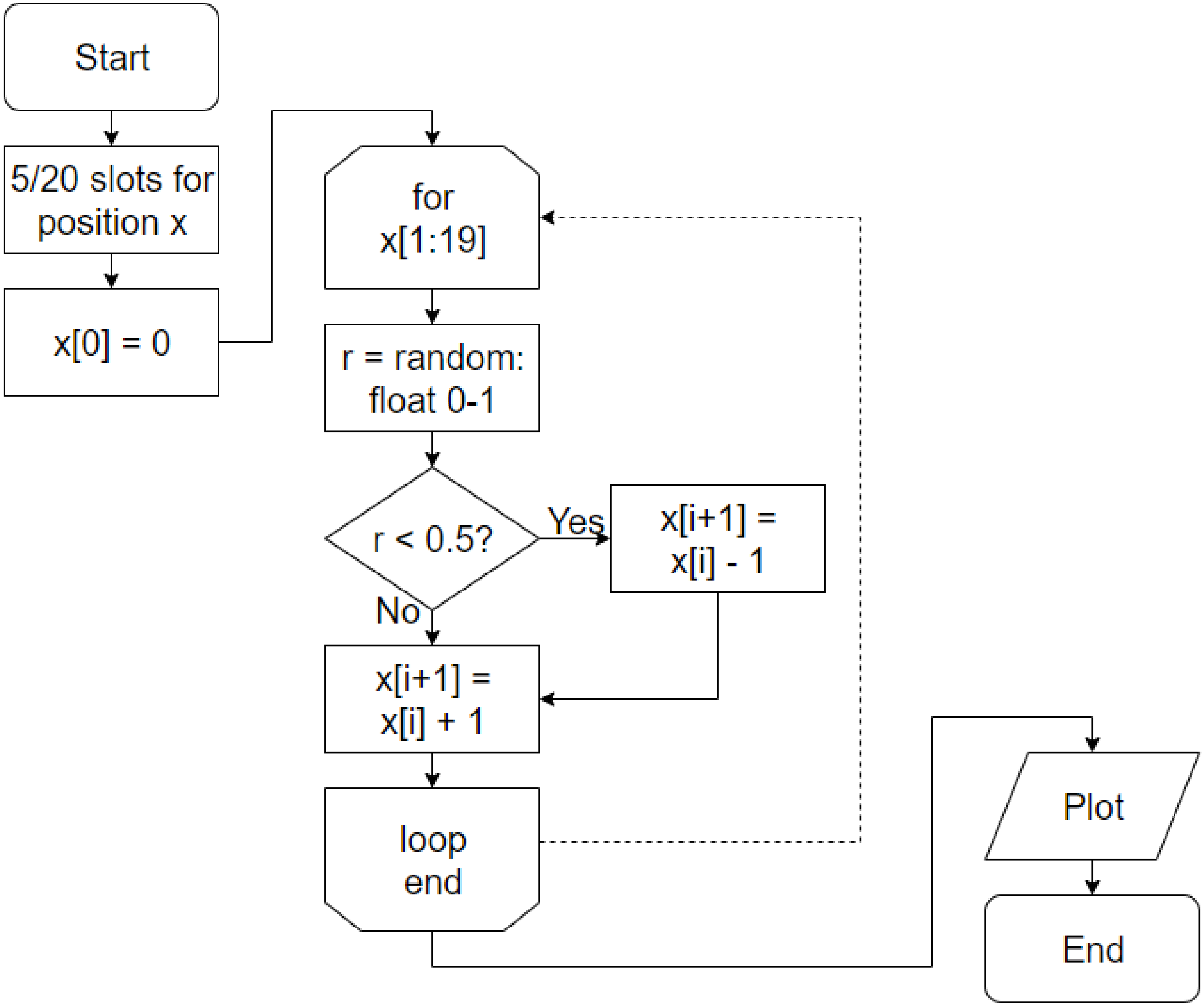
Coding seminar

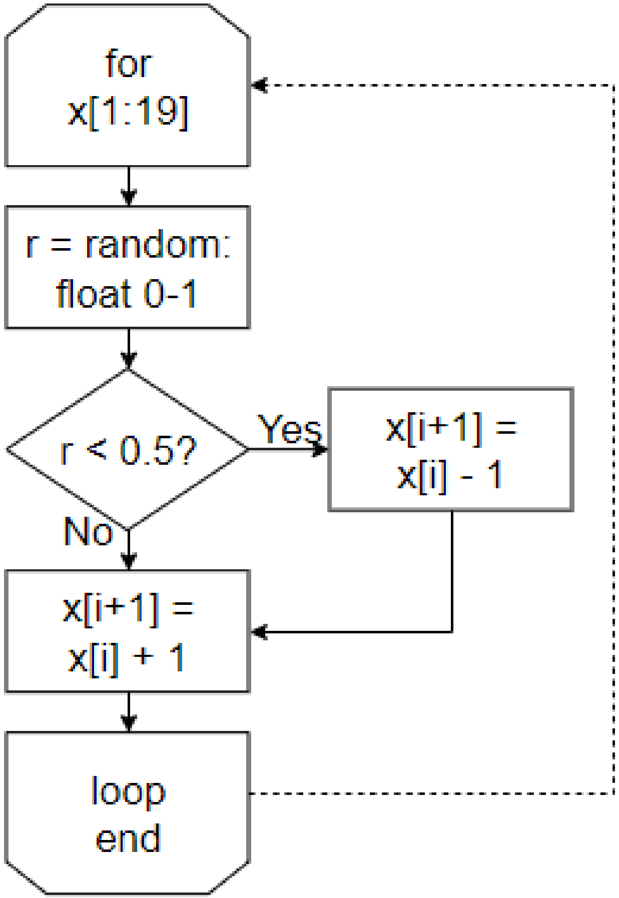
Lesson 3: Handling real-world data

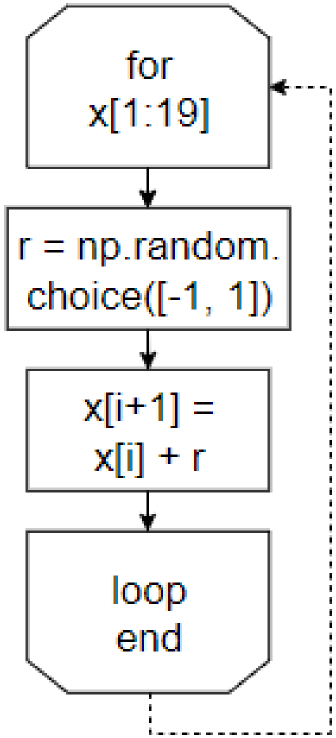
Ikue Hirata, PhD

Exercise model answer: random walk

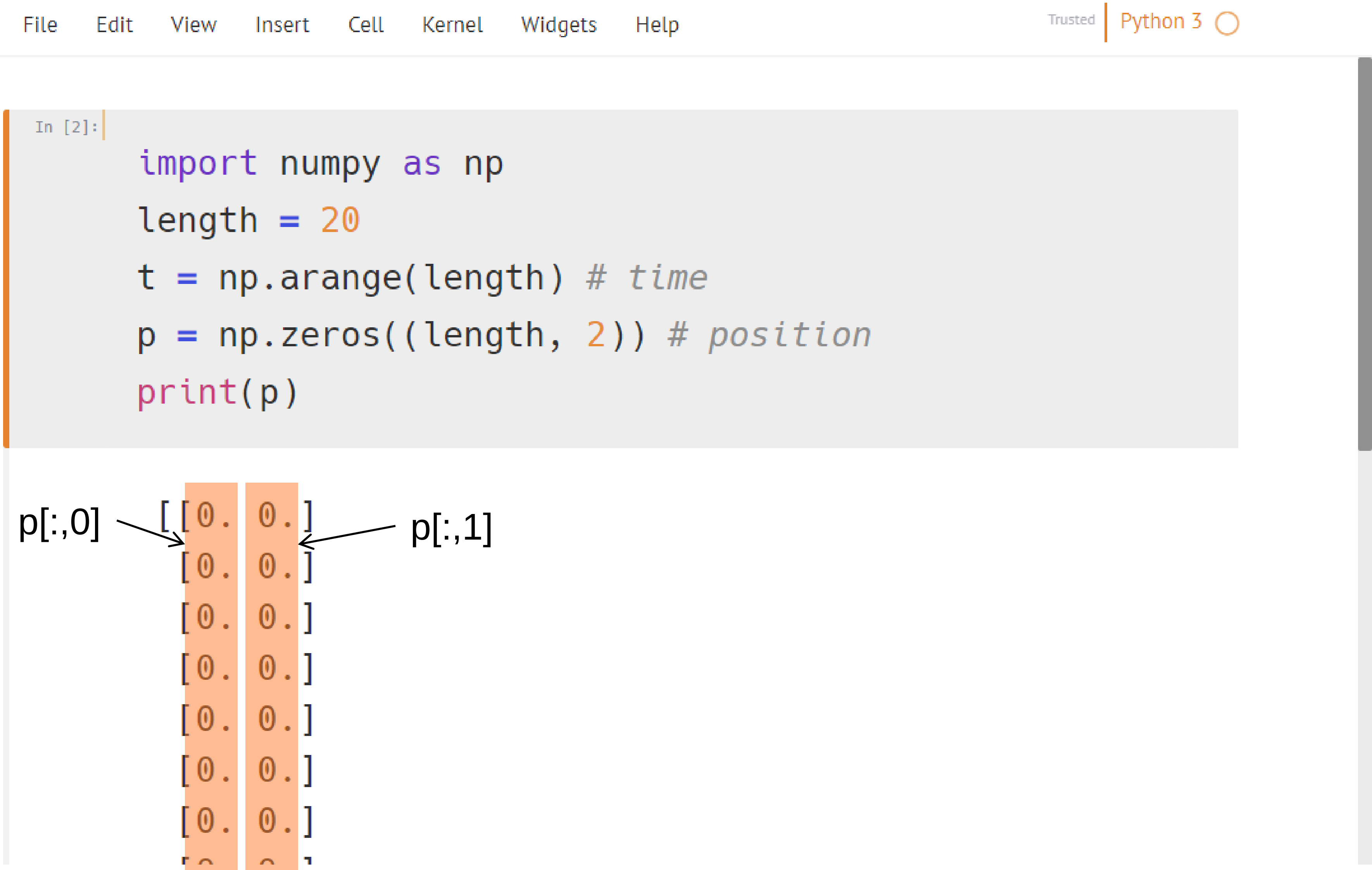


Make it simple





2-D random walk



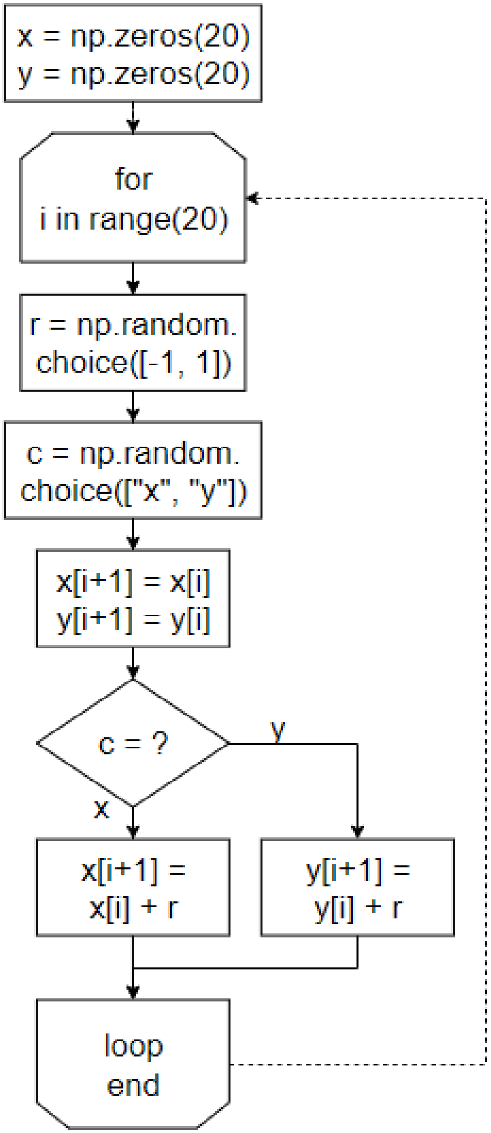
```python
import numpy as np
length = 20
t = np.arange(length) # time
p = np.zeros((length, 2)) # position
print(p)
```

```
[[0. 0.]
 [0. 0.]
 [0. 0.]
 [0. 0.]
 [0. 0.]
 [0. 0.]
 [0. 0.]
```

p[:,0]

p[:,1]

2-D random walk



x = np.zeros(20)
y = np.zeros(20)

for
i in range(20)

r = np.random.
choice([-1, 1])

c = np.random.
choice(["x", "y"])

x[i+1] = x[i]
y[i+1] = y[i]

c = ?

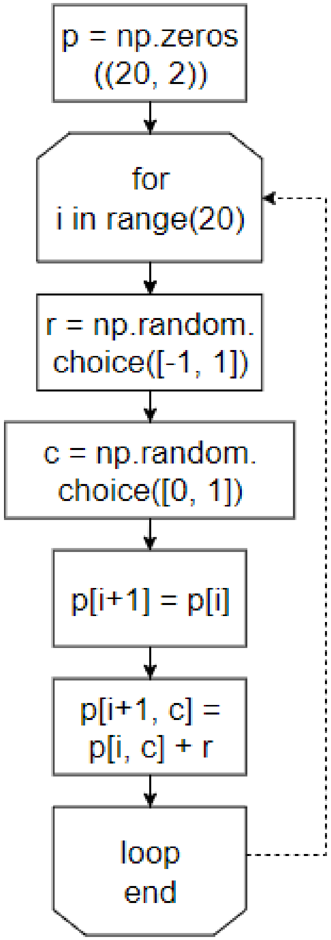
y

x

x[i+1] =
x[i] + r

y[i+1] =
y[i] + r

loop
end



p = np.zeros
((20, 2))

for
i in range(20)

r = np.random.
choice([-1, 1])

c = np.random.
choice([0, 1])

p[i+1] = p[i]

p[i+1, c] =
p[i, c] + r

loop
end

Contents

Modules: More about Numpy

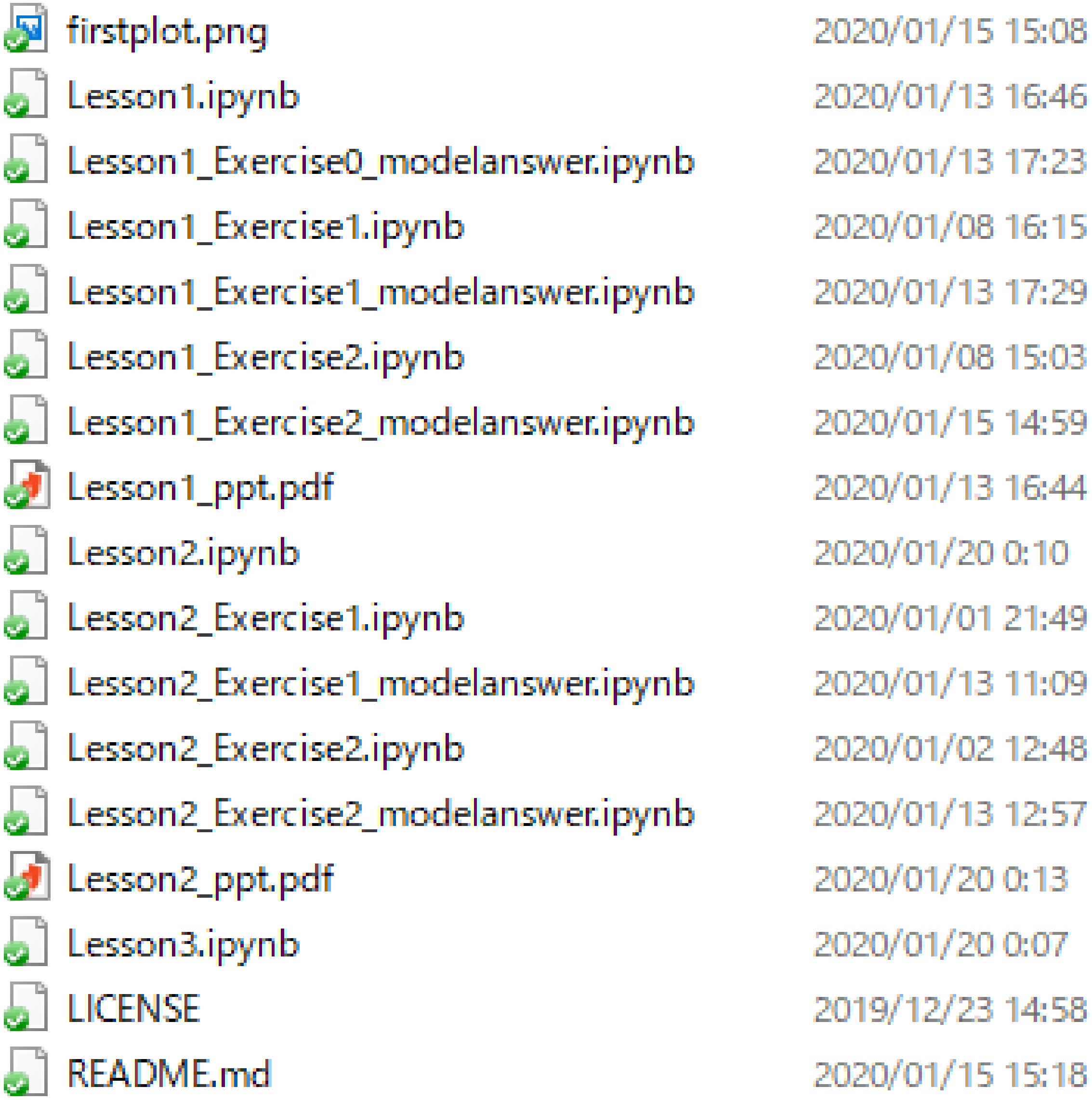
Plotting: More about Matplotlib

Editors

Debugger/debugging

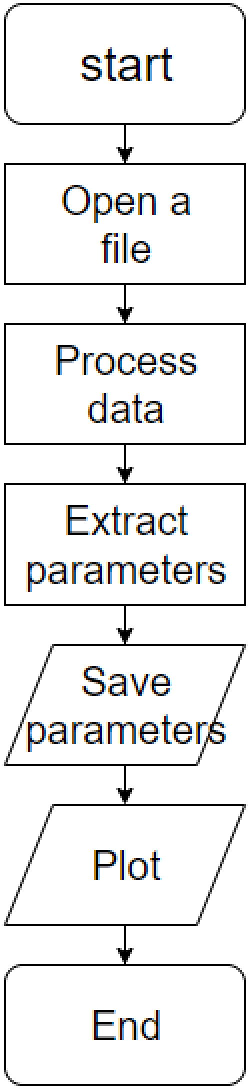
UNIX-based commands for mass-process

Extensions ready?



firstplot.png	2020/01/15 15:08
Lesson1.ipynb	2020/01/13 16:46
Lesson1_Exercise0_modelanswer.ipynb	2020/01/13 17:23
Lesson1_Exercise1.ipynb	2020/01/08 16:15
Lesson1_Exercise1_modelanswer.ipynb	2020/01/13 17:29
Lesson1_Exercise2.ipynb	2020/01/08 15:03
Lesson1_Exercise2_modelanswer.ipynb	2020/01/15 14:59
Lesson1_ppt.pdf	2020/01/13 16:44
Lesson2.ipynb	2020/01/20 0:10
Lesson2_Exercise1.ipynb	2020/01/01 21:49
Lesson2_Exercise1_modelanswer.ipynb	2020/01/13 11:09
Lesson2_Exercise2.ipynb	2020/01/02 12:48
Lesson2_Exercise2_modelanswer.ipynb	2020/01/13 12:57
Lesson2_ppt.pdf	2020/01/20 0:13
Lesson3.ipynb	2020/01/20 0:07
LICENSE	2019/12/23 14:58
README.md	2020/01/15 15:18

