Scientific Python for Matlab users

Antonio Ulloa, PhD

HPC @ NIH

antonio.ulloa@nih.gov

July 9, 2019

NIH National Institutes of Health
Turning Discovery Into Health

BIOWULF
HIGH PERFORMANCE COMPUTING AT THE NIH

Download these slides from:
https://hpc.nih.gov/training/handouts/matlab2python.pdf

Outline

- Goal
- Motivation
- Historical perspective
- Python vs Matlab
- Scientific Python
- Python environments
- Hands-on examples
- Conclusion

GOAL

Goal

- To provide Matlab users with a rough introduction to coding in Python (on Biowulf)

MOTIVATION

Motivation

- Limited Matlab licenses on Biowulf
- Matlab IDE too slow for working interactively
- Need to compile for launching batch jobs (compiler also needs a license)
- Useful to be familiar with other languages
- Need to share with more collaborators
- Difficult to use Matlab code after leaving NIH
- Need for more transferable skills

HISTORICAL PERSPECTIVE

History of Matlab

- 1970s: EISPACK (Matrix Eigensystem Package) and LINPACK (Linear Equation Package) are developed in Fortran at Argonne National Laboratory
- 1981: Cleve Moler at University of New Mexico develops interactive matrix calculator (Matrix Laboratory) in Fortran
- 1983-4: Matlab is re-written in C and Mathworks is founded
- 1984: PC-Matlab launches which includes functions, toolboxes and graphics
- 1985: Pro-Matlab (for Unix) launches

Source: A Brief History of Matlab (2018), by Cleve Moler, mathworks.com

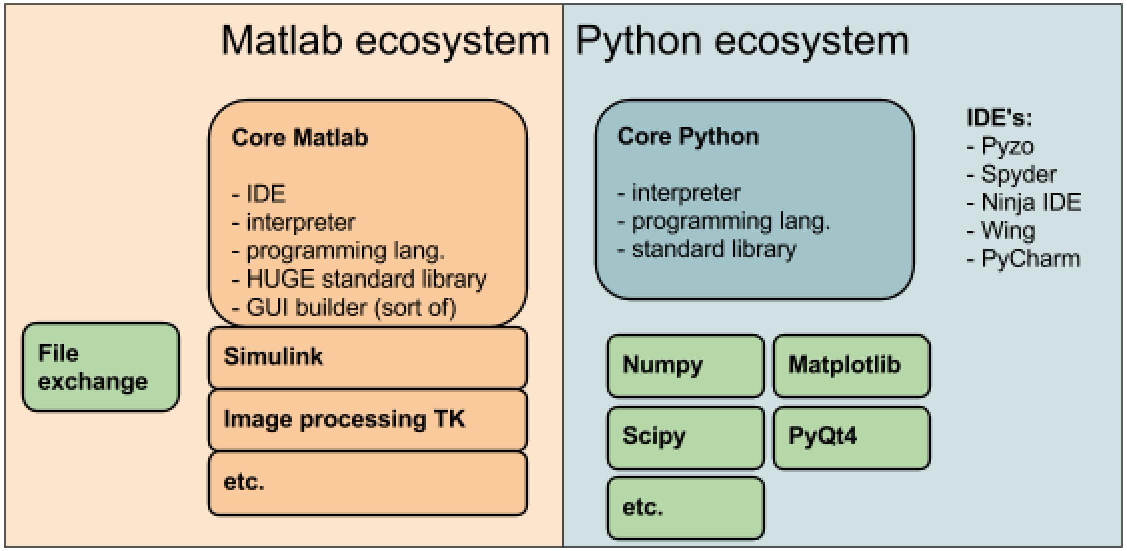
History of Python

- 1980s: Guido van Rossum develops Python at the Netherlands National Research Institute of Mathematics and Computer Science (named after Monty Python!)
- 1991: Python 0.9.0 is released which includes classes, list and strings
- 2000: Python 2.0 is released which includes list comprehensions
- 2008: Python 3.0 is released which is not backward compatible with Python 2.x
- 1985: Pro-Matlab (for Unix) launches

Source: A Brief History of Python (2018), by John Wolfe, medium.com

PYTHON VS MATLAB

Python vs Matlab



Matlab ecosystem

Core Matlab

- IDE
- interpreter
- programming lang.
- HUGE standard library
- GUI builder (sort of)

Simulink

Image processing TK

etc.

File exchange

Python ecosystem

Core Python

- interpreter
- programming lang.
- standard library

IDE's:

- Pyzo
- Spyder
- Ninja IDE
- Wing
- PyCharm

Numpy

Matplotlib

Scipy

PyQt4

etc.

Source: www.pyzo.org

Python vs Matlab

Programming language	Numerical computing environment
Free	Licenses have to be purchased
Might need additional packages (also free)	Might need toolboxes (some free, some have to be purchased)
Does not come with Integrated Development Environment (IDE)	Includes IDE
Open source	Proprietary
Can be executed as batch on Biowulf cluster directly	Needs to be compiled to be executed as batch on Biowulf cluster

SCIENTIFIC PYTHON

Scientific Python

- Python has a large community of user scientists, with easy to find help (e.g., stackoverflow).
- Availability of scientific libraries (e.g., numpy, scipy, matplotlib)
- Availability of IDEs to choose from (e.g., Spyder, pyCharm)

Source: Scientific Python Lectures (2016), by Robert Johansson [github.com/jrjohansson/scientific-python-lectures]

Scientific Python

NumPy

Base N-dimensional array package

SciPy library

Fundamental library for scientific computing

Matplotlib

Comprehensive 2D Plotting

Source: Scipy.org

Scientific Python

NumPy

Base N-dimensional array package

SciPy library

Fundamental library for scientific computing

Matplotlib

Comprehensive 2D Plotting

Source: Scipy.org

Numpy

- Python library that contains multidimensional arrays, matrices, and other objects
- It also contains functions to perform operations on arrays
- A large number of scientific computing applications in Python use Numpy
- Zero-based indexing (Matlab uses 1-based indexing

Source: docs.scipy.org/doc/numpy/user

Scientific Python

NumPy

Base N-dimensional array package

SciPy library

Fundamental library for scientific computing

Matplotlib

Comprehensive 2D Plotting

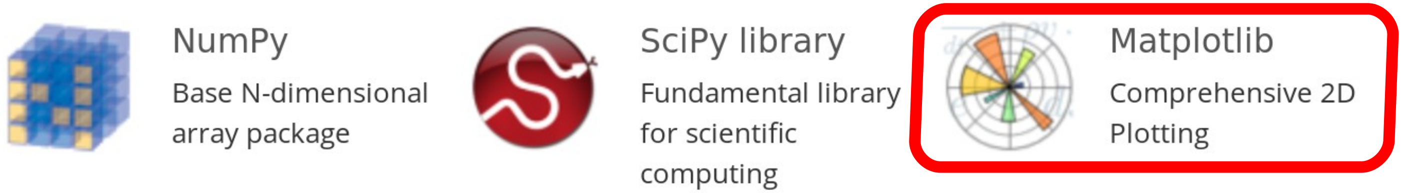
Source: Scipy.org

Scipy

- Scipy is built on top of numpy and contains many scientific computing functions
- Those examples of the areas those functions solve are: integration, optimization, interpolation, fourier transforms, signal processing, linear algebra, statistics, image processing, file Input/Ouput, etc.

Source: Scientific Python Lectures (2016), by Robert Johansson [github.com/jrjohansson/scientific-python-lectures]

Scientific Python



NumPy

Base N-dimensional array package

SciPy library

Fundamental library for scientific computing

Matplotlib

Comprehensive 2D Plotting

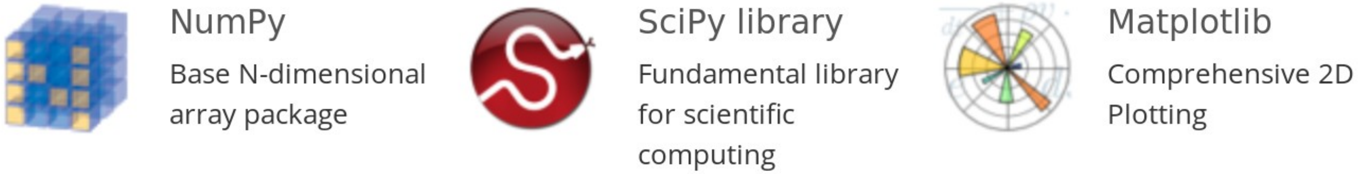
Source: Scipy.org

Matplotlib

- Matplotlib is a Python library for 2D plotting (and simple 3D plotting)
- For basic plotting one can use Matplotlib's module pyplot (similar to Matlab plotting)
- Some of the plots you can create with Matplotlib are line plots, image display, histograms, bar and pie charts, scatter plots, log plots, polar plots, etc

Source: matplotlib.org

Scientific Python



NumPy

Base N-dimensional array package

SciPy library

Fundamental library for scientific computing

Matplotlib

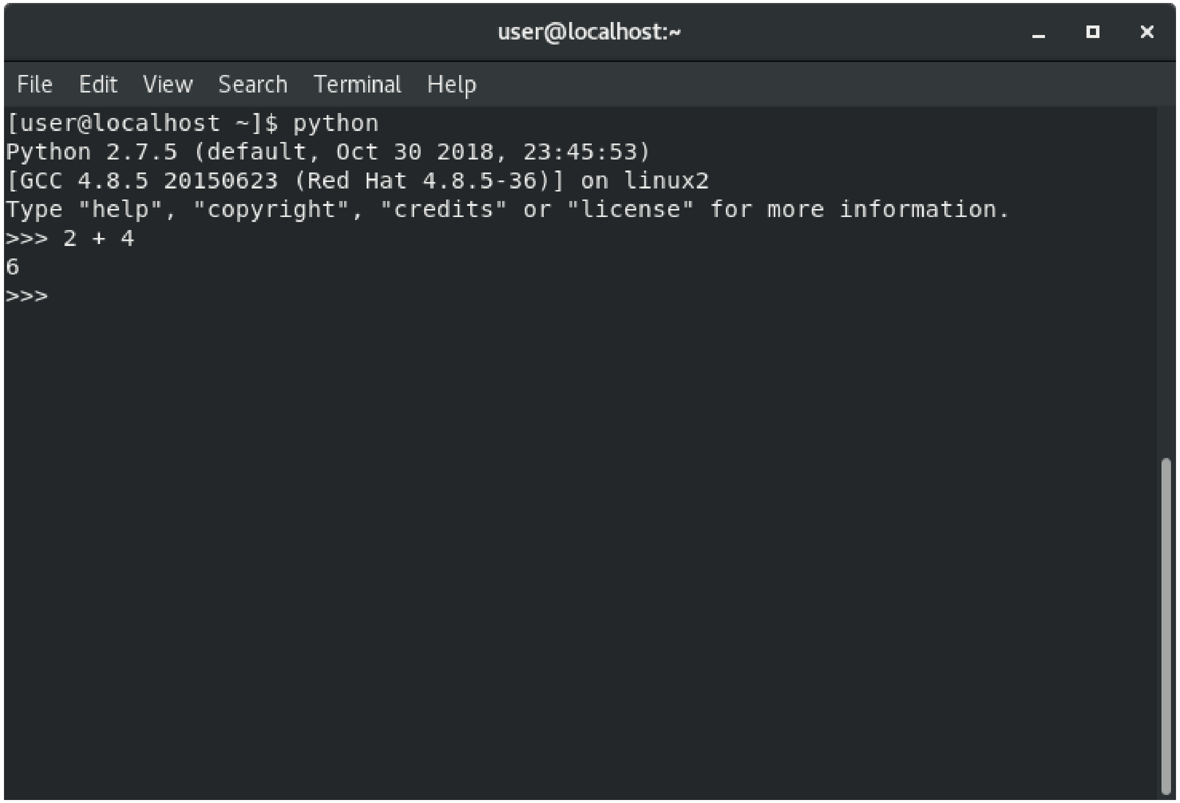
Comprehensive 2D Plotting

Python itself does not come with an IDE, but one can choose from a number of them available as a free download

Source: Scipy.org, Spyder-ide.org

PYTHON ENVIRONMENTS

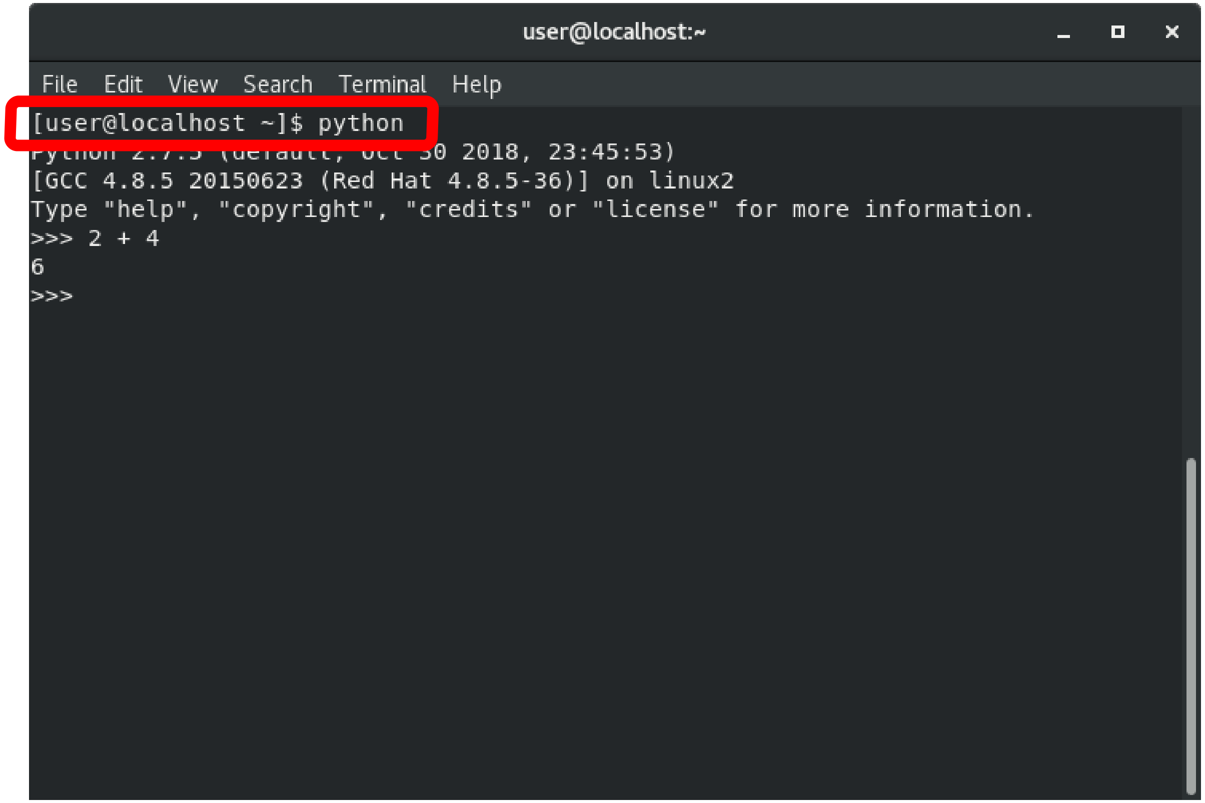
Python interpreter



```
[user@localhost ~]$ python
Python 2.7.5 (default, Oct 30 2018, 23:45:53)
[GCC 4.8.5 20150623 (Red Hat 4.8.5-36)] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> 2 + 4
6
>>>
```

