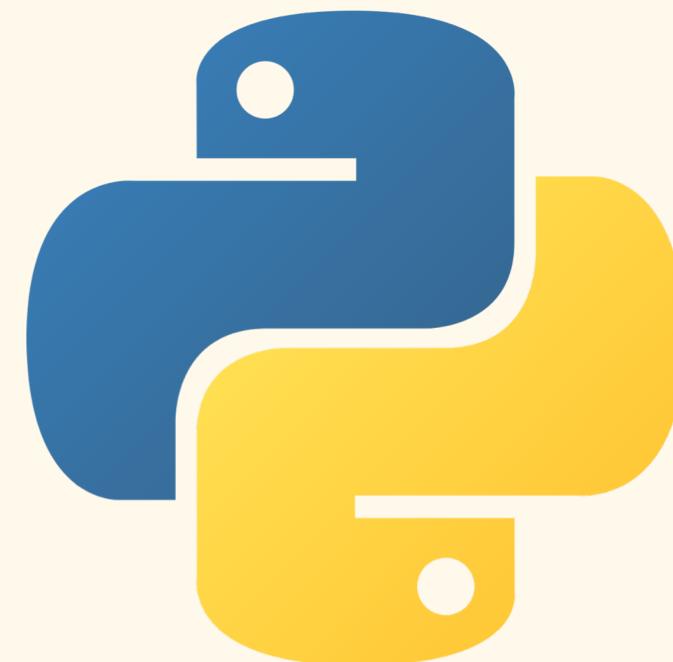


REU Python 3 Workshop

Brought to you by APS



July 23rd, 2020

Today's Agenda

Note: Speaker will go through the agenda. Main points are that the first half will be a seminar-like review of basic Python, then the second half will include three challenge problems. For the challenge problems, attendees will be broken into breakout rooms of about 4 people.

12:00 - 12:10 PM (ET)	<i>Introductions and Housekeeping</i>
12:10 - 1:00 PM (ET)	<i>Python 3 basics (Syntax, data structures, loops, and user-defined functions)</i>
1:00 - 1:10 PM (ET)	<i>[Break]</i>
1:10 - 1:25 PM (ET)	<i>Matplotlib Example with Breakout Room Work</i>
1:25 - 1:40 PM (ET)	<i>Scipy and Pandas Example with Breakout Room Work</i>
1:40 - 1:50 PM (ET)	<i>FITS Handling Example with Breakout Room Work</i>
1:50 - 2:00 PM (ET)	<i>Wrap Up</i>

Note: Speaker will show this slide, then switch to Jupyter Notebook and go through the contents of the APS-Python-Workshop-master folder. The speaker will use this time to explain how to use Jupyter Notebook.

What's in the Folder?

Note: Speaker will ask attendees to go to the “Part 1” folder and open the first notebook. Upon opening the notebook, the speaker will explain how to run notebook code and explain the purposes of markdown and code cells.

1_syntax.ipynb

Python Code Structure

Note: Speaker will go over this table of Python code structure

A line like #!/bin/bash
Comment with license and author information
Import packages
Define your own functions
Code that calls the functions

short comments sprinkled throughout

”

*Some really long comments too,
which are enclosed by triple quotes*

”

*print('Print functions whenever you want
to see code output')*

Note: Speaker will show this example and point out the elements from the previous slide

```
1 #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3 """
4 Created for APS Summer REU Python 3 Workshop
5
6 @author: victoriacatlett
7 """
8
9 #####
10 ##### Import what you need from packages at the top #####
11 #####
12
13 import numpy as np
14 import pandas as pd
15 import matplotlib.pyplot as plt
16 from scipy.optimize import curve_fit
17
18 #####
19 ##### Define a function we'll need later #####
20 #####
21
22 def mySin(x,a,b):
23     y = a*np.sin(b*x)
24     return y
25
26
27 #####
28 ##### TASK 1: Read in the Excel file OR the CSV file #####
29 #####
30
31 path_to_xlsx = '../../../../../files/pandas.xlsx'
32 #path_to_csv = '../../../../../files/pandas.csv'
33
34 data = # YOUR CODE HERE: Read in the data
35 x = # YOUR CODE HERE: Get the x data
36 y = # YOUR CODE HERE: Get the y data
37
38
39 #####
40 ##### TASK 2: Plot data (BONUS: Plot fit sine) #####
41 #####
42
43 # This creates two places to plot (ax[0] and ax[1])
44 # on a single figure called "fig"
45 fig, ax = plt.subplots(nrows=2, ncols=1)
46 plt.subplots_adjust(hspace=0.5)
47
48 # Set titles for the two plots
49 ax[0].set_title('Original Data')
50 ax[1].set_title('Fit Sine Curve')
```

Note: Speaker will ask attendees to go to the “Part 1” folder and open the second notebook

2_variables.ipynb

2 variables: Variables

Note: The speaker will explain that the next two slides are *reference slides* for this example, then they will switch to going through the Jupyter Notebook

Variables store values for future reference.