Data analysis process



start

Open a file

Process data

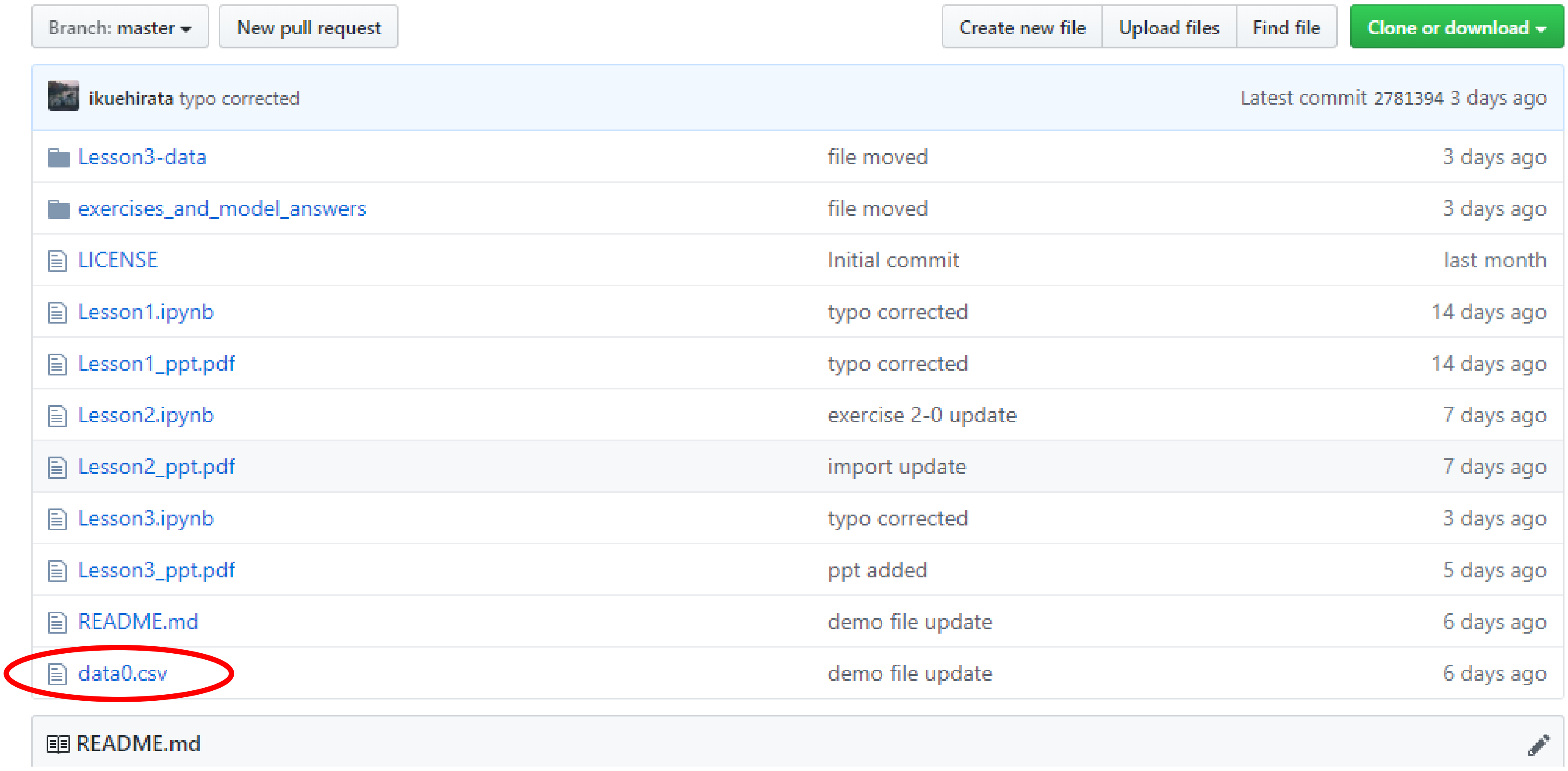
Extract parameters

Save parameters

Plot

End

Download demo files

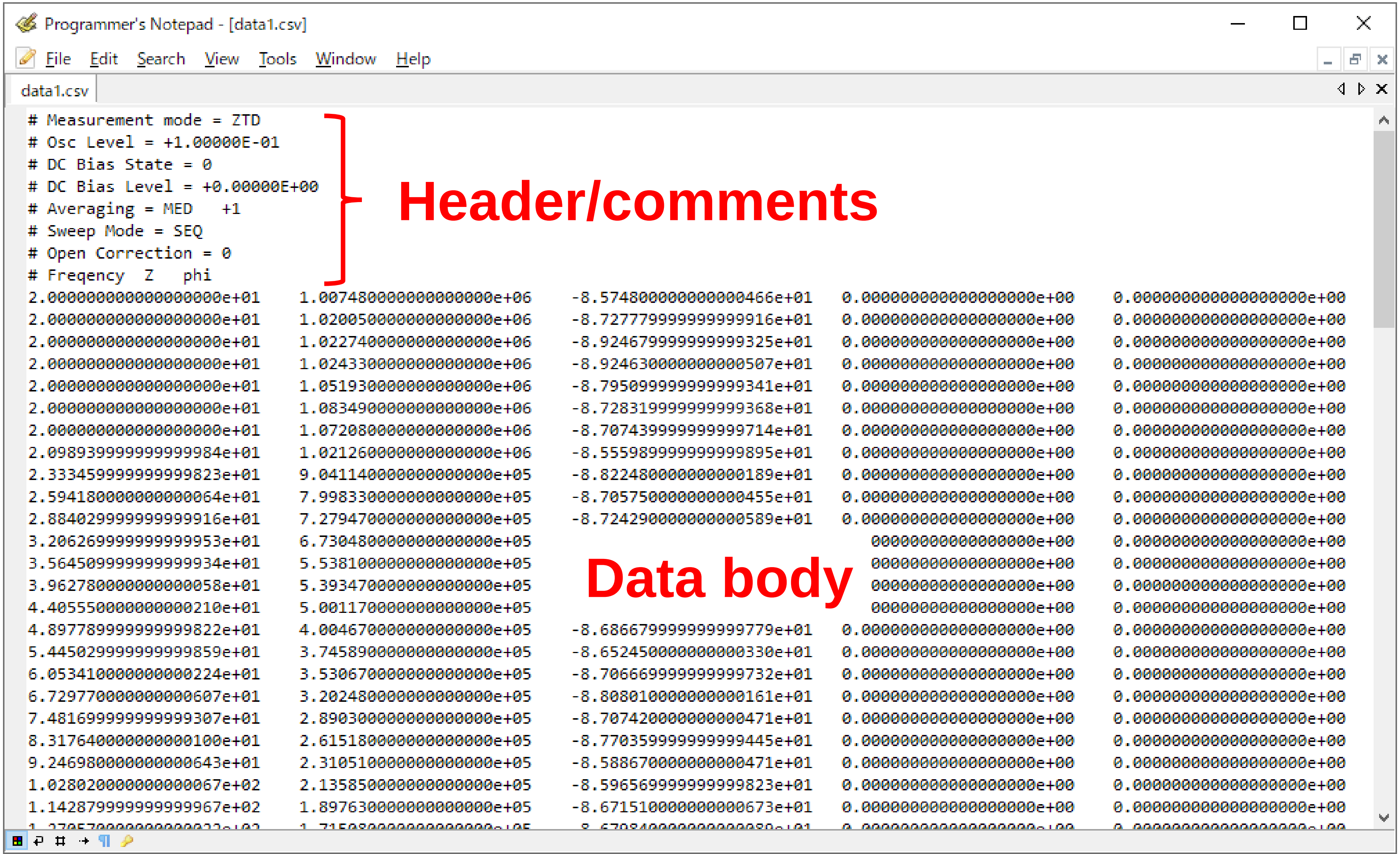


ikuehirata typo corrected Latest commit 2781394 3 days ago

Lesson3-data	file moved	3 days ago
exercises_and_model_answers	file moved	3 days ago
LICENSE	Initial commit	last month
Lesson1.ipynb	typo corrected	14 days ago
Lesson1_ppt.pdf	typo corrected	14 days ago
Lesson2.ipynb	exercise 2-0 update	7 days ago
Lesson2_ppt.pdf	import update	7 days ago
Lesson3.ipynb	typo corrected	3 days ago
Lesson3_ppt.pdf	ppt added	5 days ago
README.md	demo file update	6 days ago
data0.csv	demo file update	6 days ago

README.md

Sample data



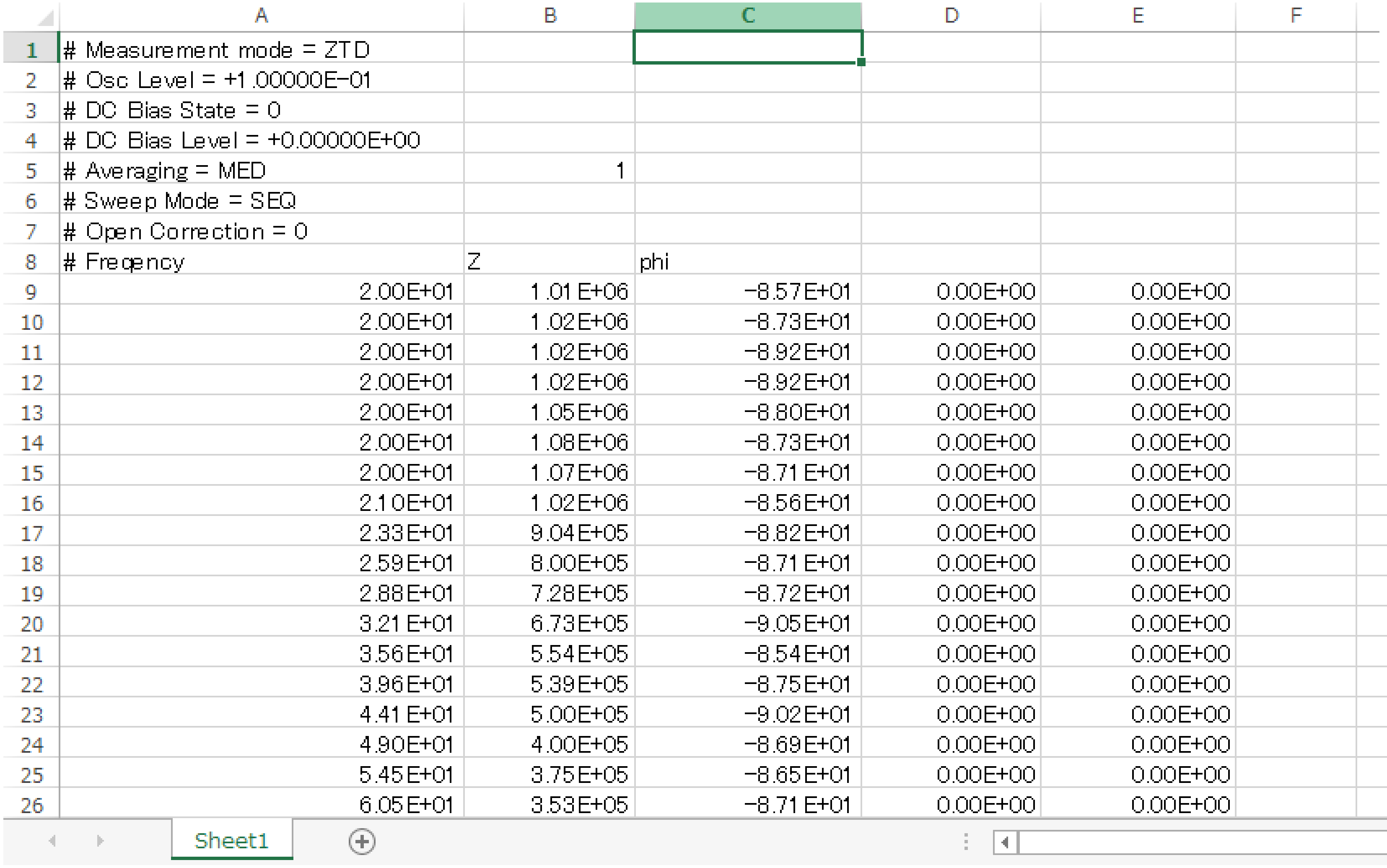
```
# Measurement mode = ZTD
# Osc Level = +1.00000E-01
# DC Bias State = 0
# DC Bias Level = +0.00000E+00
# Averaging = MED    +1
# Sweep Mode = SEQ
# Open Correction = 0
# Freqency  Z   phi
2.000000000000000000e+01    1.007480000000000000e+06    -8.574800000000000466e+01   0.000000000000000000e+00    0.000000000000000000e+00
2.000000000000000000e+01    1.020050000000000000e+06    -8.727779999999999916e+01   0.000000000000000000e+00    0.000000000000000000e+00
2.000000000000000000e+01    1.022740000000000000e+06    -8.924679999999999325e+01   0.000000000000000000e+00    0.000000000000000000e+00
2.000000000000000000e+01    1.024330000000000000e+06    -8.924630000000000507e+01   0.000000000000000000e+00    0.000000000000000000e+00
2.000000000000000000e+01    1.051930000000000000e+06    -8.795099999999999341e+01   0.000000000000000000e+00    0.000000000000000000e+00
2.000000000000000000e+01    1.083490000000000000e+06    -8.728319999999999368e+01   0.000000000000000000e+00    0.000000000000000000e+00
2.000000000000000000e+01    1.072080000000000000e+06    -8.707439999999999714e+01   0.000000000000000000e+00    0.000000000000000000e+00
2.098939999999999984e+01    1.021260000000000000e+06    -8.555989999999999895e+01   0.000000000000000000e+00    0.000000000000000000e+00
2.333459999999999823e+01    9.041140000000000000e+05    -8.822480000000000189e+01   0.000000000000000000e+00    0.000000000000000000e+00
2.594180000000000064e+01    7.998330000000000000e+05    -8.705750000000000455e+01   0.000000000000000000e+00    0.000000000000000000e+00
2.884029999999999916e+01    7.279470000000000000e+05    -8.724290000000000589e+01   0.000000000000000000e+00    0.000000000000000000e+00
3.206269999999999953e+01    6.730480000000000000e+05                                   00000000000000000e+00    0.000000000000000000e+00
3.564509999999999934e+01    5.538100000000000000e+05                                   00000000000000000e+00    0.000000000000000000e+00
3.962780000000000058e+01    5.393470000000000000e+05                                   00000000000000000e+00    0.000000000000000000e+00
4.405550000000000210e+01    5.001170000000000000e+05                                   00000000000000000e+00    0.000000000000000000e+00
4.897789999999999822e+01    4.004670000000000000e+05    -8.686679999999999779e+01   0.000000000000000000e+00    0.000000000000000000e+00
5.445029999999999859e+01    3.745890000000000000e+05    -8.652450000000000330e+01   0.000000000000000000e+00    0.000000000000000000e+00
6.053410000000000224e+01    3.530670000000000000e+05    -8.706669999999999732e+01   0.000000000000000000e+00    0.000000000000000000e+00
6.729770000000000607e+01    3.202480000000000000e+05    -8.808010000000000161e+01   0.000000000000000000e+00    0.000000000000000000e+00
7.481699999999999307e+01    2.890300000000000000e+05    -8.707420000000000471e+01   0.000000000000000000e+00    0.000000000000000000e+00
8.317640000000000100e+01    2.615180000000000000e+05    -8.770359999999999445e+01   0.000000000000000000e+00    0.000000000000000000e+00
9.246980000000000643e+01    2.310510000000000000e+05    -8.588670000000000471e+01   0.000000000000000000e+00    0.000000000000000000e+00
1.028020000000000067e+02    2.135850000000000000e+05    -8.596569999999999823e+01   0.000000000000000000e+00    0.000000000000000000e+00
1.142879999999999967e+02    1.897630000000000000e+05    -8.671510000000000673e+01   0.000000000000000000e+00    0.000000000000000000e+00
1.270570000000000022e+02    1.715080000000000000e+05    -8.670840000000000080e+01   0.000000000000000000e+00    0.000000000000000000e+00
```

Header/comments

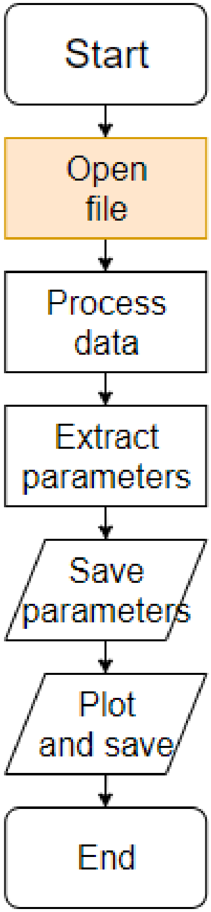
Data body

Array data – csv/tsv

comma/tab separated values



	A	B	C	D	E	F
1	# Measurement mode = ZTD					
2	# Osc Level = +1.00000E-01					
3	# DC Bias State = 0					
4	# DC Bias Level = +0.00000E+00					
5	# Averaging = MED	1				
6	# Sweep Mode = SEQ					
7	# Open Correction = 0					
8	# Freqency	Z	phi			
9	2.00E+01	1.01E+06	-8.57E+01	0.00E+00	0.00E+00	
10	2.00E+01	1.02E+06	-8.73E+01	0.00E+00	0.00E+00	
11	2.00E+01	1.02E+06	-8.92E+01	0.00E+00	0.00E+00	
12	2.00E+01	1.02E+06	-8.92E+01	0.00E+00	0.00E+00	
13	2.00E+01	1.05E+06	-8.80E+01	0.00E+00	0.00E+00	
14	2.00E+01	1.08E+06	-8.73E+01	0.00E+00	0.00E+00	
15	2.00E+01	1.07E+06	-8.71E+01	0.00E+00	0.00E+00	
16	2.10E+01	1.02E+06	-8.56E+01	0.00E+00	0.00E+00	
17	2.33E+01	9.04E+05	-8.82E+01	0.00E+00	0.00E+00	
18	2.59E+01	8.00E+05	-8.71E+01	0.00E+00	0.00E+00	
19	2.88E+01	7.28E+05	-8.72E+01	0.00E+00	0.00E+00	
20	3.21E+01	6.73E+05	-9.05E+01	0.00E+00	0.00E+00	
21	3.56E+01	5.54E+05	-8.54E+01	0.00E+00	0.00E+00	
22	3.96E+01	5.39E+05	-8.75E+01	0.00E+00	0.00E+00	
23	4.41E+01	5.00E+05	-9.02E+01	0.00E+00	0.00E+00	
24	4.90E+01	4.00E+05	-8.69E+01	0.00E+00	0.00E+00	
25	5.45E+01	3.75E+05	-8.65E+01	0.00E+00	0.00E+00	
26	6.05E+01	3.53E+05	-8.71E+01	0.00E+00	0.00E+00	



Start

Open file

Process data

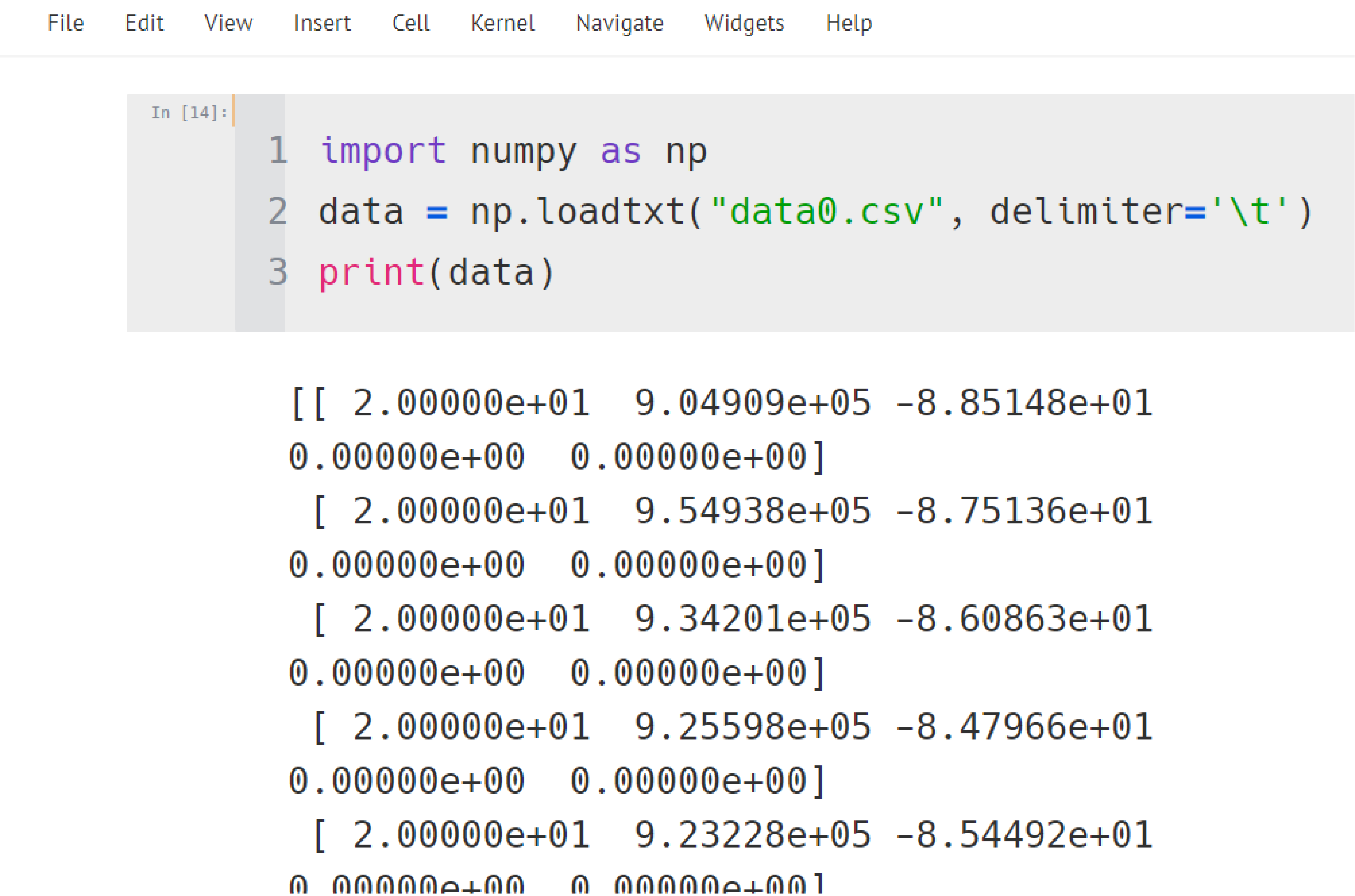
Extract parameters

Save parameters

Plot and save

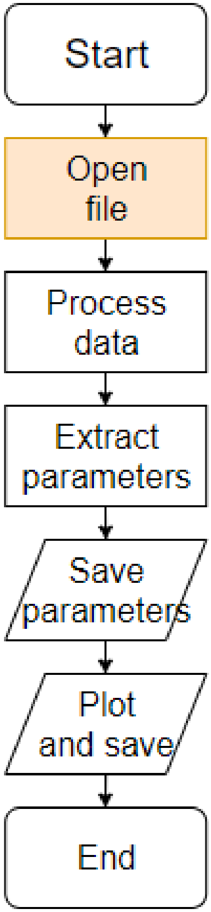
End

Read data into array

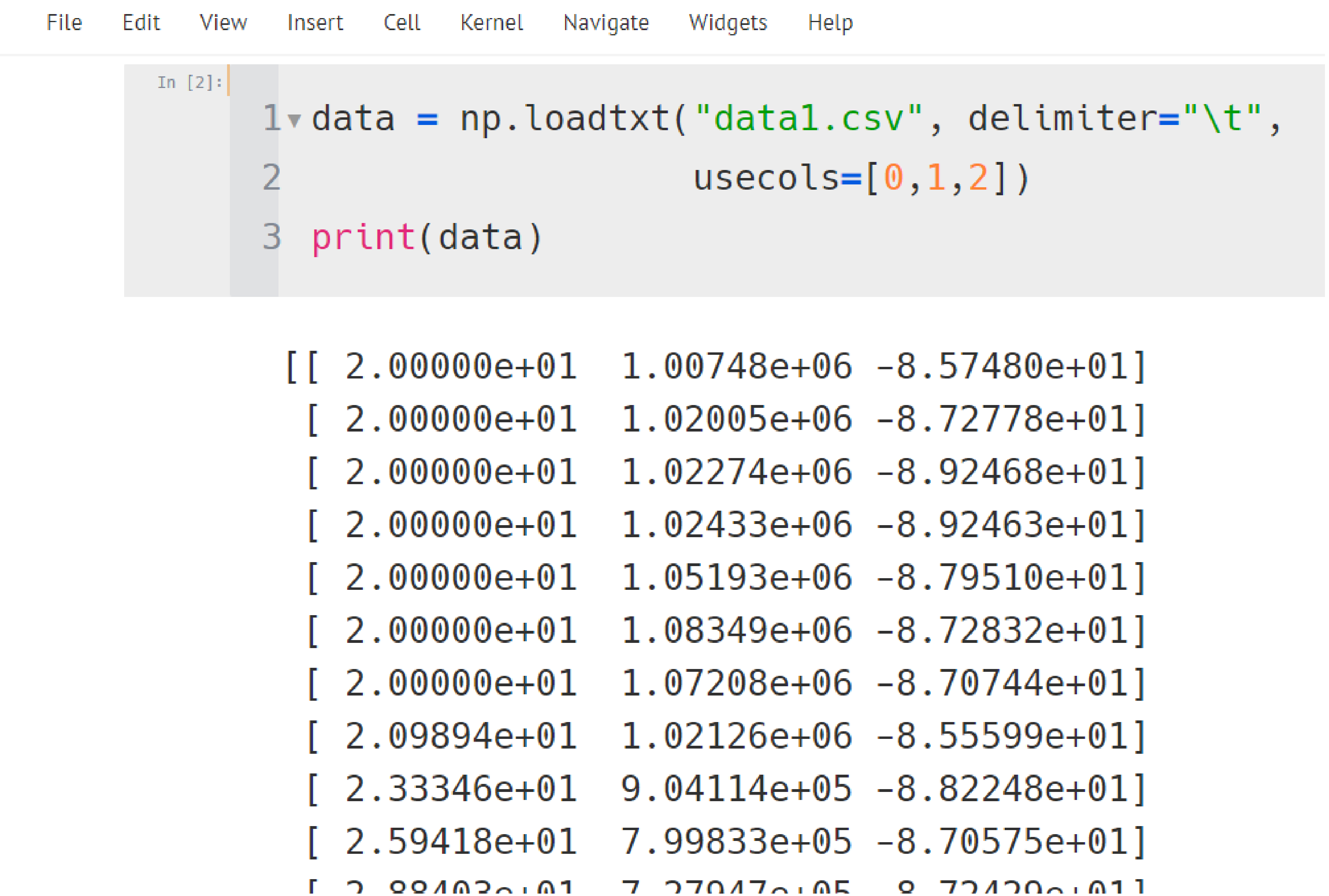


```python
import numpy as np
data = np.loadtxt("data0.csv", delimiter='\t')
print(data)
```

```
[[ 2.00000e+01  9.04909e+05 -8.85148e+01
0.00000e+00  0.00000e+00]
 [ 2.00000e+01  9.54938e+05 -8.75136e+01
0.00000e+00  0.00000e+00]
 [ 2.00000e+01  9.34201e+05 -8.60863e+01
0.00000e+00  0.00000e+00]
 [ 2.00000e+01  9.25598e+05 -8.47966e+01
0.00000e+00  0.00000e+00]
 [ 2.00000e+01  9.23228e+05 -8.54492e+01
0.00000e+00  0.00000e+00]
```