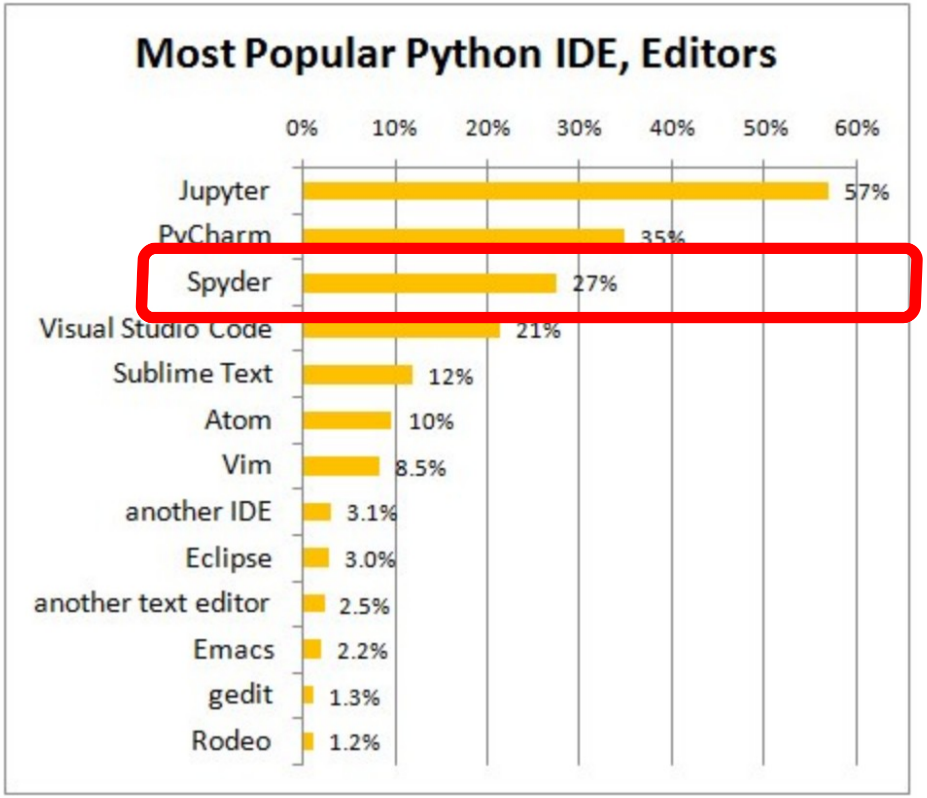
Python interpreter

Just type “python” in your terminal to have access to the interpreter



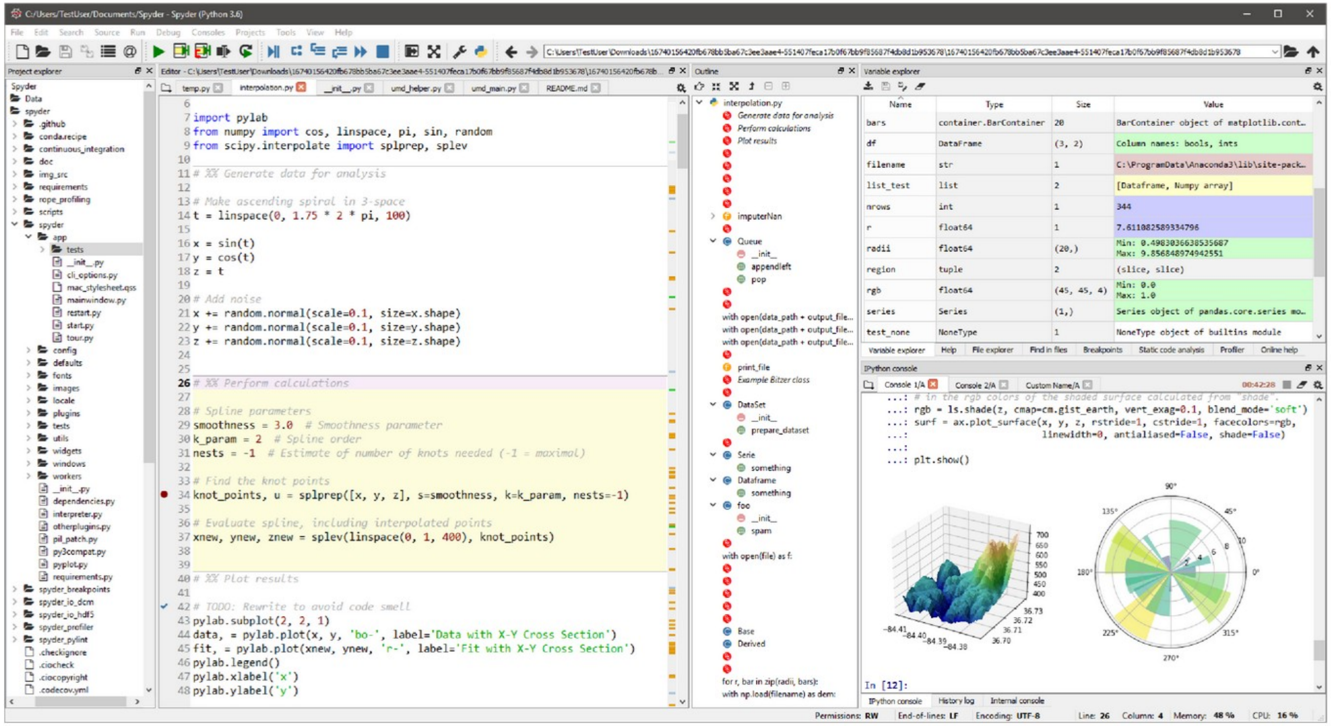
```
[user@localhost ~]$ python
Python 2.7.5 (default, Oct 30 2018, 23:45:53)
[GCC 4.8.5 20150623 (Red Hat 4.8.5-36)] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> 2 + 4
6
>>>
```

Integrated Development Environments



Source: Most popular Python IDEs / Editors, 2018, by Gregory Platetsky, Kdnuggets, https://www.kdnuggets.com/2018/12/most-popular-python-ide-editor.html

Spyder IDE



```python

import pylab
from numpy import cos, linspace, pi, sin, random
from scipy.interpolate import splprep, splev

# %% Generate data for analysis

# Make ascending spiral in 3-space
t = linspace(0, 1.75 * 2 * pi, 100)

x = sin(t)
y = cos(t)
z = t

# Add noise
x += random.normal(scale=0.1, size=x.shape)
y += random.normal(scale=0.1, size=y.shape)
z += random.normal(scale=0.1, size=z.shape)


# %% Perform calculations

# Spline parameters
smoothness = 3.0  # Smoothness parameter
k_param = 2  # Spline order
nests = -1  # Estimate of number of knots needed (-1 = maximal)

# Find the knot points
knot_points, u = splprep([x, y, z], s=smoothness, k=k_param, nests=-1)

# Evaluate spline, including interpolated points
xnew, ynew, znew = splev(linspace(0, 1, 400), knot_points)


# %% Plot results

# TODO: Rewrite to avoid code smell
pylab.subplot(2, 2, 1)
data, = pylab.plot(x, y, 'bo-', label='Data with X-Y Cross Section')
fit, = pylab.plot(xnew, ynew, 'r-', label='Fit with X-Y Cross Section')
pylab.legend()
pylab.xlabel('x')
pylab.ylabel('y')
```

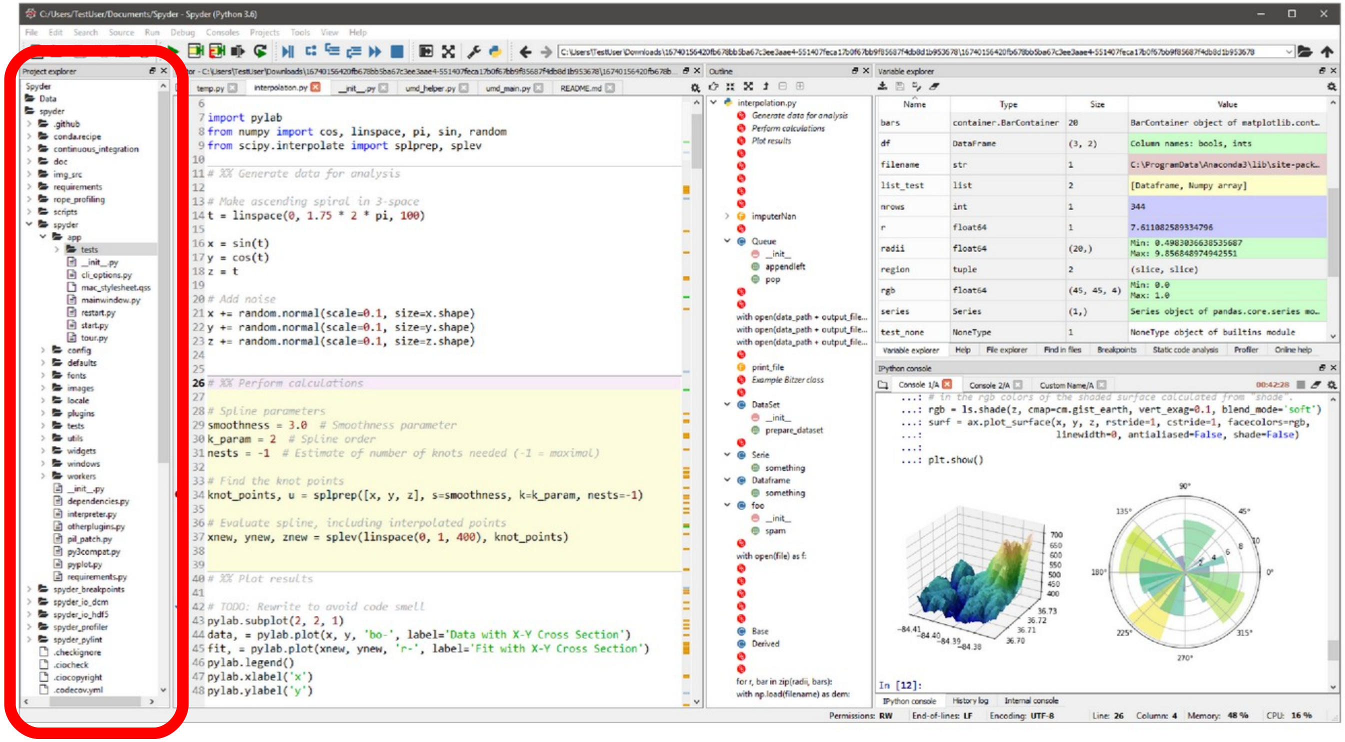
Name	Type	Size	Value
bars	container.BarContainer	20	BarContainer object of matplotlib.cont…
df	DataFrame	(3, 2)	Column names: bools, ints
filename	str	1	C:\ProgramData\Anaconda3\lib\site-pack…
list_test	list	2	[Dataframe, Numpy array]
nrows	int	1	344
r	float64	1	7.611082589334796
radii	float64	(20,)	Min: 0.4983036638535687 Max: 9.856848974942551
region	tuple	2	(slice, slice)
rgb	float64	(45, 45, 4)	Min: 0.0 Max: 1.0
series	Series	(1,)	Series object of pandas.core.series mo…
test_none	NoneType	1	NoneType object of builtins module

```
    ...: # in the rgb colors of the shaded surface calculated from "shade".
    ...: rgb = ls.shade(z, cmap=cm.gist_earth, vert_exag=0.1, blend_mode='soft')
    ...: surf = ax.plot_surface(x, y, z, rstride=1, cstride=1, facecolors=rgb,
    ...:                        linewidth=0, antialiased=False, shade=False)
    ...:
    ...: plt.show()

In [12]:
```

Source: Spyder-ide.org

Spyder IDE

File explorer



```python

import pylab
from numpy import cos, linspace, pi, sin, random
from scipy.interpolate import splprep, splev

# %% Generate data for analysis

# Make ascending spiral in 3-space
t = linspace(0, 1.75 * 2 * pi, 100)

x = sin(t)
y = cos(t)
z = t

# Add noise
x += random.normal(scale=0.1, size=x.shape)
y += random.normal(scale=0.1, size=y.shape)
z += random.normal(scale=0.1, size=z.shape)


# %% Perform calculations

# Spline parameters
smoothness = 3.0  # Smoothness parameter
k_param = 2  # Spline order
nests = -1  # Estimate of number of knots needed (-1 = maximal)

# Find the knot points
knot_points, u = splprep([x, y, z], s=smoothness, k=k_param, nests=-1)

# Evaluate spline, including interpolated points
xnew, ynew, znew = splev(linspace(0, 1, 400), knot_points)


# %% Plot results

# TODO: Rewrite to avoid code smell
pylab.subplot(2, 2, 1)
data, = pylab.plot(x, y, 'bo-', label='Data with X-Y Cross Section')
fit, = pylab.plot(xnew, ynew, 'r-', label='Fit with X-Y Cross Section')
pylab.legend()
pylab.xlabel('x')
pylab.ylabel('y')
```

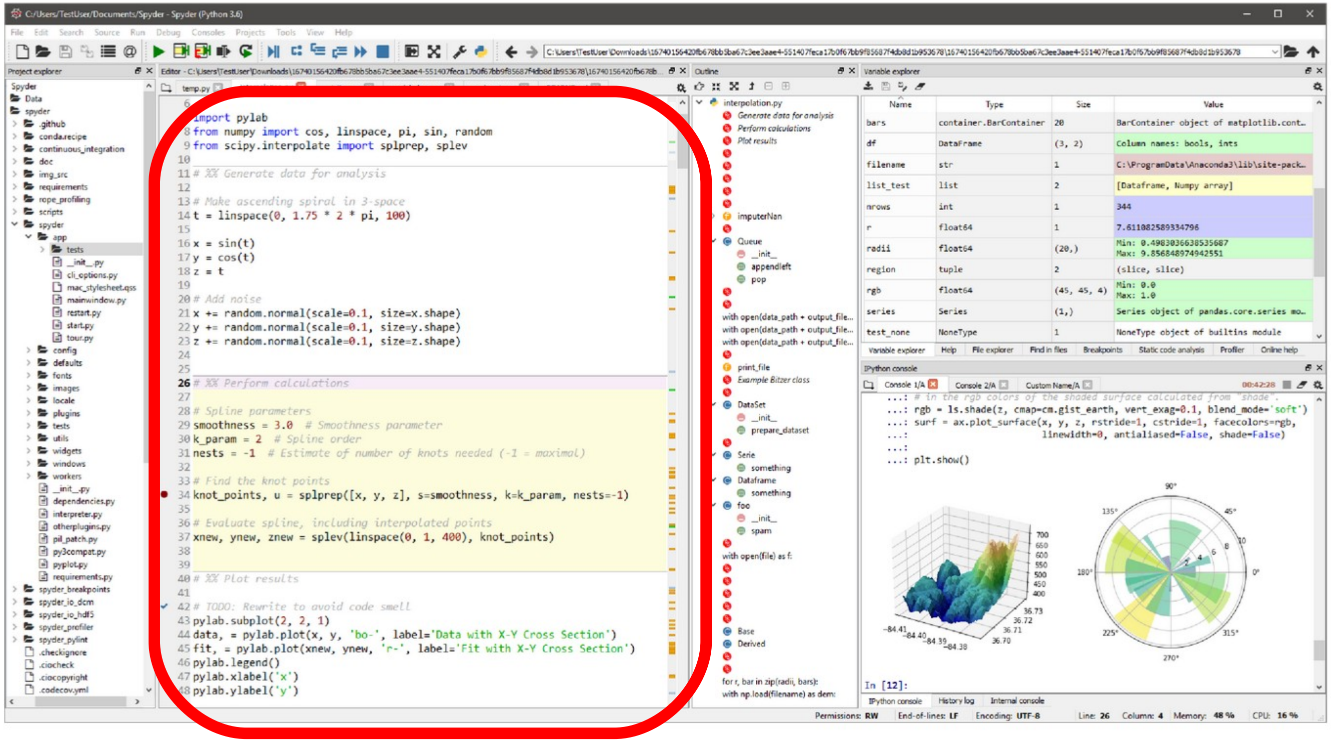
Name	Type	Size	Value
bars	container.BarContainer	20	BarContainer object of matplotlib.cont…
df	DataFrame	(3, 2)	Column names: bools, ints
filename	str	1	C:\ProgramData\Anaconda3\lib\site-pack…
list_test	list	2	[Dataframe, Numpy array]
nrows	int	1	344
r	float64	1	7.611082589334796
radii	float64	(20,)	Min: 0.4983036638535687 Max: 9.856848974942551
region	tuple	2	(slice, slice)
rgb	float64	(45, 45, 4)	Min: 0.0 Max: 1.0
series	Series	(1,)	Series object of pandas.core.series mo…
test_none	NoneType	1	NoneType object of builtins module

```
   ...: # in the rgb colors of the shaded surface calculated from "shade".
   ...: rgb = ls.shade(z, cmap=cm.gist_earth, vert_exag=0.1, blend_mode='soft')
   ...: surf = ax.plot_surface(x, y, z, rstride=1, cstride=1, facecolors=rgb,
   ...:                        linewidth=0, antialiased=False, shade=False)
   ...:
   ...: plt.show()

In [12]:
```

Source: Spyder-ide.org

Spyder IDE

Code editor



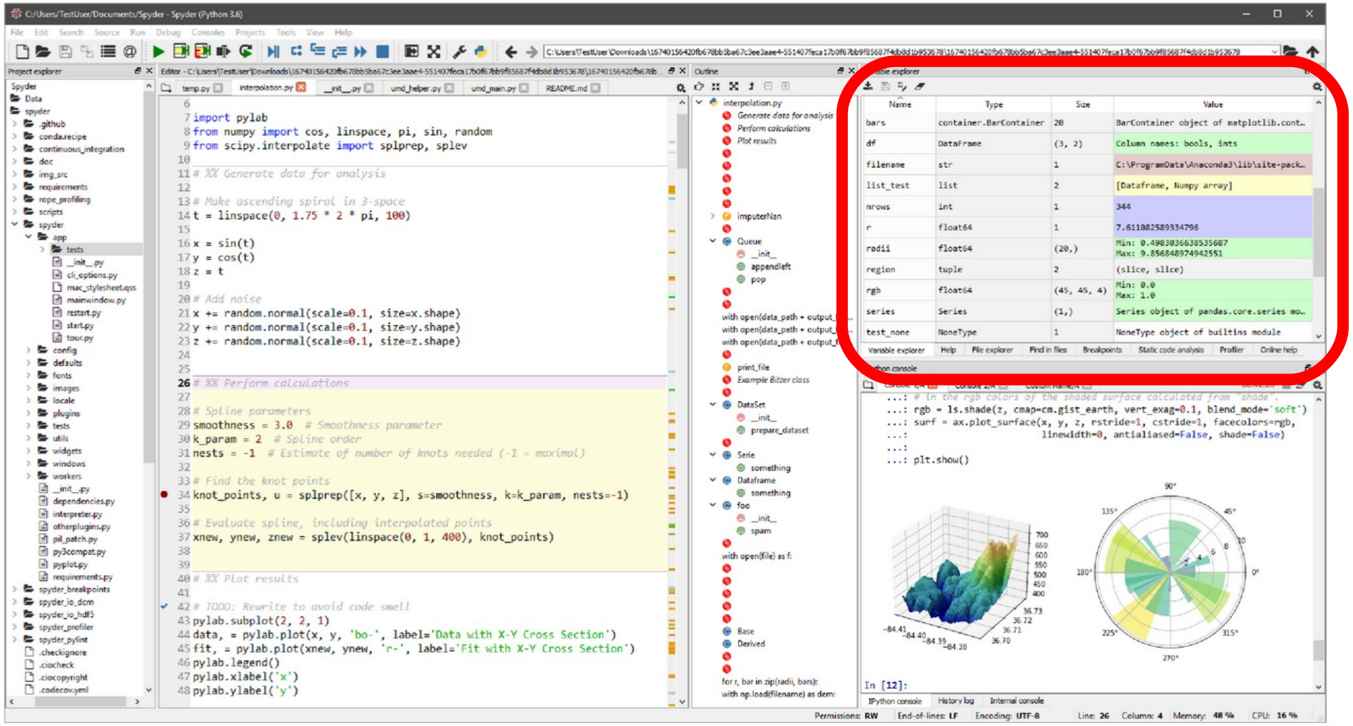
```python

import pylab
from numpy import cos, linspace, pi, sin, random
from scipy.interpolate import splprep, splev

# %% Generate data for analysis

# Make ascending spiral in 3-space
t = linspace(0, 1.75 * 2 * pi, 100)

x = sin(t)
y = cos(t)
z = t

# Add noise
x += random.normal(scale=0.1, size=x.shape)
y += random.normal(scale=0.1, size=y.shape)
z += random.normal(scale=0.1, size=z.shape)


# %% Perform calculations

# Spline parameters
smoothness = 3.0  # Smoothness parameter
k_param = 2  # Spline order
nests = -1  # Estimate of number of knots needed (-1 = maximal)

# Find the knot points
knot_points, u = splprep([x, y, z], s=smoothness, k=k_param, nests=-1)

# Evaluate spline, including interpolated points
xnew, ynew, znew = splev(linspace(0, 1, 400), knot_points)


# %% Plot results

# TODO: Rewrite to avoid code smell
pylab.subplot(2, 2, 1)
data, = pylab.plot(x, y, 'bo-', label='Data with X-Y Cross Section')
fit, = pylab.plot(xnew, ynew, 'r-', label='Fit with X-Y Cross Section')
pylab.legend()
pylab.xlabel('x')
pylab.ylabel('y')
```