Variable Type	Definition	Examples
boolean	True or False	True, False
integer	A whole number <i>without</i> a decimal point	... -2, -1, 0, 1, 2 ...
float	Numbers <i>with</i> a decimal point	3.14, 100.0001, 7.
string	collections of characters surrounded by quotation marks	'Hello!', "1234", "etc."

Ex) Say you have a code that tells you how old you'll be in certain years. It'll need your current age.
To create a variable called `my_age` which stores your age, just type the line

`my_age = 20`

Python knows this is an integer, and now you don't have to type 20 everywhere!

2 variables: Data Structures

Data Structures can hold more than one value at once

Variable Type	Definition	Examples
list	A list of variables	<code>my_list = [1, 'two', 3.0]</code>
array	A matrix. (Essentially a list of lists.)	$\begin{matrix} \text{my_array} = [[1,2,3],[4,5,6],[7,8,9]] \\ \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix} \end{matrix}$
dictionary	A set of “keys” and their associated “values” (like a word and its definitions). Keys must be integers or strings.	<code>my_dictionary = {'Year':2019, 'School':'UTD', 'Some_Numbers':[1,2,3]}</code>

`my_list[0] = 1`

`my_array[1][2] = 6`

`my_dictionary['School'] = 'UTD'`

Note: Speaker will ask attendees to go to the “Part 1” folder and open the third notebook

3_functions.ipynb

3 functions: Built-In Functions

Note: The speaker will explain that the next five slides are *reference slides* for this example, then they will switch to going through the Jupyter Notebook

Function	Output
print(x)	Displays string x in the console
type(x)	The variable type of x
abs(x)	The absolute value of a number x
range(n)	The sequence {0, 1, ..., n-1}
list(range(n))	The list [0, 1, ..., n-1]

This is a short list of common functions you may encounter.
See online Python documentation for the full list.

3 functions: Boolean Operators

Boolean operators create an expression that is either
True or False.

Python Operator	Meaning
and	and
or	or
not	not

That's easy to remember! Here are some examples:

True **or** False = True

True **and** False = False

3 functions: Comparison Operators

Python Operator	Meaning
<	less than
<=	less than or equal to
>	greater than
>=	greater than or equal to
==	is equal to
!=	is not equal to

$7 == 8$ is False

$3 < 4$ is True

3 functions: Loops

For Loop: Repeats a process a specific number of times

```
forSum = 0  
  
for i in range(11):  
    forSum = forSum + i
```

While Loop: Repeats a process over and over until the given condition is false

```
whileSum = 0  
x = 0  
  
while x < 11:  
    whileSum = whileSum + x  
    x = x + 1
```

Both of these loops evaluate the sum of 0-10
Note: range(n) returns the list [0, 1, ..., n-1]

3 functions: User-Defined Functions

General Format

```
def functionName(inputs):  
    output = [do something to the inputs]  
    return output  
  
specificOutput = functionName(specificInputs)
```

Example: $y = mx+b$

```
def line(x, m, b):  
    y = m * x + b  
    return y  
  
y0 = line(0, 3, 2)
```

Note: Speaker will ask attendees to go to the “Part 1” folder and open the final notebook

4_libraries.ipynb

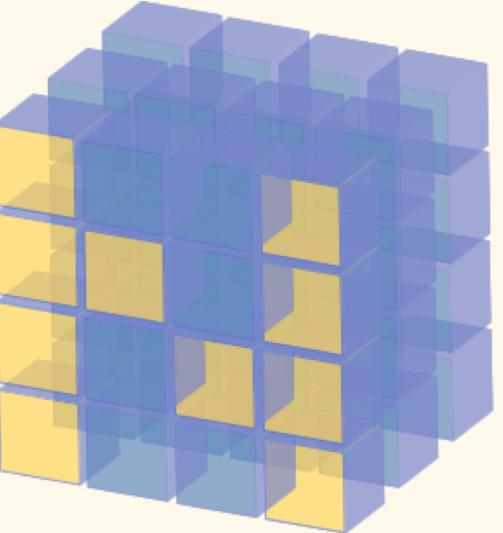
Mathematics Packages

NumPy and **SciPy** are packages for mathematics and computation.

You will almost *always* need NumPy, as it improves and adds to the built-in Python math.

SciPy contains many common scientific constants and functions.

Note: Speaker will go through the next three slides and provide brief descriptions of the packages. On this page, they will explain how these packages supplement the built-in Python math to provide a lot more functionality for STEM work



NumPy



SciPy

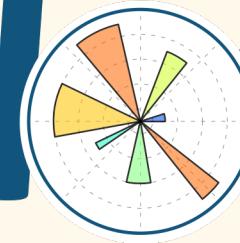
Plotting Packages

Matplotlib and **Seaborn** are two of the most common (and powerful) data visualization libraries.

We will only work with Matplotlib today.