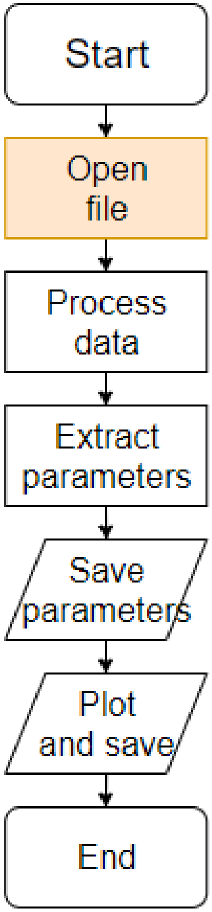


Read data into array

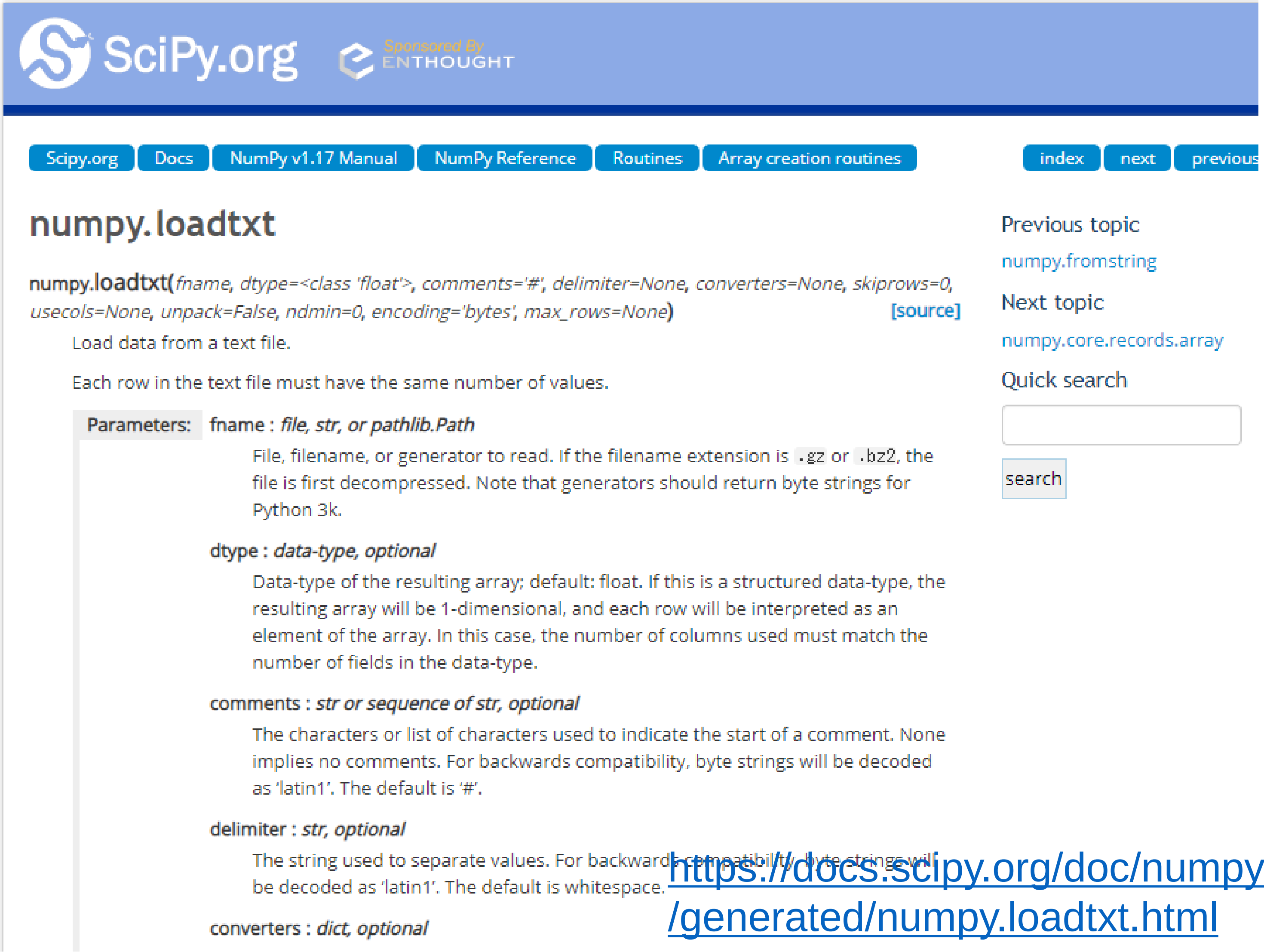


```
data = np.loadtxt("data1.csv", delimiter="\t",
                          usecols=[0,1,2])
print(data)
```

```
[[ 2.00000e+01  1.00748e+06 -8.57480e+01]
 [ 2.00000e+01  1.02005e+06 -8.72778e+01]
 [ 2.00000e+01  1.02274e+06 -8.92468e+01]
 [ 2.00000e+01  1.02433e+06 -8.92463e+01]
 [ 2.00000e+01  1.05193e+06 -8.79510e+01]
 [ 2.00000e+01  1.08349e+06 -8.72832e+01]
 [ 2.00000e+01  1.07208e+06 -8.70744e+01]
 [ 2.09894e+01  1.02126e+06 -8.55599e+01]
 [ 2.33346e+01  9.04114e+05 -8.82248e+01]
 [ 2.59418e+01  7.99833e+05 -8.70575e+01]
 [ 2.88403e+01  7.27947e+05 -8.72429e+01]
```



For more options…



SciPy.org Sponsored By ENTHOUGHT

numpy.loadtxt

numpy.loadtxt(*fname, dtype=<class 'float'>, comments='#', delimiter=None, converters=None, skiprows=0, usecols=None, unpack=False, ndmin=0, encoding='bytes', max_rows=None*) [source]

Load data from a text file.

Each row in the text file must have the same number of values.

Parameters: fname : *file, str, or pathlib.Path*

File, filename, or generator to read. If the filename extension is `.gz` or `.bz2`, the file is first decompressed. Note that generators should return byte strings for Python 3k.

dtype : *data-type, optional*

Data-type of the resulting array; default: float. If this is a structured data-type, the resulting array will be 1-dimensional, and each row will be interpreted as an element of the array. In this case, the number of columns used must match the number of fields in the data-type.

comments : *str or sequence of str, optional*

The characters or list of characters used to indicate the start of a comment. None implies no comments. For backwards compatibility, byte strings will be decoded as 'latin1'. The default is '#'.

delimiter : *str, optional*

The string used to separate values. For backwards compatibility, byte strings will be decoded as 'latin1'. The default is whitespace.

converters : *dict, optional*

Previous topic

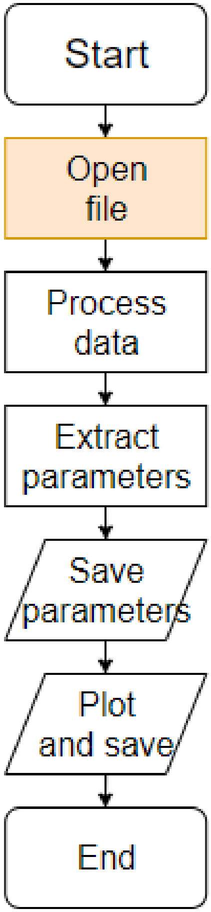
numpy.fromstring

Next topic

numpy.core.records.array

Quick search

https://docs.scipy.org/doc/numpy/reference/generated/numpy.loadtxt.html



Start

Open file

Process data

Extract parameters

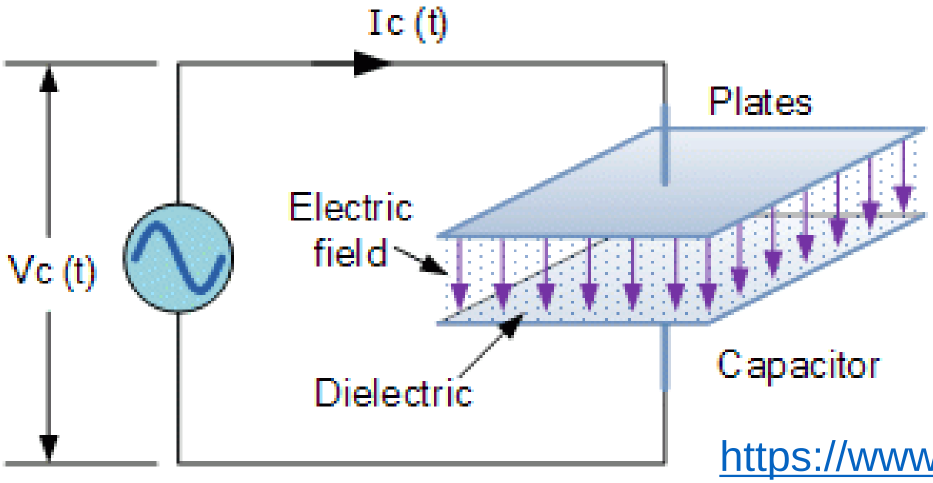
Save parameters

Plot and save

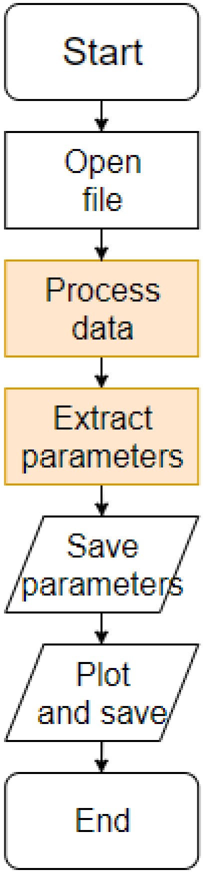
End

What is this?

Complex impedance of a capacitor



https://www.electronics-tutorials.ws/filter/filter_1.html



Start

Open file

Process data

Extract parameters

Save parameters

Plot and save

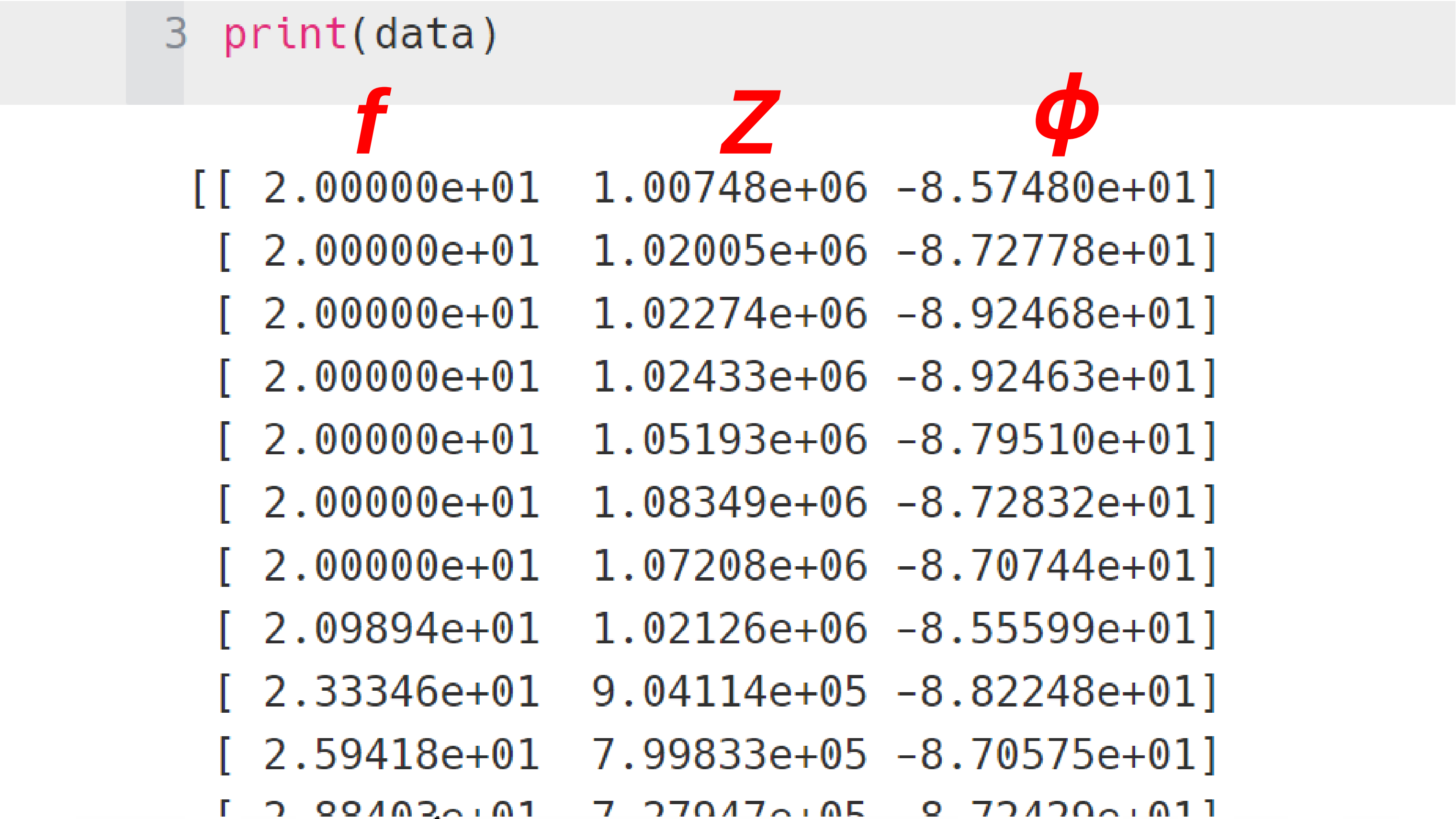
End

$$Z = \frac{1}{j\omega C}, \omega = 2\pi f, \phi = \arg Z$$

We want to get this value

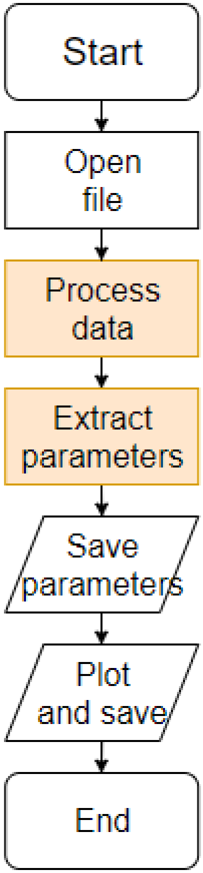
j as imaginary number

Which value is which?



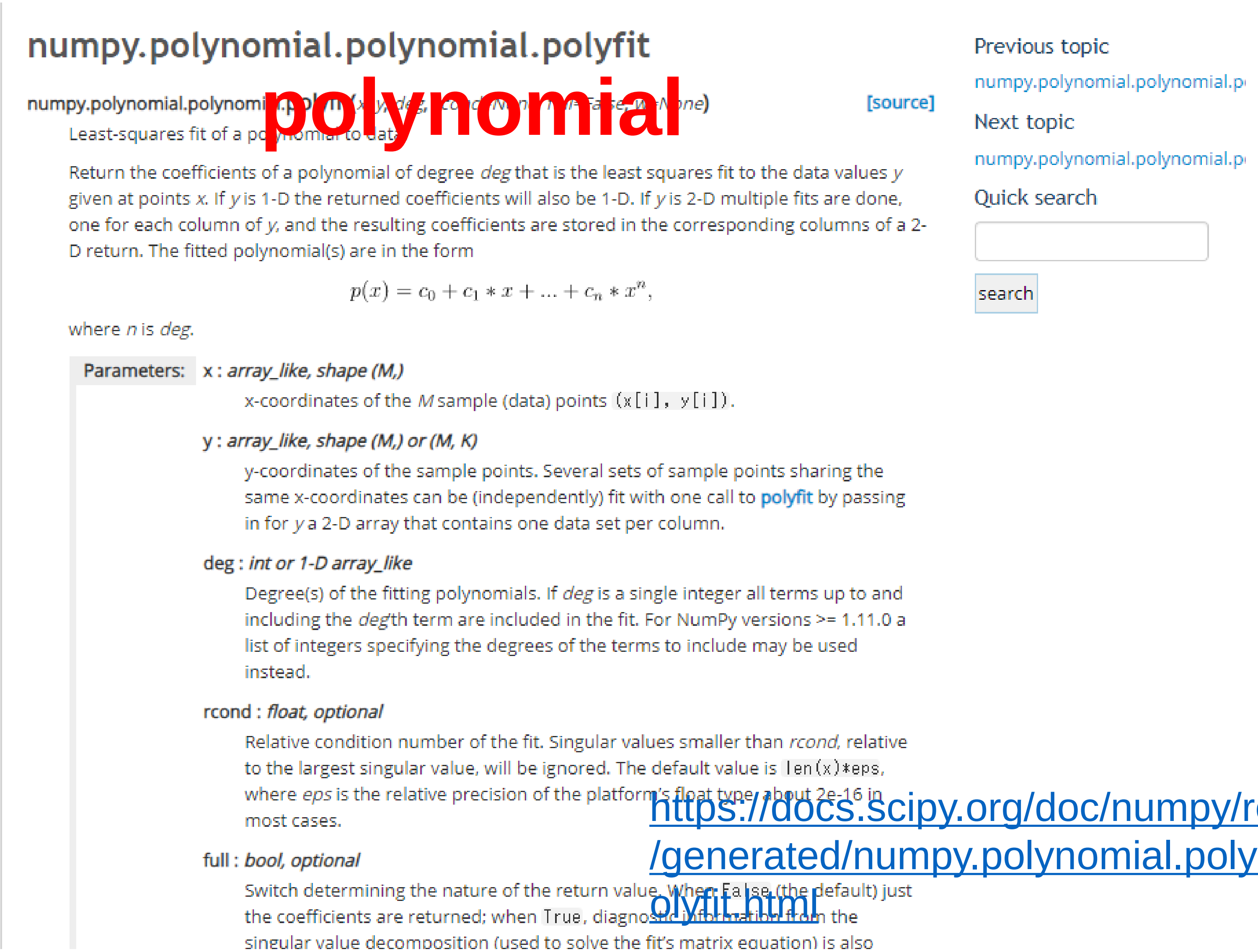
```
print(data)
```

```
[[ 2.00000e+01  1.00748e+06 -8.57480e+01]
 [ 2.00000e+01  1.02005e+06 -8.72778e+01]
 [ 2.00000e+01  1.02274e+06 -8.92468e+01]
 [ 2.00000e+01  1.02433e+06 -8.92463e+01]
 [ 2.00000e+01  1.05193e+06 -8.79510e+01]
 [ 2.00000e+01  1.08349e+06 -8.72832e+01]
 [ 2.00000e+01  1.07208e+06 -8.70744e+01]
 [ 2.09894e+01  1.02126e+06 -8.55599e+01]
 [ 2.33346e+01  9.04114e+05 -8.82248e+01]
 [ 2.59418e+01  7.99833e+05 -8.70575e+01]
 [ 2.88403e+01  7.27947e+05 -8.72439e+01]
```



$$Z = \frac{1}{j\omega C}, \omega = 2\pi f, \phi = \arg Z$$

Fitting function



numpy.polynomial.polynomial.polyfit

numpy.polynomial.polynomial.polyfit(*x, y, deg, rcond=None, full=False, w=None*) [source]

Least-squares fit of a polynomial to data.