Name	Type	Size	Value
bars	container.BarContainer	20	BarContainer object of matplotlib.cont…
df	DataFrame	(3, 2)	Column names: bools, ints
filename	str	1	C:\ProgramData\Anaconda3\lib\site-pack…
list_test	list	2	[Dataframe, Numpy array]
nrows	int	1	344
r	float64	1	7.611082589334796
radii	float64	(20,)	Min: 0.4983036638535687 Max: 9.856848974942551
region	tuple	2	(slice, slice)
rgb	float64	(45, 45, 4)	Min: 0.0 Max: 1.0
series	Series	(1,)	Series object of pandas.core.series mo…
test_none	NoneType	1	NoneType object of builtins module

```
   ...: # in the rgb colors of the shaded surface calculated from "shade".
   ...: rgb = ls.shade(z, cmap=cm.gist_earth, vert_exag=0.1, blend_mode='soft')
   ...: surf = ax.plot_surface(x, y, z, rstride=1, cstride=1, facecolors=rgb,
   ...:                        linewidth=0, antialiased=False, shade=False)
   ...:
   ...: plt.show()

In [12]:
```

Source: Spyder-ide.org

Spyder IDE



```python

import pylab
from numpy import cos, linspace, pi, sin, random
from scipy.interpolate import splprep, splev

# %% Generate data for analysis

# Make ascending spiral in 3-space
t = linspace(0, 1.75 * 2 * pi, 100)

x = sin(t)
y = cos(t)
z = t

# Add noise
x += random.normal(scale=0.1, size=x.shape)
y += random.normal(scale=0.1, size=y.shape)
z += random.normal(scale=0.1, size=z.shape)


# %% Perform calculations

# Spline parameters
smoothness = 3.0  # Smoothness parameter
k_param = 2  # Spline order
nests = -1  # Estimate of number of knots needed (-1 = maximal)

# Find the knot points
knot_points, u = splprep([x, y, z], s=smoothness, k=k_param, nests=-1)

# Evaluate spline, including interpolated points
xnew, ynew, znew = splev(linspace(0, 1, 400), knot_points)


# %% Plot results

# TODO: Rewrite to avoid code smell
pylab.subplot(2, 2, 1)
data, = pylab.plot(x, y, 'bo-', label='Data with X-Y Cross Section')
fit, = pylab.plot(xnew, ynew, 'r-', label='Fit with X-Y Cross Section')
pylab.legend()
pylab.xlabel('x')
pylab.ylabel('y')
```

Name	Type	Size	Value
bars	container.BarContainer	20	BarContainer object of matplotlib.cont…
df	DataFrame	(3, 2)	Column names: bools, ints
filename	str	1	C:\ProgramData\Anaconda3\lib\site-pack…
list_test	list	2	[Dataframe, Numpy array]
nrows	int	1	344
r	float64	1	7.611082589334796
radii	float64	(20,)	Min: 0.4983036638535687 Max: 9.856848974942551
region	tuple	2	(slice, slice)
rgb	float64	(45, 45, 4)	Min: 0.0 Max: 1.0
series	Series	(1,)	Series object of pandas.core.series mo…
test_none	NoneType	1	NoneType object of builtins module

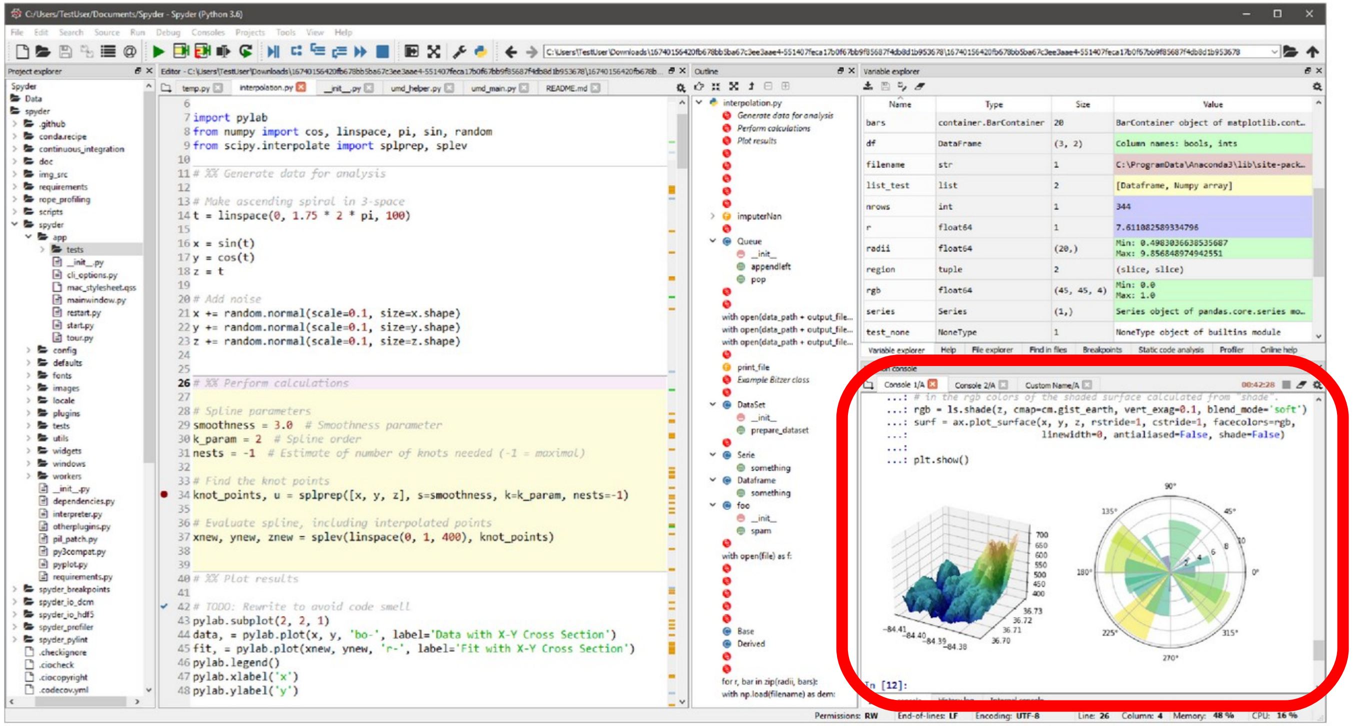
```
   ...: # in the rgb colors of the shaded surface calculated from "shade".
   ...: rgb = ls.shade(z, cmap=cm.gist_earth, vert_exag=0.1, blend_mode='soft')
   ...: surf = ax.plot_surface(x, y, z, rstride=1, cstride=1, facecolors=rgb,
   ...:                        linewidth=0, antialiased=False, shade=False)
   ...:
   ...: plt.show()

In [12]:
```

Variable explorer

Source: Spyder-ide.org

NIH BIOWULF

Spyder IDE



```python
import pylab
from numpy import cos, linspace, pi, sin, random
from scipy.interpolate import splprep, splev

# %% Generate data for analysis

# Make ascending spiral in 3-space
t = linspace(0, 1.75 * 2 * pi, 100)

x = sin(t)
y = cos(t)
z = t

# Add noise
x += random.normal(scale=0.1, size=x.shape)
y += random.normal(scale=0.1, size=y.shape)
z += random.normal(scale=0.1, size=z.shape)


# %% Perform calculations

# Spline parameters
smoothness = 3.0  # Smoothness parameter
k_param = 2  # Spline order
nests = -1  # Estimate of number of knots needed (-1 = maximal)

# Find the knot points
knot_points, u = splprep([x, y, z], s=smoothness, k=k_param, nests=-1)

# Evaluate spline, including interpolated points
xnew, ynew, znew = splev(linspace(0, 1, 400), knot_points)


# %% Plot results

# TODO: Rewrite to avoid code smell
pylab.subplot(2, 2, 1)
data, = pylab.plot(x, y, 'bo-', label='Data with X-Y Cross Section')
fit, = pylab.plot(xnew, ynew, 'r-', label='Fit with X-Y Cross Section')
pylab.legend()
pylab.xlabel('x')
pylab.ylabel('y')
```

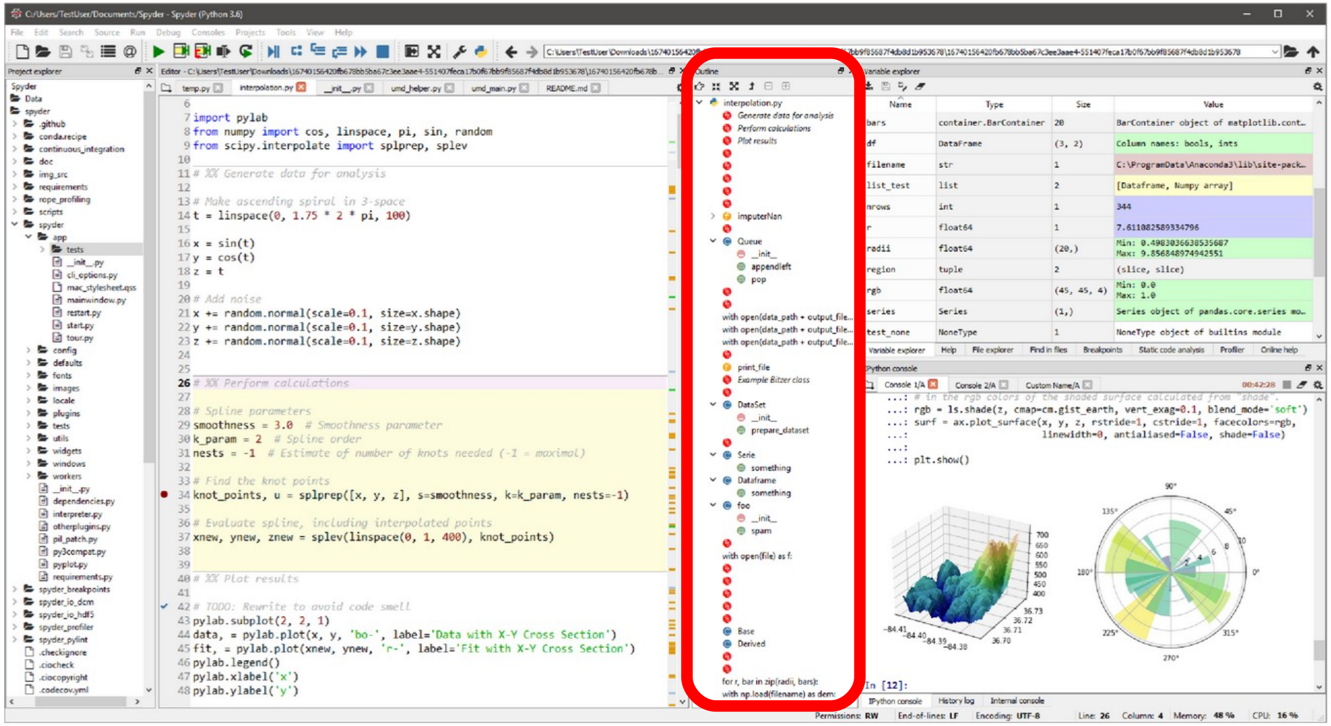
Name	Type	Size	Value
bars	container.BarContainer	20	BarContainer object of matplotlib.cont…
df	DataFrame	(3, 2)	Column names: bools, ints
filename	str	1	C:\ProgramData\Anaconda3\lib\site-pack…
list_test	list	2	[Dataframe, Numpy array]
nrows	int	1	344
r	float64	1	7.611082589334796
radii	float64	(20,)	Min: 0.4983036638535687 Max: 9.856848974942551
region	tuple	2	(slice, slice)
rgb	float64	(45, 45, 4)	Min: 0.0 Max: 1.0
series	Series	(1,)	Series object of pandas.core.series mo…
test_none	NoneType	1	NoneType object of builtins module

```
   ...: # in the rgb colors of the shaded surface calculated from "shade".
   ...: rgb = ls.shade(z, cmap=cm.gist_earth, vert_exag=0.1, blend_mode='soft')
   ...: surf = ax.plot_surface(x, y, z, rstride=1, cstride=1, facecolors=rgb,
   ...:                        linewidth=0, antialiased=False, shade=False)
   ...:
   ...: plt.show()

In [12]:
```

Interactive console

Source: Spyder-ide.org

Spyder IDE



```python

import pylab
from numpy import cos, linspace, pi, sin, random
from scipy.interpolate import splprep, splev

# %% Generate data for analysis

# Make ascending spiral in 3-space
t = linspace(0, 1.75 * 2 * pi, 100)

x = sin(t)
y = cos(t)
z = t

# Add noise
x += random.normal(scale=0.1, size=x.shape)
y += random.normal(scale=0.1, size=y.shape)
z += random.normal(scale=0.1, size=z.shape)


# %% Perform calculations

# Spline parameters
smoothness = 3.0  # Smoothness parameter
k_param = 2  # Spline order
nests = -1  # Estimate of number of knots needed (-1 = maximal)

# Find the knot points
knot_points, u = splprep([x, y, z], s=smoothness, k=k_param, nests=-1)

# Evaluate spline, including interpolated points
xnew, ynew, znew = splev(linspace(0, 1, 400), knot_points)


# %% Plot results

# TODO: Rewrite to avoid code smell
pylab.subplot(2, 2, 1)
data, = pylab.plot(x, y, 'bo-', label='Data with X-Y Cross Section')
fit, = pylab.plot(xnew, ynew, 'r-', label='Fit with X-Y Cross Section')
pylab.legend()
pylab.xlabel('x')
pylab.ylabel('y')
```

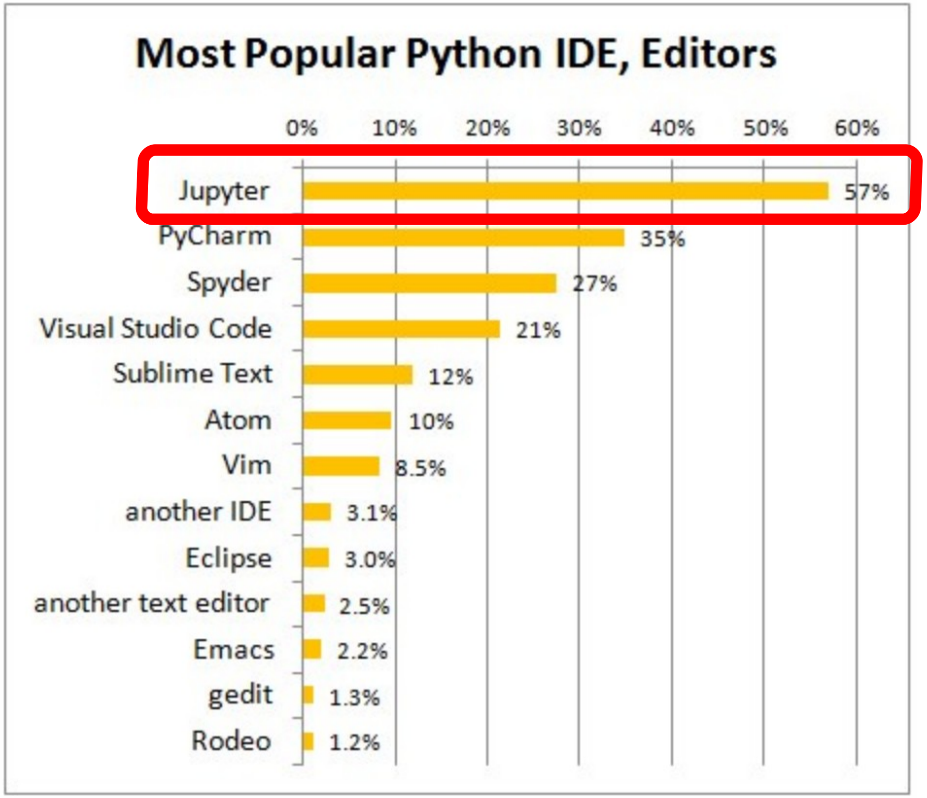
Name	Type	Size	Value
bars	container.BarContainer	20	BarContainer object of matplotlib.cont…
df	DataFrame	(3, 2)	Column names: bools, ints
filename	str	1	C:\ProgramData\Anaconda3\lib\site-pack…
list_test	list	2	[Dataframe, Numpy array]
nrows	int	1	344
r	float64	1	7.611082589334796
radii	float64	(20,)	Min: 0.4983036638535687 Max: 9.856848974942551
region	tuple	2	(slice, slice)
rgb	float64	(45, 45, 4)	Min: 0.0 Max: 1.0
series	Series	(1,)	Series object of pandas.core.series mo…
test_none	NoneType	1	NoneType object of builtins module

```
   ...: # in the rgb colors of the shaded surface calculated from "shade".
   ...: rgb = ls.shade(z, cmap=cm.gist_earth, vert_exag=0.1, blend_mode='soft')
   ...: surf = ax.plot_surface(x, y, z, rstride=1, cstride=1, facecolors=rgb,
   ...:                        linewidth=0, antialiased=False, shade=False)
   ...:
   ...: plt.show()

In [12]:
```