Note: Speaker will note that these are two common packages to create plots, and that many people use them to make final plots for research papers or professional presentations. Matplotlib is the most common, but Seaborn is good to be aware of because it can make really beautiful data representations.

matplotlib



Pandas

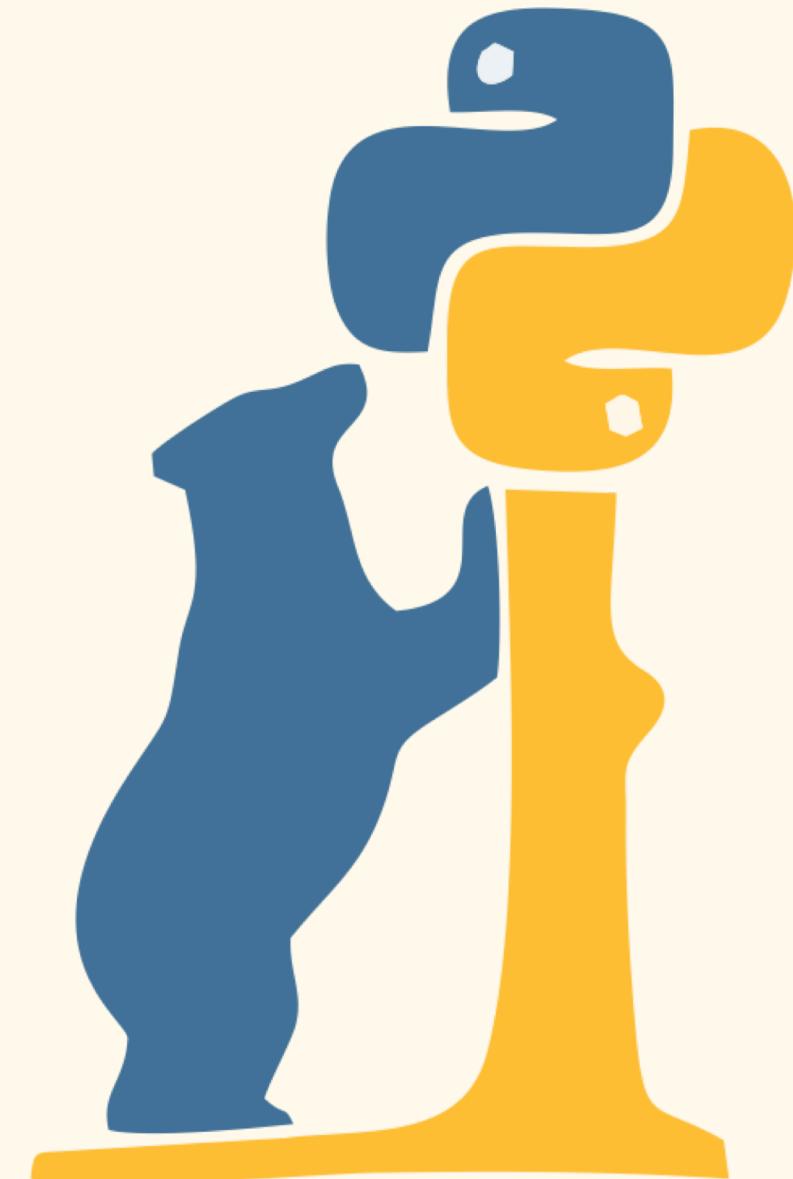
Note: Speaker will explain how pandas works with data in tabular form, that it, data with columns of the same length

Pandas is a powerful data analysis library which works with **Data Frames**.

Data Frames are data in a **table format**, such as Excel spreadsheets or Comma Separated Values (CSV) files.

	A	B	C
1	x	y	
2		0	0.04746958
3	0.01	0.23932802	
4	0.02	0.15274071	
5	0.03	0.16962677	
6	0.04	0.30859576	
7	0.05	0.38257724	
8	0.06	0.3292564	
9	0.07	0.38616885	
10	0.08	0.38549168	
11	0.09	0.43197485	
12	0.1	0.51758194	

```
index,x,y
0,0.0,0.04659049277507954
1,0.01,0.06642411315729269
2,0.02,0.13102361953897132
3,0.03,0.22867759086606038
4,0.04,0.3468186080452961
5,0.05,0.36043418819287615
6,0.06,0.3438464959254184
7,0.07,0.36819030282979937
8,0.08,0.38670691500471466
9,0.09,0.42290156458823214
10,0.1,0.48125929691252844
```



Note: Speaker will announce a 10-minute break

[Break]

Note: Speaker will welcome everyone back and ask everyone to open the “Part 2” folder. They will go over the contents of the folder (three breakout room examples and an extra one for people who work quickly)

