toml  setup.cfg  src/  tests/

palme@DESKTOP-G1A6VGK MINGW64 /d/Documents/MATH 4610/Tasksheet 3/maceps_packa
ge
$ cd dist

palme@DESKTOP-G1A6VGK MINGW64 /d/Documents/MATH 4610/Tasksheet 3/maceps_packa
ge/dist
$ ls
maceps_package-palmeredholm-0.0.1.tar.gz
maceps_package_palmeredholm-0.0.1-py3-none-any.whl
```

Figure 10: Archived file

Task 6 From [2] and [3], I learned a fair amount about the advantages and disadvantages of shared libraries. First of all, a shared library is a dynamic library that uses dynamic linking when compiling. The opposite of a dynamic library is a static library. While shared libraries are slower than static libraries, a single copy of the shared library is kept in RAM reducing the size of the executable.

References

[1] https://nickolasteixeira.medium.com/shared-dynamic-libraries-vs-static-libraries-differences-in-performance-2716f5b3c826

[2] https://nickolasteixeira.medium.com/what-are-static-libraries-in-c-and-why-do-software-engineers-use-them-eb7022d89135

[3] https://packaging.python.org/tutorials/packaging-projects/