Function browser

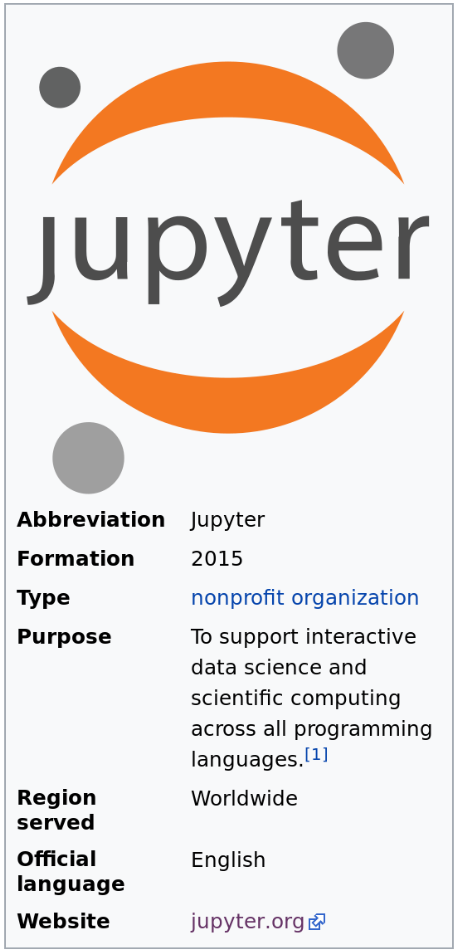
Source: Spyder-ide.org

Integrated Development Environments



Source: Most popular Python IDEs / Editors, 2018, by Gregory Platetsky, Kdnuggets, https://www.kdnuggets.com/2018/12/most-popular-python-ide-editor.html

Jupyter

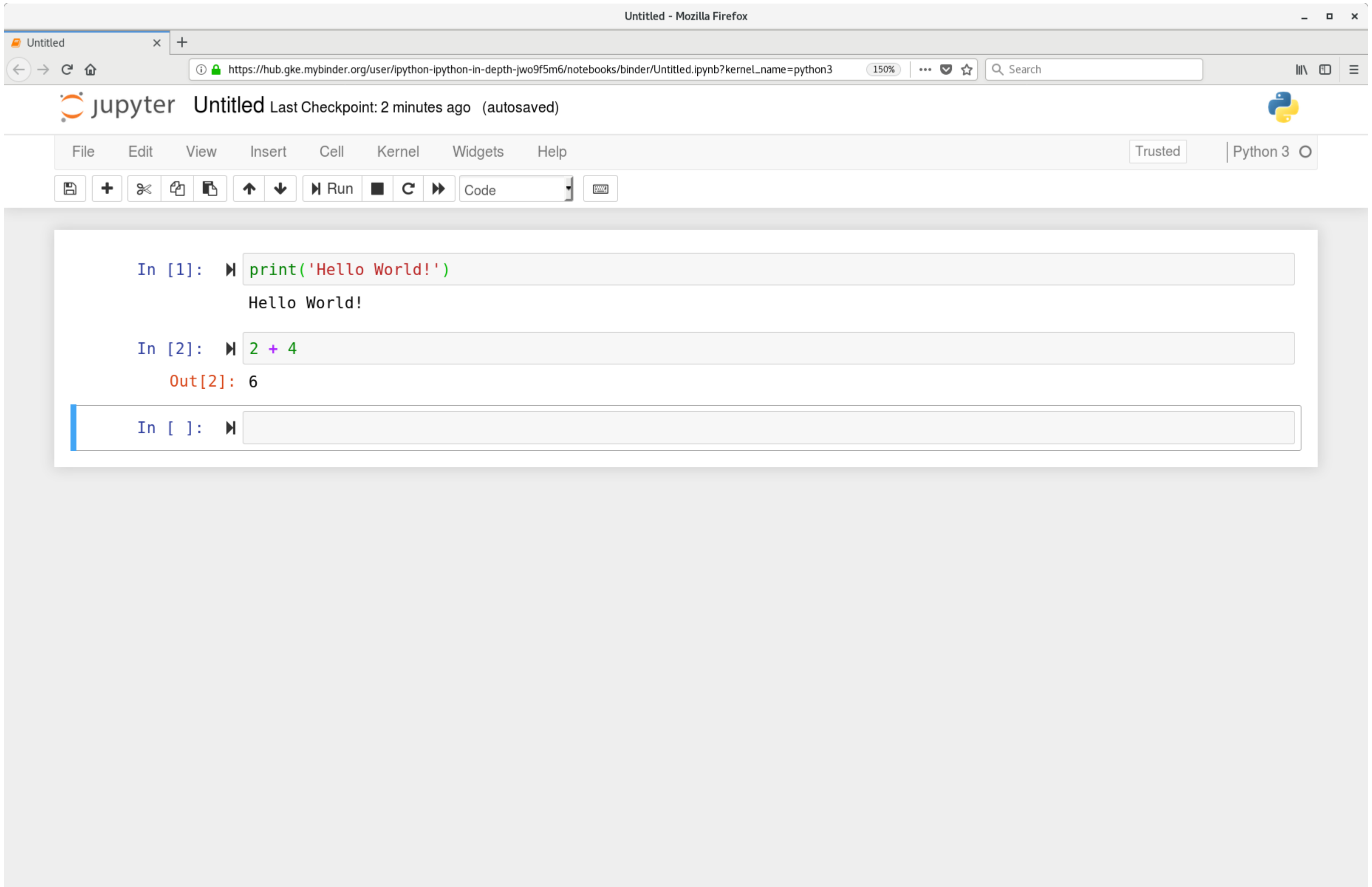


Abbreviation	Jupyter
Formation	2015
Type	nonprofit organization
Purpose	To support interactive data science and scientific computing across all programming languages.[1]
Region served	Worldwide
Official language	English
Website	jupyter.org

Jupyter is an interactive web-based execution environment that allows to create and share code. Jupyter supports several languages but the core ones are Julia, Python, and R.

Source:https://en.wikipedia.org/wiki/Project_Jupyter

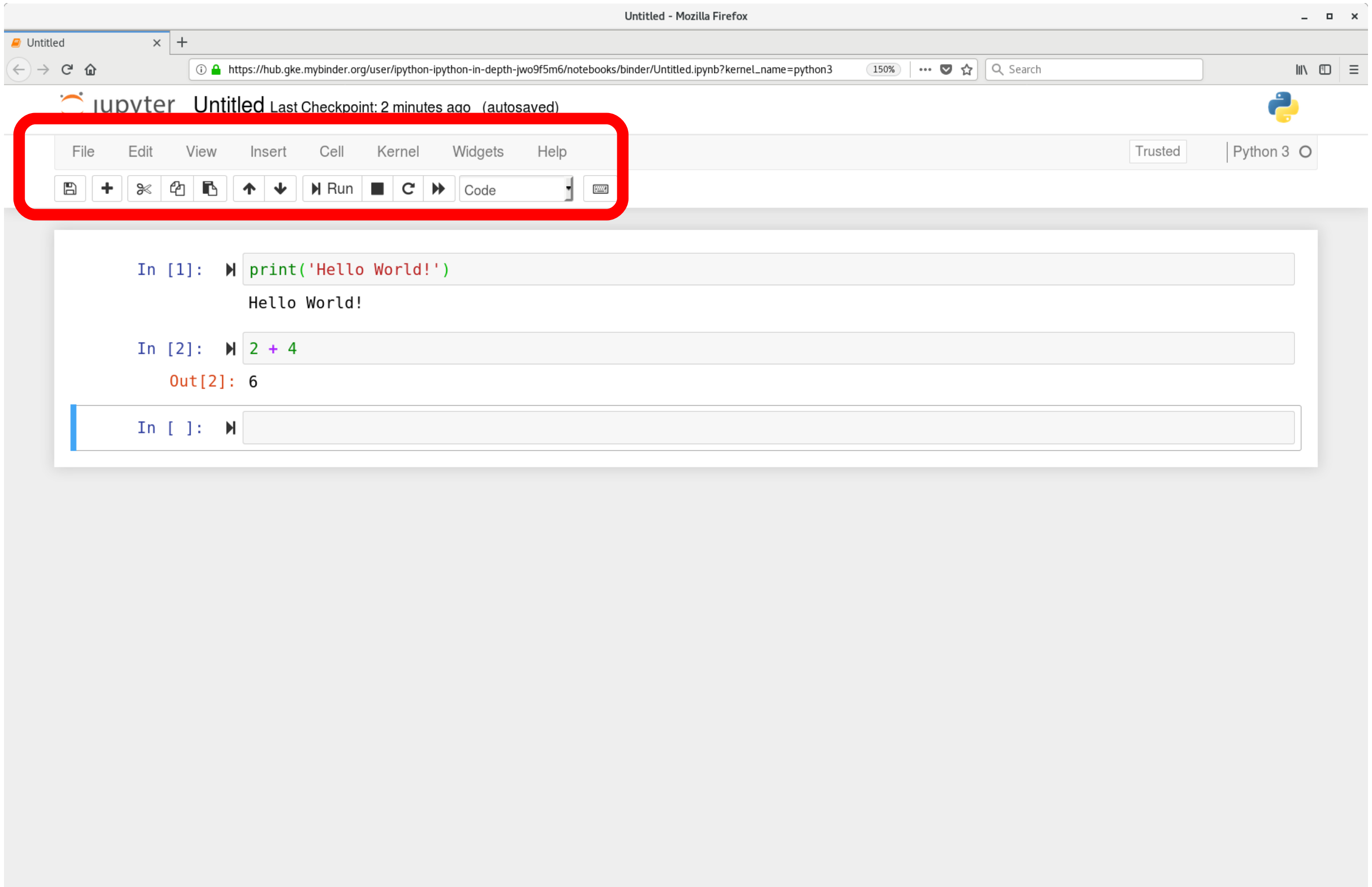
Jupyter notebook


Untitled - Mozilla Firefox
Untitled
https://hub.gke.mybinder.org/user/ipython-ipython-in-depth-jwo9f5m6/notebooks/binder/Untitled.ipynb?kernel_name=python3
150%
Search
jupyter
Untitled Last Checkpoint: 2 minutes ago (autosaved)
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Trusted
Python 3
Run
Code

```
In [1]:  print('Hello World!')

         Hello World!

In [2]:  2 + 4

Out[2]:  6

In [ ]:
```

Jupyter notebook

Menu bar and toolbar with editor common utilities

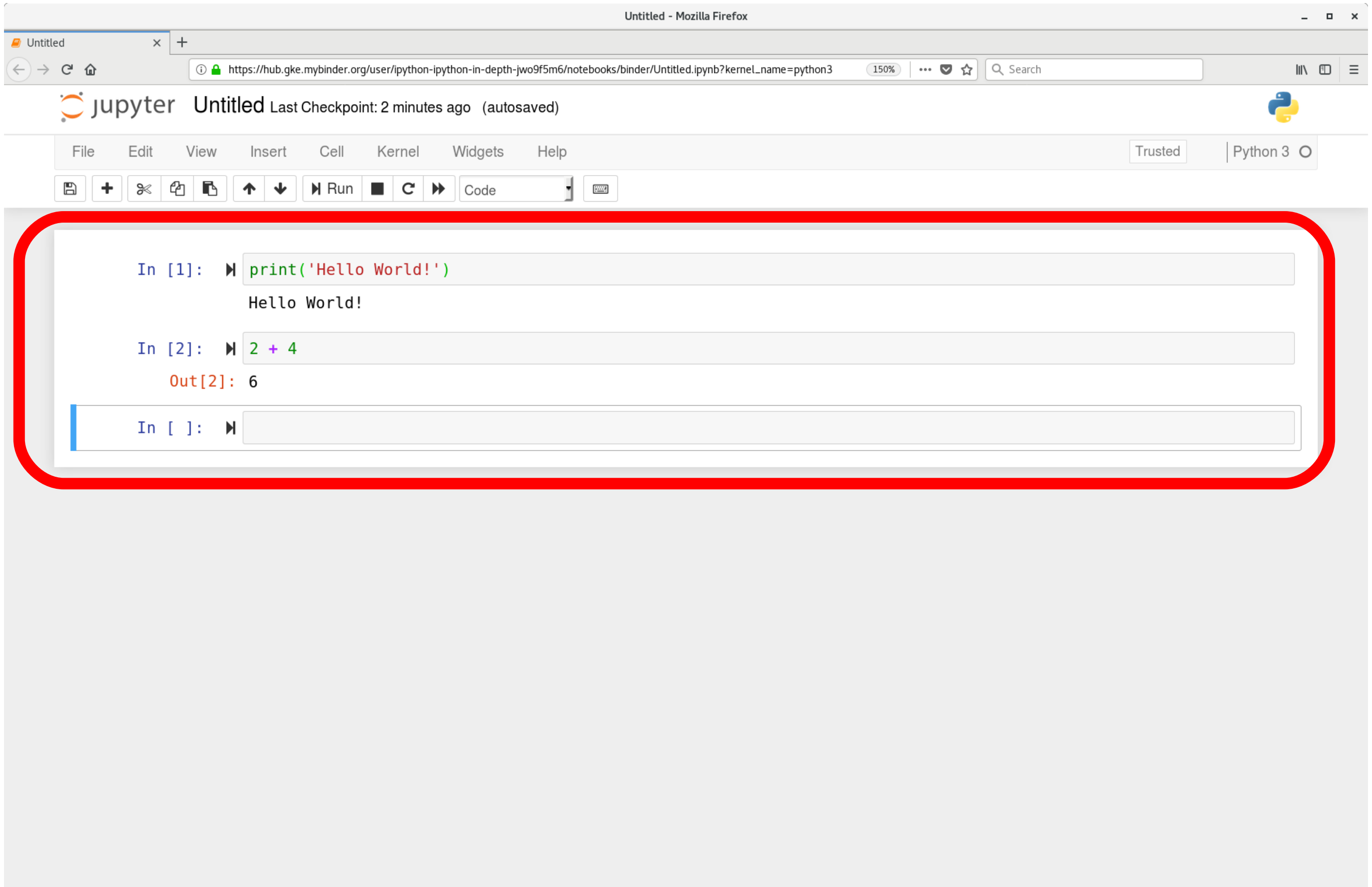


Untitled - Mozilla Firefox
Untitled
https://hub.gke.mybinder.org/user/ipython-ipython-in-depth-jwo9f5m6/notebooks/binder/Untitled.ipynb?kernel_name=python3
150%
Search
jupyter
Untitled Last Checkpoint: 2 minutes ago (autosaved)
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Trusted
Python 3
Run
Code

```
In [1]:   print('Hello World!')
          Hello World!

In [2]:   2 + 4
Out[2]:   6

In [ ]:
```

Jupyter notebook

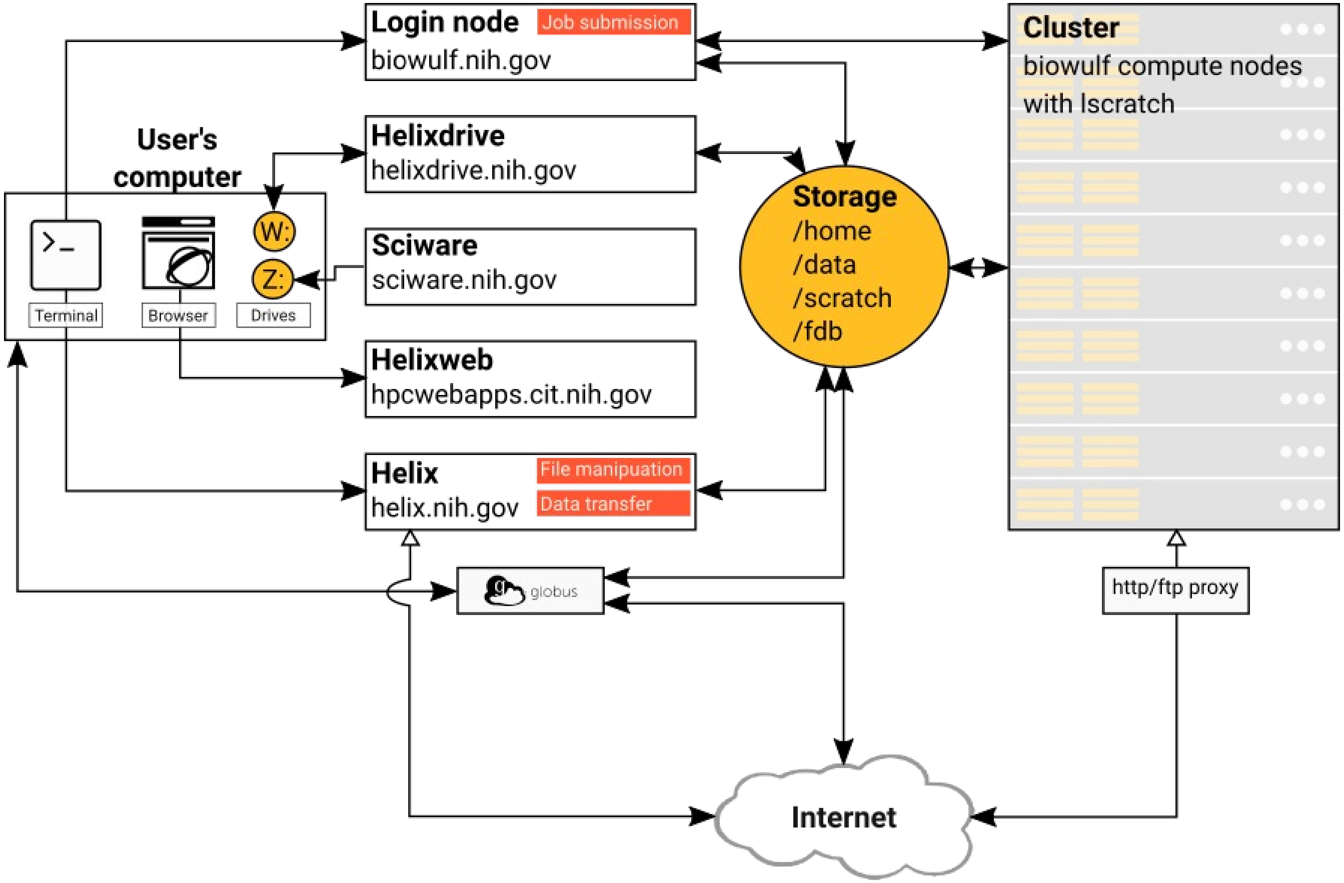
The body of the notebook contains “cells” which consists of code and output