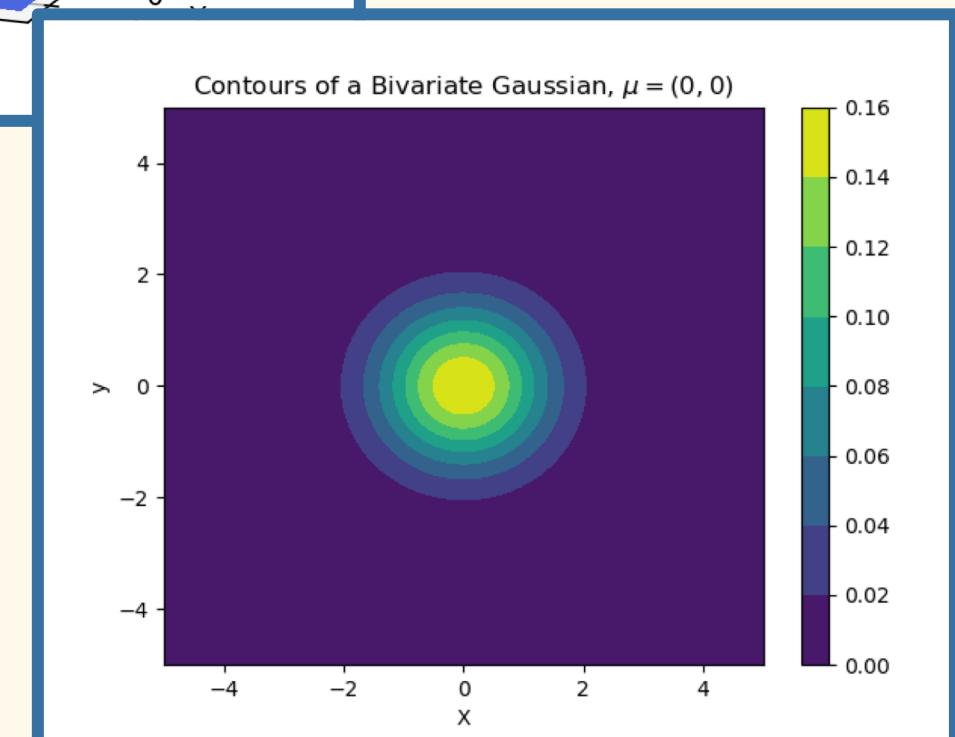
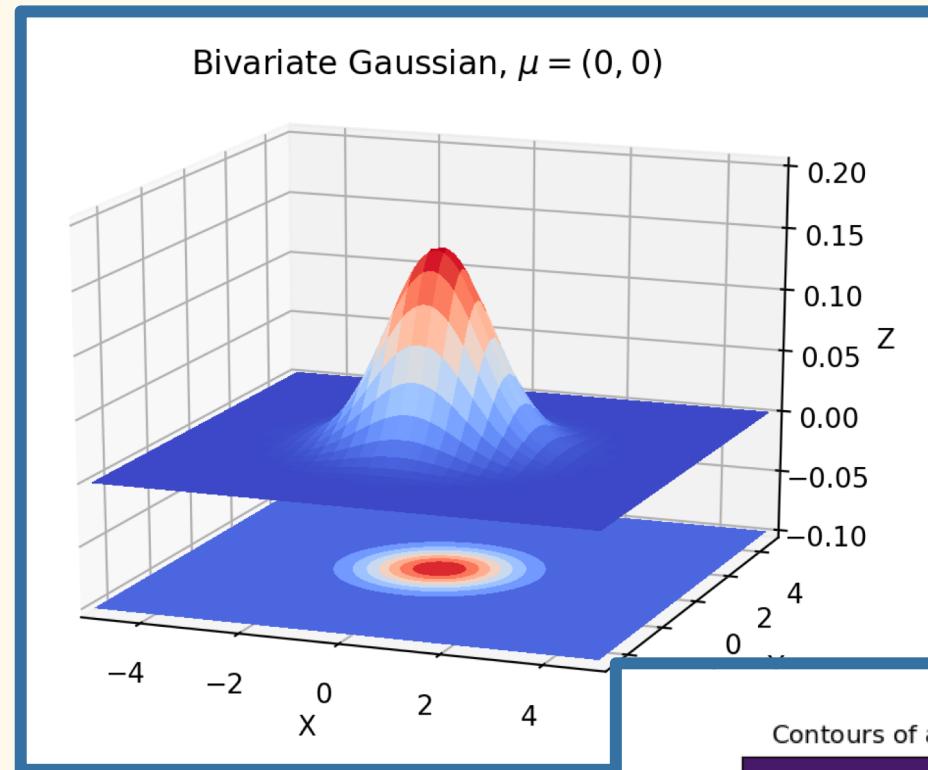
Part 2: Breakout Room Python Examples

extra_plot3d: Create 3D Plots

Note: Speaker will point out the extra example for people who work ahead

This example will guide you through:

- 1) Plotting a bivariate Gaussian
- 2) Plotting its x-y projection in 3D
- 3) Plotting its x-y projection in 2D



Note: Speaker will invite everyone to open the first example in the “Part 2” folder.