Return the coefficients of a polynomial of degree *deg* that is the least squares fit to the data values *y* given at points *x*. If *y* is 1-D the returned coefficients will also be 1-D. If *y* is 2-D multiple fits are done, one for each column of *y*, and the resulting coefficients are stored in the corresponding columns of a 2-D return. The fitted polynomial(s) are in the form

$$p(x) = c_0 + c_1 * x + ... + c_n * x^n,$$

where *n* is *deg*.

Parameters:

x : *array_like, shape (M,)*

x-coordinates of the *M* sample (data) points `(x[i], y[i])`.

y : *array_like, shape (M,) or (M, K)*

y-coordinates of the sample points. Several sets of sample points sharing the same x-coordinates can be (independently) fit with one call to polyfit by passing in for *y* a 2-D array that contains one data set per column.

deg : *int or 1-D array_like*

Degree(s) of the fitting polynomials. If *deg* is a single integer all terms up to and including the *deg*th term are included in the fit. For NumPy versions >= 1.11.0 a list of integers specifying the degrees of the terms to include may be used instead.

rcond : *float, optional*

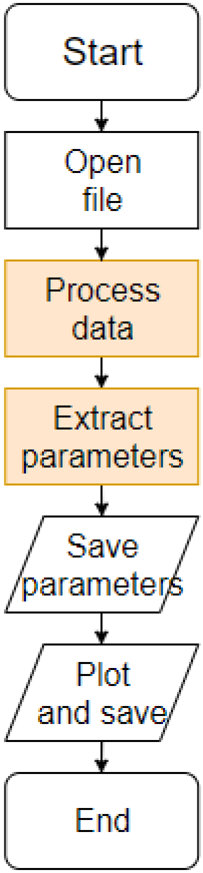
Relative condition number of the fit. Singular values smaller than *rcond*, relative to the largest singular value, will be ignored. The default value is `len(x)*eps`, where *eps* is the relative precision of the platform's float type, about 2e-16 in most cases.

full : *bool, optional*

Switch determining the nature of the return value. When `False` (the default) just the coefficients are returned; when `True`, diagnostic information from the singular value decomposition (used to solve the fit's matrix equation) is also

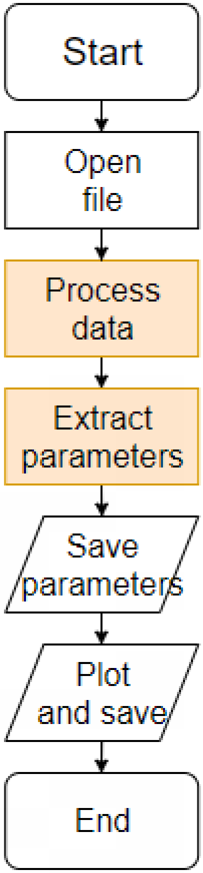
polynomial

https://docs.scipy.org/doc/numpy/reference/generated/numpy.polynomial.polynomial.polyfit.html



Small trick

$$\begin{aligned} Z &= \frac{1}{j \cdot 2\pi f C} \\ \log Z &= -\log(2\pi f C) \\ \log Z &= -\log(2\pi C) - \log f \end{aligned}$$



Start

Open file

Process data

Extract parameters

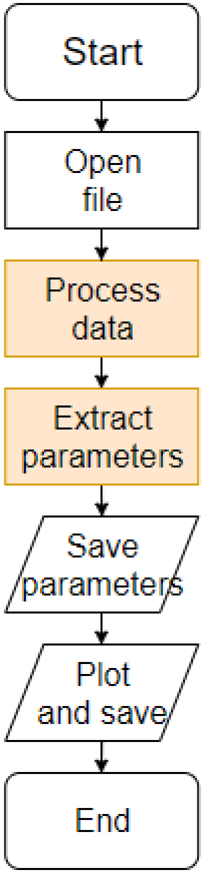
Save parameters

Plot and save

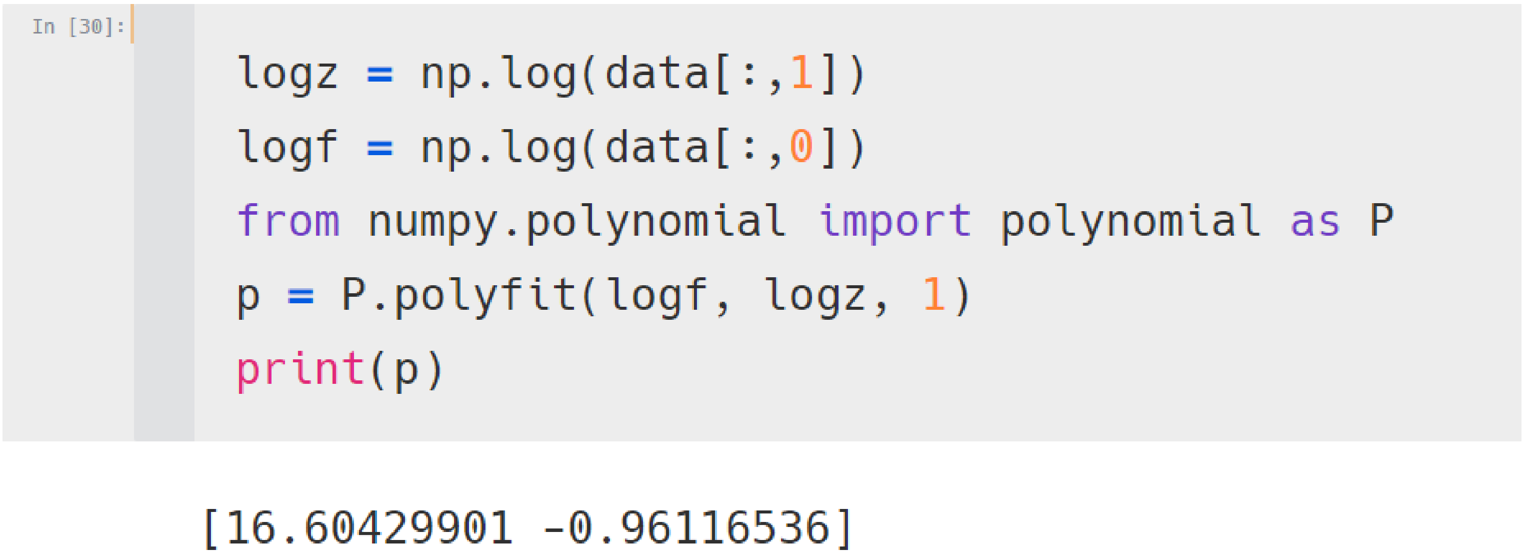
End

Fitting 1 – Exercise 0-1

$$y = -0.96x + 16.6$$



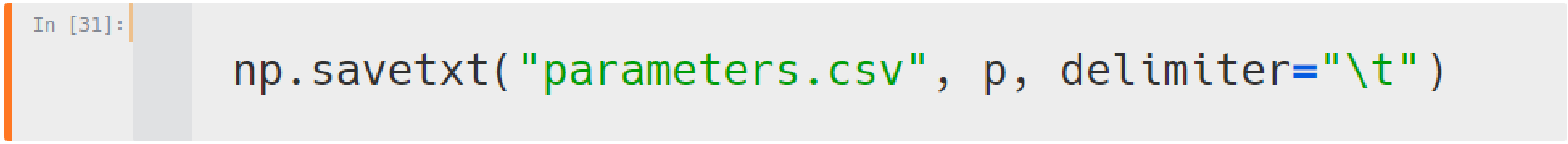
Save parameter



In [30]:

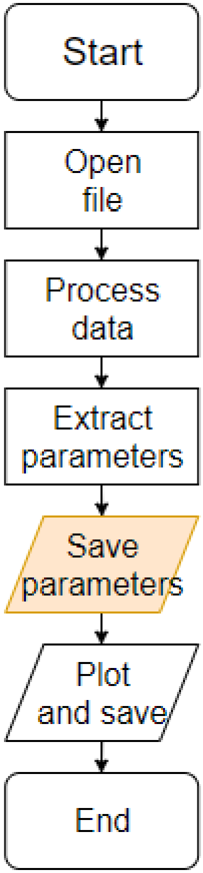
```python
logz = np.log(data[:,1])
logf = np.log(data[:,0])
from numpy.polynomial import polynomial as P
p = P.polyfit(logf, logz, 1)
print(p)
```

```
[16.60429901 -0.96116536]
```



In [31]:

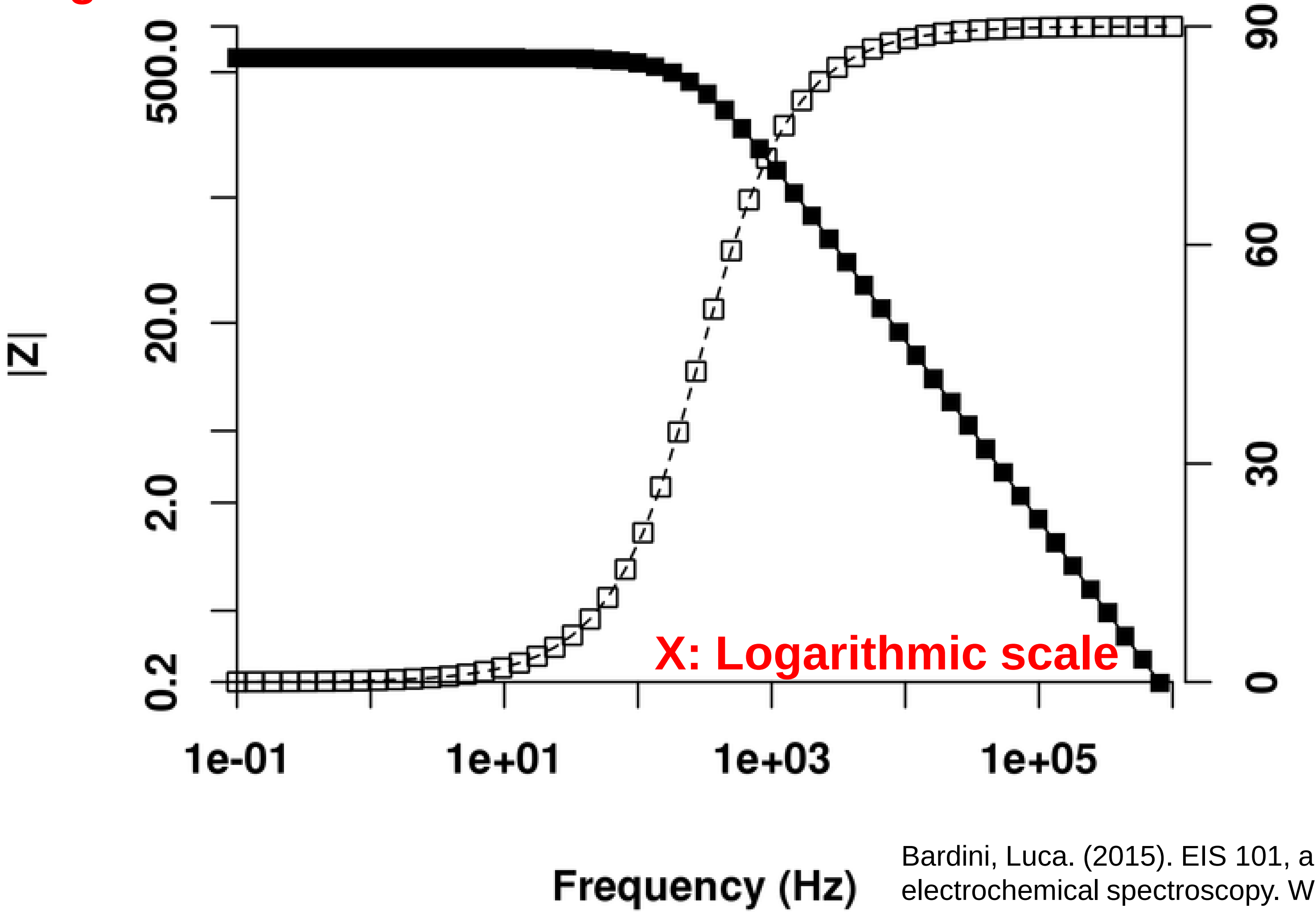
```python
np.savetxt("parameters.csv", p, delimiter="\t")
```



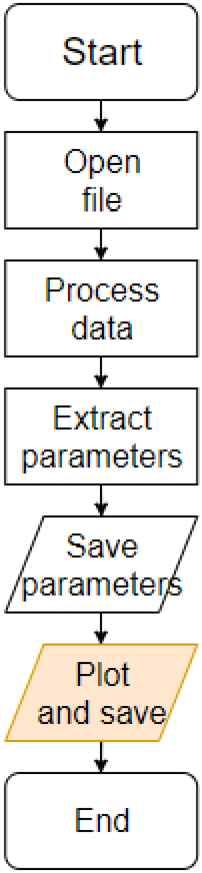
Bode plot

Y2: Linear scale –
I prefer ϕ not inverted

Y1: Logarithmic scale

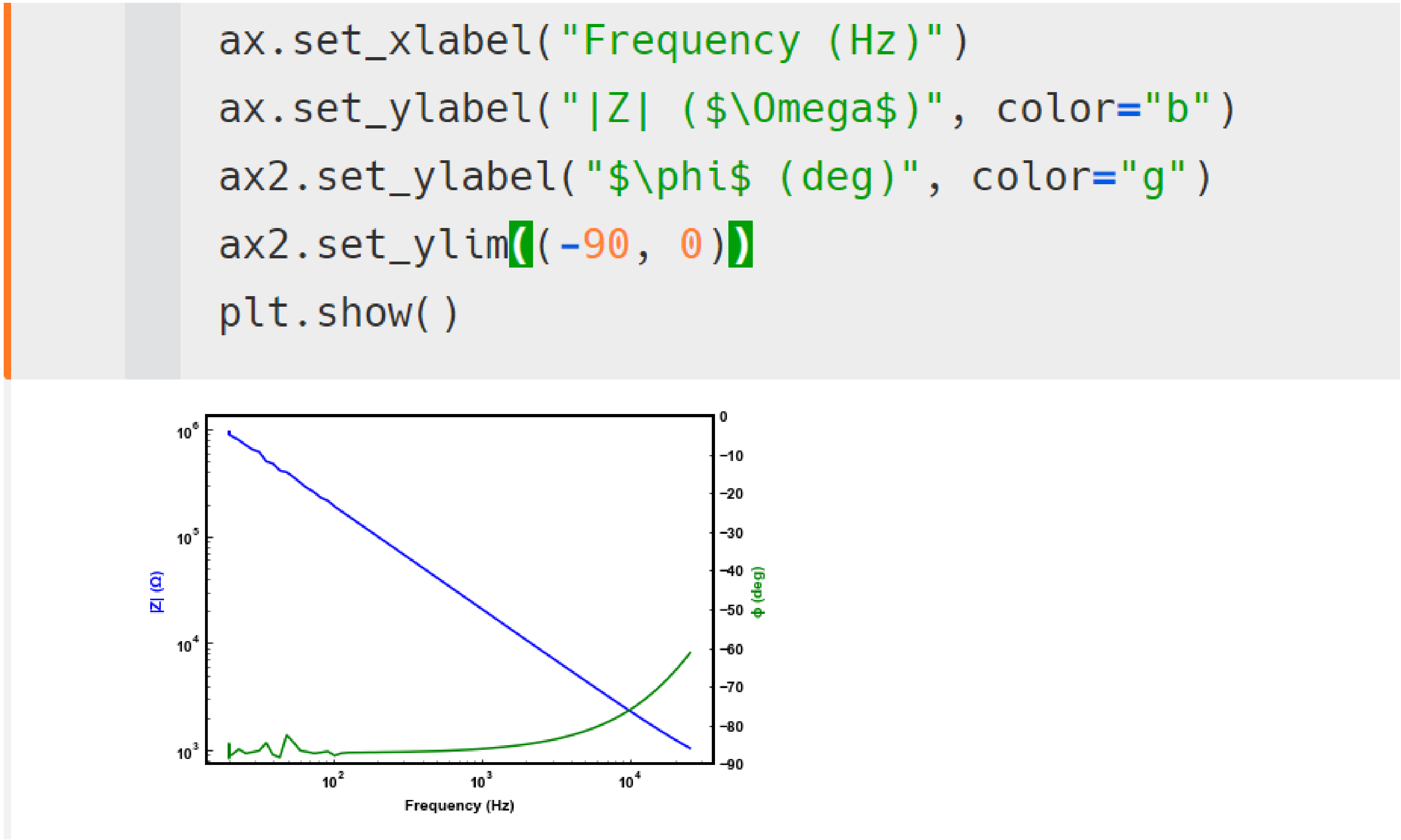


X: Logarithmic scale

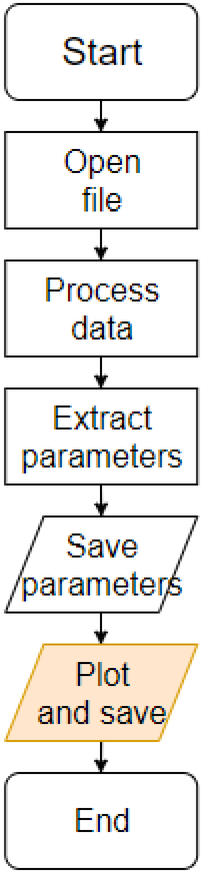


Bardini, Luca. (2015). EIS 101, an introduction to electrochemical spectroscopy. What was a website is now available as a self-contained PDF.

Plotting – Exercise 0-2



```
ax.set_xlabel("Frequency (Hz)")
ax.set_ylabel("|Z| ($\Omega$)", color="b")
ax2.set_ylabel("$\phi$ (deg)", color="g")
ax2.set_ylim((-90, 0))
plt.show()
```



Start

Open file

Process data

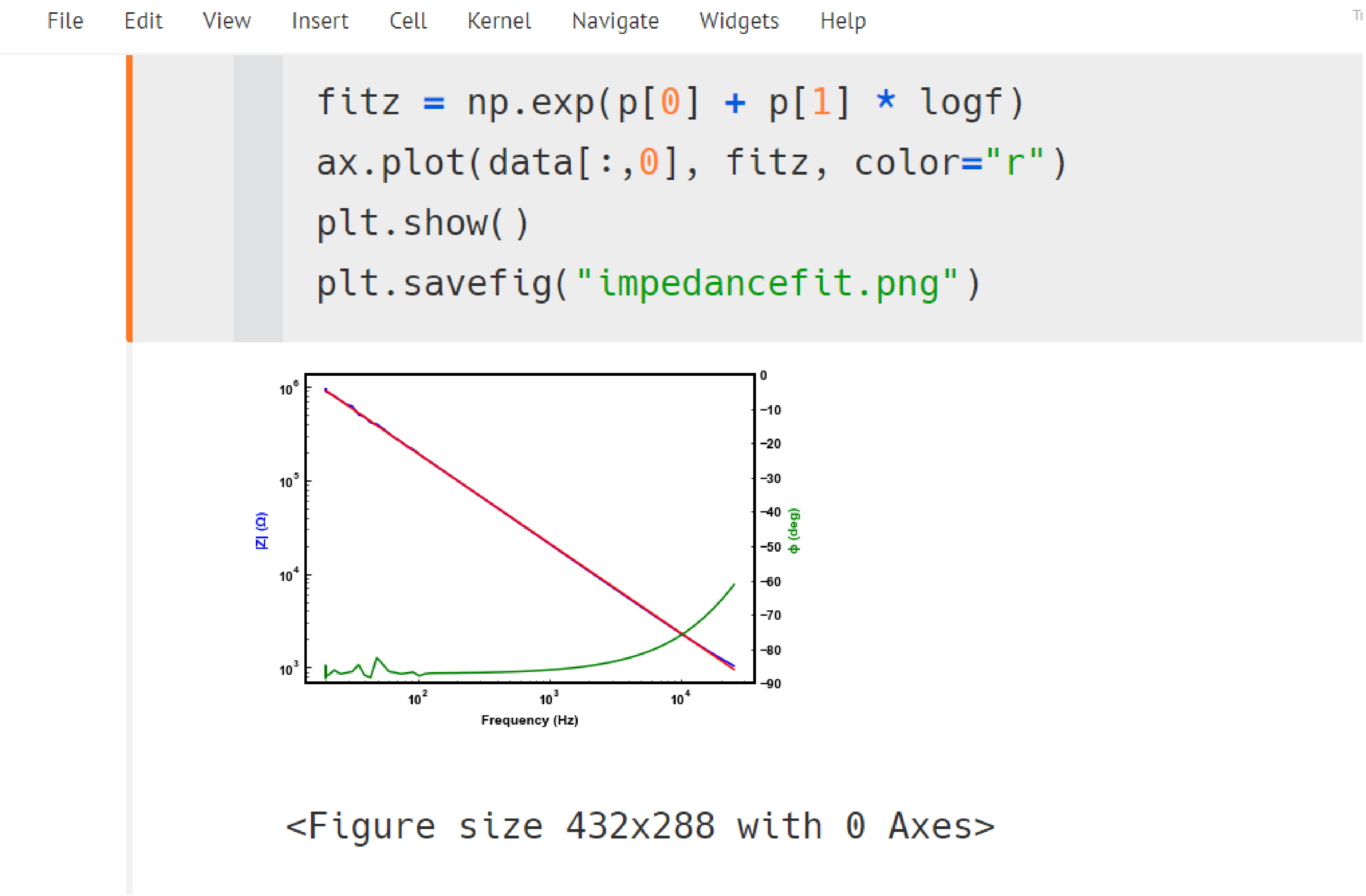
Extract parameters

Save parameters

Plot and save

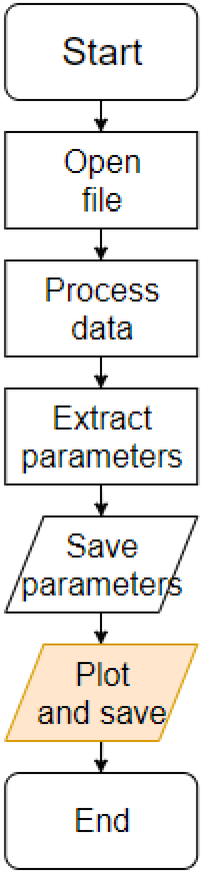
End

Add fit line

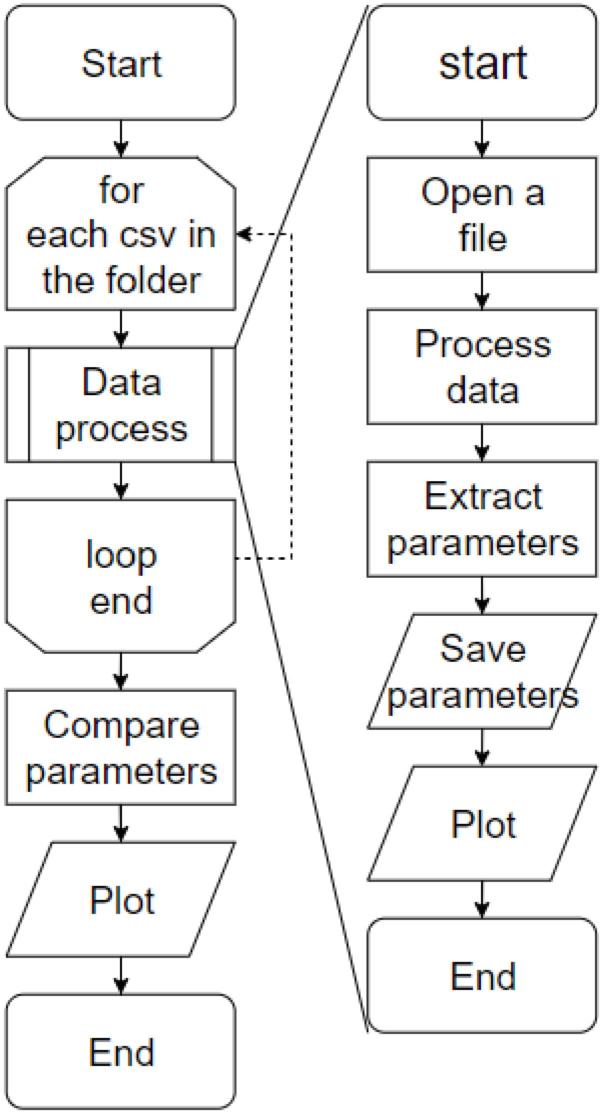


```
fitz = np.exp(p[0] + p[1] * logf)
ax.plot(data[:,0], fitz, color="r")
plt.show()
plt.savefig("impedancefit.png")
```