Untitled - Mozilla Firefox
Untitled
https://hub.gke.mybinder.org/user/ipython-ipython-in-depth-jwo9f5m6/notebooks/binder/Untitled.ipynb?kernel_name=python3
150%
Search
jupyter
Untitled Last Checkpoint: 2 minutes ago (autosaved)
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Trusted
Python 3
Run
Code

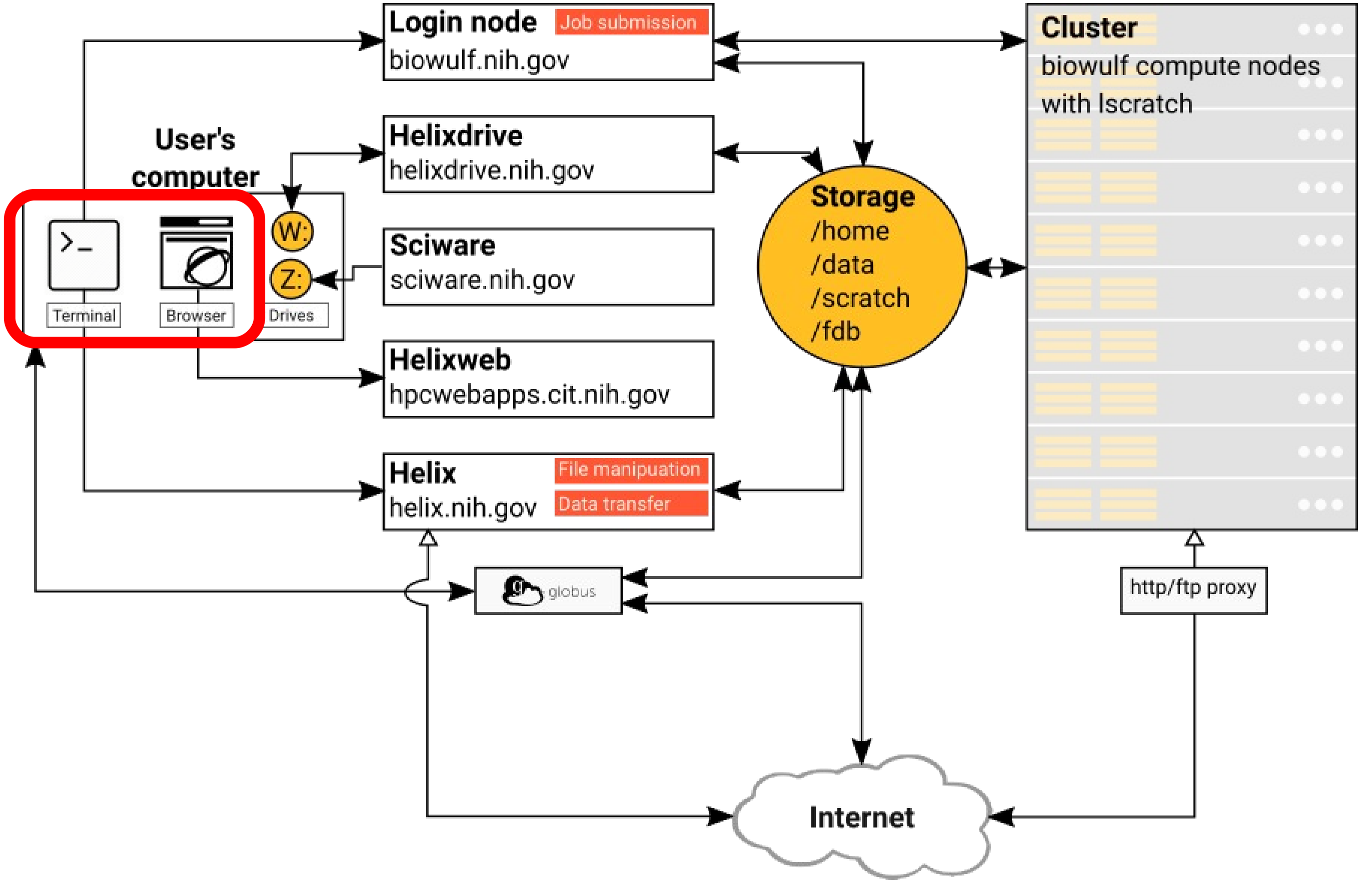
```
In [1]:  print('Hello World!')
         Hello World!

In [2]:  2 + 4
Out[2]:  6

In [ ]:
```