Challenge 1: 1_matplotlib.ipynb

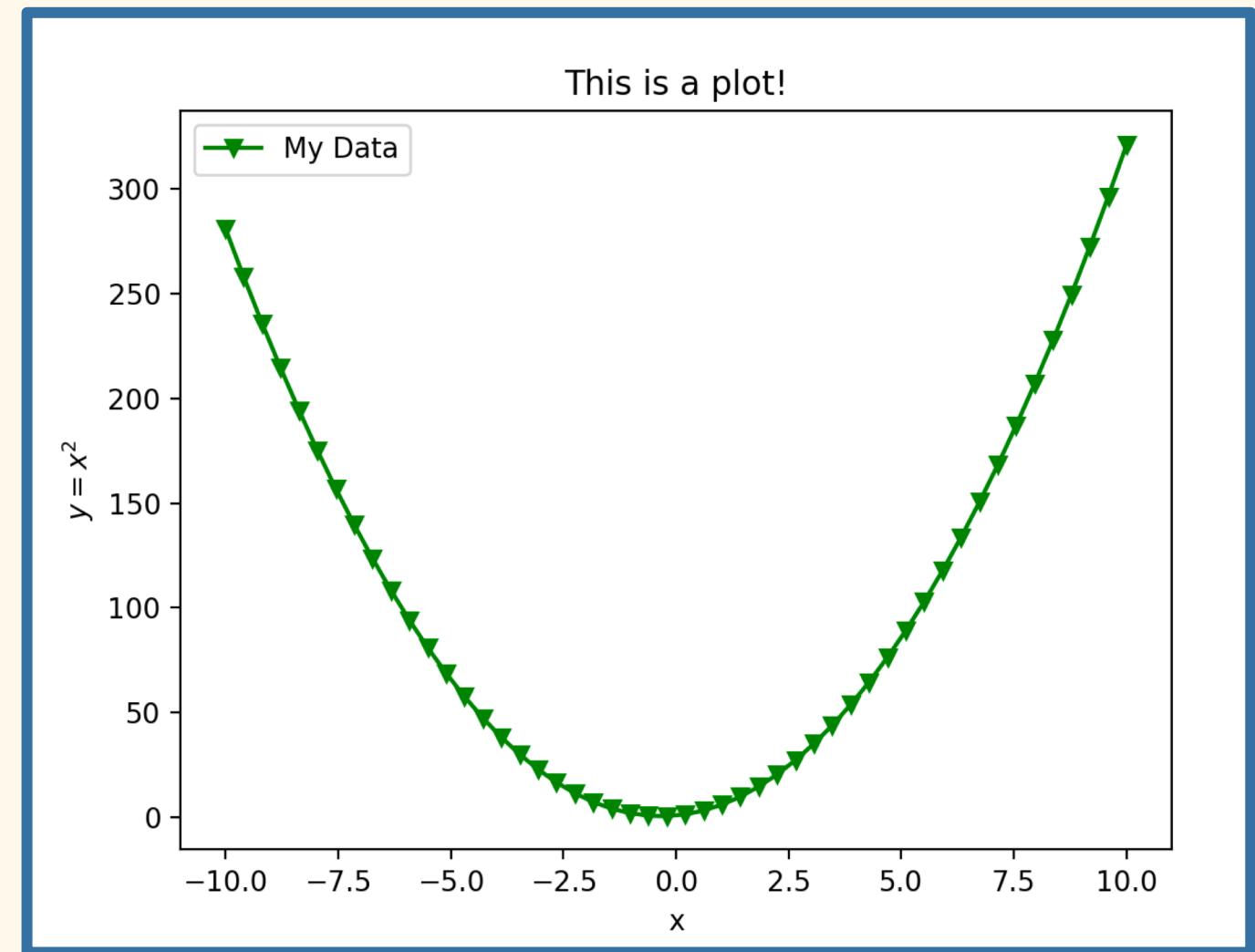
1 matplotlib: Plotting with Matplotlib

In this example, we will:

- 1) Plot the function
 $y = 3x^2 + 2x + 1$
- 2) Add an equation to an axis label
- 3) Save the figure as a PNG

Note: Speaker will quickly go over the goals of this challenge, then split everyone into breakout rooms.

After 5-10 minutes, the host will invite everyone back into the main room, and the speaker will ask people to raise their hand if they'd like to share their answer.



Note: Speaker will invite everyone to open the second example in the “Part 2” folder.