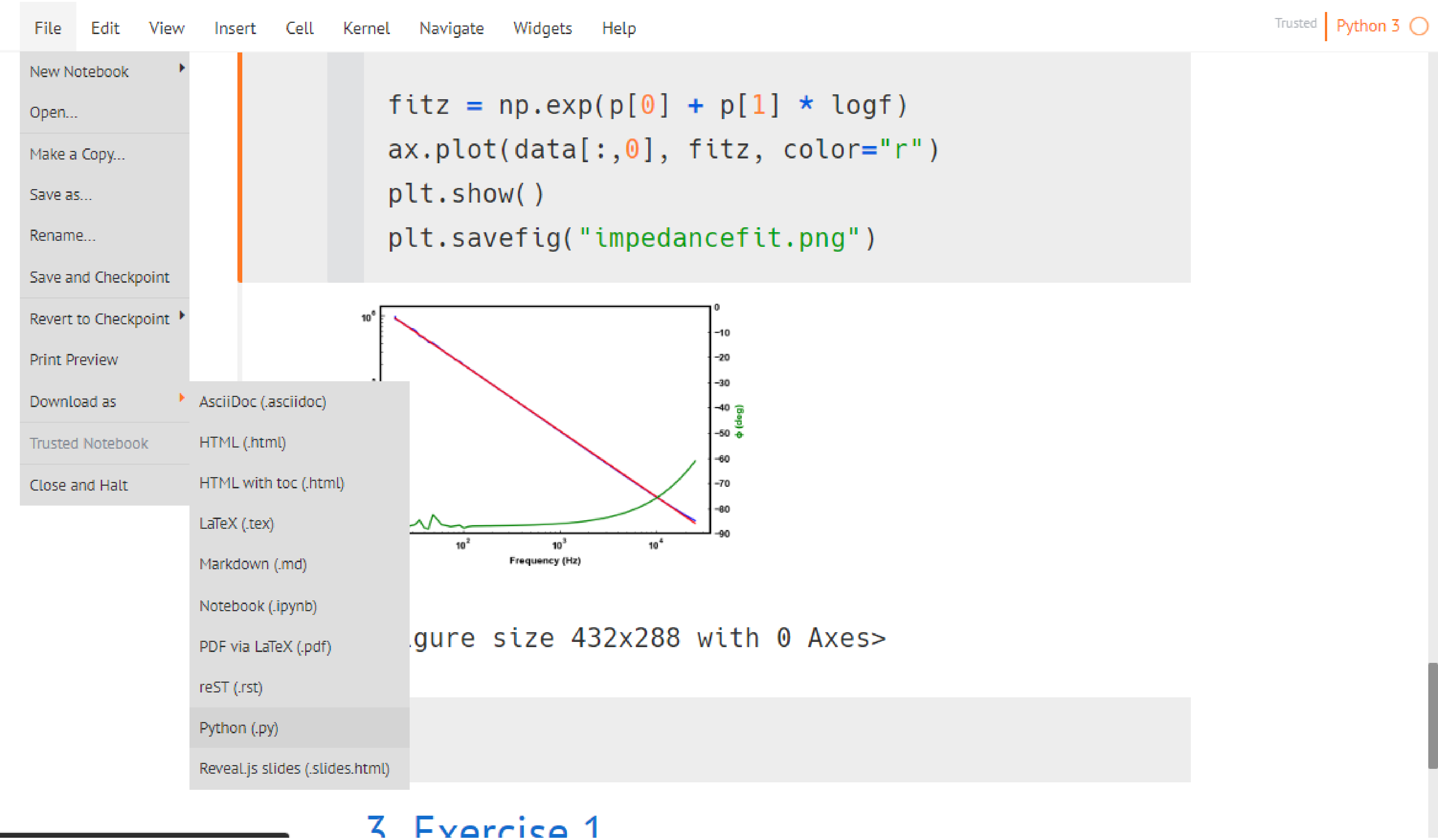
```
<Figure size 432x288 with 0 Axes>
```



That’s not all!



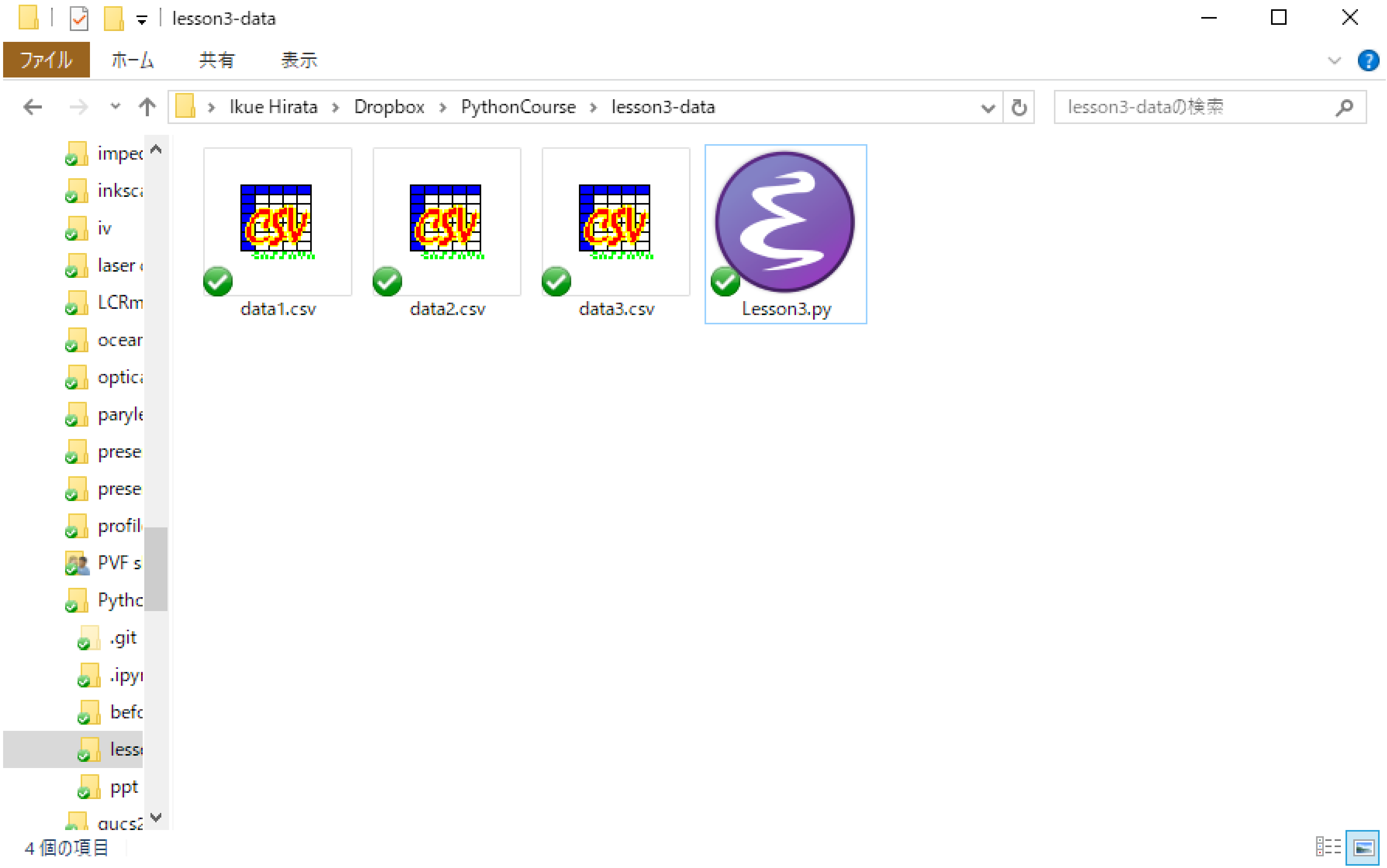
Download the code



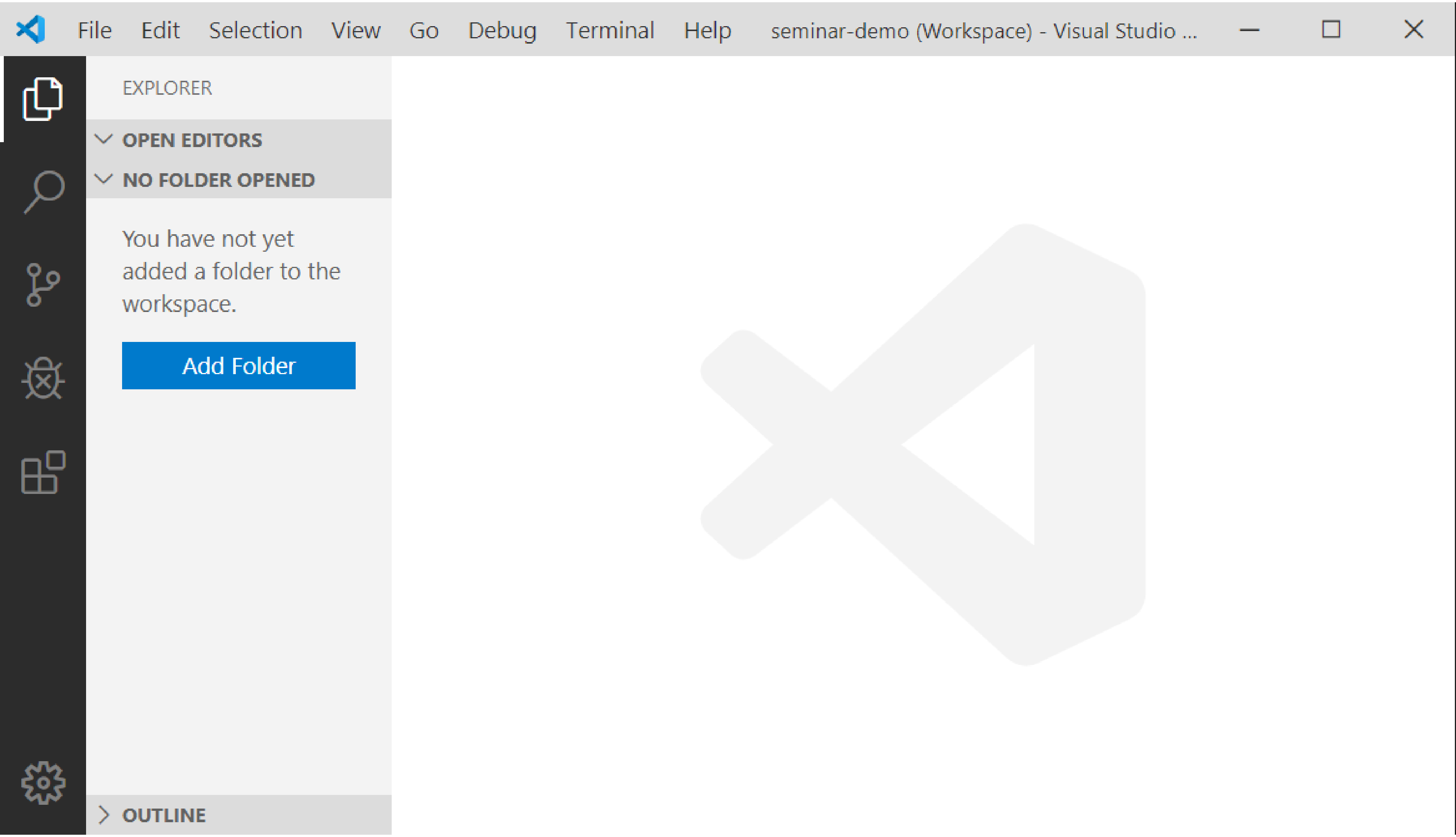
```
fitz = np.exp(p[0] + p[1] * logf)
ax.plot(data[:,0], fitz, color="r")
plt.show()
plt.savefig("impedancefit.png")
```

```
.gure size 432x288 with 0 Axes>
```

Store it where other data are

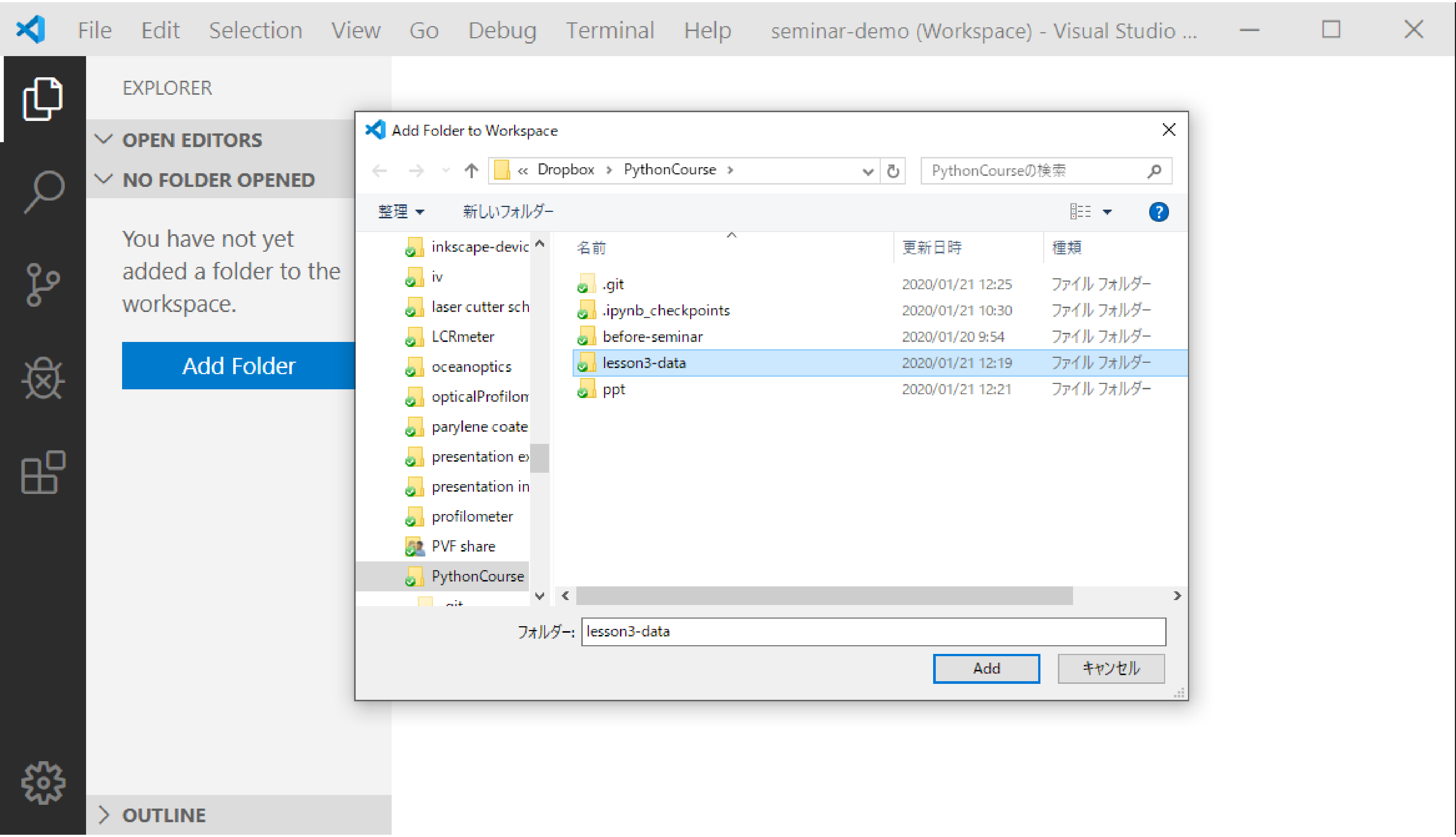


Open Visual Studio Code



You have not yet added a folder to the workspace.

Add folder – workspace

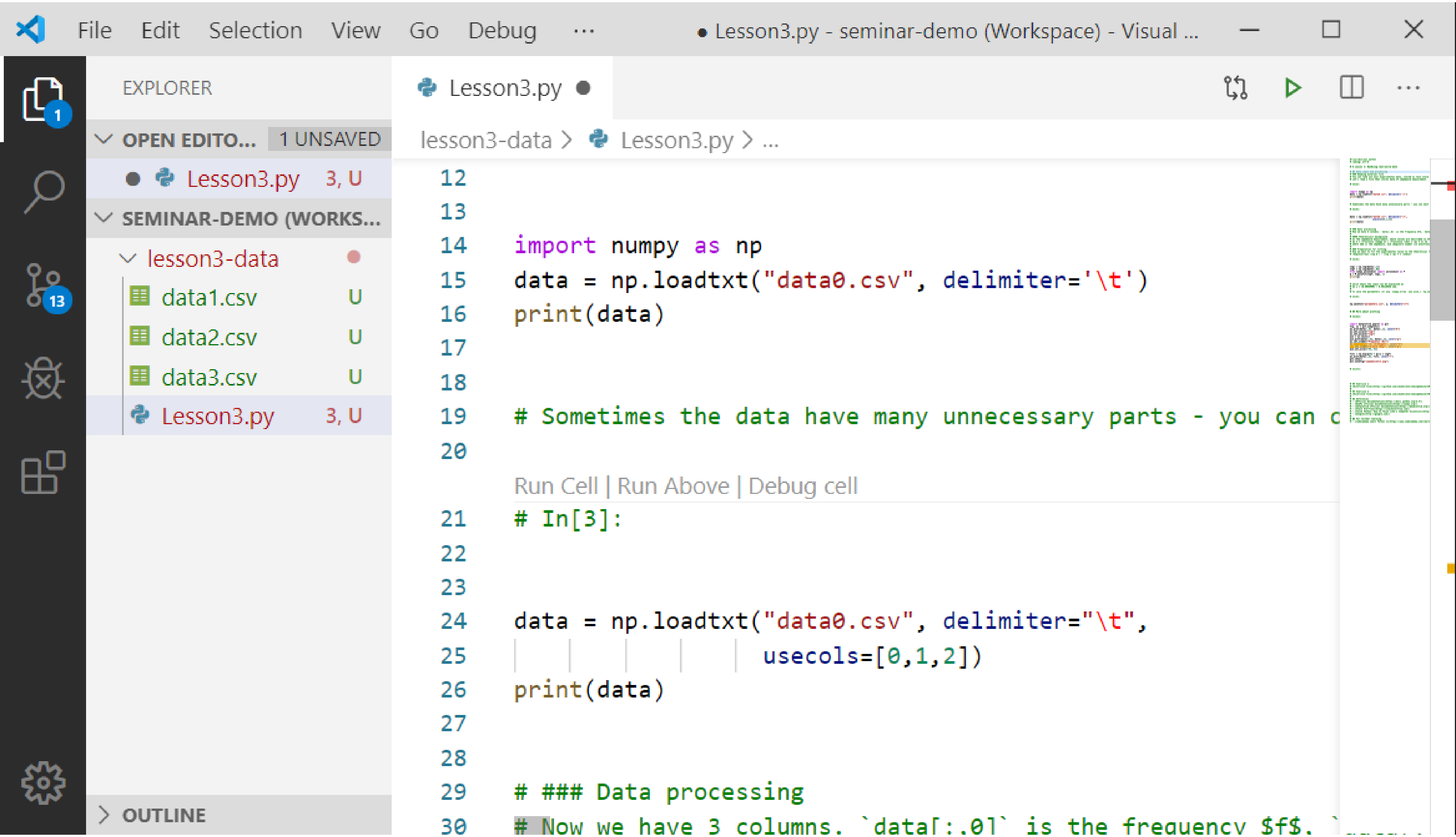


You have not yet added a folder to the workspace.