HANDS-ON EXAMPLES

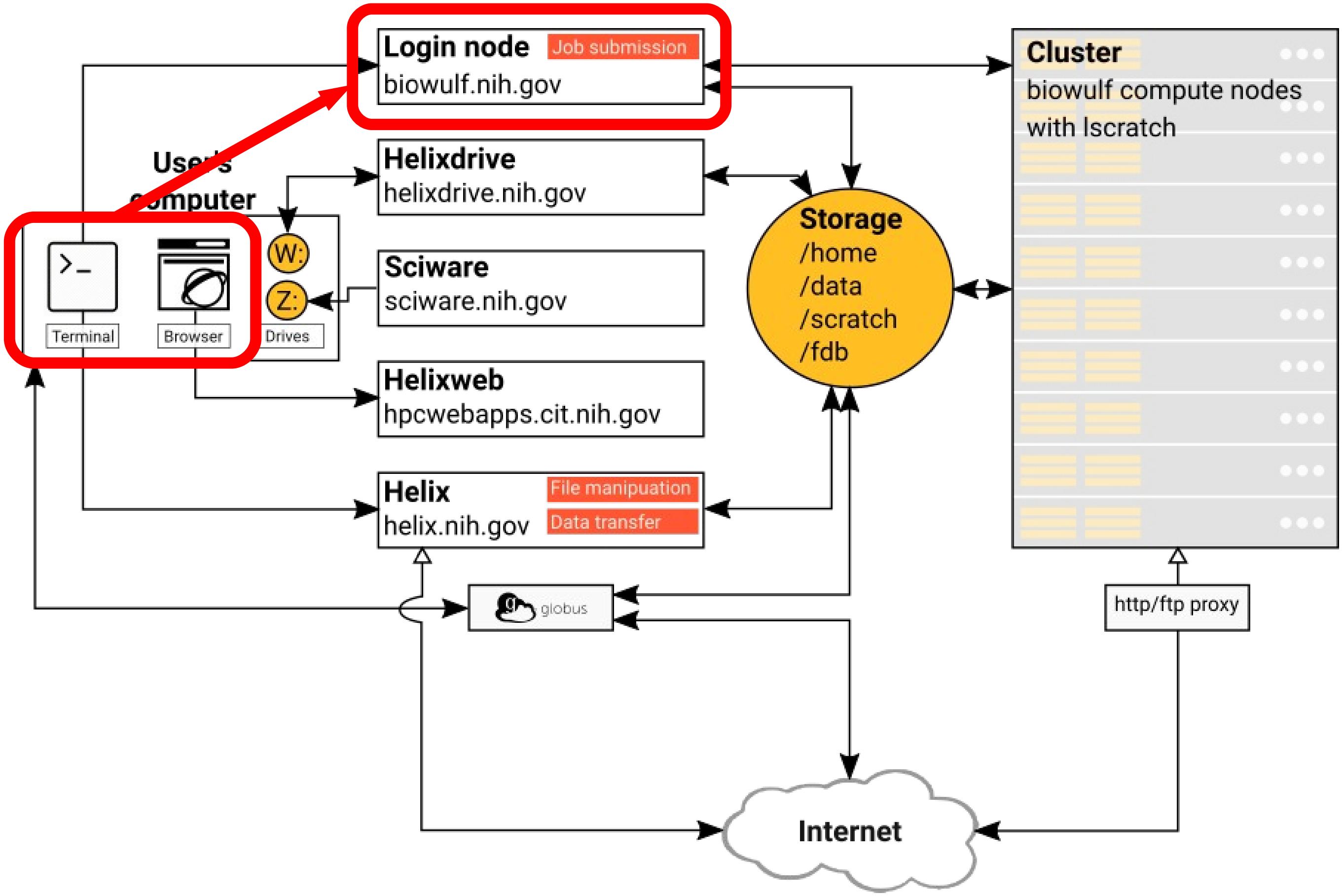
Jupyter on Biowulf setup



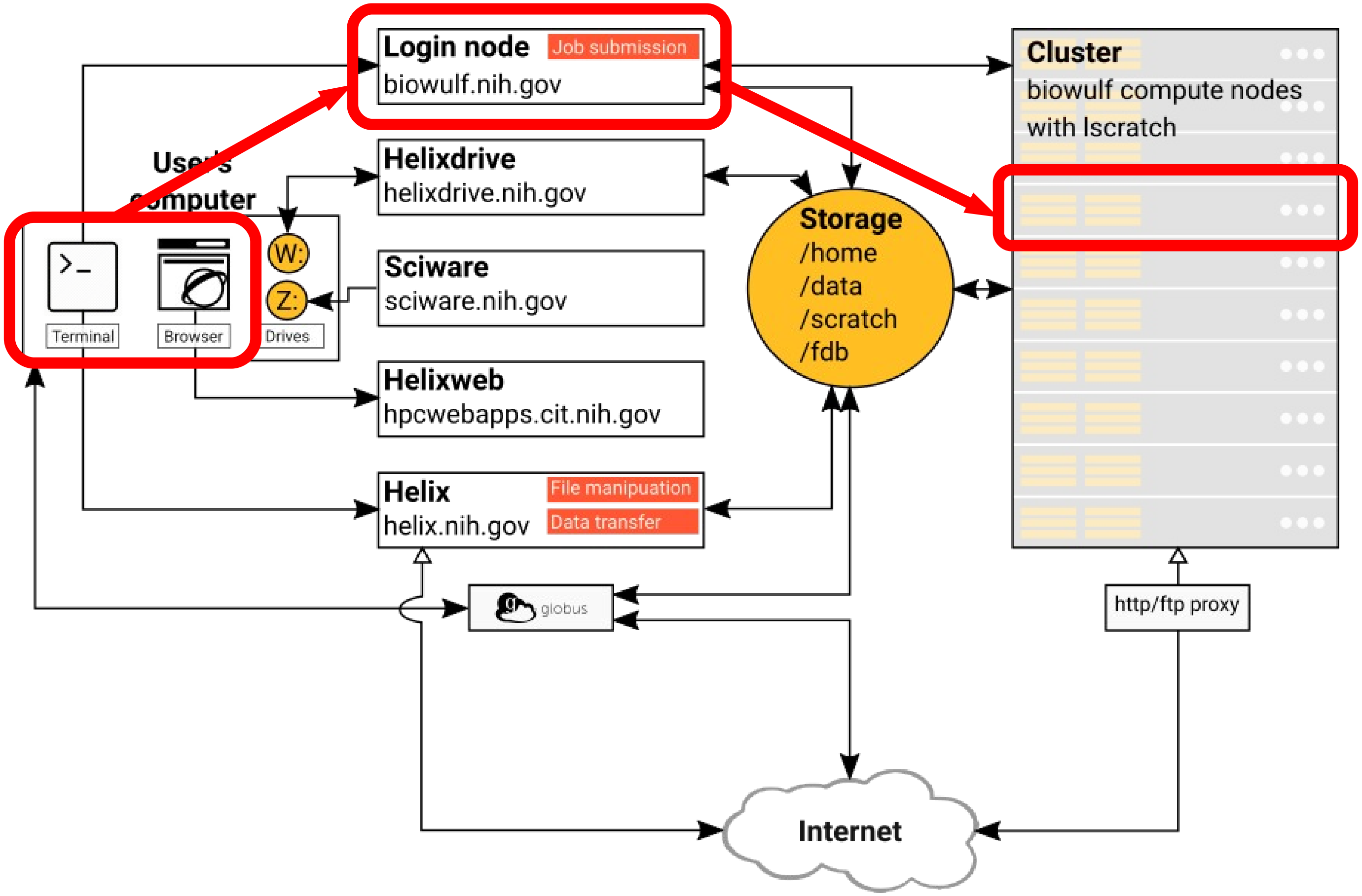
Jupyter on Biowulf setup



Jupyter on Biowulf setup

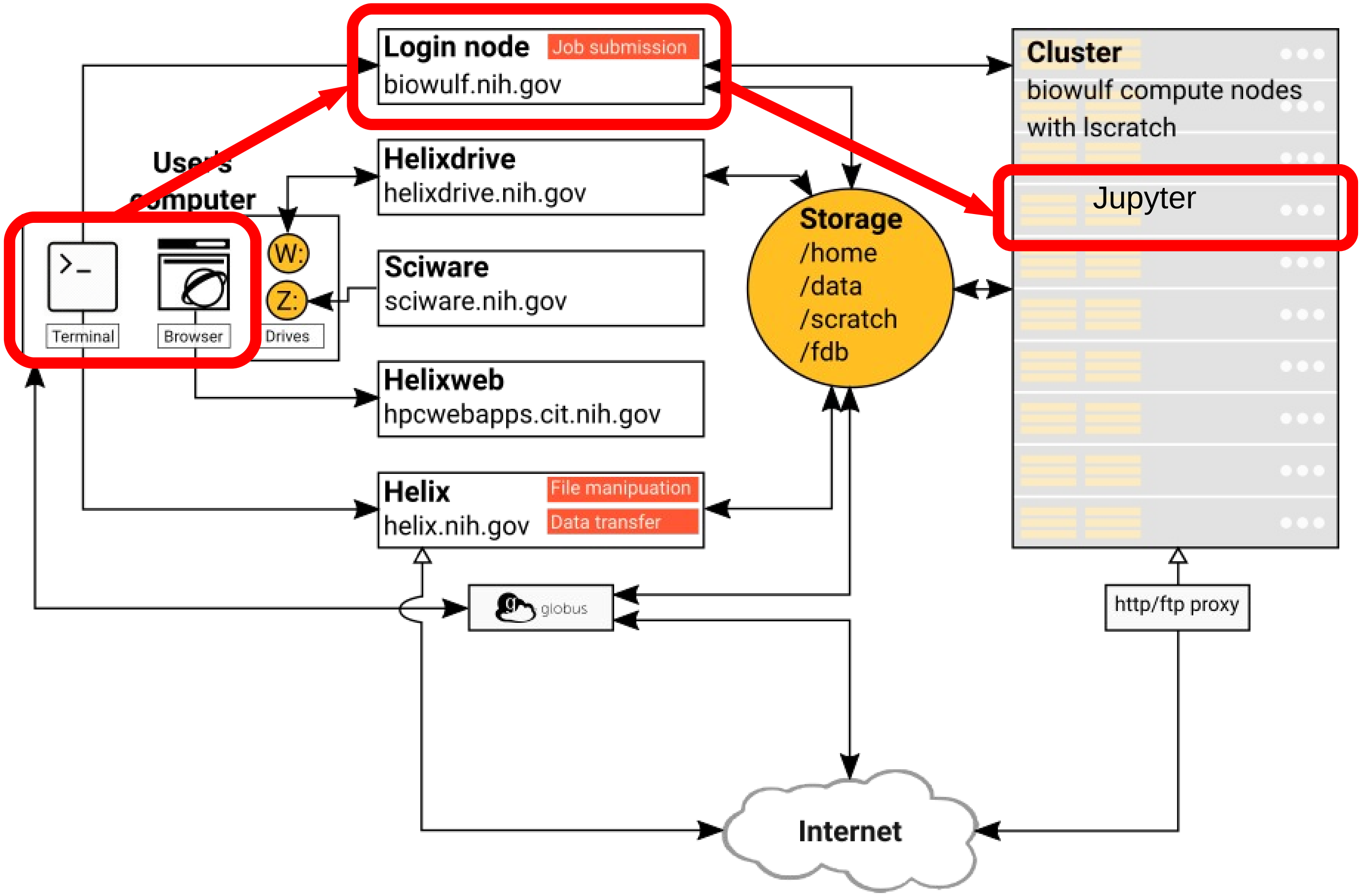


Jupyter on Biowulf setup



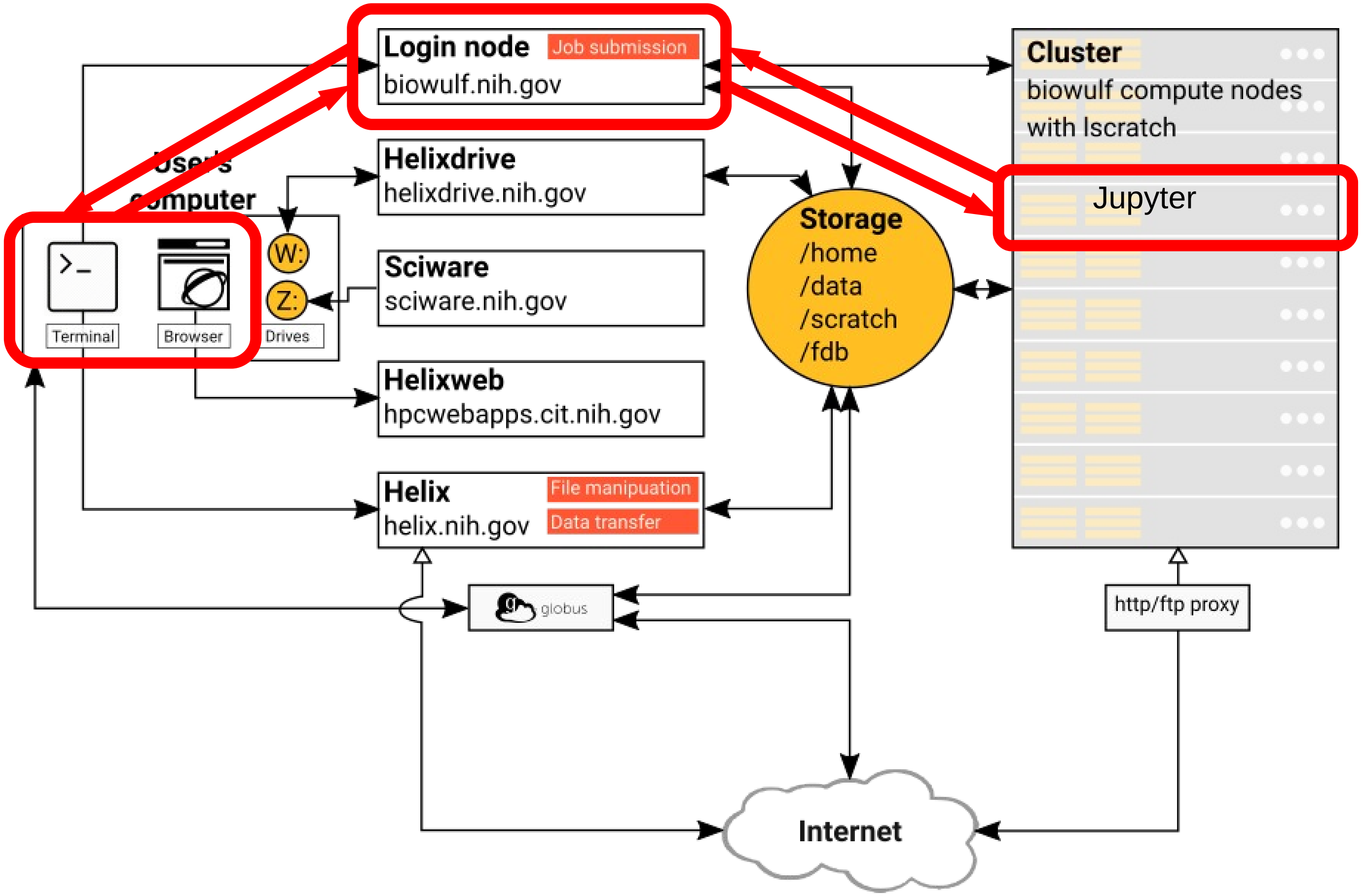
NIH BIOWULF

Jupyter on Biowulf setup



Jupyter on Biowulf setup

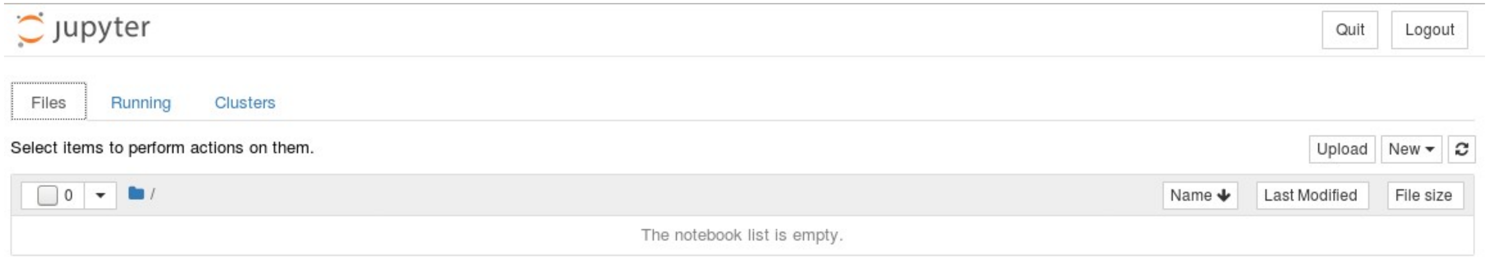
Jupyter notebook on your browser



Jupyter on Biowulf setup

1) Open a terminal on your desktop	(varies by platform)
2) Log into Biowulf	local$> ssh -Y user@biowulf.nih.gov
3) Launch screen	biowulf$> screen
4) Request interactive session	biowulf$> sinteractive --gres=lscratch:1 --tunnel
5) Change to local directory	cn1234$> cd /lscratch/$SLURM_JOB_ID
6) Load Jupyter module	cn1234$> module load jupyter
7) Launch Jupyter notebooks	cn1234$> jupyter notebook --ip localhost \\ --port $PORT1 \\ --no-browser
8) Open 2nd terminal on your desktop	(varies by platform)
9) Open tunnel from desktop to Biowulf	local$> ssh -L 12345:localhost:12345 user@biowulf.nih.gov
10) Enter Jupyter URL into local browser	(copy/paste)

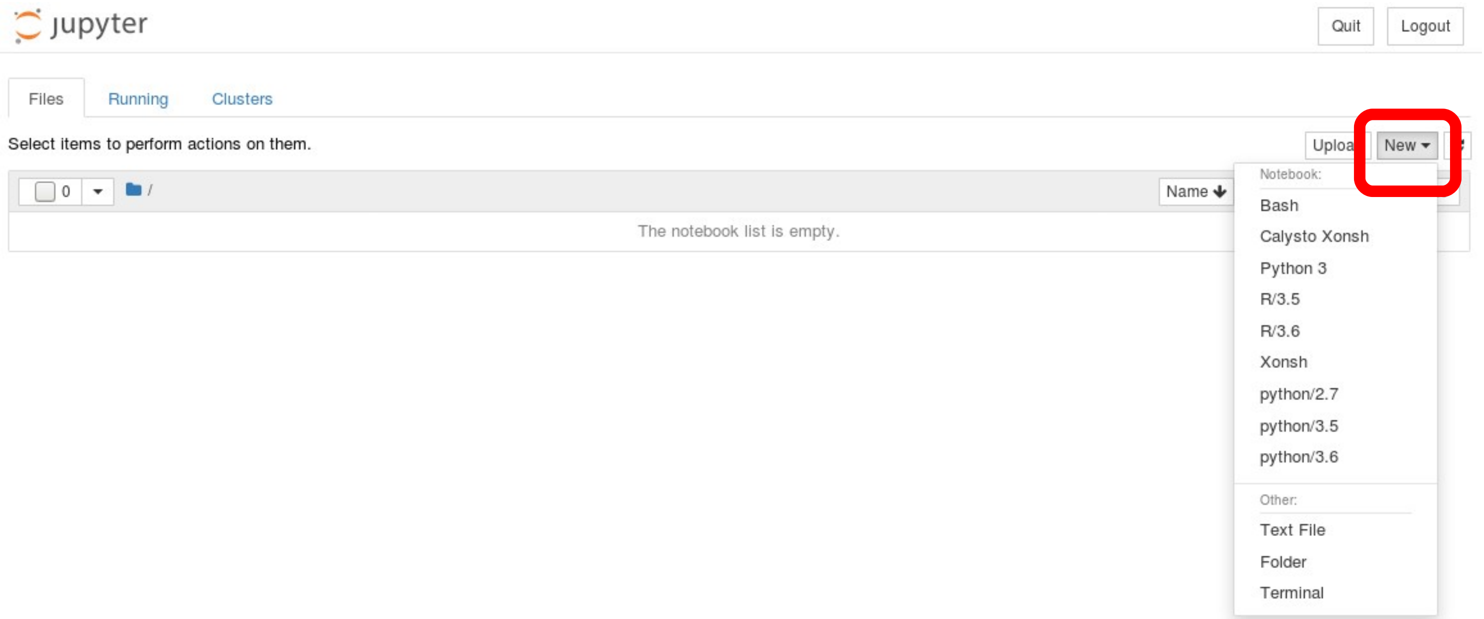
Jupyter on Biowulf setup



Select items to perform actions on them.

The notebook list is empty.

Jupyter on Biowulf setup

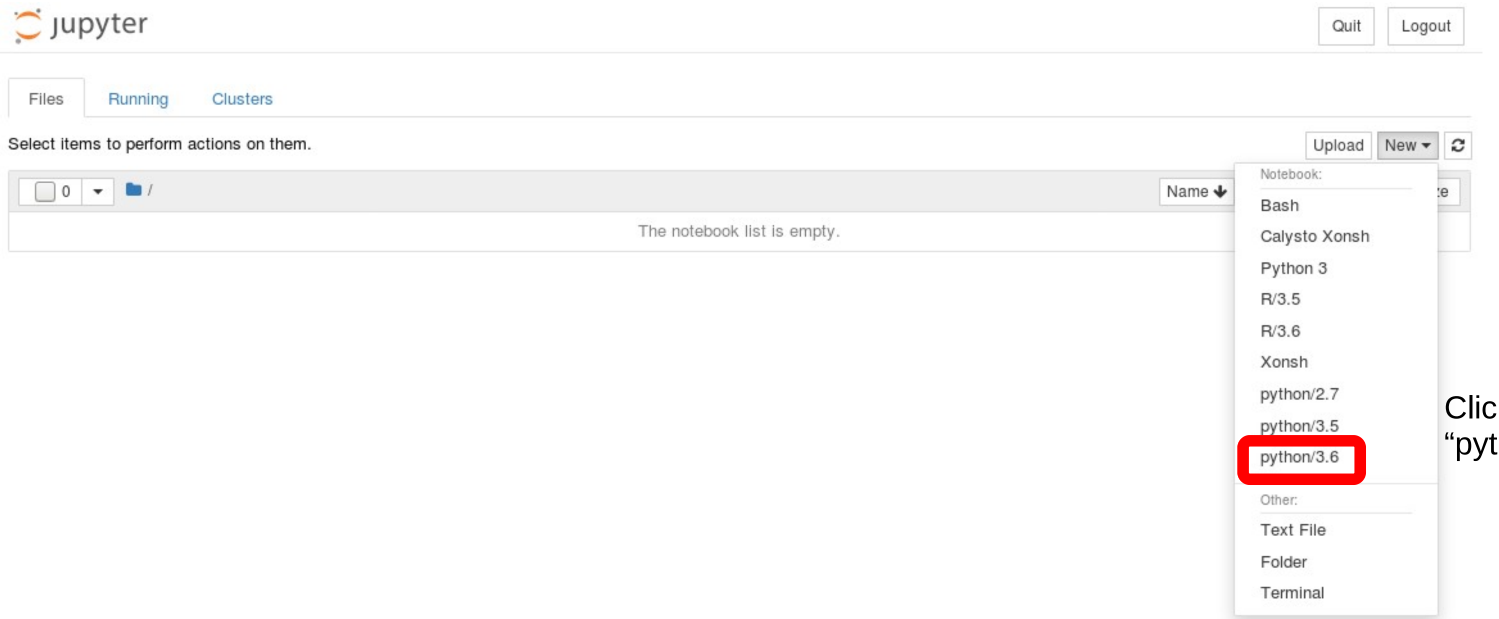


Select items to perform actions on them.

The notebook list is empty.

Click on “New”

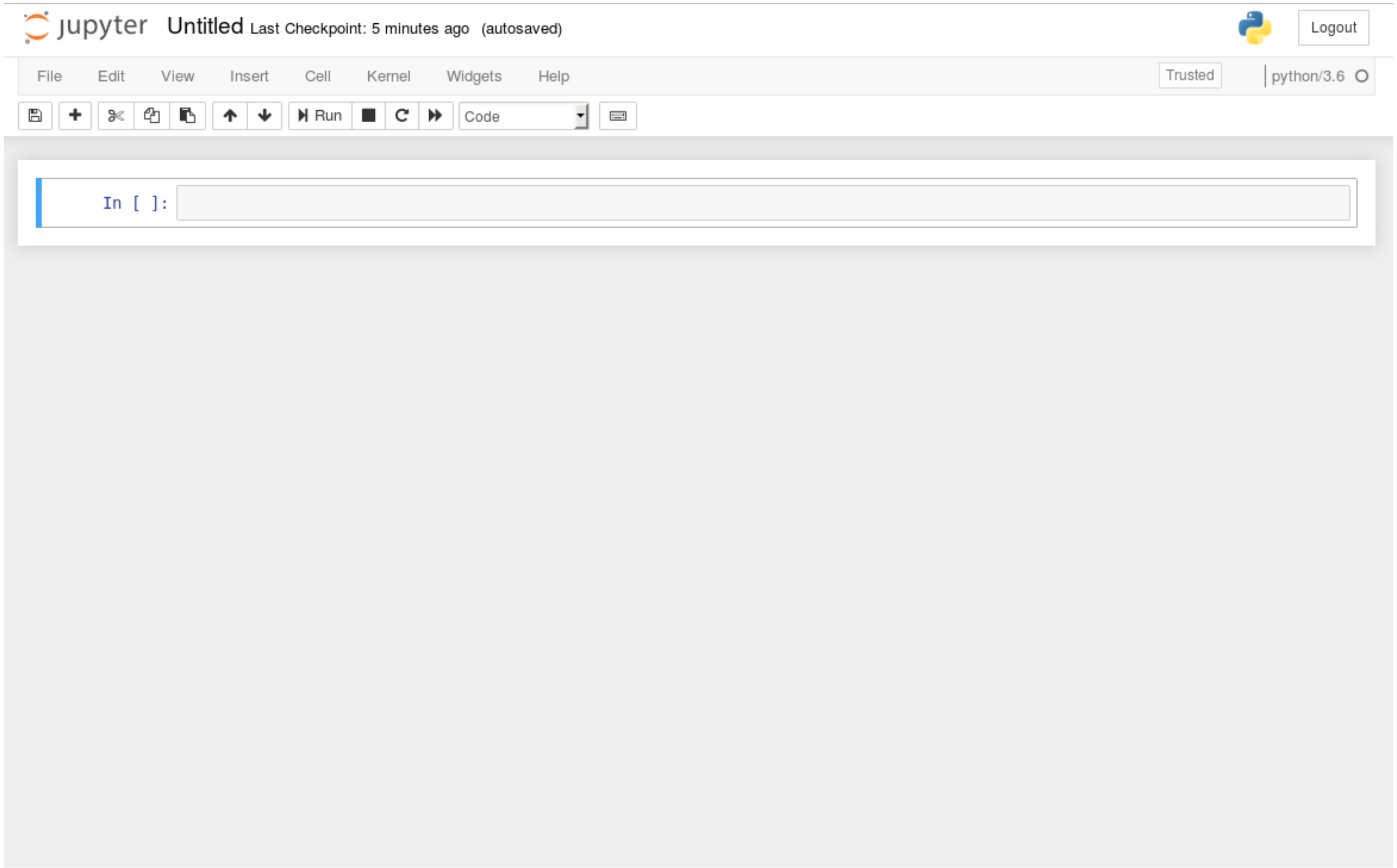
Jupyter on Biowulf setup



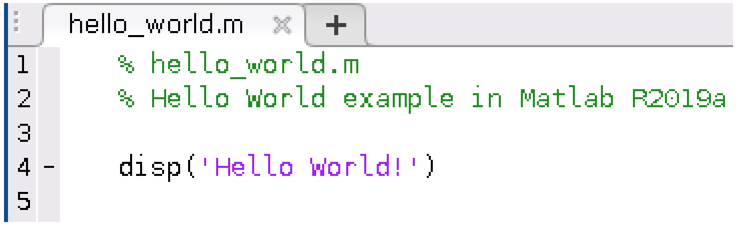
Select items to perform actions on them.

Click on "python/3.6"

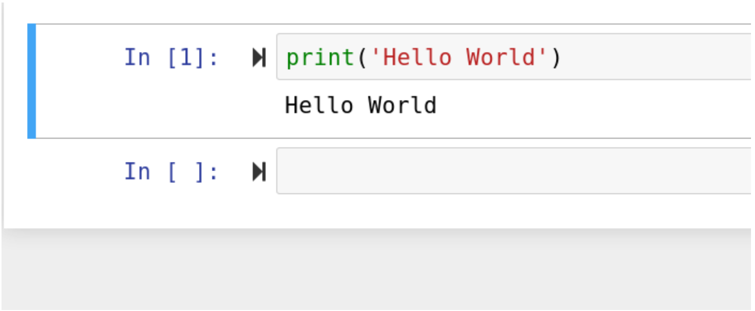
Jupyter on Biowulf setup



Example 1: hello world!