Challenge 2: **2_scipy_pandas.ipynb**

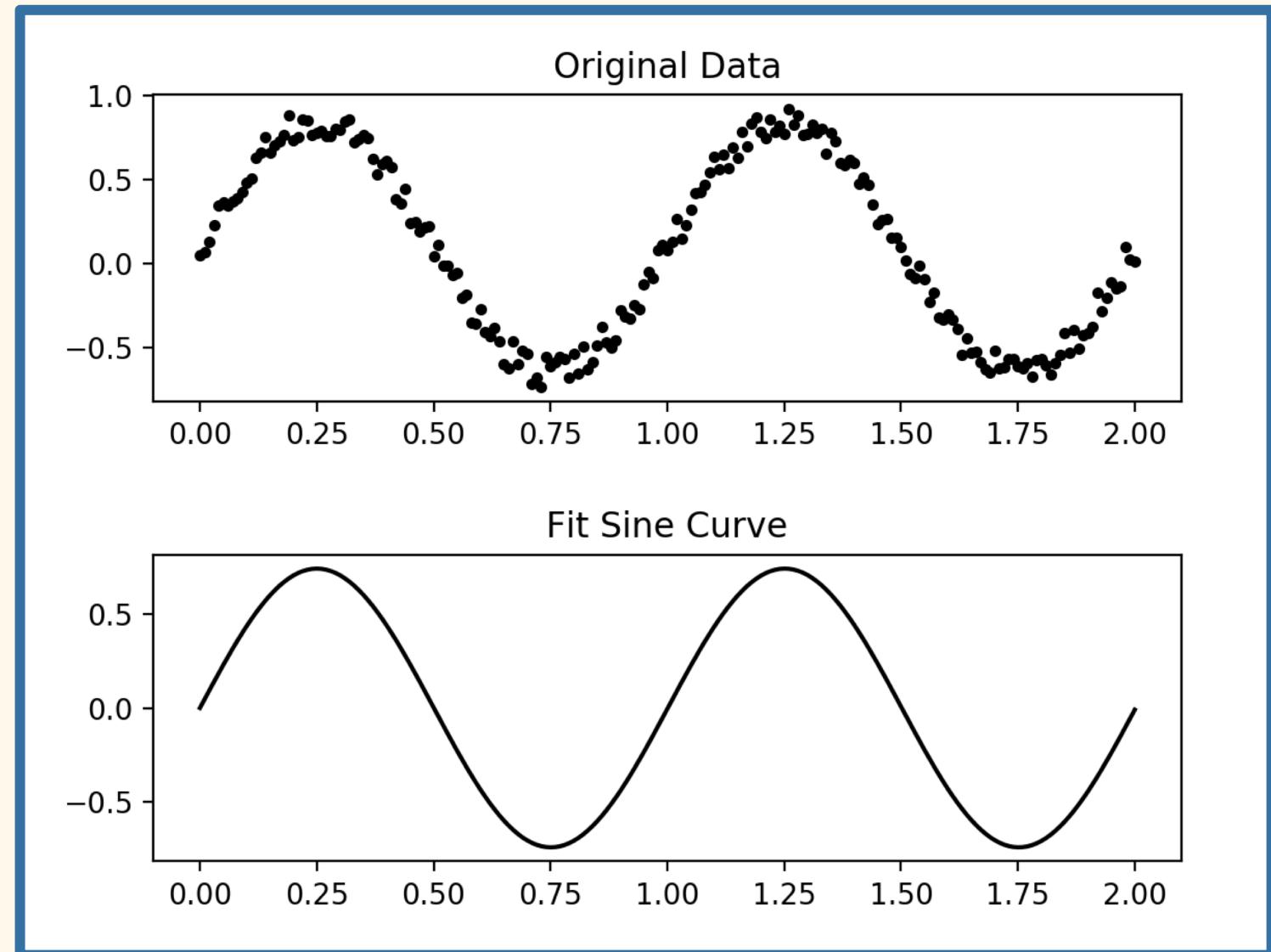
2 scipy_pandas

In this example, you will:

- 1) Pull data from an Excel sheet (or CSV)
- 2) Fit a sine curve to the data
- 3) Plot everything
- 4) Save the data to a new file

Note: Speaker will quickly go over the goals of this challenge, then split everyone into breakout rooms.

After 5-10 minutes, the host will invite everyone back into the main room, and the speaker will ask people to raise their hand if they'd like to share their answer.



Note: Speaker will invite everyone to open the third example in the “Part 2” folder.