名前	更新日時	種類
.git	2020/01/21 12:25	ファイル フォルダー
.ipynb_checkpoints	2020/01/21 10:30	ファイル フォルダー
before-seminar	2020/01/20 9:54	ファイル フォルダー
lesson3-data	2020/01/21 12:19	ファイル フォルダー
ppt	2020/01/21 12:21	ファイル フォルダー

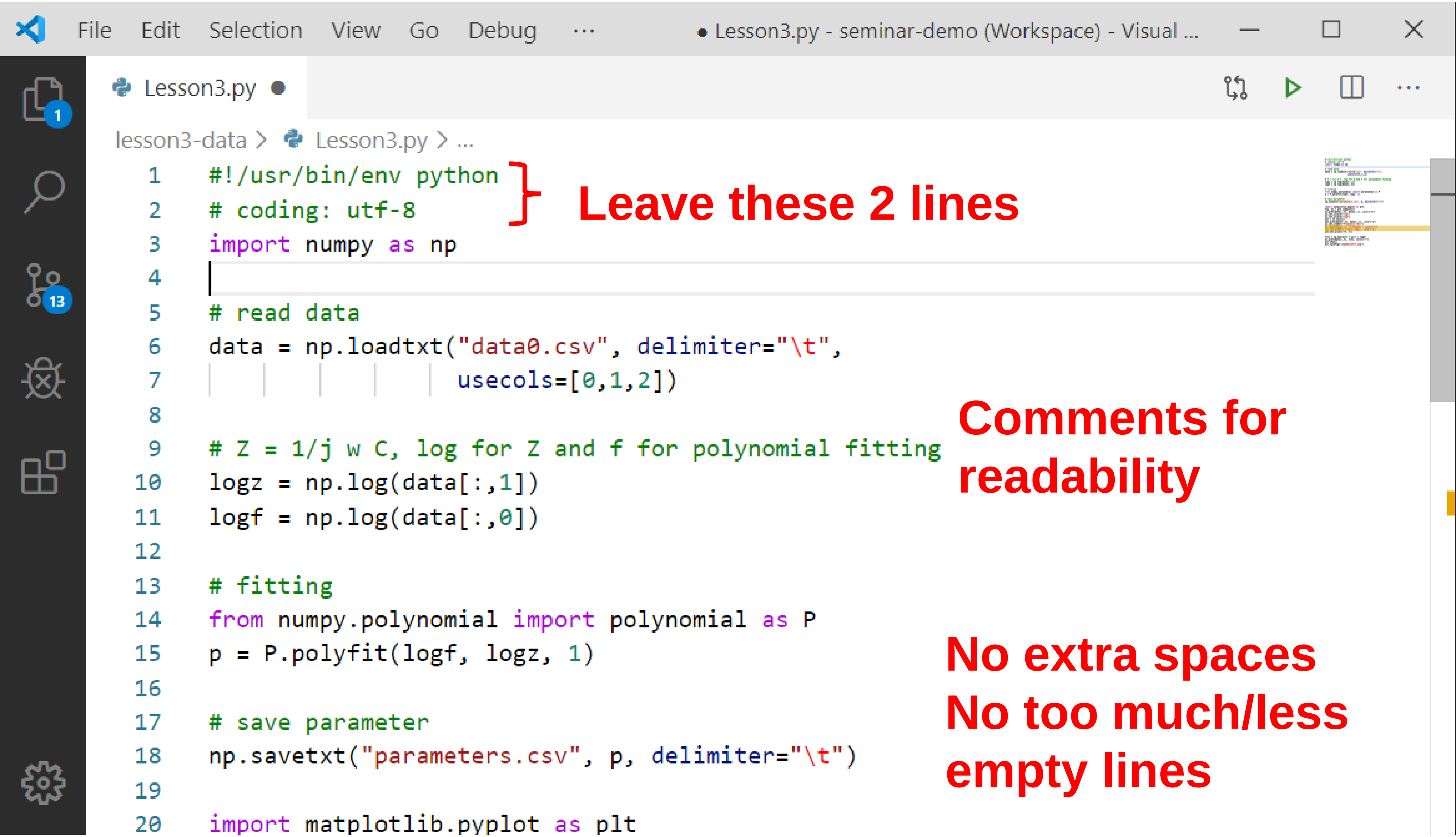
フォルダー: lesson3-data

Open your code



```


import numpy as np
data = np.loadtxt("data0.csv", delimiter='\t')
print(data)


# Sometimes the data have many unnecessary parts - you can c

# In[3]:


data = np.loadtxt("data0.csv", delimiter="\t",
                  usecols=[0,1,2])
print(data)


# ### Data processing
# Now we have 3 columns. `data[:,0]` is the frequency $f$. `
```

Code aesthetics – readability


File
Edit
Selection
View
Go
Debug
···
● Lesson3.py - seminar-demo (Workspace) - Visual ...
Lesson3.py ●
lesson3-data > Lesson3.py > ...
1
13

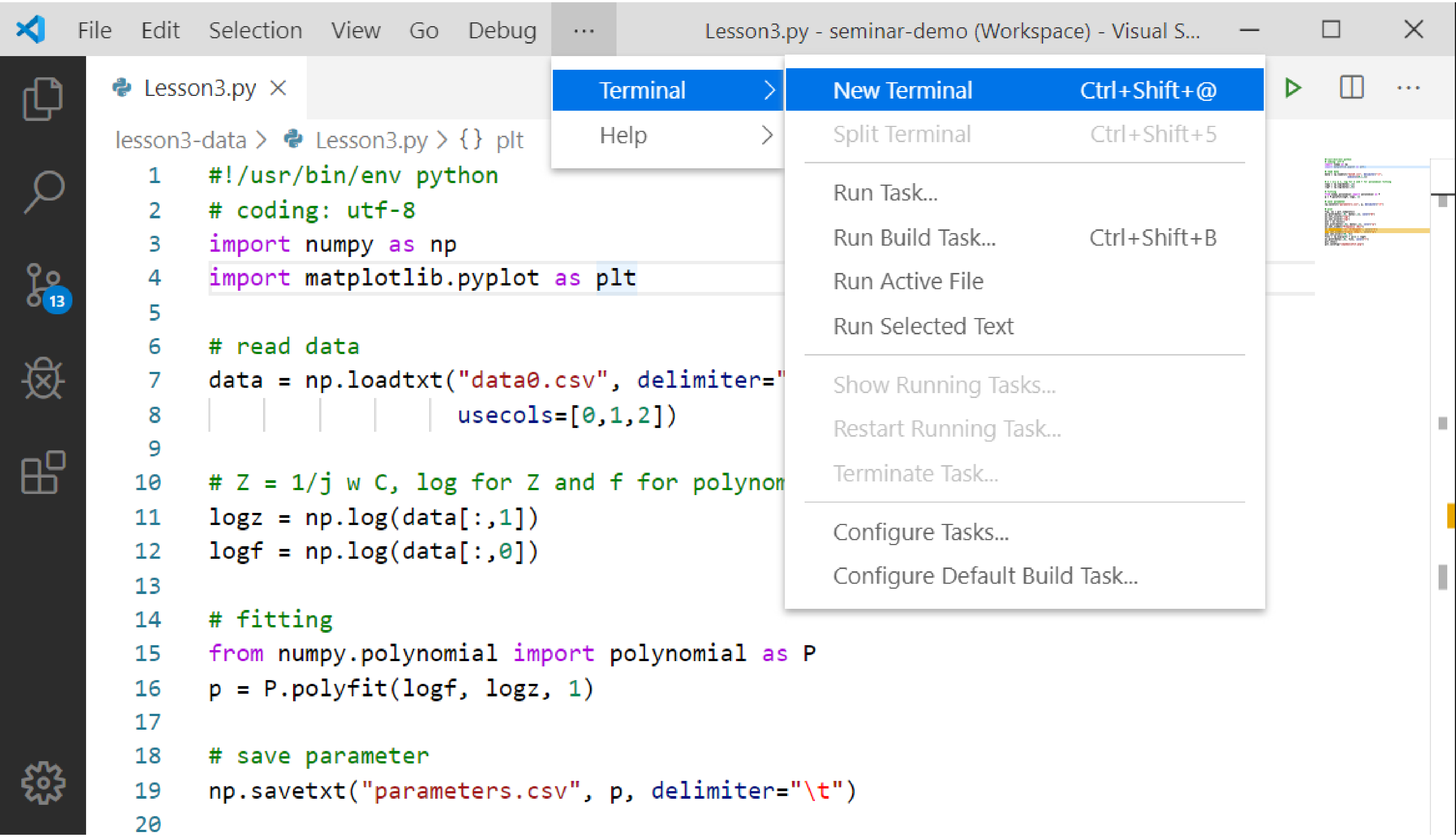
```python
#!/usr/bin/env python
# coding: utf-8
import numpy as np

# read data
data = np.loadtxt("data0.csv", delimiter="\t",
                  usecols=[0,1,2])

# Z = 1/j w C, log for Z and f for polynomial fitting
logz = np.log(data[:,1])
logf = np.log(data[:,0])

# fitting
from numpy.polynomial import polynomial as P
p = P.polyfit(logf, logz, 1)

# save parameter
np.savetxt("parameters.csv", p, delimiter="\t")

import matplotlib.pyplot as plt
```

Leave these 2 lines

Comments for readability

No extra spaces
No too much/less empty lines

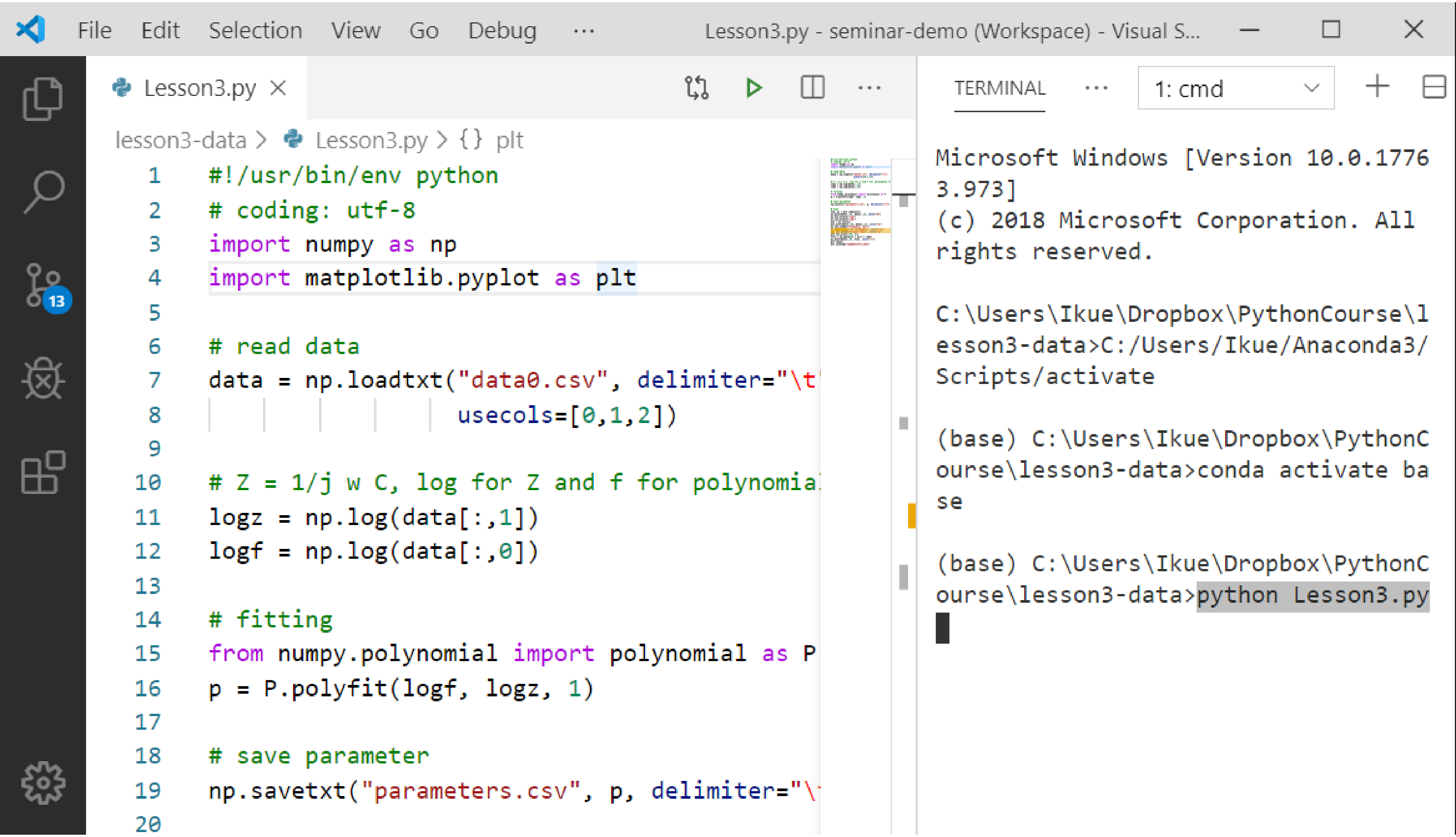
Open Terminal



File
Edit
Selection
View
Go
Debug
···
Lesson3.py - seminar-demo (Workspace) - Visual S...
Lesson3.py
lesson3-data > Lesson3.py > {} plt
13
Terminal
Help
New Terminal Ctrl+Shift+@
Split Terminal Ctrl+Shift+5
Run Task...
Run Build Task... Ctrl+Shift+B
Run Active File
Run Selected Text
Show Running Tasks...
Restart Running Task...
Terminate Task...
Configure Tasks...
Configure Default Build Task...

```python
#!/usr/bin/env python
# coding: utf-8
import numpy as np
import matplotlib.pyplot as plt

# read data
data = np.loadtxt("data0.csv", delimiter="
                    usecols=[0,1,2])

# Z = 1/j w C, log for Z and f for polynom
logz = np.log(data[:,1])
logf = np.log(data[:,0])

# fitting
from numpy.polynomial import polynomial as P
p = P.polyfit(logf, logz, 1)

# save parameter
np.savetxt("parameters.csv", p, delimiter="\t")

```

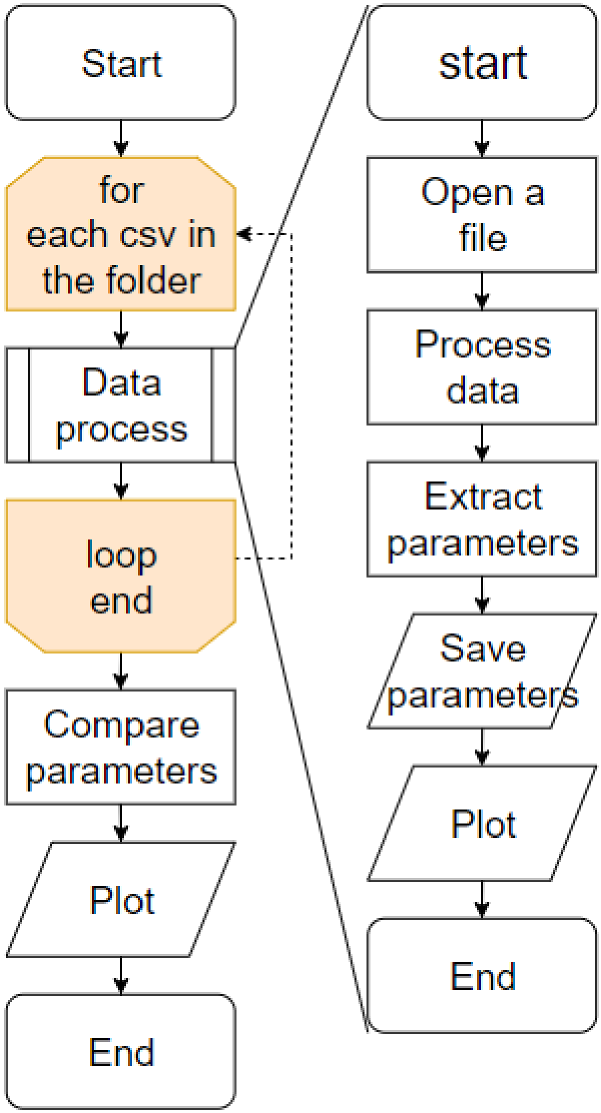
Run the code


File
Edit
Selection
View
Go
Debug
···
Lesson3.py - seminar-demo (Workspace) - Visual S...
Lesson3.py
13
lesson3-data > Lesson3.py > {} plt
TERMINAL
1: cmd

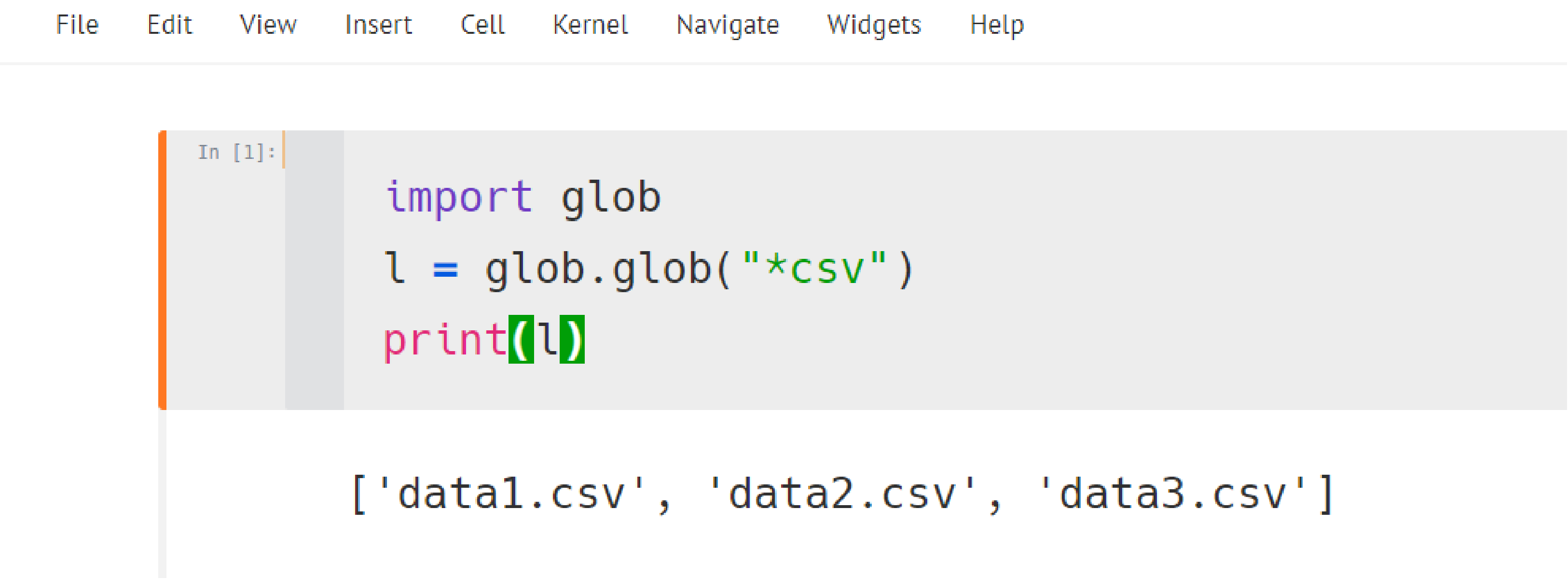
```python
#!/usr/bin/env python
# coding: utf-8
import numpy as np
import matplotlib.pyplot as plt

# read data
data = np.loadtxt("data0.csv", delimiter="\t
                    usecols=[0,1,2])

# Z = 1/j w C, log for Z and f for polynomia
logz = np.log(data[:,1])
logf = np.log(data[:,0])

# fitting
from numpy.polynomial import polynomial as P
p = P.polyfit(logf, logz, 1)

# save parameter
np.savetxt("parameters.csv", p, delimiter="\
```

```
Microsoft Windows [Version 10.0.1776
3.973]
(c) 2018 Microsoft Corporation. All
rights reserved.

C:\Users\Ikue\Dropbox\PythonCourse\l
esson3-data>C:/Users/Ikue/Anaconda3/
Scripts/activate

(base) C:\Users\Ikue\Dropbox\PythonC
ourse\lesson3-data>conda activate ba
se

(base) C:\Users\Ikue\Dropbox\PythonC
ourse\lesson3-data>python Lesson3.py
```

Getting multiple files

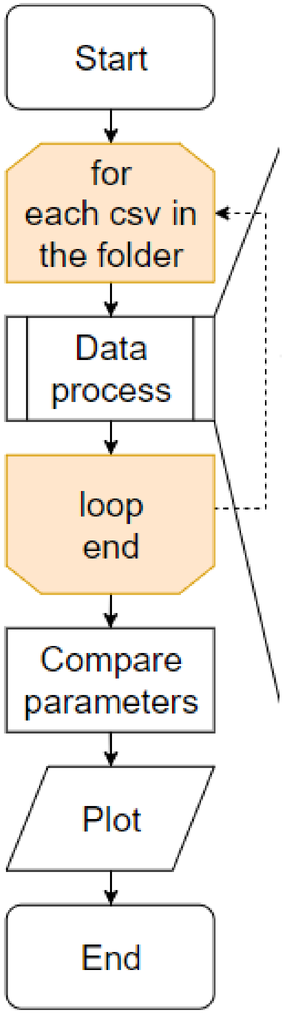


glob

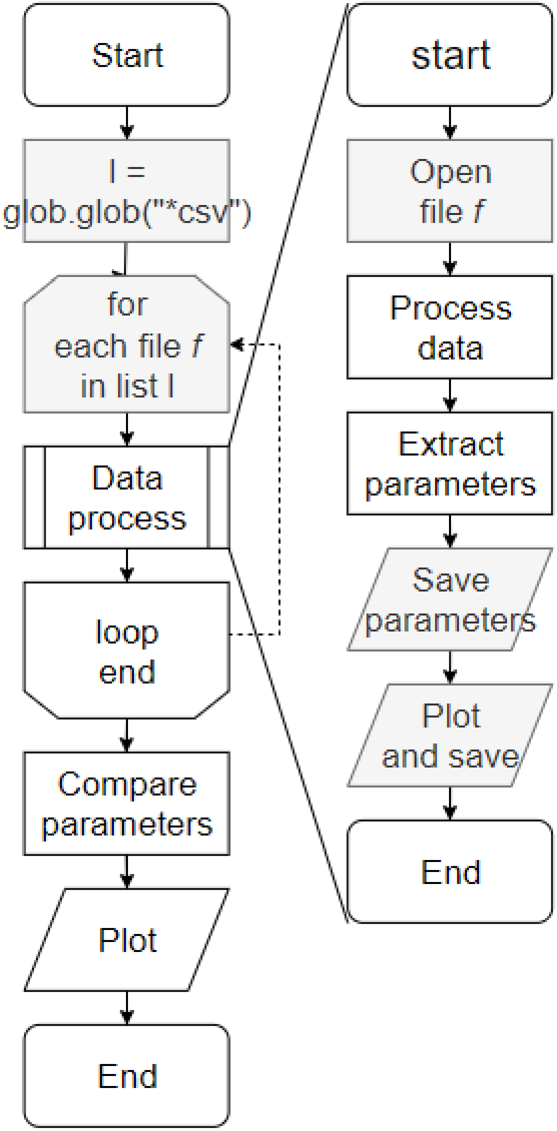


```python
import glob
l = glob.glob("*csv")
print(l)
```