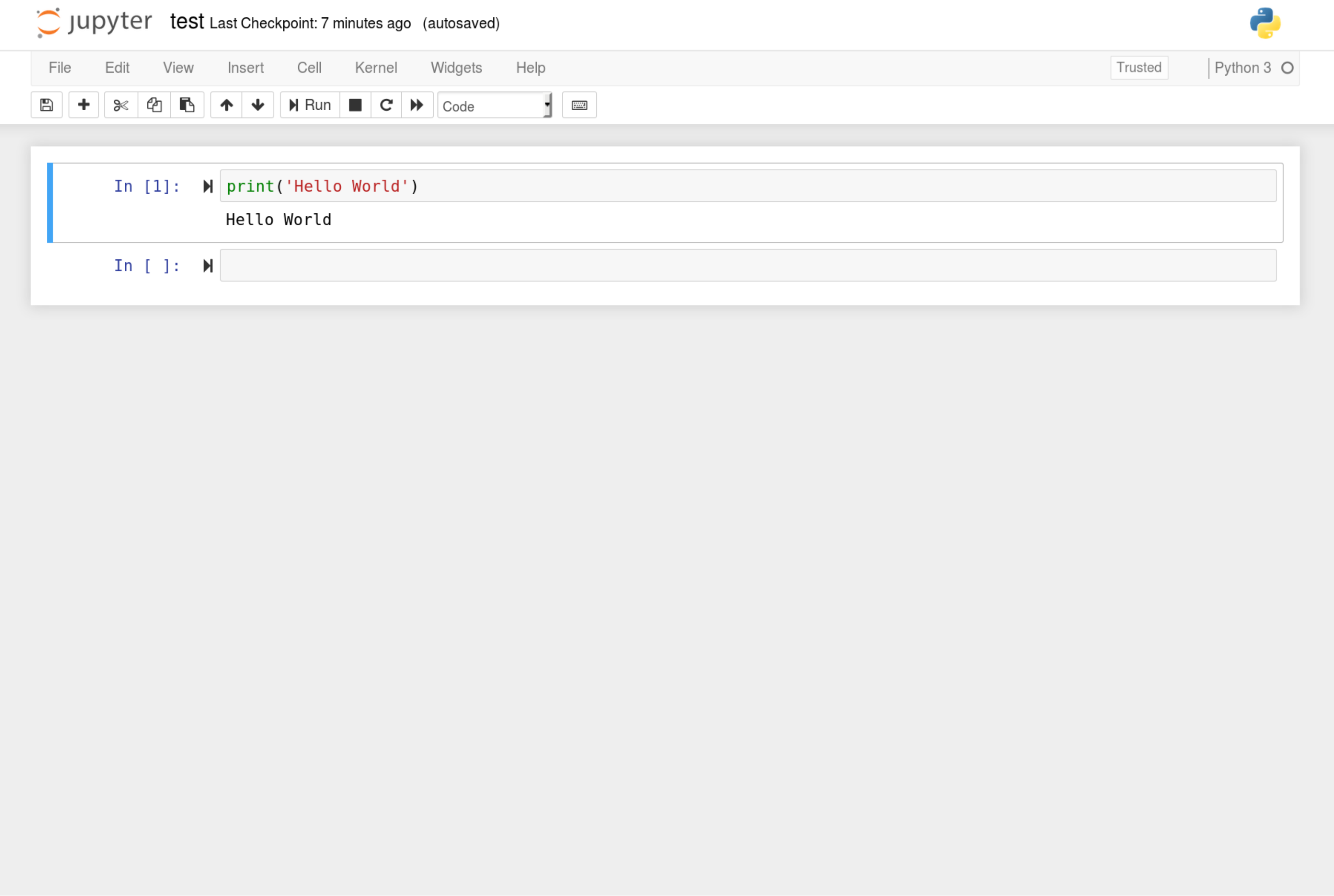


```
% hello_world.m
% Hello World example in Matlab R2019a

disp('Hello World!')
```



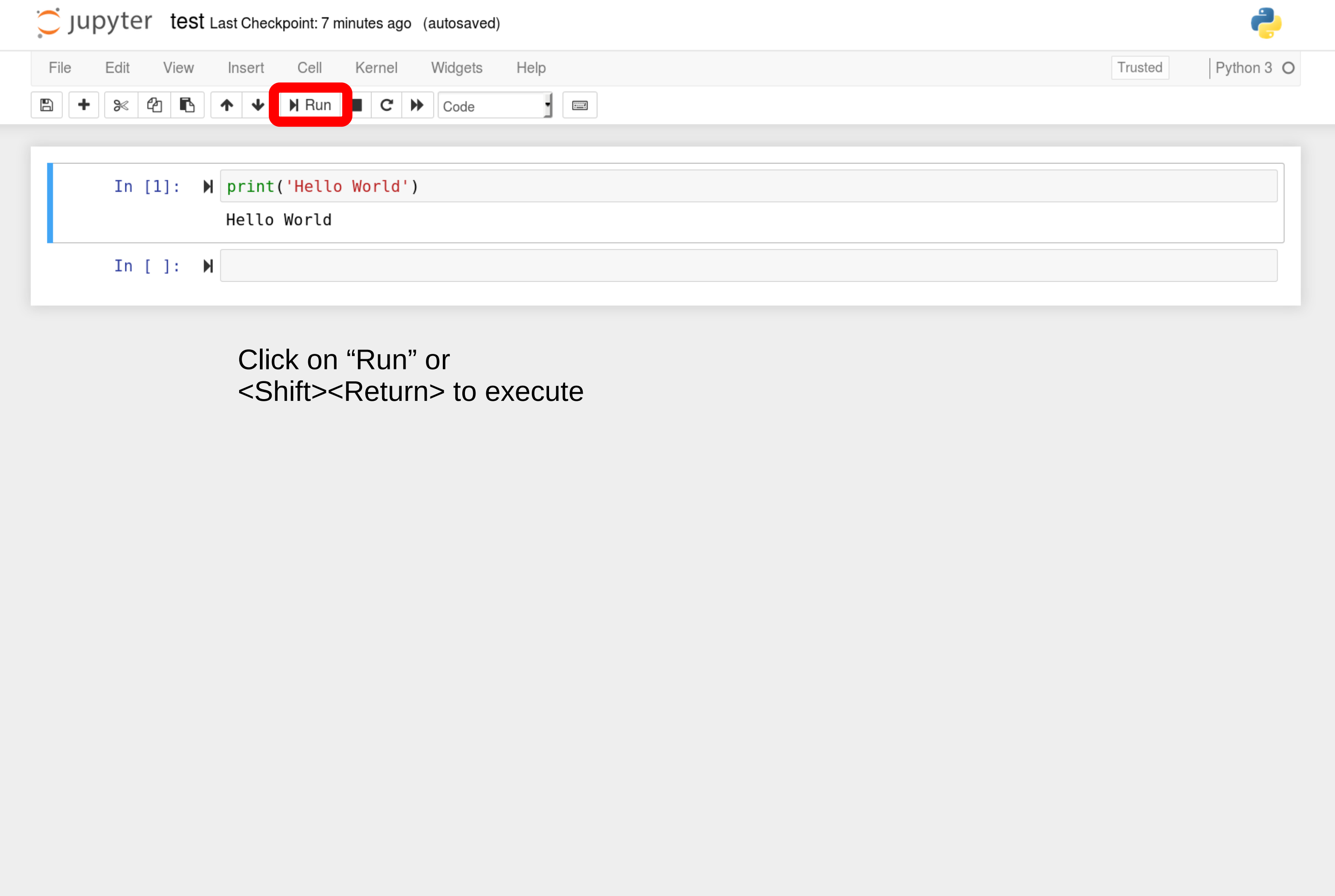
```
In [1]:  print('Hello World')
         Hello World
In [ ]:
```

Example 1: hello world!



```
In [1]: print('Hello World')
        Hello World

In [ ]:
```

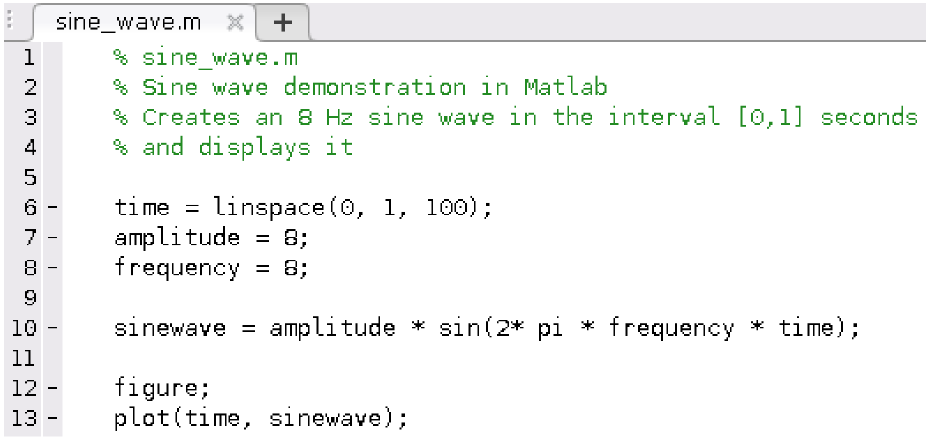
Example 1: hello world!



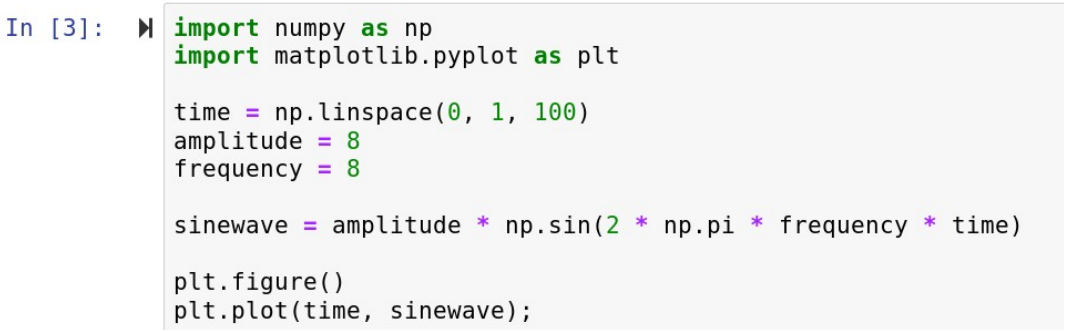
```
In [1]:  print('Hello World')
         Hello World

In [ ]:
```

Click on “Run” or
<Shift><Return> to execute

Example 2: Sine wave

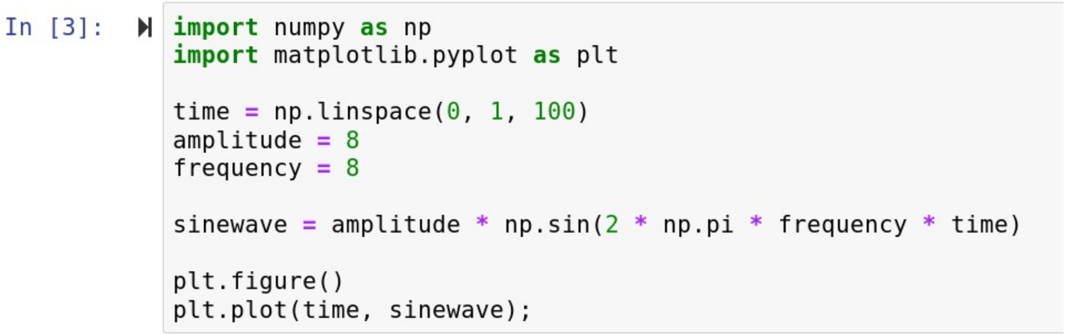


```matlab
% sine_wave.m
% Sine wave demonstration in Matlab
% Creates an 8 Hz sine wave in the interval [0,1] seconds
% and displays it

time = linspace(0, 1, 100);
amplitude = 8;
frequency = 8;

sinewave = amplitude * sin(2* pi * frequency * time);

figure;
plot(time, sinewave);
```



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

time = np.linspace(0, 1, 100)
amplitude = 8
frequency = 8

sinewave = amplitude * np.sin(2 * np.pi * frequency * time)

plt.figure()
plt.plot(time, sinewave);
```

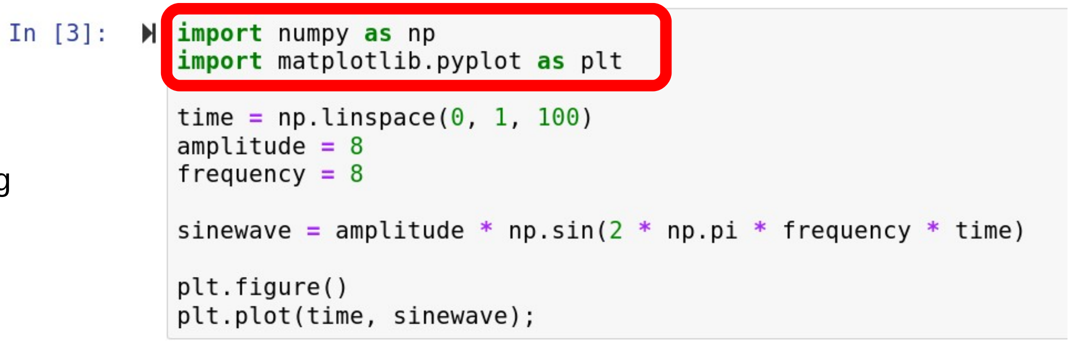
Example 2: Sine wave



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

time = np.linspace(0, 1, 100)
amplitude = 8
frequency = 8

sinewave = amplitude * np.sin(2 * np.pi * frequency * time)

plt.figure()
plt.plot(time, sinewave);
```

Example 2: Sine wave

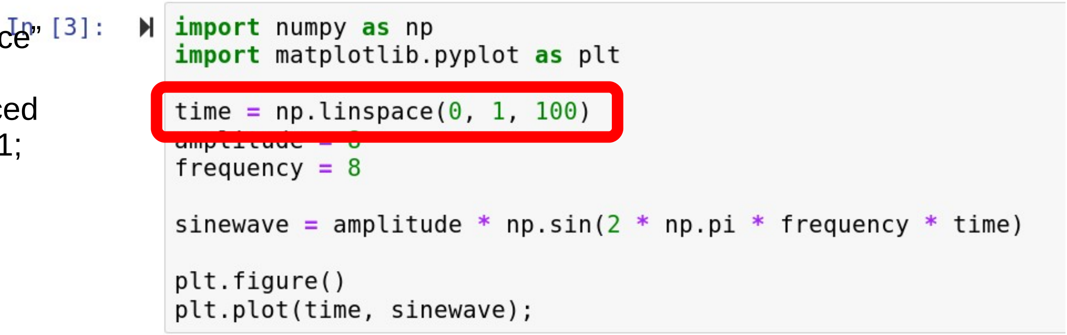
Extra functions in Python are contained in packages; to import a package we need to use the “import” command followed by package name (e.g., “numpy”, “matplotlib”); we can give an internal name (alias) to package by using “as”.