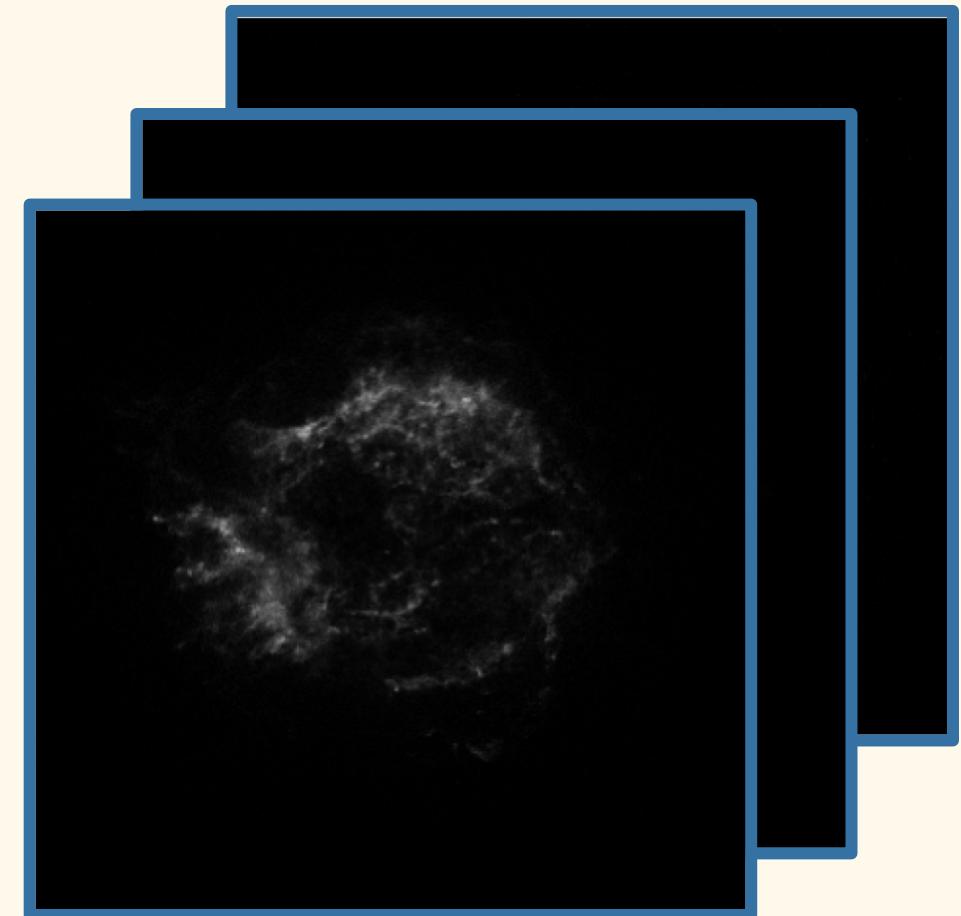
Challenge 3: **3_fits.ipynb**

Flexible Image Transport System (FITS)

FITS is the standard data type for astronomical images, and it can be used in a variety of other applications.

FITS files contain one or more Header Data Units (HDUs), which each contain an image (the data) and keyword descriptions (the header).

Essentially, the files are stacks of images which each have some text to describe them.



Note: FITS files are a fickle file type unique to astronomy, so the speaker will use this slide to explain what exactly they are.