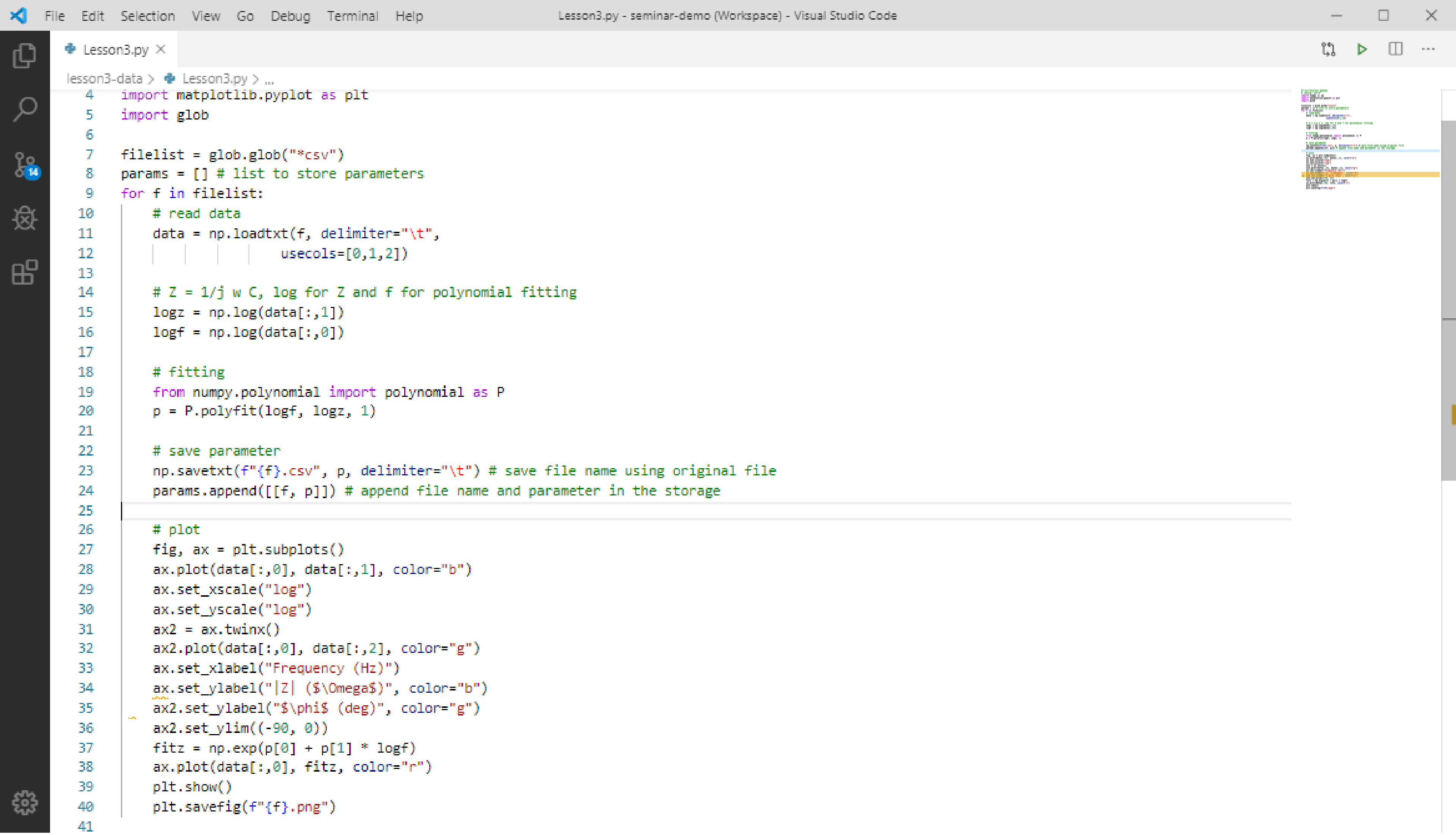
```
['data1.csv', 'data2.csv', 'data3.csv']
```



Pay attention to save file name



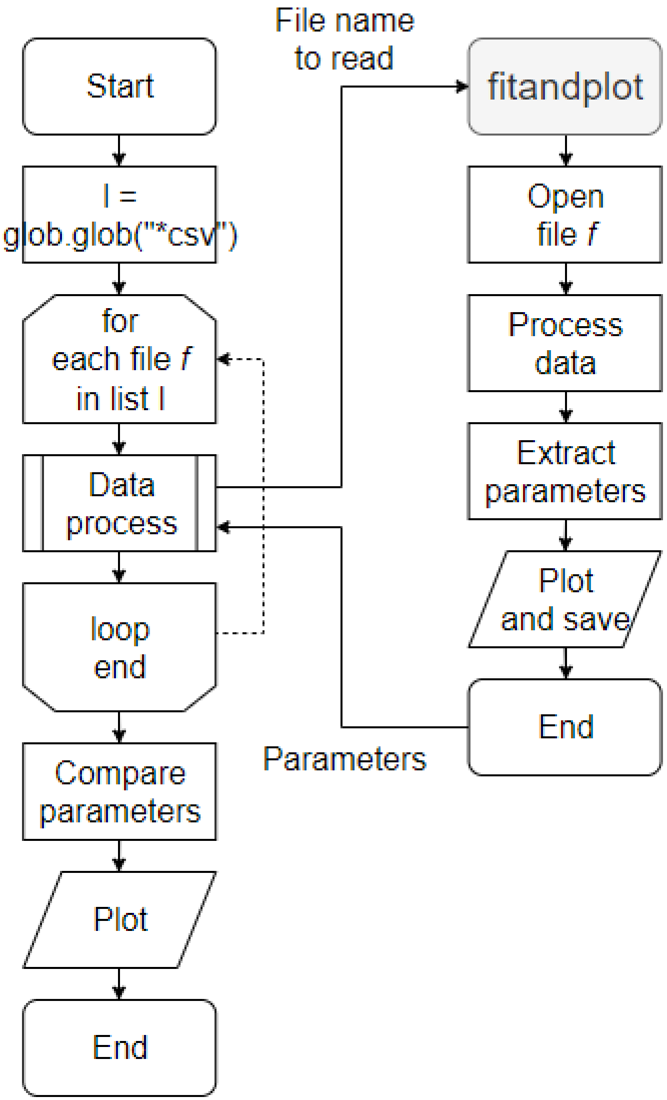
“Spaghetti” code



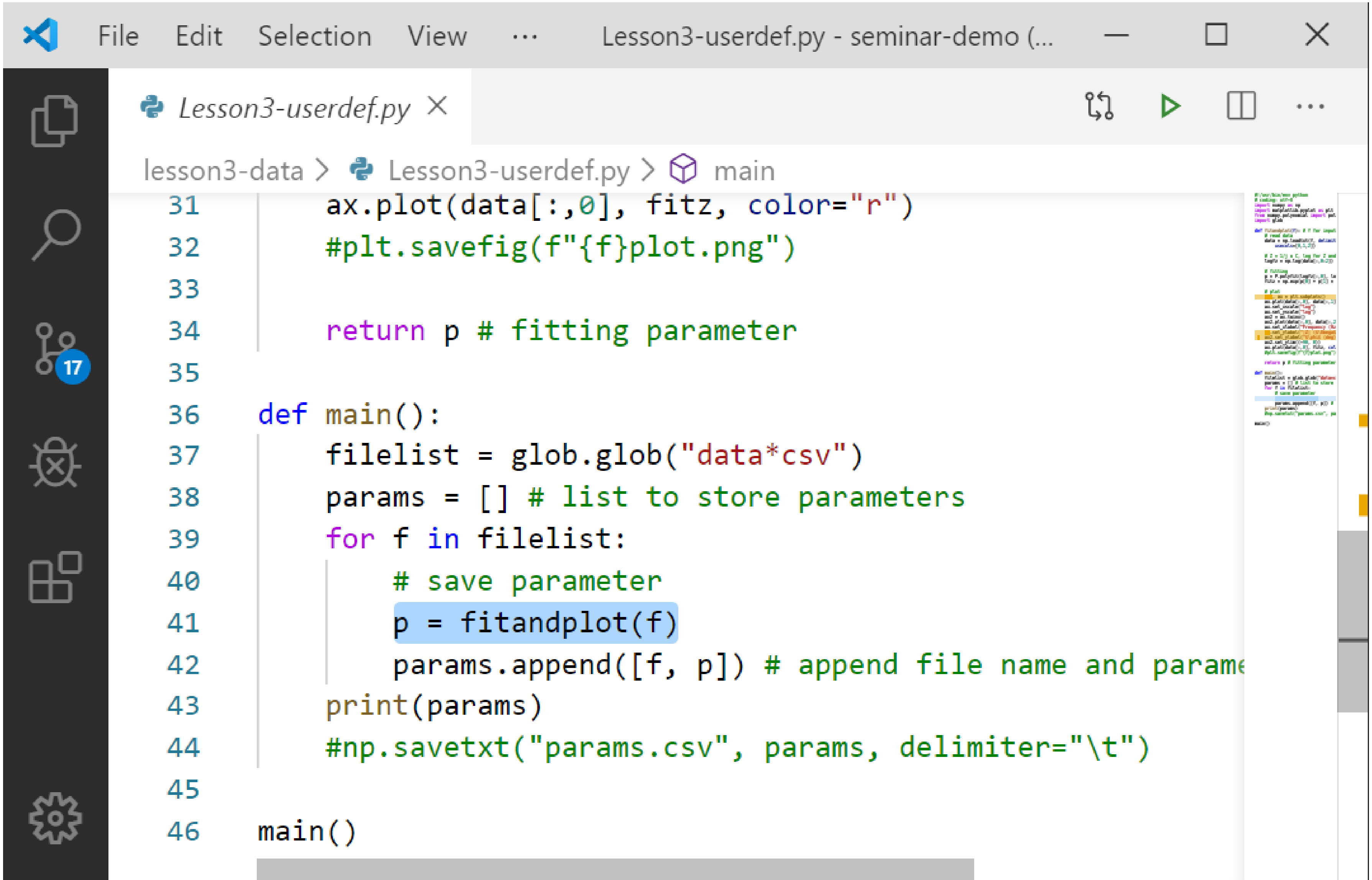
File
Edit
Selection
View
Go
Debug
Terminal
Help
Lesson3.py - seminar-demo (Workspace) - Visual Studio Code
Lesson3.py
lesson3-data > Lesson3.py > ...
14

```python
import matplotlib.pyplot as plt
import glob

filelist = glob.glob("*csv")
params = [] # list to store parameters
for f in filelist:
    # read data
    data = np.loadtxt(f, delimiter="\t",
                    usecols=[0,1,2])

    # Z = 1/j w C, log for Z and f for polynomial fitting
    logz = np.log(data[:,1])
    logf = np.log(data[:,0])

    # fitting
    from numpy.polynomial import polynomial as P
    p = P.polyfit(logf, logz, 1)

    # save parameter
    np.savetxt(f"{f}.csv", p, delimiter="\t") # save file name using original file
    params.append([[f, p]]) # append file name and parameter in the storage

    # plot
    fig, ax = plt.subplots()
    ax.plot(data[:,0], data[:,1], color="b")
    ax.set_xscale("log")
    ax.set_yscale("log")
    ax2 = ax.twinx()
    ax2.plot(data[:,0], data[:,2], color="g")
    ax.set_xlabel("Frequency (Hz)")
    ax.set_ylabel("|Z| ($\Omega$)", color="b")
    ax2.set_ylabel("$\phi$ (deg)", color="g")
    ax2.set_ylim((-90, 0))
    fitz = np.exp(p[0] + p[1] * logf)
    ax.plot(data[:,0], fitz, color="r")
    plt.show()
    plt.savefig(f"{f}.png")

```

User defined function

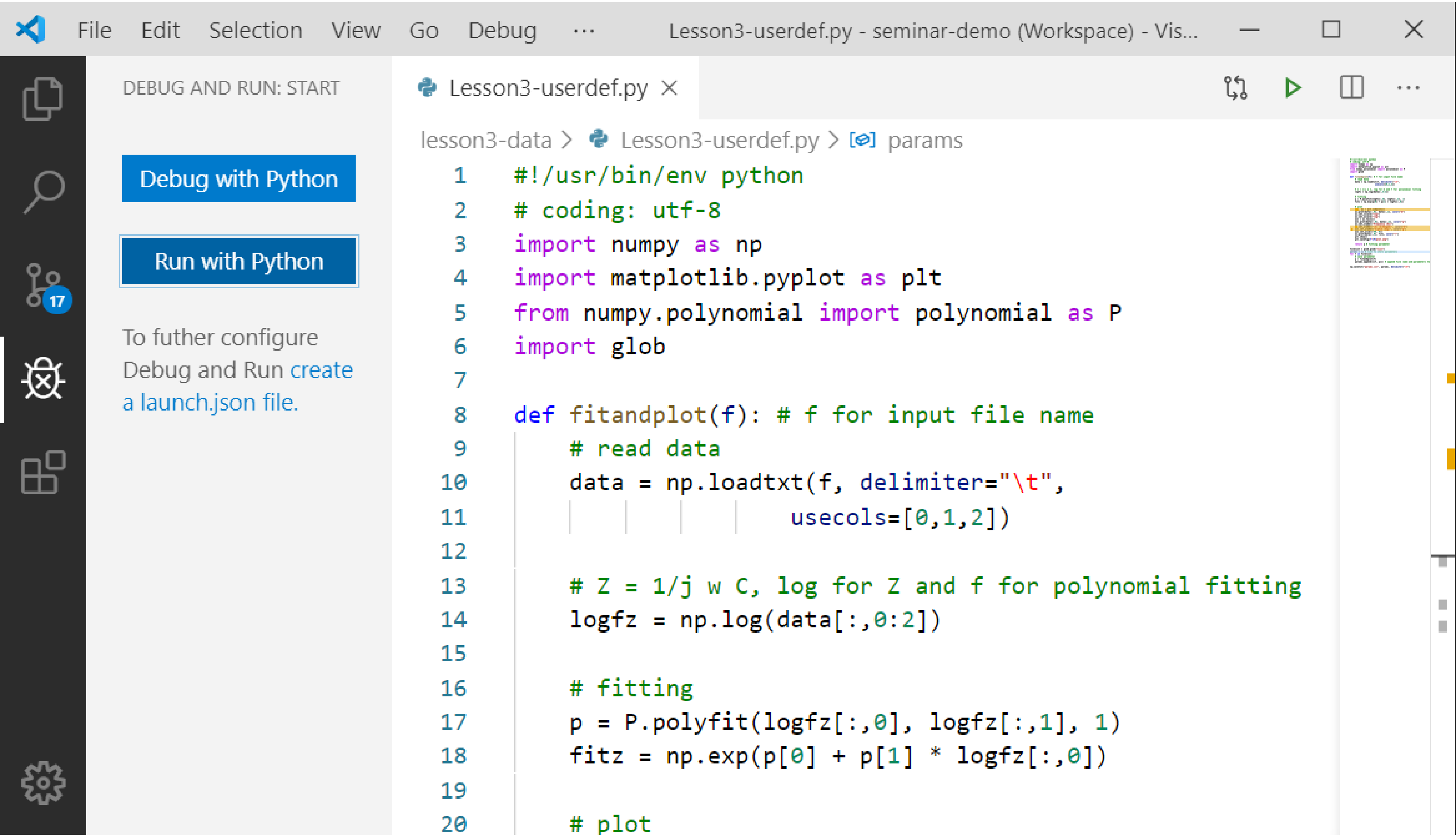


Using user defined function



```python
    ax.plot(data[:,0], fitz, color="r")
    #plt.savefig(f"{f}plot.png")

    return p # fitting parameter

def main():
    filelist = glob.glob("data*csv")
    params = [] # list to store parameters
    for f in filelist:
        # save parameter
        p = fitandplot(f)
        params.append([f, p]) # append file name and parame
    print(params)
    #np.savetxt("params.csv", params, delimiter="\t")

main()
```

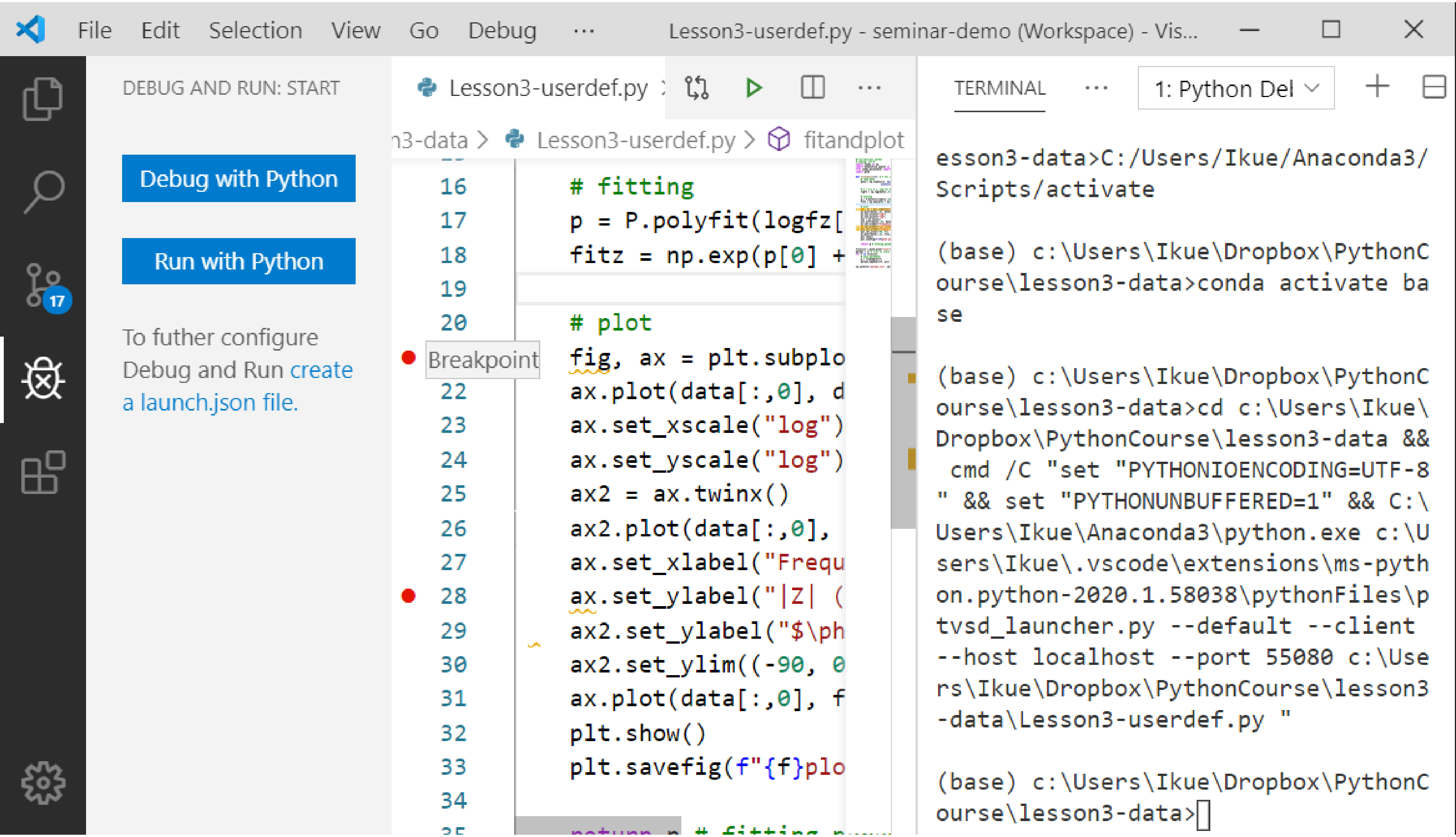
Debugging



To futher configure Debug and Run create a launch.json file.

```
#!/usr/bin/env python
# coding: utf-8
import numpy as np
import matplotlib.pyplot as plt
from numpy.polynomial import polynomial as P
import glob

def fitandplot(f): # f for input file name
    # read data
    data = np.loadtxt(f, delimiter="\t",
                usecols=[0,1,2])

    # Z = 1/j w C, log for Z and f for polynomial fitting
    logfz = np.log(data[:,0:2])

    # fitting
    p = P.polyfit(logfz[:,0], logfz[:,1], 1)
    fitz = np.exp(p[0] + p[1] * logfz[:,0])

    # plot
```