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

time = np.linspace(0, 1, 100)
amplitude = 8
frequency = 8

sinewave = amplitude * np.sin(2 * np.pi * frequency * time)

plt.figure()
plt.plot(time, sinewave);
```

Example 2: Sine wave

Here we are using the alias “np” which corresponds to the package “numpy” that contains functions that perform array manipulations; “np.linspace” creates an array of 100 ordered and equally spaced numbers between 0 and 1;



```
In [3]:  import numpy as np
         import matplotlib.pyplot as plt

         time = np.linspace(0, 1, 100)
         amplitude = 8
         frequency = 8

         sinewave = amplitude * np.sin(2 * np.pi * frequency * time)

         plt.figure()
         plt.plot(time, sinewave);
```

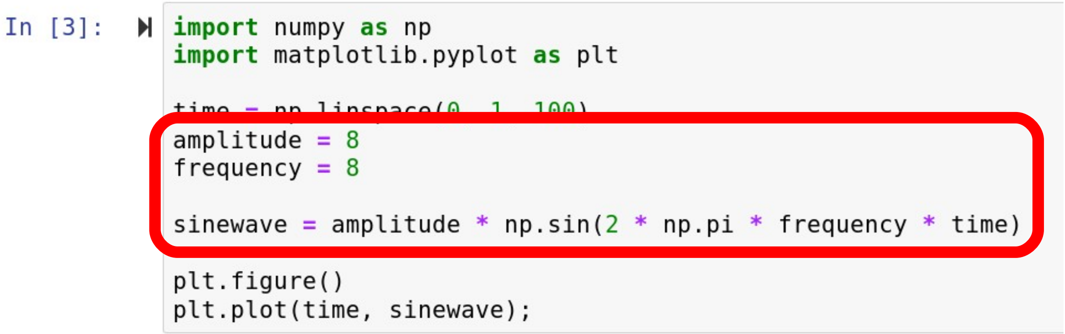
Example 2: Sine wave

MATLAB ARRAYS	NUMPY ARRAYS
Multidimensional	Multidimensional
1 (one) based indexing	0 (zero) based indexing
Elements are accessed using parentheses, e.g., a(1)	Elelements are accessed using brackets, e.g., a[0]
Slicing is inclusive at both ends of array	Slicing is left inclusive and right exclusive

Source: Numpy for Matlab users [docs.scipy.org/doc/numpy/user/numpy-for-matlab-users.html]

Example 2: Sine wave

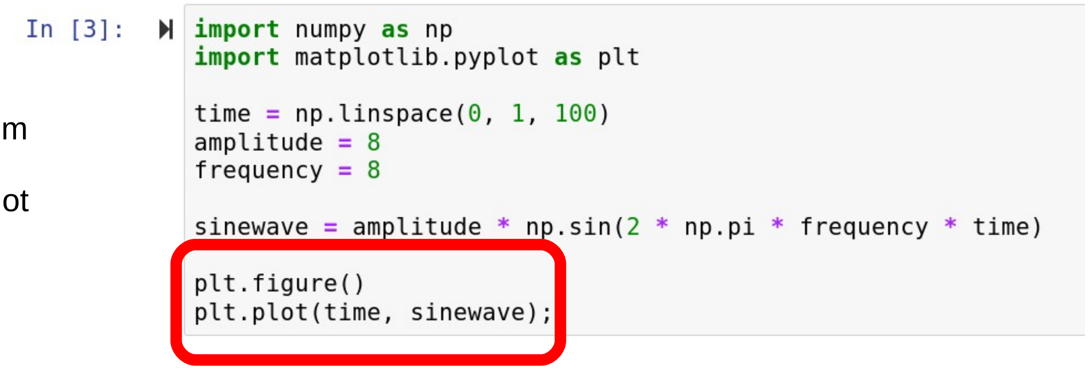
This is just the sine wave equation in Python using the function “sin” from the package “np”



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

time = np.linspace(0, 1, 100)
amplitude = 8
frequency = 8

sinewave = amplitude * np.sin(2 * np.pi * frequency * time)

plt.figure()
plt.plot(time, sinewave);
```

Example 2: Sine wave

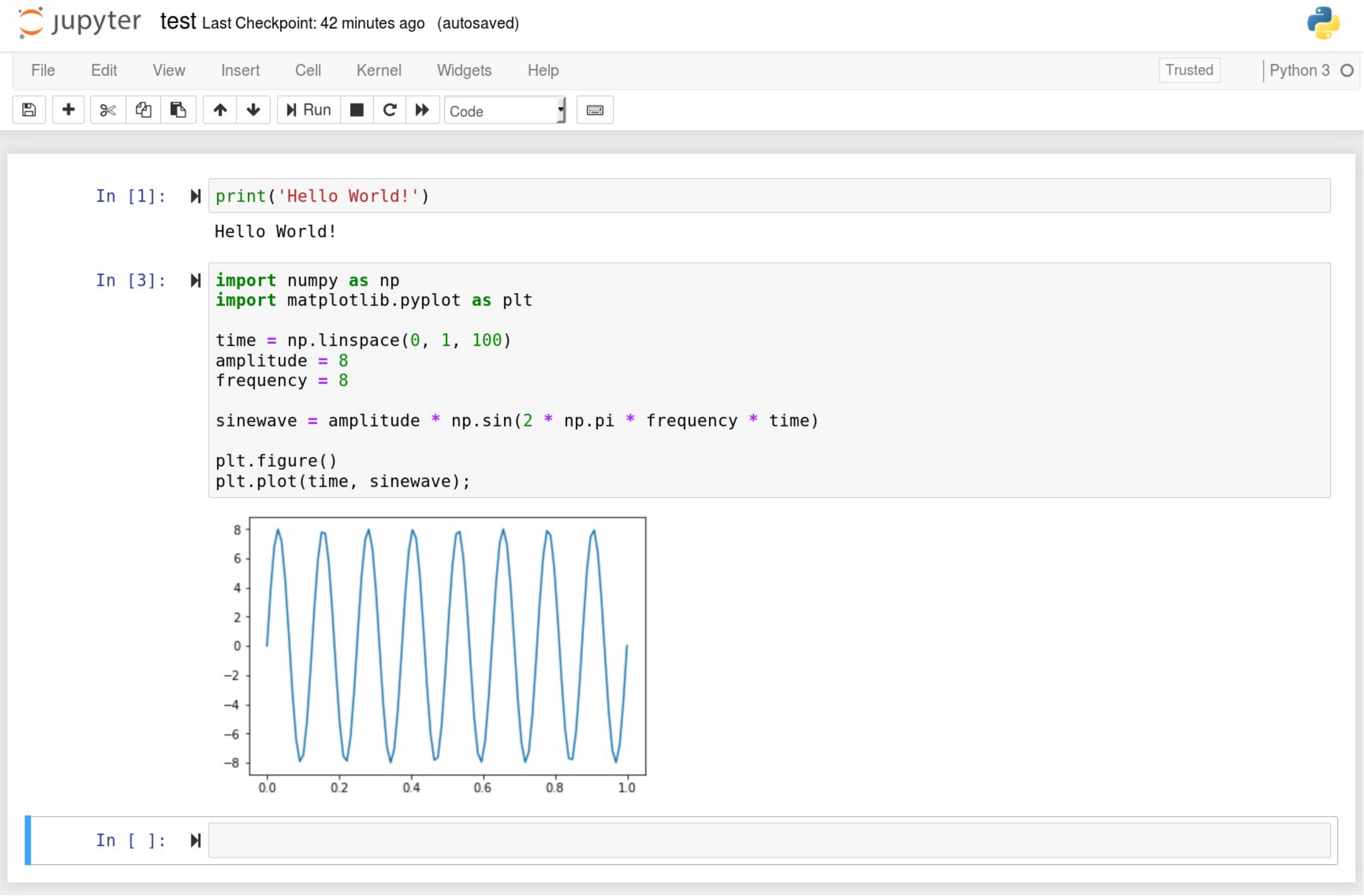
Finally, using functions from the pyplot package ("plt"), we create a figure, then plot the contents of the array "sinewave" against the array "time"



In [3]:

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

time = np.linspace(0, 1, 100)
amplitude = 8
frequency = 8

sinewave = amplitude * np.sin(2 * np.pi * frequency * time)

plt.figure()
plt.plot(time, sinewave);
```

Example 2: Sine wave

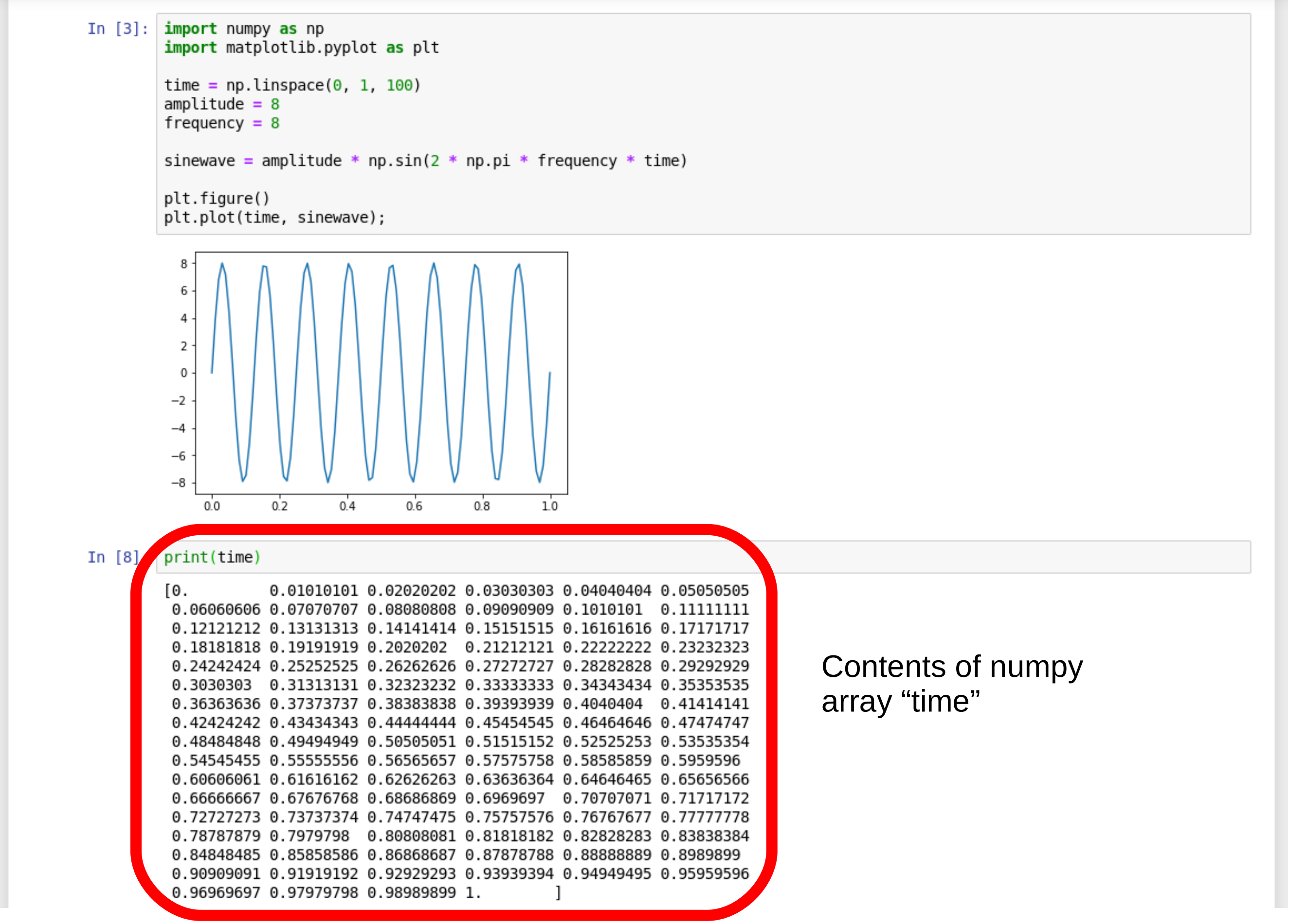


```
print('Hello World!')
```

```
Hello World!
```

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

time = np.linspace(0, 1, 100)
amplitude = 8
frequency = 8

sinewave = amplitude * np.sin(2 * np.pi * frequency * time)

plt.figure()
plt.plot(time, sinewave);
```

Example 2: Sine wave



In [3]:

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

time = np.linspace(0, 1, 100)
amplitude = 8
frequency = 8

sinewave = amplitude * np.sin(2 * np.pi * frequency * time)

plt.figure()
plt.plot(time, sinewave);
```

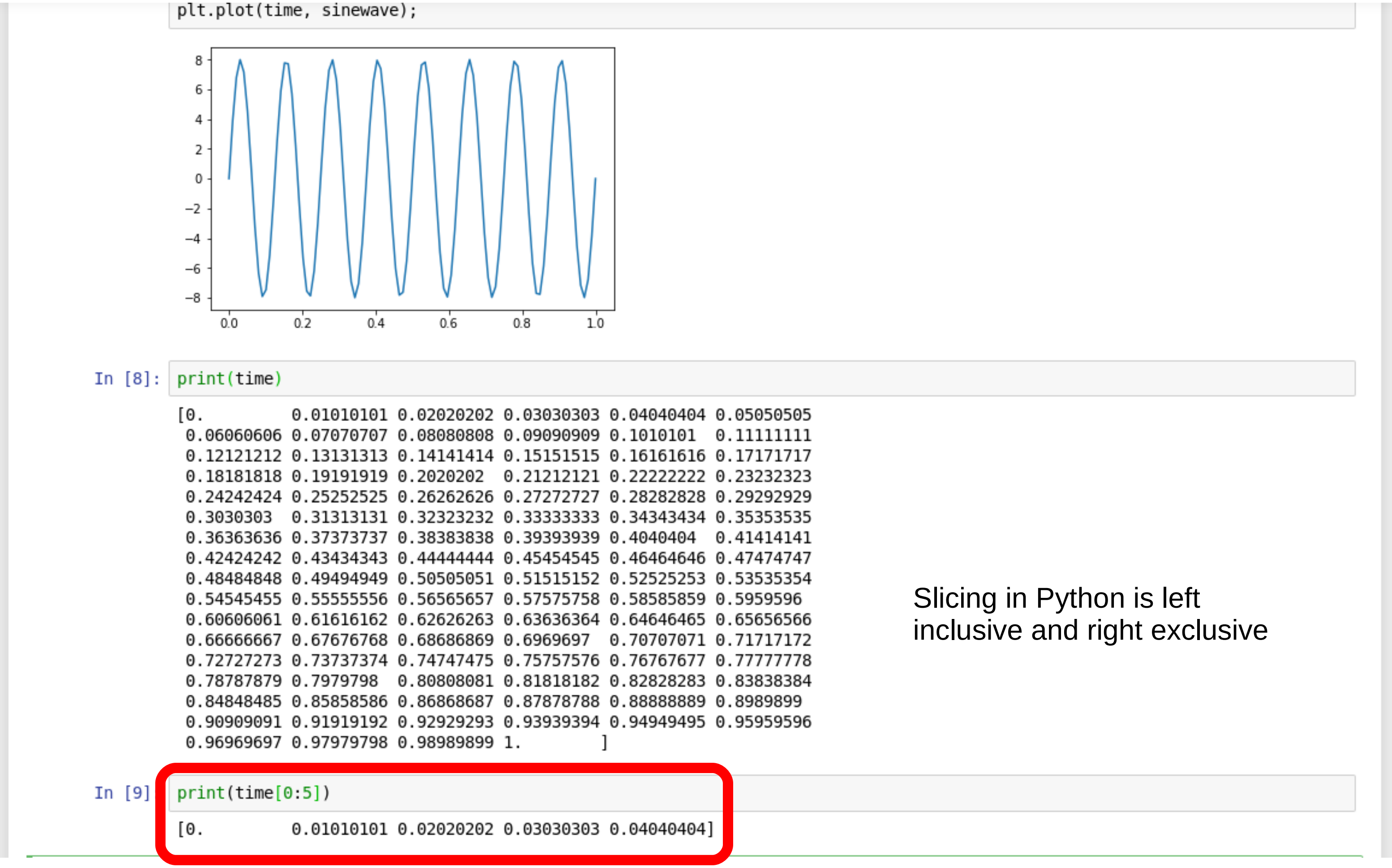
In [8]:

```
print(time)
```

```
[0.         0.01010101 0.02020202 0.03030303 0.04040404 0.05050505
 0.06060606 0.07070707 0.08080808 0.09090909 0.1010101  0.11111111
 0.12121212 0.13131313 0.14141414 0.15151515 0.16161616 0.17171717
 0.18181818 0.19191919 0.2020202  0.21212121 0.22222222 0.23232323
 0.24242424 0.25252525 0.26262626 0.27272727 0.28282828 0.29292929
 0.3030303  0.31313131 0.32323232 0.33333333 0.34343434 0.35353535
 0.36363636 0.37373737 0.38383838 0.39393939 0.4040404  0.41414141
 0.42424242 0.43434343 0.44444444 0.45454545 0.46464646 0.47474747
 0.48484848 0.49494949 0.50505051 0.51515152 0.52525253 0.53535354
 0.54545455 0.55555556 0.56565657 0.57575758 0.58585859 0.5959596
 0.60606061 0.61616162 0.62626263 0.63636364 0.64646465 0.65656566
 0.66666667 0.67676768 0.68686869 0.6969697  0.70707071 0.71717172
 0.72727273 0.73737374 0.74747475 0.75757576 0.76767677 0.77777778
 0.78787879 0.7979798  0.80808081 0.81818182 0.82828283 0.83838384
 0.84848485 0.85858586 0.86868687 0.87878788 0.88888889 0.8989899
 0.90909091 0.91919192 0.92929293 0.93939394 0.94949495 0.95959596
 0.96969697 0.97979798 0.98989899 1.        ]
```

Contents of numpy array “time”

Example 2: Sine wave



```
plt.plot(time, sinewave);
```

```
In [8]: print(time)
```

```
[0.         0.01010101 0.02020202 0.03030303 0.04040404 0.05050505
 0.06060606 0.07070707 0.08080808 0.09090909 0.1010101  0.11111111
 0.12121212 0.13131313 0.14141414 0.15151515 0.16161616 0.17171717
 0.18181818 0.19191919 0.2020202  0.21212121 0.22222222 0.23232323
 0.24242424 0.25252525 0.26262626 0.27272727 0.28282828 0.29292929
 0.3030303  0.31313131 0.32323232 0.33333333 0.34343434 0.35353535
 0.36363636 0.37373737 0.38383838 0.39393939 0.4040404  0.41414141
 0.42424242 0.43434343 0.44444444 0.45454545 0.46464646 0.47474747
 0.48484848 0.49494949 0.50505051 0.51515152 0.52525253 0.53535354
 0.54545455 0.55555556 0.56565657 0.57575758 0.58585859 0.5959596
 0.60606061 0.61616162 0.62626263 0.63636364 0.64646465 0.65656566
 0.66666667 0.67676768 0.68686869 0.6969697  0.70707071 0.71717172
 0.72727273 0.73737374 0.74747475 0.75757576 0.76767677 0.77777778
 0.78787879 0.7979798  0.80808081 0.81818182 0.82828283 0.83838384
 0.84848485 0.85858586 0.86868687 0.87878788 0.88888889 0.8989899
 0.90909091 0.91919192 0.92929293 0.93939394 0.94949495 0.95959596
 0.96969697 0.97979798 0.98989899 1.        ]
```

```
In [9] print(time[0:5])
```

```
[0.         0.01010101 0.02020202 0.03030303 0.04040404]
```

Slicing in Python is left inclusive and right exclusive

CONCLUSION

Conclusion

- (Hopefully) we have provided Matlab users with a rough introduction to coding in Python on Biowulf

Useful links

- www.scipy.org
- www.spyder-ide.org
- www.jupyter.org
- www.anaconda.com/distribution
- pyzo.org/python_vs_matlab.html
- docs.scipy.org/doc/numpy/user/numpy-for-matlab-users.html
- github.com/jrjohansson/scientific-python-lectures

Questions? Suggestions?

staff@hpc.nih.gov