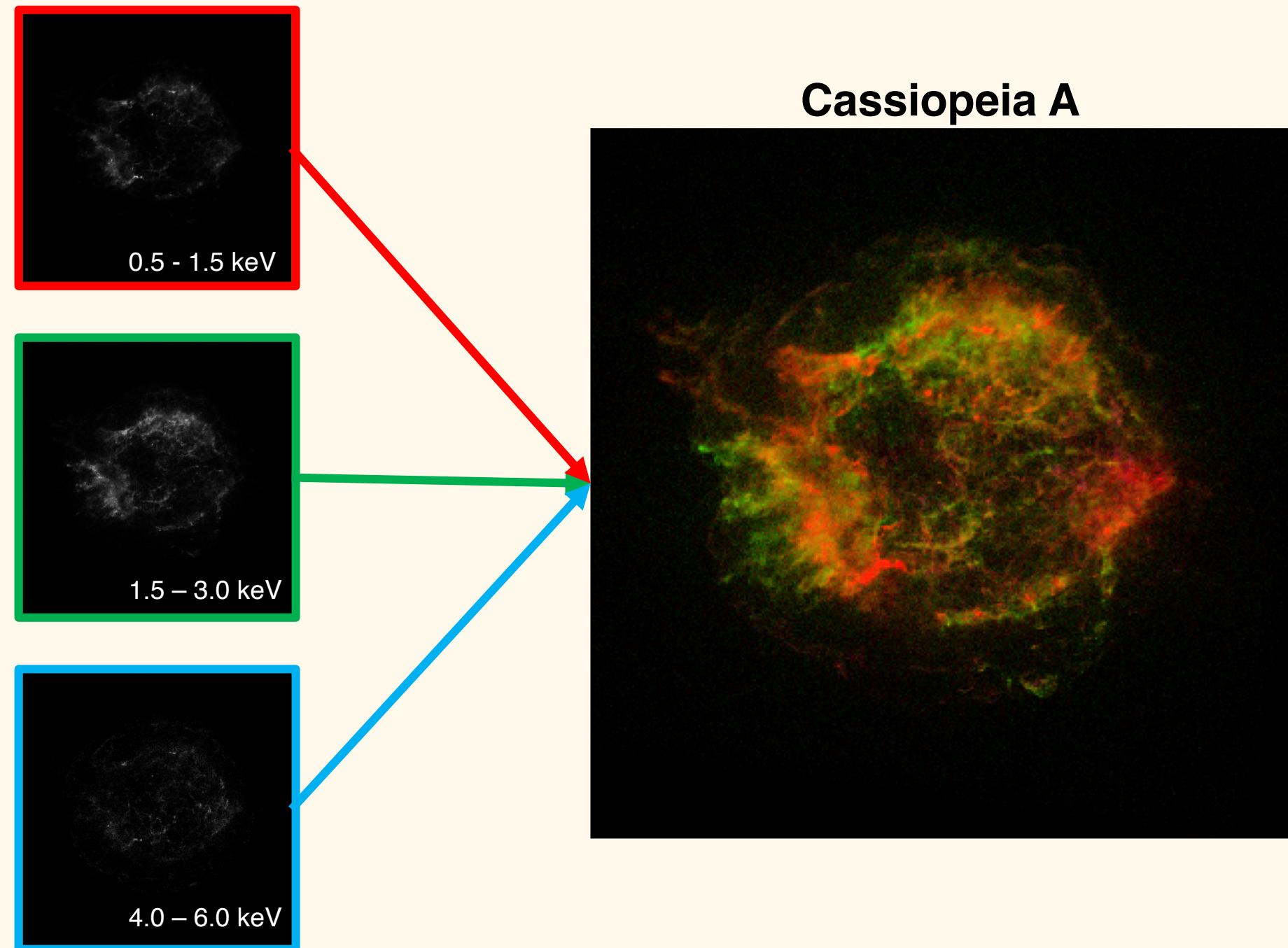
3 fits: Use Astropy to work with FITS

In this example, you will:

- 1) Open a FITS file with 3 HDUs
- 2) Read header information
- 3) Turn the 3 images into an RGB image

Note: Speaker will quickly go over the goals of this challenge, then split everyone into breakout rooms.

After 5-10 minutes, the host will invite everyone back into the main room, and the speaker will ask people to raise their hand if they'd like to share their answer.



Data Source: [Chandra Photo Album](#)

Note: Speaker will acknowledge the hard work of everyone who attended, then comment on how Python is capable of doing awesome things. The next few slides will show examples.

Wrap Up: Other Python Packages

Other Potentially Useful Packages



[TkInter](#), [Kivy](#), and [PyQt](#) will help you create cross-platform desktop applications



[OpenCV](#) is used for image recognition and processing, such as facial recognition or object detection.

Other Potentially Useful Packages



Scikits are toolkit extensions to Scipy. There are all sorts of them, one of the most notable being Scikit-learn, which is commonly used for machine learning.

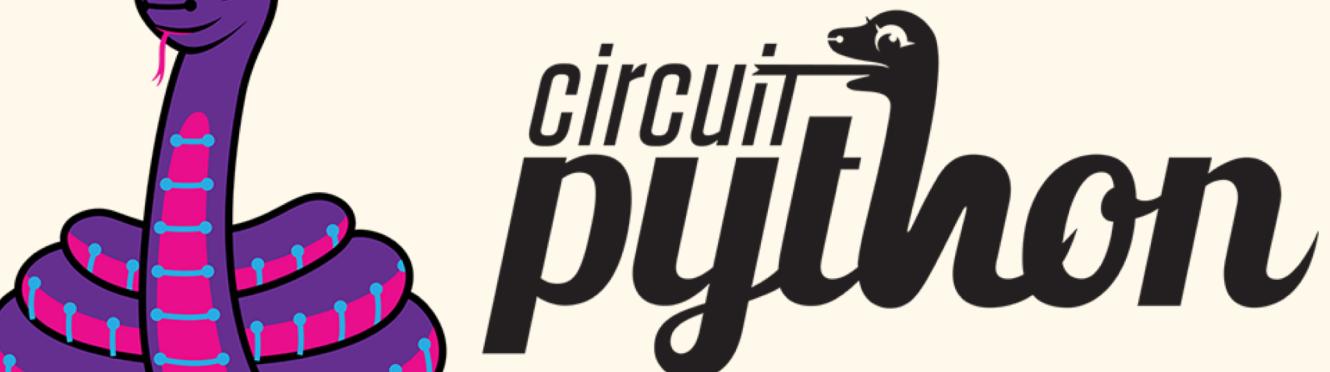
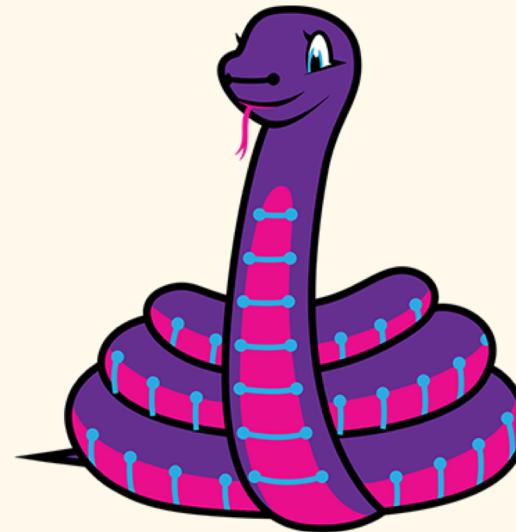


TensorFlow is another Python package for machine learning. It is especially useful for deep learning and making neural networks.

Fun Packages



[**Twilio**](#) allows you to send and receive phone calls and text messages via your program.



[**CircuitPython**](#) is almost exactly the same as Python, but it has support for hardware. It is produced by Adafruit, which makes compatible microcontrollers.

Thank you for joining this workshop!

If you want to reach out later, my email is
vec17000@utdallas.edu