Breakpoints

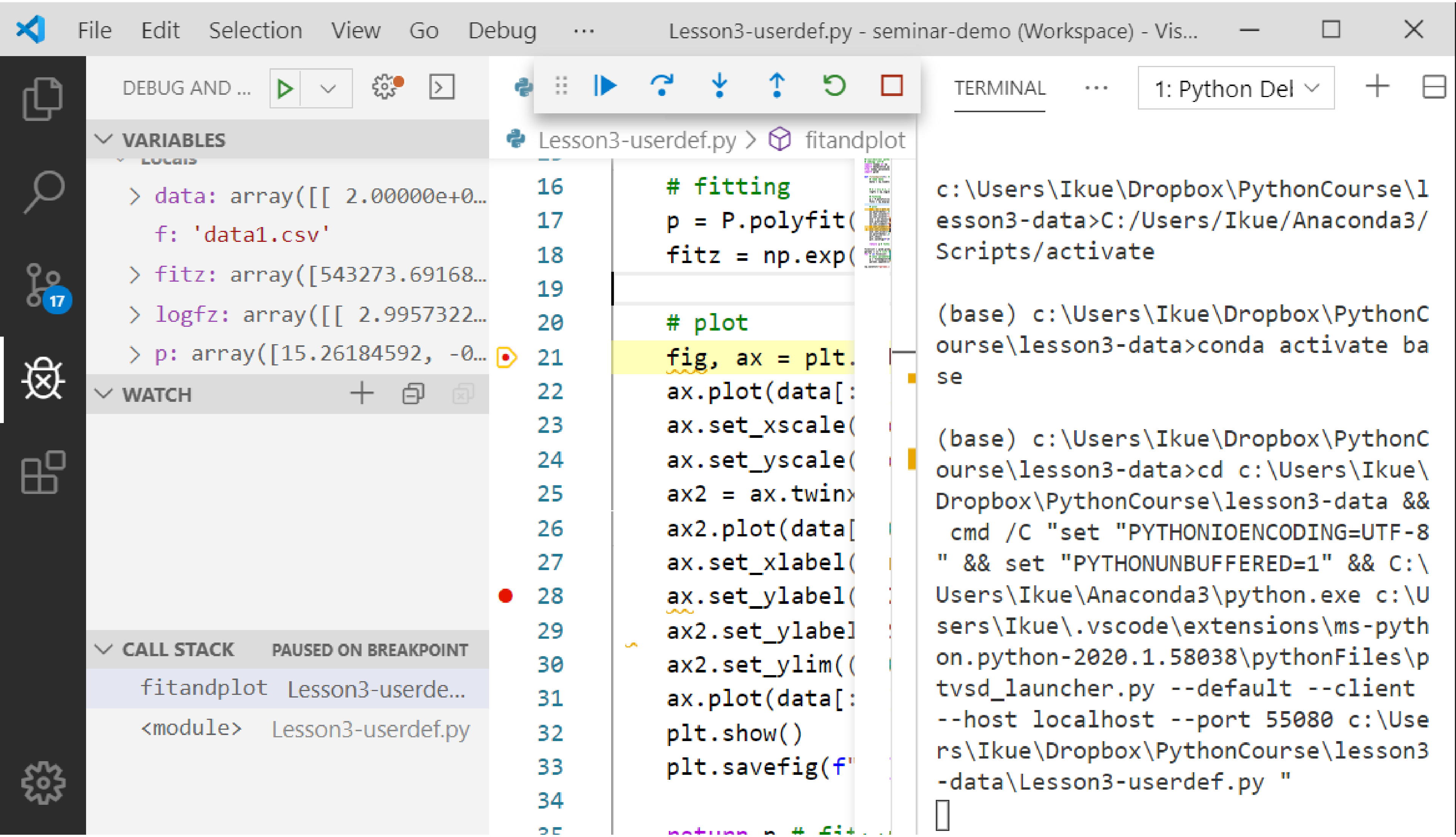


To futher configure Debug and Run create a launch.json file.

```
    # fitting
    p = P.polyfit(logfz[
    fitz = np.exp(p[0] +

    # plot
    fig, ax = plt.subplo
    ax.plot(data[:,0], d
    ax.set_xscale("log")
    ax.set_yscale("log")
    ax2 = ax.twinx()
    ax2.plot(data[:,0],
    ax.set_xlabel("Frequ
    ax.set_ylabel("|Z| (
    ax2.set_ylabel("$\ph
    ax2.set_ylim((-90, 0
    ax.plot(data[:,0], f
    plt.show()
    plt.savefig(f"{f}plo
```

```
esson3-data>C:/Users/Ikue/Anaconda3/Scripts/activate

(base) c:\Users\Ikue\Dropbox\PythonCourse\lesson3-data>conda activate base

(base) c:\Users\Ikue\Dropbox\PythonCourse\lesson3-data>cd c:\Users\Ikue\Dropbox\PythonCourse\lesson3-data && cmd /C "set "PYTHONIOENCODING=UTF-8" && set "PYTHONUNBUFFERED=1" && C:\Users\Ikue\Anaconda3\python.exe c:\Users\Ikue\.vscode\extensions\ms-python.python-2020.1.58038\pythonFiles\ptvsd_launcher.py --default --client --host localhost --port 55080 c:\Users\Ikue\Dropbox\PythonCourse\lesson3-data\Lesson3-userdef.py "

(base) c:\Users\Ikue\Dropbox\PythonCourse\lesson3-data>
```

Run



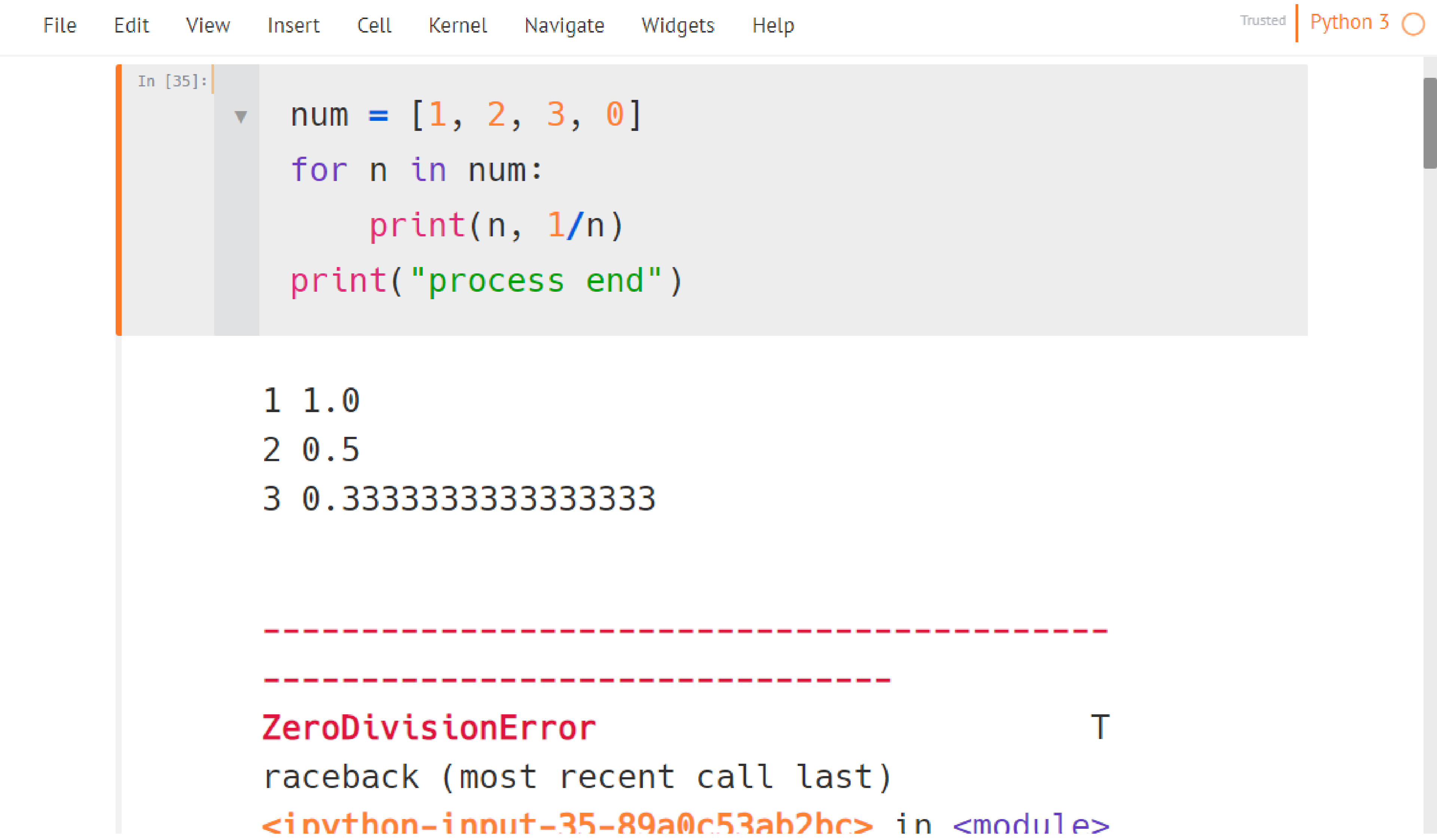
```
data: array([[ 2.00000e+0...
f: 'data1.csv'
fitz: array([543273.69168...
logfz: array([[ 2.9957322...
p: array([15.26184592, -0...
```

```
fitandplot    Lesson3-userde...
<module>      Lesson3-userdef.py
```

```
    # fitting
    p = P.polyfit(
    fitz = np.exp(

    # plot
    fig, ax = plt.
    ax.plot(data[:
    ax.set_xscale(
    ax.set_yscale(
    ax2 = ax.twinx
    ax2.plot(data[
    ax.set_xlabel(
    ax.set_ylabel(
    ax2.set_ylabel
    ax2.set_ylim((
    ax.plot(data[:
    plt.show()
    plt.savefig(f"
```

```
c:\Users\Ikue\Dropbox\PythonCourse\lesson3-data>C:/Users/Ikue/Anaconda3/Scripts/activate

(base) c:\Users\Ikue\Dropbox\PythonCourse\lesson3-data>conda activate base

(base) c:\Users\Ikue\Dropbox\PythonCourse\lesson3-data>cd c:\Users\Ikue\Dropbox\PythonCourse\lesson3-data && cmd /C "set "PYTHONIOENCODING=UTF-8" && set "PYTHONUNBUFFERED=1" && C:\Users\Ikue\Anaconda3\python.exe c:\Users\Ikue\.vscode\extensions\ms-python.python-2020.1.58038\pythonFiles\ptvsd_launcher.py --default --client --host localhost --port 55080 c:\Users\Ikue\Dropbox\PythonCourse\lesson3-data\Lesson3-userdef.py "
```

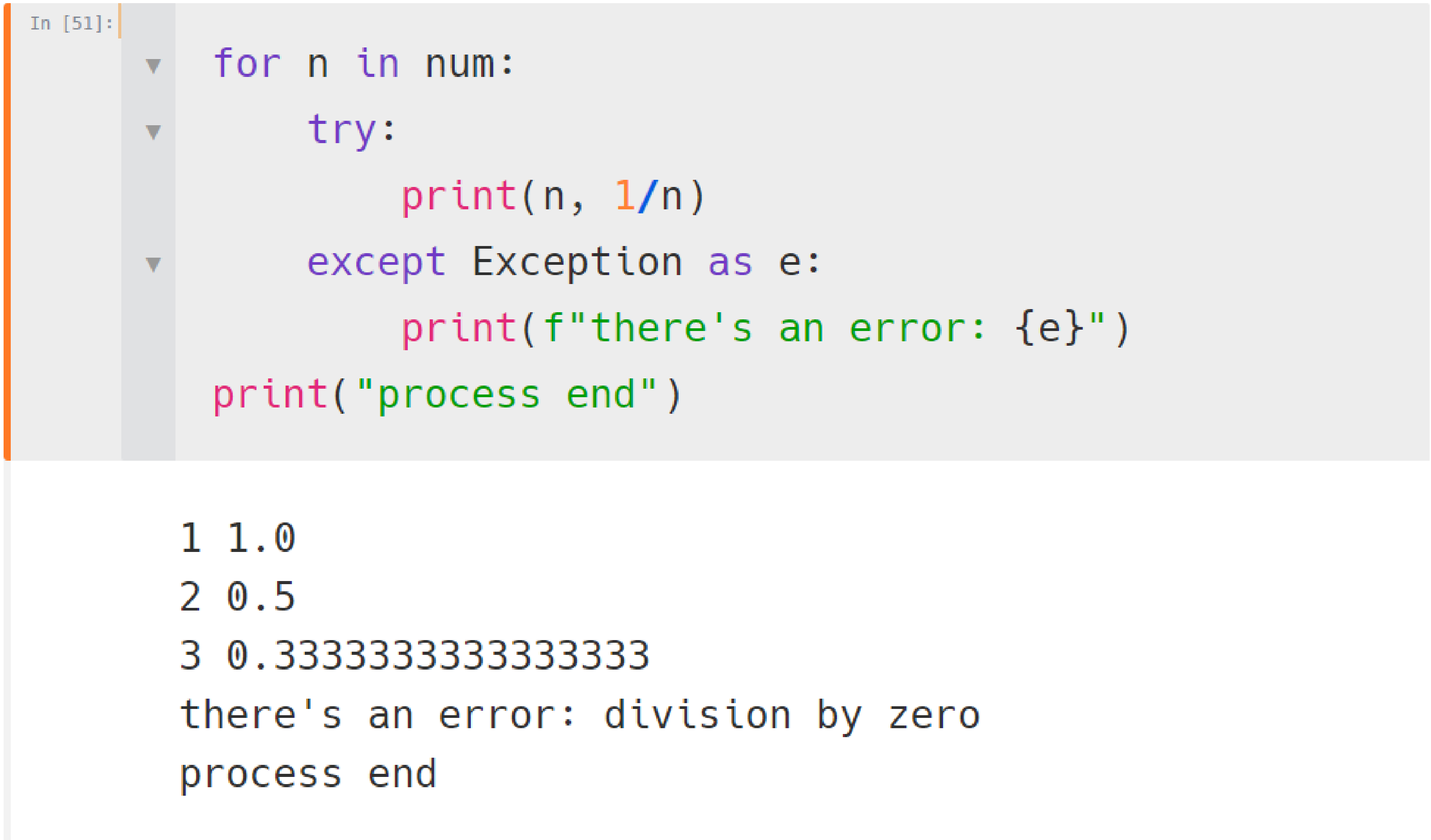
Exception handling



```
num = [1, 2, 3, 0]
for n in num:
    print(n, 1/n)
print("process end")
```

```
1 1.0
2 0.5
3 0.3333333333333333


-------------------------------------------
--------------------------------
ZeroDivisionError                         T
raceback (most recent call last)
<ipython-input-35-89a0c53ab2bc> in <module>
```

try and except

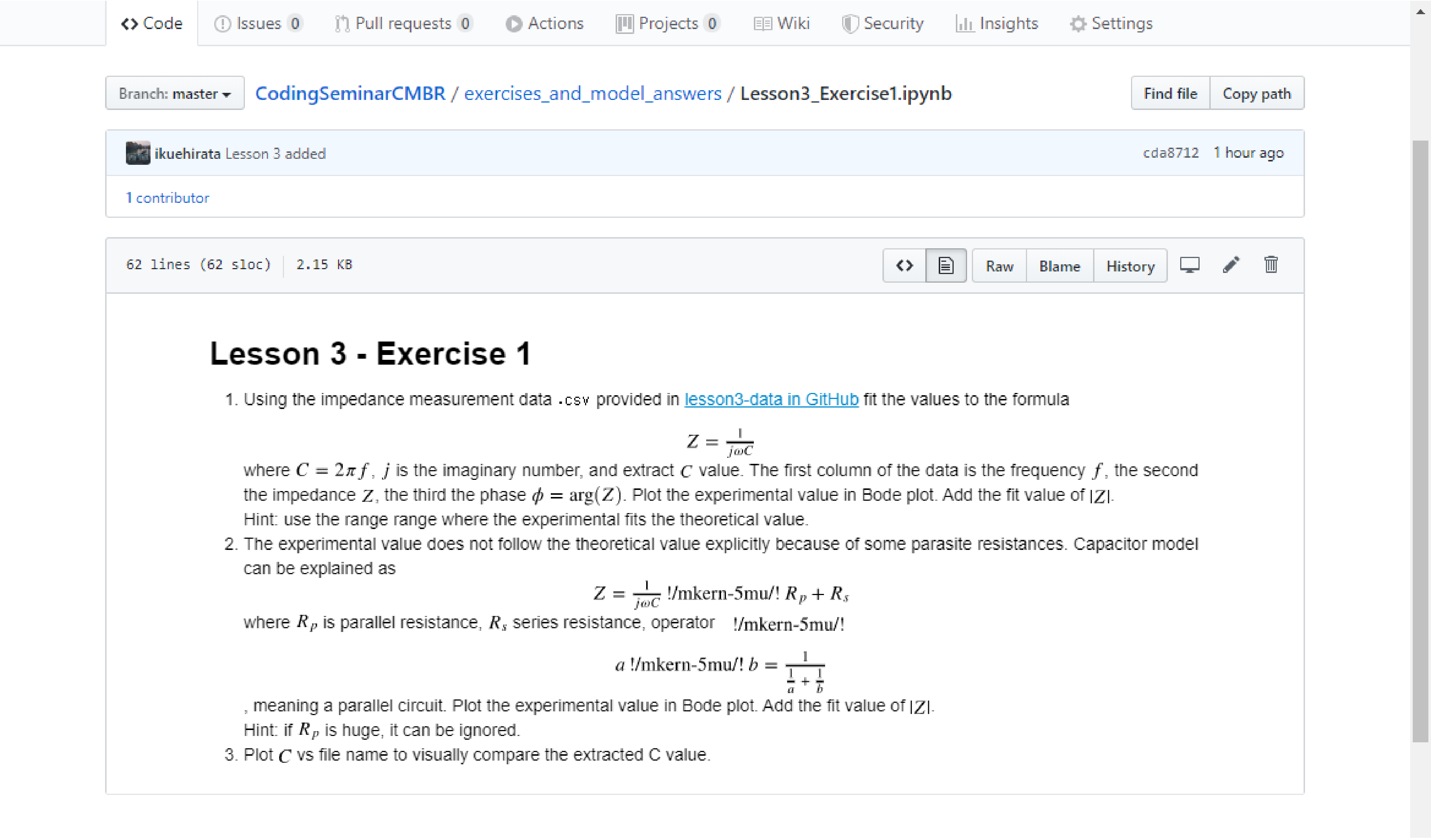


In [51]:

```
for n in num:
    try:
        print(n, 1/n)
    except Exception as e:
        print(f"there's an error: {e}")
print("process end")
```

```
1 1.0
2 0.5
3 0.3333333333333333
there's an error: division by zero
process end
```

Exercise 1



CodingSeminarCMBR / exercises_and_model_answers / Lesson3_Exercise1.ipynb

ikuehirata Lesson 3 added cda8712 1 hour ago

1 contributor

62 lines (62 sloc) 2.15 KB

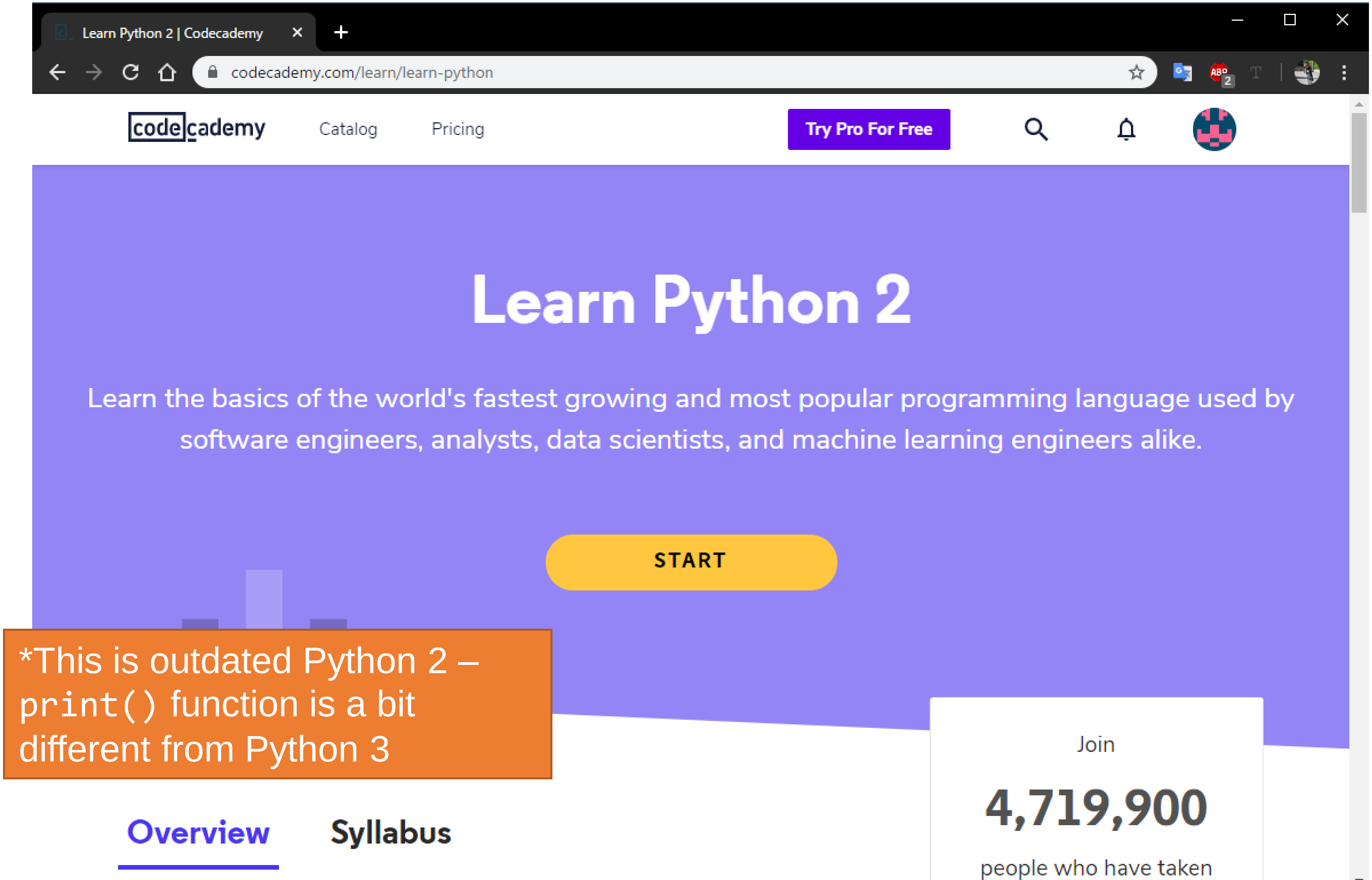
Lesson 3 - Exercise 1

1. Using the impedance measurement data `.csv` provided in lesson3-data in GitHub fit the values to the formula
$$Z = \frac{1}{j\omega C}$$
where $C = 2\pi f$, j is the imaginary number, and extract C value. The first column of the data is the frequency f, the second the impedance Z, the third the phase $\phi = \arg(Z)$. Plot the experimental value in Bode plot. Add the fit value of $|Z|$.
Hint: use the range range where the experimental fits the theoretical value.
2. The experimental value does not follow the theoretical value explicitly because of some parasite resistances. Capacitor model can be explained as
$$Z = \frac{1}{j\omega C} \text{ !/mkern-5mu/! } R_p + R_s$$
where R_p is parallel resistance, R_s series resistance, operator !/mkern-5mu/!
$$a \text{ !/mkern-5mu/! } b = \frac{1}{\frac{1}{a} + \frac{1}{b}}$$
, meaning a parallel circuit. Plot the experimental value in Bode plot. Add the fit value of $|Z|$.
Hint: if R_p is huge, it can be ignored.
3. Plot C vs file name to visually compare the extracted C value.

Exercise 2

Plan, code, and run your own program for your own experiments.

To study by yourself: codecademy



Learn Python 2

Learn the basics of the world's fastest growing and most popular programming language used by software engineers, analysts, data scientists, and machine learning engineers alike.

Join

4,719,900

people who have taken

*This is outdated Python 2 – `print()` function is a bit different from Python 3